LNCS 3138

Ana Fred
Terry Caelli
Robert P.W. Duin
Aurélio Campilho
Dick de Ridder (Eds.)

Structural, Syntactic, and Statistical Pattern Recognition

Joint IAPR International Workshops
SSPR 2004 and SPR 2004
Lisbon, Portugal, August 2004, Proceedings

S+SSPR 2004

IAPR

Springer

Ana Fred   Terry Caelli
Robert P.W. Duin   Aurélio Campilho
Dick de Ridder (Eds.)

Structural, Syntactic, and Statistical Pattern Recognition

Joint IAPR International Workshops
SSPR 2004 and SPR 2004
Lisbon, Portugal, August 18-20, 2004
Proceedings

Springer

Volume Editors

Ana Fred
Technical University of Lisbon, Telecommunications Institute
Instituto Superior TÃ©cnico, Department of Electrical and Computer Engineering
Av. Rovisco Pais, 1049-001 Lisbon, Portugal
E-mail: afred@lx.it.pt

Terry Caelli
University of Alberta, Department of Computing Science
Athabasca Hall, Edmonton, T6G 2E8 Alberta, Canada
E-mail: tcaelli@ualberta.ca

Robert P.W. Duin
Delft University of Technology
Department of Electrical Engineering, Mathematics and Computer Science
P.O. Box 5031, 2600 GA Delft, The Netherlands
E-mail: R.P.W.Duin@ewi.tudelft.nl

Aurélio Campilho
University of Porto, Institute of Biomedical Engineering, Faculty of Engineering
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
E-mail: campilho@fe.up.pt

Dick de Ridder
Delft University of Technology
Department of Electrical Engineering, Mathematics and Computer Science
P.O. Box 5031, 2600 GA Delft, The Netherlands
E-mail: D.deRidder@ewi.tudelft.nl

Library of Congress Control Number: 2004109599

CR Subject Classification (1998): I.5, I.4, I.2.10, I.3.5, G.2

ISSN 0302-9743
ISBN 3-540-22570-6 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik
Printed on acid-free paper SPIN: 11019022 06/3142 5 4 3 2 1 0

Preface

This volume contains all papers presented at SSPR 2004 and SPR 2004, hosted by the Instituto de Telecomunicações/Instituto Superior Técnico, Lisbon, Portugal, August 18–20, 2004.

This was the fourth time that the two workshops were held back-to-back. The SSPR was the tenth International Workshop on Structural and Syntactic Pattern Recognition, and the SPR was the fifth International Workshop on Statistical Techniques in Pattern Recognition. These workshops have traditionally been held in conjunction with ICPR (International Conference on Pattern Recognition), and are the major events for technical committees TC2 and TC1, respectively, of the International Association for Pattern Recognition (IAPR).

The workshops were closely coordinated, being held in parallel, with plenary talks and a common session on hybrid systems. This was an attempt to resolve the dilemma of how to deal with the need for narrow-focus specialized workshops yet accommodate the presentation of new theories and techniques that blur the distinction between the statistical and the structural approaches.

A total of 219 papers were received from many countries, with the submission and reviewing processes being carried out separately for each workshop. A total of 59 papers were accepted for oral presentation and 64 for posters. In addition, four invited speakers presented informative talks and overviews of their research. They were: Alberto Sanfeliu, from the Technical University of Catalonia, Spain; Marco Gori, from the University of Siena, Italy; Nello Cristianini, from the University of California, USA; and Erkki Oja, from Helsinki University of Technology, Finland, winner of the 2004 Pierre Devijver Award.

SSPR 2004 and SPR 2004 were sponsored by the IAPR, the Instituto Superior Técnico, Technical University of Lisbon and the Fundação Luso-Americana para o Desenvolvimento (FLAD).

We would like to express our sincere gratitude to all the members of the program committees for performing the hard work of reviewing the many submissions which led to a selection of high-quality papers. We would like to thank everyone who made this meeting possible: the authors for submitting papers, the invited speakers for accepting our invitation, the organizing committee, and the sponsors for their support.

We also appreciate the help of the editorial staff at Springer-Verlag and, in particular, Anna Kramer and Alfred Hofmann, for supporting this publication in the LNCS series; and the help of Piotr Juszczak during the final editing process.

August 2004

Ana Fred
Terry Caelli
Robert P.W. Duin
Aurélio Campilho
Dick de Ridder

SSPR and SPR 2004

General Chair

Ana Fred

Telecommunications Institute
Dept. of Electrical and Computer Engineering
Instituto Superior Técnico
Technical University of Lisbon
Lisbon, Portugal
afred@lx.it.pt

Local Chair

Mário Figueiredo

Telecommunications Institute
Dept. of Electrical and Computer Engineering
Instituto Superior Técnico
Technical University of Lisbon
Lisbon, Portugal
mtf@lx.it.pt

Webmaster

Dick de Ridder

Information & Communication Theory Group
Dept. of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Delft, The Netherlands
D.deRidder@ewi.tudelft.nl

Supported by

International Association for Pattern Recognition (IAPR)
Instituto Superior Técnico, Technical University of Lisbon
Fundação Luso-Americana para o Desenvolvimento (FLAD)

SSPR Committee

Co-chairmen

Ana Fred	Terry Caelli
Telecommunications Institute Dept. of Electrical and Computer Engineering Instituto Superior Técnico Technical University of Lisbon Lisbon, Portugal	Dept. of Computing Science The University of Alberta Edmonton, Alberta, Canada
afred@lx.it.pt	tcaelli@ualberta.ca

Program Committee

Keiichi Abe (Japan)	Gady Agam (USA)
Adnan Amin (Australia)	Walter F. Bischof (Canada)
Horst Bunke (Switzerland)	Kim Boyer (USA)
Francisco Casacuberta (Spain)	Juan Andrade Cetto (Spain)
Sven Dickinson (Canada)	Georgy Gimel'farb (New Zealand)
Edwin Hancock (UK)	Colin de la Higuera (France)
Jose M. Iñesta (Spain)	Xiaoyi Jiang (Germany)
Jean-Michel Jolion (France)	Walter Kropatsch (Austria)
Brian C. Lovell (Australia)	B. John Oommen (Canada)
Petra Perner (Germany)	Alberto Sanfeliu (Spain)
Gabriella Sanniti di Baja (Italy)	Karl Tombre (France)
Svetha Venkatesh (Australia)	Mario Vento (Italy)

SPR Committee

Co-chairmen

Aurélio Campilho	Robert P.W. Duin
Institute of Biomedical Engineering Faculty of Engineering University of Porto Porto, Portugal	Information & Communication Theory Group Dept. of Electrical Engineering, Mathematics and Computer Science Delft University of Technology Delft, The Netherlands
campilho@fe.up.pt	R.P.W.Duin@ewi.tudelft.nl

Program Committee

Reviewers

The Program Committees for both SPR and SSPR were kindly assisted by:

Luís A. Alexandre (Portugal)
Rene Alquezar (Spain)
Xaro Benavent (Spain)
Gustavo Carneiro (Canada)
Enric Celaya (Spain)
Francesco Colace (Italy)
Donatello Conte (Italy)
Massimo De Santo (Italy)
Claudio De Stefano (Italy)
Jan Flusser (Czech Republic)
Pasquale Foggia (Italy)
Jiří Grim (Czech Republic)
Corrado Guidobaldi (Italy)
Michael Haindl (Czech Republic)
Allan Hanbury (Austria)
Yll Haxhimusa (Austria)
Piotr Juszczak (The Netherlands)
Thomas Landgrebe (The Netherlands)
Marco Loog (The Netherlands)
Angelo Marcelli (Italy)
Jocelyn Marchadier (Austria)
Ana Mendonç (Portugal)
Shunji Mori (Japan)
Hiroshi Murase (Japan)
Hiromasa Nakatani (Japan)
Michel Neuhaus (Switzerland)
Pavel Paclík (The Netherlands)
Elżbieta Pękalska (The Netherlands)
Pedro Pina (Portugal)
Marcel J.T. Reinders (The Netherlands)
Hitoshi Saji (Japan)
Carlo Sansone (Italy)
Michael S. Brown (Hong Kong)
Cristian Sminchisescu (Canada)
Petr Somol (Czech Republic)
Takahiro Sugiyama (Japan)
Chew Lim Tan (Singapore)
David M.J. Tax (The Netherlands)
Francesco Tortorella (Italy)
John Tsotsos (Canada)
Serguei Verzakov (The Netherlands)
Teresa Vidal (Spain)
Lodewyk F.A. Wessels (The Netherlands)
Ahmed M. Zeki (Malaysia)

Table of Contents

Poster Papers

Finding Clusters and Components by Unsupervised Learning

Erkki Oja[⋆]

Helsinki University of Technology, Neural Networks Research Centre
P.O.Box 5400, 02015 HUT, Finland
erkki.oja@hut.fi

Abstract. We present a tutorial survey on some recent approaches to unsupervised machine learning in the context of statistical pattern recognition. In statistical PR, there are two classical categories for unsupervised learning methods and models: first, variations of Principal Component Analysis and Factor Analysis, and second, learning vector coding or clustering methods. These are the starting-point in this article. The more recent trend in unsupervised learning is to consider this problem in the framework of probabilistic generative models. If it is possible to build and estimate a model that explains the data in terms of some latent variables, key insights may be obtained into the true nature and structure of the data. This approach is also reviewed, with examples such as linear and nonlinear independent component analysis and topological maps.

1 Introduction: Supervised and Unsupervised Learning from Data

In statistical pattern recognition, machine learning from a training set is an essential technique. If the classes of the training vectors are known, supervised methods are used to build the classifiers. If class information does not exist, one has to resort to unsupervised methods. Also in the preliminary stage of feature extraction unsupervised methods are mostly used [13, 30].

The optimality of supervised classifiers is given by the theoretical limit of the Bayes decision rule. For unsupervised methods, no such clear optimality criterion exists. Usually, the result of unsupervised learning is a new explanation or representation of the observation data, which will then lead to improved future decisions. In statistical pattern recognition, the representation may be a clustering of the data, a discrete map, or a continuous lower-dimensional manifold in the vector space of observations, which explains their structure and may reveal their underlying causes [13].

Unsupervised learning seems to be the basic mechanism for sensory adaptation in the animal brain, e.g. in the visual pathway [4]. In pattern recognition, it is a highly powerful and promising approach to some practical problems like

[⋆] This work was supported by the Academy of Finland as part of its Center of Excellence project "New Information Processing Principles".

data mining and knowledge discovery from very large databases, or new modes of human-computer interactions in which the software adapts to the requirements and habits of the human user by observing her behaviour. For an excellent collection of recent articles on unsupervised learning, see [23].

The current trend in unsupervised learning is to consider this problem in the framework of probabilistic generative models. The concept of a generative model is very general and potentially powerful. In fact, as discussed by Roweis and Ghahramani [48], a large number of central techniques like FA, PCA, mixtures of Gaussians, vector quantization, and also dynamical models like Kalman filters or Hidden Markov Models, can be presented in a unified framework of unsupervised learning under a single basic generative model. If it is possible to build and estimate a model that explains the data in terms of some latent variables, key insights may be obtained into the true nature and structure of the data. Operations like prediction and compression become easier and rigorously justifiable. In this paper, we take a brief look at such models, which reveal the structure of the data by projections on linear or nonlinear structures, spanned by components or clusters hidden in the data.

The first class of unsupervised learning methods we consider in Section 2 is motivated by standard statistical methods like PCA or FA, which give a reduced subset of linear combinations of the original input variables. Also nonlinear variants have been suggested, such as autoassociative neural networks, kernel PCA, principal curves and surfaces, and mixtures of local PCA's. A more recent model in this category is that of independent components, which would maximally reduce the redundancy between the latent variables even in the case that gaussianity does not hold. This leads to the techniques of Independent Component Analysis (ICA) and Blind Source Separation (BSS) [27]. In the latter technique, a set of parallel time signals such as speech waveforms, electromagnetic measurements from the brain, or financial time series, are assumed to be linear combinations of underlying independent latent variables. The variables, called independent components, are found by efficient ICA learning rules. ICA is a linear technique, but nonlinear variants have been proposed recently, and some approaches along Nonlinear ICA or Nonlinear FA are also pointed out in Section 2.

The second class of methods is close to clustering or visualization by projecting the data on a nonlinear low-dimensional grid. A typical application is data mining or profiling from massive databases. It is of interest to find out what kind of typical clusters there are among the data records, and what is the relation between the clusters. A competitive learning algorithm gives an efficient solution to this problem. Section 3 briefly reviews a well-known competitive learning system, the Self-Organizing Map (SOM) [36], and a related generative latent variable model GTM [7].

2 Finding Independent Components

2.1 Principal Component Analysis

Principal component analysis (PCA) and the closely related Karhunen-Loève Transform, or the Hotelling Transform, as well as Factor Analysis (FA), are classical techniques in statistical data analysis, feature extraction, and data compression [14, 40, 61]. Given a set of multivariate measurement vectors $\mathbf{x}(1), \dots \mathbf{x}(T)$, the purpose is to find a smaller set of variables with less redundancy, that would give as good a representation as possible. The redundancy is measured through second-order statistics only and is removed by decorrelation. This means rotating the data into a new coordinate system given by the eigenvectors of the data covariance matrix.

It is not always feasible to solve the eigenvectors by standard numerical methods. In an on-line data compression application like image or speech coding, the data samples $\mathbf{x}(t)$ arrive at high speed, and it may not be possible to estimate the covariance matrix and solve the eigenvector-eigenvalue problem once and for all.

An alternative is to derive gradient ascent algorithms or other on-line methods for PCA. The algorithms will then converge to the solution of the problem, that is, to the eigenvectors. The advantage of this approach is that such algorithms work on-line, using each input vector $\mathbf{x}(t)$ once as it becomes available and making an incremental change to the eigenvector estimates, without computing the covariance matrix at all. This approach is the basis of the PCA neural network learning rules introduced by the author [39, 42]. Other related on-line algorithms have been introduced in [16, 49, 14, 60]. Some of them, like the APEX algorithm by Diamantaras and Kung [14], are based on a feedback neural network. Also minor components defined by the eigenvectors corresponding to the smallest eigenvalues can be computed by similar algorithms [42].

Another possibility for PCA computation in neural networks is the Multi-Layer Perceptron network, which learns using the back-propagation algorithm (see [20]) in unsupervised autoassociative mode. In autoassociative mode, the same vectors $\mathbf{x}$ are used both as inputs and as desired outputs in back-propagation learning. This network with nonlinear hidden layer was suggested for data compression by [11], and it was shown to be closely connected to the theoretical PCA by [8]. It is not equivalent to PCA, however, as shown by [31], unless the hidden layer is linear. A much more powerful network is obtained if more hidden layers are added. For instance, a 5 - layer autoassociative MLP is able to compute in principle any smooth nonlinear mapping between the inputs and the central hidden layer, and another mapping between the central hidden layer and the outputs. This is due to the two extra nonlinear hidden layers; see e.g. [41]. This network is one way to compute a nonlinear PCA expansion.

Other prominent approaches to extend PCA to nonlinearities are the kernel PCA [52] and the method of Principal Curves [19]. PCA can be "kernelized" because it is a second-order statistical technique. Yet another approach to construct nonlinear PCA manifolds is to combine the two major unsupervised learning paradigms - PCA and vector coding (VQ) - using mixtures of local linear models, for example PCA's, in which the data cloud is first clustered or parcelled using VQ, and then a separate linear model is fitted to each of the clusters around the code vector. This notion has been formalized by several authors [22, 47, 48, 55, 62]. It is closely related to the conventional technique of semiparametric density estimation, the Mixture of Gaussians (MoG) model widely used in clustering and data modelling. However, instead of using full covariance matrices for the component gaussians, the local linear models constrain the covariances in a natural and easily adjustable way.

Another linear projection technique is Factor Analysis (FA) [18], in which a generative latent variable model is assumed for $\mathbf{x}$:

$$\mathbf{x} = \mathbf{A}\mathbf{y} + \mathbf{n}. \tag{1}$$

With certain assumptions on the additive noise, FA and PCA produce the same solution. PCA, too, can be derived from a generative model in the technique called Probabilistic PCA [55] or Principal Factor Analysis [18].

2.2 Independent Component Analysis

In Independent Component Analysis (ICA) [1, 5, 9, 10, 25, 27, 32, 34, 43] the same model (1) is assumed, but now the assumption on y_i is much stronger: we require that they are statistically independent and nongaussian. Interestingly, then the ambiguity in Factor Analysis disappears and the solution, if we can find one, is (almost) unique.

In the simplest form of ICA, the additive noise $\mathbf{n}$ is not included and the standard notation for the independent components or sources is s_i; thus the ICA model for observation vectors $\mathbf{x}$ is

$$\mathbf{x} = \mathbf{A}\mathbf{s}. \tag{2}$$

It is assumed that both $\mathbf{x}$ and $\mathbf{s}$ are zero mean. The observations x_i are now linear combinations or mixtures of the sources s_j. The matrix $\mathbf{A}$ is called in ICA the mixing matrix. The model looks deceptively simple but is not, because both $\mathbf{A}$ and $\mathbf{s}$ are unknown and must be estimated from a sample of the observations $\mathbf{x}$.

We may further assume that the dimensions of $\mathbf{x}$ and $\mathbf{s}$ are the same. If originally $\dim \mathbf{x} < \dim \mathbf{s}$, or there are more sources than observed variables, then the problem becomes quite difficult - see [27]. If, on the other hand, $m = \dim \mathbf{x} > \dim \mathbf{s} = n$, then model (2) implies that there is redundancy in $\mathbf{x}$ which is revealed and can be removed by performing PCA on $\mathbf{x}$. This is done as follows.

We can write the $m \times m$ covariance matrix of $\mathbf{x}$ as

$$\mathbf{C}_{\mathbf{x}} = \mathbf{A}\mathrm{E}\{\mathbf{s}\mathbf{s}^T\}\mathbf{A}^T = \mathbf{A}\mathbf{A}^T. \tag{3}$$

We have used the knowledge that matrix $\mathrm{E}\{\mathbf{s}\mathbf{s}^T\}$ is diagonal, due to the fact that the elements of $\mathbf{s}$ are zero mean and independent; if we further absorb their variances to matrix $\mathbf{A}$ and assume that $\mathrm{E}\{s_i^2\} = 1$, then it holds $\mathrm{E}\{\mathbf{s}\mathbf{s}^T\} = \mathbf{I}$. Now, matrix $\mathbf{A}$ is an $m \times n$ matrix and so matrix $\mathbf{C}_{\mathbf{x}} = \mathbf{A}\mathbf{A}^T$ is an $m \times m$ matrix with rank n. It will have only n nonzero eigenvalues. Let us denote the diagonal $n \times n$ matrix of the nonzero eigenvalues of $\mathbf{C}_{\mathbf{x}}$ by $\mathbf{D}$, the orthonormal eigenvectors of $\mathbf{C}_{\mathbf{x}}$ by $\mathbf{e}_1, ..., \mathbf{e}_m$, and the orthogonal matrix that has the n first

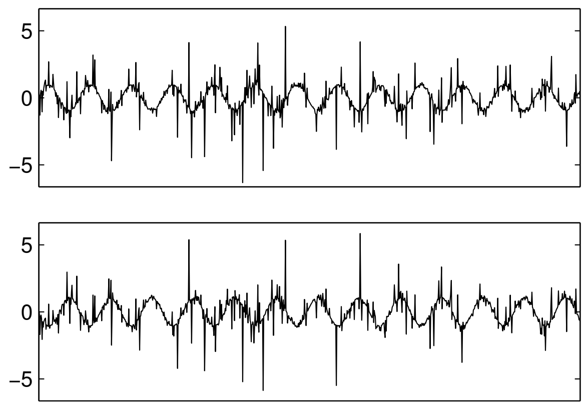


Fig. 1. Mixed signals

ones as columns by $\mathbf{E}$. Thus $\mathbf{E}$ is $m \times n$. Make now a linear transformation for the m - dimensional observation vectors $\mathbf{x}$:

$$\mathbf{x}' = \mathbf{D}^{-1/2}\mathbf{E}^T\mathbf{x}. \tag{4}$$

For the covariance matrix of the transformed n - dimensional vector $\mathbf{x}'$ it holds:

$$\mathrm{E}\{\mathbf{x}'\mathbf{x}'^T\} = \mathbf{D}^{-1/2}\mathbf{E}^T\mathbf{C}_\mathbf{x}\mathbf{E}\mathbf{D}^{-1/2} = \mathbf{D}^{-1/2}\mathbf{E}^T\mathbf{E}\mathbf{D}\mathbf{D}^{-1/2} = \mathbf{I}. \tag{5}$$

This transformation is called whitening. Let us assume in the following that whitening has always been performed in the model, and denote simply by $\mathbf{x}$ the whitened observation vector whose dimension is the same as that of the source vector $\mathbf{s}$.

Whitening has another desirable side-effect, which can be seen by noting from eq. (3) that now $\mathbf{A}\mathbf{A}^T = \mathbf{I}$. But this means that matrix $\mathbf{A}$ is an orthogonal matrix, for which $\mathbf{A}^{-1} = \mathbf{A}^T$. So, if we knew matrix $\mathbf{A}$, we could directly solve the unknown source vector $\mathbf{s}$ from the model by

$$\mathbf{s} = \mathbf{A}^T\mathbf{x}.$$

It is an interesting finding that very few assumptions suffice for solving the mixing matrix and, hence, the sources. All we need is the assumption that the sources s_i are statistically independent and nongaussian. Consider the following simple example: we have two signals, shown in Fig. 1, that are linear combinations or mixtures of two underlying independent nongaussian source signals. This example is related to model (2) in such a way that the elements x_1, x_2 of the random vector $\mathbf{x}$ in (2) are the amplitudes of the two signals in Fig. 1. The signals provide a sample $\mathbf{x}(1), \ldots \mathbf{x}(T)$ from this two-dimensional random vector. The joint histogram of the sample vectors is plotted in Fig. 2; each point in the scatter plot corresponds to one time point in Fig. 1. The vector $\mathbf{x}$ is now white in the sense that x_1 and x_2 are zero mean, uncorrelated, and have unit variance.

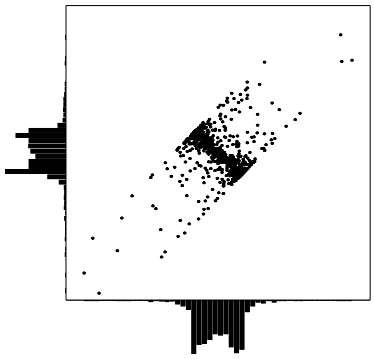


Fig. 2. Histogram of the two amplitudes of the mixed signals $x.,x.$

This may not be apparent from the histogram but can be verified by estimating the covariance matrix of all the points.

The example suggests a method that in fact is highly useful and forms the basis of some practical ICA algorithms. Consider a line passing through the origin at the center of the data cloud in Fig. 2. Denote a unit vector defining the direction of the line by $\mathbf{w}$. Then the projection of a data point $\mathbf{x}$ on the line is given by $y = \mathbf{w}^T\mathbf{x}$. This can be considered as a random variable whose density is approximated by the histogram of the projections of all the data points in the cloud on this line. No matter what is the orientation of the line, it always holds that y has zero mean and unit variance. The unit variance is due to $\mathrm{E}\{y^2\} = \mathrm{E}\{(\mathbf{w}^T\mathbf{x})^2\} = \mathbf{w}^T\mathrm{E}\{\mathbf{x}\mathbf{x}^T\}\mathbf{w} = \mathbf{w}^T\mathbf{w} = 1$ where we have used the facts that $\mathbf{x}$ is white and $\mathbf{w}$ has unit norm.

However, it is easy to see from Fig. 2 that the density of y will certainly vary as the orientation of the line varies, meaning that all the moments of y cannot stay constant. In fact, any other moment than the first and second ones is not constant. What is most important is that any such moment, say, $\mathrm{E}\{y^3\}$ or $\mathrm{E}\{y^4\}$ or in fact $\mathrm{E}\{G(y)\}$, with $G(y)$ a nonlinear and non-quadratic function, will attain a number of maxima and minima when the orientation of the line goes full circle, and some of these extrema coincide with orientations in which the 2-dimensional density factorizes into the product of its marginal densities - meaning independence.

In Fig. 3, the coordinate system has been rotated so that the fourth moment $\mathrm{E}\{y^4\}$ is maximal in the vertical direction and minimal in the horizontal direction. We have found two new variables $y_1 = \mathbf{w}_1^T\mathbf{x}$ and $y_2 = \mathbf{w}_2^T\mathbf{x}$, with $\mathbf{w}_1, \mathbf{w}_2$ orthonormal, that satisfy

$$p(y_1, y_2) = p(y_1)p(y_2)$$

with $p(.)$ the appropriate probability densities. The variables are thus independent and it holds

$$\mathbf{y} = \mathbf{W}\mathbf{x}$$

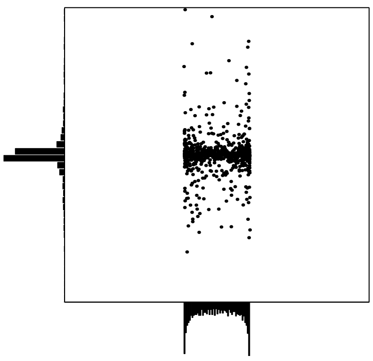


Fig. 3. Histogram of the two amplitudes of the separated signals $y_{\bullet}, y_{\bullet}$

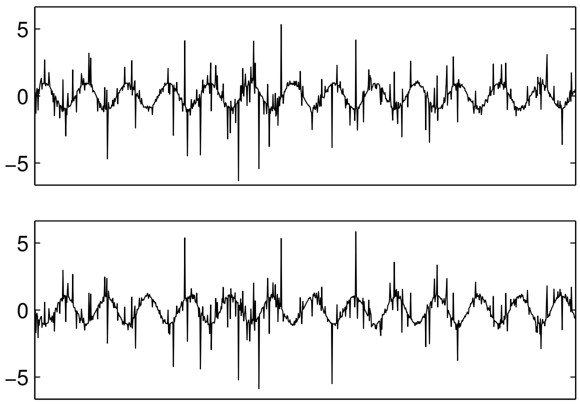


Fig. 4. Separated signals

where $\mathbf{W} = (\mathbf{w}_1\mathbf{w}_2)^T$. We have solved the inverse of the model (2) and obviously found the mixing matrix: $\mathbf{A} = \mathbf{W}^T$.

Fig. 4 shows y_1, y_2 again arranged in their correct time order. It is seen that they form two signals, one a random nongaussian noise and the other one a deterministic sinusoid. These were in fact the original signals that were used to make the artificial mixtures in Fig. 1. In the context of separating time series or signals, the ICA technique is an example of blind signal separation.

The above illustrative example can be formalized to an efficient mathematical algorithm. What we need is a numerical method to maximize, say, the fourth moment $\mathrm{E}\{y^4\}$ in terms of a unit norm weight vector $\mathbf{w}$. A possibility is gradient ascent: the gradient of $\mathrm{E}\{y^4\}$ with respect to $\mathbf{w}$ is $4\mathrm{E}\{y^3\mathbf{x}\} = 4\mathrm{E}\{(\mathbf{w}^T\mathbf{x})^3\mathbf{x}\}$. However, gradient methods are notoriously slow. A better idea is a fast algorithm with higher-order convergence speed. Such a method is provided by the FAstICA algorithm. For finding one independent component (one weight vector $\mathbf{w}$), the algorithm is as follows:

1. Choose the initial value randomly for the weight vector $\mathbf{w}$.
2. Repeat Steps 3,4 until the algorithm has converged:
3. Normalize $\mathbf{w}$ to unit norm.
4. Update $\mathbf{w}$ by

$$\mathbf{w} \leftarrow \mathrm{E}\{(\mathbf{w}^T\mathbf{x})^3\mathbf{x}\} - 3\mathbf{w} \tag{6}$$

This algorithm was introduced in [25] and further extended and analyzed in [26]; for a detailed review, see [27]. The FastICA algorithm is available in public-domain software [15] from the author's web pages. The algorithm can be run either in a deflation mode, in which the orthogonal weight vectors (columns of the mixing matrix $\mathbf{A}$) can be found one at a time, or in a parallel mode, in which all the independent components and the whole matrix $\mathbf{A}$ are solved in one iteration.

An analysis of the local maxima and minima of a general cost function $\mathrm{E}\{G(y)\} = \mathrm{E}\{G(\mathbf{w}^T\mathbf{x})\}$ over the unit sphere $\|\mathbf{w}\| = 1$ was made by the author in [45]. The result is

Theorem. Under the linear mixing model $\mathbf{x} = \mathbf{A}\mathbf{s}$, with whitened $\mathbf{x}$ (hence: orthogonal $\mathbf{A}$), the local maxima (resp. minima) of $\mathrm{E}\{G(\mathbf{w}^T\mathbf{x})\}$ under the constraint $\|\mathbf{w}\| = 1$ include those columns $\mathbf{a}_i$ of the mixing matrix $\mathbf{A}$ such that the corresponding sources s_i satisfy

$$\mathrm{E}\{s_i g(s_i) - g'(s_i)\} > 0 \text{ (resp. } < 0) \tag{7}$$

where $g(.)$ is the derivative of $G(.)$.

The Theorem essentially says that all the columns of the mixing matrix will be among the local minima or maxima of $\mathrm{E}\{G(\mathbf{w}^T\mathbf{x})\}$, but there may be also other extrema. The condition (7) states that some columns (and the corresponding sources) are found by minimizing, others by maximizing. For the case $G(y) = y^4$, (7) becomes

$$\mathrm{E}\{s_i^4 - 3\} > 0$$

(note that the sources have unit variances). The term on the left hand side is the kurtosis of s_i. Thus, the positively kurtotic sources are found at the local maxima of $\mathrm{E}\{(\mathbf{w}^T\mathbf{x})^4\}$ and vice versa. For other cost functions $G(y)$, the condition (7) always splits the sources in two groups, too.

In [27], the above method of fourth order moment maximization is shown to be an example of a powerful criterion of finding maximally nongaussian orthogonal directions through the multidimensional density $p(\mathbf{x})$. Cost functions like maximum likelihood or minimal mutual information are shown to be intimately related to this basic criterion. Other algorithms to solving the basic linear ICA model have been reported e.g. by [1, 5, 9, 10, 32], as reviewed in [27]. Especially, if the sources are actually signals with time structure, not just samples of random variables, then blind separation can be achieved using either temporal correlations [6] or nonstationarity [46].

2.3 Nonlinear Factor and Independent Component Analysis

The model (2) is extremely simple and can be extended in several directions. If the additive noise cannot be assumed to be zero, we have the noisy ICA model, also termed independent factor analysis [2]. This is due to the fact that it is otherwise similar to the factor analysis model (1), with the difference that the factors y_i are not uncorrelated (thus independent) gaussians, but rather independent nongaussians. Some solution methods are reviewed in [27].

Another extension is nonlinear ICA and FA. Instead of the linear models (2),(1), consider

$$\mathbf{x} = \mathbf{f}(\mathbf{y}, \mathbf{M}) + \mathbf{n} \tag{8}$$

with $\mathbf{f}$ a nonlinearity parameterized by an array of parameters $\mathbf{M}$. Vector $\mathbf{y}$ gives a number of latent variables and $\mathbf{n}$ is again gaussian noise. If we assume that the prior $p(\mathbf{y})$ for $\mathbf{y}$ is gaussian with unit (or diagonal) covariance, making the elements y_i independent, the model (14) may be called nonlinear factor analysis. A further extension would be $p(\mathbf{y})$ that is nongaussian but factorizable so that the y_i are independent; then the model becomes nonlinear independent component analysis.

Several authors have attacked this problem. The baseline is that the problem is ill-defined. Under very general assumptions, a random vector can be transformed nonlinearly into another random vector that has independent elements [28], but there is no guarantee that the independent elements are the original sources. Therefore, the solution can only be sought with restrictions that somehow regularize the problem. Typical such restrictions are post-nonlinear mixtures and some special cases that can be reduced to linear mixtures with simple mappings. For general nonlinearities, there are a variety of methods, some of them rather ad hoc; for a review, see [33].

Recently, Valpola [56] used an approximation for the nonlinear function $\mathbf{f}(\mathbf{y}, \mathbf{M})$ in the model, that was based on a Multilayer Perceptron (MLP) network with one hidden layer. It is well-known [24, 17] that this function can approximate uniformly any continuous functions on compact input domains and it is therefore suitable for this task. Then the model becomes

$$\mathbf{x} = \mathbf{B}\phi(\mathbf{A}\mathbf{y} + \mathbf{a}) + \mathbf{b} + \mathbf{n} \tag{9}$$

where $\mathbf{A}, \mathbf{a}$ are the weight matrix and offset vector of the hidden layer, ϕ is the sigmoidal nonlinearity, typically a tanh or $\sinh^{-1}$ function, and $\mathbf{B}, \mathbf{b}$ are the weight matrix and offset vector of the linear output layer. It is understood that ϕ is applied to its argument vector element by element. In practice, there is a training sample $\mathbf{x}(1), ..., \mathbf{x}(T)$, and we wish to solve from the model the corresponding source or factor vectors $\mathbf{y}(1), ..., \mathbf{y}(T)$.

The problem now is that, contrary to the usual supervised learning situations, the inputs to the MLP are not known and therefore back-propagation type of learning rules cannot be used for finding the unknown parameters. The idea in [56] is to use a purely Bayesian approach called ensemble learning. The cost function is the Kullback - Leibler divergence between the true posterior probability for the parameters, given the observations, and an approximation of that density. Denote the set of all the unknown parameters by $\mathbf{\Theta} = \{\mathbf{Y}, \mathbf{M}\}$. There the vector $\mathbf{Y}$ contains all the unknown source vectors $\mathbf{y}(1), ..., \mathbf{y}(T)$, while $\mathbf{M}$ contains the weights of the MLP network that define the unknown nonlinear function $\mathbf{f}$, and also the parameters of the gaussian noise $\mathbf{n}$. In addition, because this is a Bayesian model, it includes hyperparameters defining the distributions of the weights. Denote the sample of observations by $\mathbf{X} = \mathbf{x}(1), ..., \mathbf{x}(T)$.

We can write for the posterior density of the parameters

$$p(\mathbf{\Theta}|\mathbf{X}) = p(\mathbf{Y}, \mathbf{M}|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{Y}, \mathbf{M})p(\mathbf{Y}|\mathbf{M})p(\mathbf{M})}{p(\mathbf{X})}. \tag{10}$$

The first term $p(\mathbf{X}|\mathbf{Y}, \mathbf{M})$ is obtained from the data model (9); it is simply a product of gaussians with means $\mathbf{B}\phi(\mathbf{A}\mathbf{y}(t) + \mathbf{a}) + \mathbf{b}$. Likewise, the terms $p(\mathbf{Y}|\mathbf{M})$ and $p(\mathbf{M})$ are obtained as products of gaussians, when we assume mutually independent gaussian priors for all the parameters. The term $p(\mathbf{X})$ does not contain any unknown parameters and can be omitted.

This density is now approximated by another density $q(\mathbf{\Theta})$ - the ensemble - that has a simple form [56]: it is a gaussian with diagonal covariance. Then the KL divergence

$$C_{KL} = \int d\mathbf{\Theta} q(\mathbf{\Theta}) \log \frac{q(\mathbf{\Theta})}{p(\mathbf{\Theta}|\mathbf{X})} \tag{11}$$

also obtains a relatively simple form, splitting into the expectations of many simple terms. It can be minimized by a suitable numerical method.

In [56], several applications with real data are shown. The model is also extended to a dynamical model, similar to an extended Kalman filter but with unknown parameters, and very promising results are obtained in case studies [57, 29].

3 The Self-organizing Map

3.1 The Basic SOM

One of the best-known learning systemss in the unsupervised category is the Self-Organizing Map (SOM) introduced by Kohonen [36]. It belongs to the class of vector coding algorithms. In vector coding, the problem is to place a fixed number of vectors, called codewords, into the input space which is usually a high-dimensional vector space. The dimension of the data vectors is determined by the problem and can be very large. In the WEBSOM system [37] for organizing collections of text documents, the dimensionality of the data in the largest applications is about $n = 50,000$ and the size of the training sample is about $T = 7,000,000$.

A well-known method for vector coding is the Linde-Buzo-Gray (LBG) algorithm, which is very similar to the k - means clustering algorithm [13]. Assume a set of nodes which are numbered by index $i = 1, \ldots, k$, and assume that each node i has a weight vector $\mathbf{w}_i$ that has the same dimension as the input vectors $\mathbf{x}$ that we wish to cluster. In k - means clustering, the goal is to place the weight vectors (codewords) into the input space in such a way that the average squared distance from each $\mathbf{x}$ to its closest codeword is minimized. In the Self - Organizing Map (SOM), there is an extra feature compared to mere clustering: nodes are spatially arranged to a 1-, 2- or multidimensional lattice, such that each node has a set of neighbors. The goal of SOM learning is not only to find the most representative code vectors for the input training set in the sense of minimum distance, but at the same time to form a topological mapping from the input space to the grid of nodes. This idea originally stems from the modelling of the topographic maps on the sensory cortical areas of the brain. A related early work in neural modelling is [38].

For any data point $\mathbf{x}$ in the input space, one or several of the codewords are closest to it. Assume that $\mathbf{w}_i$ is the closest among all codewords:

$$\|\mathbf{x} - \mathbf{w}_i\| = min\|\mathbf{x} - \mathbf{w}_j\|, j = 1, ..., k \tag{12}$$

The unit i having the weight vector $\mathbf{w}_i$ is then called the best-matching unit (BMU) for vector $\mathbf{x}$. The well-known Kohonen algorithm for self-organization of the code vectors is as follows [36]:

1. Choose initial values for the weight vectors $\mathbf{w}_i$.
2. Repeat Steps 3,4 until the algorithm has converged:
3. Draw a sample vector $\mathbf{x}$ from the training set and find the best matching unit $i = i(\mathbf{x})$ according to Eq. (12).
4. Adjust the weight vectors of all units by

$$\mathbf{w}_j \leftarrow \mathbf{w}_j + \gamma * h_r * (\mathbf{x} - \mathbf{w}_j) \tag{13}$$

where γ is a gain factor and h_r is a function of the distance $r = \|i - j\|$ of units i and j measured along the lattice.

There are several choices for the initial values and for the neighborhood function h_r; these, as well as the convergence and the mathematical properties of this algorithm have been considered by several authors, e.g. [36, 47, 44, 58]. For SOM learning, topology preservation, and its relation to a cost function, see [59, 12, 21]. A more efficient learning rule for the SOM is the batch algorithm, covered e.g. in [36]. The 2-dimensional map is also a powerful tool for data visualization: e.g., a color code can be used in which each unit has its own characteristic color. For a public domain software implementation of the SOM, with various graphical tools for map presentations as well as with preprocessing methods, see [54]. A database of well over four thousand applications of SOM is given by [53].

3.2 The Generative Topographic Map

There is a probabilistic generative model that is close to the SOM, the Generative Topographic Map (GTM) [7], in which the vectors $\mathbf{x}$ are expressed in terms of a number of latent variables, which are defined on a similar lattice or grid as the nodes in the SOM. Assume a grid with dimension l (usually, this would be equal to 2, at most 3), and assume there are k nodes $\mathbf{y}_i,\ i = 1, \ldots, k$ on the grid. Assume a random latent variable $\mathbf{y}$, whose values are concentrated at these nodes. Let us make a nonlinear mapping from the l-dimensional random variable $\mathbf{y}$ to the original n -dimensional vectors $\mathbf{x}$:

$$\mathbf{x} = \mathbf{f}(\mathbf{y}, \mathbf{M}) + \mathbf{n} \tag{14}$$

where $\mathbf{M}$ is an array of parameters of the nonlinear function $\mathbf{f}$, and $\mathbf{n}$ is additive noise. The form of the function $\mathbf{f}$ is assumed to be determined except for the unknown parameters. The model (14) is the generative latent variable model of the GTM method. It means that the data $\mathbf{x}$ are basically concentrated on an l-dimensional nonlinear manifold in the data space, except for the additive noise. The k vectors $\mathbf{w}_i = \mathbf{f}(\mathbf{y}_i, \mathbf{M})$ that are the images of the node points $\mathbf{y}_i$ are analogous to the weight vectors or codewords of the SOM. If $\mathbf{f}$ is smooth, a topographic ordering for the codewords is automatically guaranteed, because such an ordering is valid for the points $\mathbf{y}_i$. The GTM also has the advantage that it postulates a smooth manifold that naturally interpolates between the code vectors $\mathbf{w}_i$.

If we assume that the noise has a radially symmetrical gaussian density, then the density of $\mathbf{x}$, given $\mathbf{y}$, becomes a mixture of gaussians, having a separate gaussian density around each of the code vectors $\mathbf{w}_i = \mathbf{f}(\mathbf{y}_i, \mathbf{M})$. From this, the likelihood function for the parameters $\mathbf{M}, \beta$ follows immediately. The EM algorithm can now be used to numerically solve the parameters by maximum likelihood, due to the mixture of gaussians form of the density - for details, see [7]. The reference also discusses the similarities and differences between GTM and SOM.

4 Conclusions

The two main paradigms of unsupervised machine learning in statistical pattern recognition have been reviewed: the extensions to the Principal Component Analysis technique, and the clustering, vector coding, and topological mapping technique. The first class of methods form a continuous linear or nonlinear transformation of the original input vectors to feature vectors of lower dimensionality, and are especially useful in feature extraction. The reduced representation given by the feature vectors would typically be input to a classifier.

The second class of methods are able to map highly nonlinear input data manifolds onto low dimensional lattices, preserving optimally the mutual topological relations of input vectors. Thus these methods, notably the Self-Organizing Map (SOM), are suitable for data clustering and visualization. The applications range from industrial quality control to financial data mining. Also generative latent variable versions for these basic models and their combinations were reviewed.

This paper was a review of the essential principles and theory underlying unsupervised learning, with some central references cited. It is not possible here to give even a rudimentary list of applications of these techniques. There are available good text-books that cover some of the major approaches [23, 36, 14, 27].

References

1. Amari, S.- I., Cichocki, A. and Yang, H., "A new learning algorithm for blind source separation". In *Advances in Neural Information Processing Systems 8*, Cambridge: MIT Press, 1996, pp. 757 - 763.
2. Attias, H., "Independent factor analysis", *Neural Computation 11 (4)*, 1999, pp. 803 - 851.
3. Baldi, P. and Hornik, K., "Learning in linear neural networks: a survey", *IEEE Trans. Neural Networks 6 (4)*, 1995, pp. 837 - 858.
4. Barlow, H., "Unsupervised learning", *Neural Computation 1*, 1989, pp. 295 - 311.
5. Bell, A. and Sejnowski, T., "An information-maximization approach to blind separation and blind deconvolution", *Neural Computation 7*, 1995, pp. 1129 - 1159.
6. Belouchrani, A., Meraim, K., Cardoso, J-F., and Moulines, E., "A blind source separation technique based on second order statistics", *IEEE Trans. Signal Proc. 45*, 1997, pp. 434 - 444.
7. Bishop, C., Svensen, M and Williams, C., "GTM: the generative topographic mapping", *Neural Computation 10*, 1998, pp. 215 - 234.
8. Bourlard, H. and Kamp, Y., "Auto-association by multilayer Perceptrons and singular value decomposition", *Biol. Cybernetics 59*, 1988, pp. 291 - 294.
9. Cardoso, J.- F., "Blind signal separation: statistical principles", *Proc. of the IEEE 9 (10)*, 1998, pp. 2009 - 2025.
10. Cichocki, A. and Unbehauen, R., "Robust neural networks with on-line learning for blind identification and blind separation of sources", *IEEE Trans. on Circuits and Systems 43 (11)*, 1996, pp. 894 - 906.
11. Cottrell, G., Munro, P. and Zipser, D., "Learning internal representations from gray-scale images: an example of extensional programming", *Proc. 9th Ann. Conf. of the Cognitive Science Society*, 1987, pp. 462 - 473.
12. Der, R. and Herrmann, M., "Second-order learning in Self-Organizing Maps", in Oja, E. and Kaski, S. (Eds.),*Kohonen Maps.* Amsterdam: Elsevier, 1999, pp. 293 - 302.
13. Devijver, P. and Kittler, J., *Pattern Recognition - a Statistical Approach.* Englewood Cliffs, NJ: Prentice-Hall, 1982.
14. Diamantaras, K. and Kung, S.: *Principal Component Neural Networks: Theory and Applications.* New York: J. Wiley & Sons, 1996.
15. The FastICA package. Available from `www.cis.hut.fi/projects/ica/`
16. Foldiak, P., "Adaptive network for optimal linear feature extraction", *Proc. Int. J. Conf. on Neural Networks*, Washington, DC, 1989, pp. 401 - 406.
17. Funahashi, K., "On the approximate realization of continuous mappings by neural networks", *Neural Networks 2*, 1989, pp. 183-192.
18. Harman, H.H., *Modern Factor Analysis.* Univ. of Chicago Press, 1967.
19. Hastie, T. and Stuetzle, W., "Principal curves", *J. Am. Statist. Assoc. 84*, 1989, pp. 502 - 516.
20. Haykin, S., *Neural Networks - a Comprehensive Foundation.* New York: MacMillan College Publ. Co., 1998.

21. Heskes, T., "Energy functions for Self-Organizing Maps", in Oja, E. and Kaski, S. (Eds.),*Kohonen Maps.* Amsterdam: Elsevier, 1999, pp. 303 - 316.
22. Hinton, G., Revow, M. and Dayan, P., "Recognizing handwritten digits using mixtures of linear models", in G. Tesauro, D. Touretzky, and T. Leen (Eds.), *Advances in Neural Information Processing Systems 6*, San Mateo: Kauffman, 1995, pp. 1015 - 1022.
23. Hinton, G. and Sejnowski, T.J., *Unsupervised Learning - Foundations of Neural Computation.* Cambridge, MA: MIT Press, 1999.
24. Hornik, M., Stinchcombe, M., and White, H., "Multilayer Feedforward Networks are Universal Approximators", *Neural Networks 2*, 1989, pp. 359-368.
25. Hyvärinen, A. and Oja, E., "A fast fixed-point algorithm for Independent Component Analysis", *Neural Computation 9 (7)*, 1997, pp. 1483 - 1492.
26. Hyvärinen, A., "Fast and robust fixed-point algorithms for Independent Component Analysis", *IEEE Trans. Neural Networks 10 (3)*, 1999, pp. 626 - 634.
27. Hyvärinen, A., Karhunen, J. and Oja, E., *Independent Component Analysis.* New York, Wiley, 2001.
28. Hyvärinen, A. and Pajunen, P., "Nonlinear independent component analysis: existence and uniqueness results", *Neural Networks 12*, 1999, pp. 429 - 439.
29. Ilin, A., Valpola, H. and Oja, E., "Nonlinear dynamical factor analysis for state change detection", *IEEE Trans. Neural Networks 15*, No. 3, May 2004.
30. Jain, A. K., Duin, P. W., and Mao, J., "Statistical pattern recognition: a review", *IEEE Trans. Pattern Analysis and Machine Intelligence 22*, 2000, pp. 4 - 37.
31. Japkowitz, N, Hanson, S and Gluck, A., "Nonlinear autoassociation is not equivalent to PCA", *Neural Computation 12 (3)*, 2000, pp. 531 - 545.
32. Jutten, C. and Herault, J., "Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture", *Signal Processing 24*, 1991, pp. 1 - 10.
33. Jutten, C. and Karhunen, J., "Advances in nonlinear blind source separation", *Proc. 4th Int. Symp. on ICA and BSS*, Nara, Japan, April 1-4, 2003, pp. 245 - 256.
34. Karhunen, J., Oja, E., Wang, L., Vigario, R., and Joutsensalo, J., "A class of neural networks for independent component analysis", *IEEE Trans. on Neural Networks 8 (3)*, 1997, pp. 486 - 504.
35. Kendall, M and Stuart, A., *The Advanced Theory of Statistics, Vols. 1 - 3.* MacMillan, 1976 - 1979.
36. Kohonen, T., *Self-Organizing Maps.* Berlin: Springer, 1995. 3rd Edition, 2001.
37. Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Paatero, V. and saarela, A., "Self organization of massive document collection", *IEEE Trans. Neural Networks 11 (3)*, 2000, pp. 574 - 585.
38. von der Malsburg, C., "Self-organization of orientation sensitive cells in the striate cortex", *Kybernetik 14*, 1973, pp. 85 - 100.
39. Oja, E. "A Simplified Neuron Model as a Principal Components Analyzer", *J. Math. Biol. 15*, 1982, pp. 267-273.
40. Oja, E., *Subspace Methods of Pattern Recognition.* Letchworth: RSP and J. Wiley, 1983.
41. Oja, E., "Data Compression, Feature Extraction, and Autoassociation in Feedforward Neural Networks", *Proc. ICANN-91*, Espoo, Finland, June 24 - 28, 1991, pp. 737 - 745.
42. Oja, E., "Principal Components, Minor Components, and Linear Neural Networks", *Neural Networks 5*, 1992, pp. 927 - 935.
43. Oja, E., "The nonlinear PCA learning rule in independent component analysis", *Neurocomputing 17 (1)*, 1997, 25 - 46.

44. Oja, E. and Kaski, S. (Eds.),*Kohonen Maps.* Amsterdam: Elsevier, 1999.
45. Oja, E. and Wang, L., "Neural fitting: robustness by anti-Hebbian learning", *Neurocomputing 12*, 1976, pp. 155 - 170.
46. Pham, D. and Cardoso, J-F., "Blind separation of instantaneous mixtures of non-stationary sources", *IEEE Trans. Signal Proc. 49*, 2001, pp. 1837 - 1848.
47. Ritter, H., Martinetz, T., and Schulten, K., *Neural Computation and Self-Organizing Maps: an Introduction.* Reading: Addison-Wesley, 1992.
48. Roweis, S. and Ghahramani, Z., "A unifying review of linear gaussian models", *Neural Computation 11 (2)*, 1999, pp. 305 - 346.
49. Rubner, J. and Tavan, P., "A self-organizing network for Principal Component Analysis", *Europhysics Letters 10 (7)*, 1989, pp. 693 - 698.
50. Sanger, T., "Optimal unsupervised learning in a single-layered linear feedforward network", *Neural Networks 2*, 1989, pp. 459 - 473.
51. Schalkoff, R., *Pattern Recognition - Statistical, Structural, and Neural Approaches.* J. Wiley, 1992.
52. Schölkopf, B., smola, A. J. and Müller, K.-R., "Nonlinear component analysis as a kernel eigenvalue problem", *Neural Comput. 10*, 1998, 1299 - 1319.
53. Bibliography of SOM papers: a reference list of over 4000 studies on the Self-Organizing Map. Available at `www.cis.hut.fi/research/som-bibl/`.
54. The SOM Toolbox for Matlab. Available at `www.cis.hut.fi/projects/somtoolbox/`.
55. Tipping, M. E. and Bishop, C. M., "Mixtures of probabilistic principal component analyzers", *Neural Computation 11 (2)*, 1999, pp. 443 - 482.
56. Valpola, H., "Bayesian ensemble learning for nonlinear factor analysis", *Acta Polyt. Scand. Ma 108*, Espoo, 2000. D.Sc. Thesis, Helsinki University of Technology.
57. Valpola, H., Oja, E., Ilin, A., Honkela, A. and KArhunen, J., "Nonlinear blind source separation by variational Bayesian learning", *IEICE Trans. E86-A*, 2003, pp. 532 - 541.
58. VanHulle, M., *Faithful Representations and Topographic Maps.* New York: J. Wiley & Sons, 2000.
59. Villmann, T., "Topology preservation in Self-Organizing Maps". In Oja, E. and Kaski, S. (Eds.),*Kohonen Maps.* Amsterdam: Elsevier, 1999, pp. 267 - 292.
60. Wang, L. and Karhunen, J., "A unified neural bigradient algorithm for robust PCA and MCA", *Int. J. of Neural Systems 7 (1)*, 1996, pp. 53 - 67.
61. Webb, A., *Statistical Pattern Recognition.* Arnold, 1999.
62. Xu, L., "Temporal BYY learning for state space approach, hidden Markov model, and blind source separation", *IEEE Trans. Signal Proc. 48*, 2000, pp. 2132 - 2144.

Kernel Methods for Exploratory Pattern Analysis: A Demonstration on Text Data

Tijl De Bie[1] and Nello Cristianini[2]

[1] K.U.Leuven, ESAT-SCD
Kasteelpark Arenberg 10, 3001 Leuven, Belgium
tijl.debie@esat.kuleuven.ac.be
www.esat.kuleuven.ac.be/~tdebie
[2] U.C.Davis, Statistics Dept.
360 Kerr Hall, One Shields Ave., Davis, CA 95616, USA
nello@support-vector.net
www.kernel-methods.net

Abstract. Kernel Methods are a class of algorithms for pattern analysis with a number of convenient features. They can deal in a uniform way with a multitude of data types and can be used to detect many types of relations in data. Importantly for applications, they have a modular structure, in that any kernel function can be used with any kernel-based algorithm. This means that customized solutions can be easily developed from a standard library of kernels and algorithms. This paper demonstrates a case study in which many algorithms and kernels are mixed and matched, for a cross-language text analysis task. All the software is available online.

1 Introduction

Kernel Methods (KMs) offer a very general framework for performing pattern analysis on many types of data. In this paper we focus on text data, where text is chosen as an example of non-numeric data, and we demonstrate the versatility of this approach by performing cluster analysis, classification, correlation analysis and visualization on this data. What is more important, we do this by using different representations of our data defined by different kernel functions, as will be explained below. Overall, the purpose of this work is to make clear how different components can be combined together, to easily produce a wide variety of data analysis algorithms.

The main idea of kernel methods is to embed the dataset $S \subseteq X$ into a (possibly high dimensional) vector space $\Re^N$, and then to use linear pattern analysis algorithms to detect relations in the embedded data. Linear algorithms are extremely efficient and well-understood, both from a statistical and computational perspective. The embedding map is denoted here by $\phi : X \rightarrow \Re^N$, and it is understood that X can be any set.

An important point is that the embedding does not need to be performed explicitly: we do not actually need the coordinates of all the image vectors of the data in the embedding space $\Re^N$, we can perform a number of algorithms just knowing their relative positions in it. To be more accurate, if we know all the pairwise inner products $\langle \phi(x), \phi(z) \rangle$ between image vectors for all pairs of datapoints $x, z \in X$, we can perform most linear pattern discovery methods known from multivariate statistics and machine learning without ever needing the coordinates of such data points.

This point is important because it turns out that it is often easy to compute the inner product in the embedding space, even when the dimensionality N is high and so the coordinate vectors would be very large. It is often possible to find a (cheaply computable) function that returns the inner product between the images of any two data points in the feature space, and we call it a kernel. Formally, if we have data $x, z \in S \subseteq X$ and a map $\phi : X \rightarrow \Re^N$, we call kernel a function such that

$$K(x, z) = \langle \phi(x), \phi(z) \rangle$$

for every $x, z \in \Re^N$. As mentioned above, x and z can be elements of any set, and in this case study they will be text documents. On the other hand, their image $\phi(x)$ is a vector in $\Re^N$. The matrix $K_{ij} = K(x_i, x_j)$ is called the kernel matrix. Armed with this tool, we can look for linear relations in very high dimensional spaces at a very low computational cost. If the map ϕ is non-linear, then this will provide us with an efficient way to discover non-linear relations in the data, by using well understood linear algorithms in a different space. What is even more powerful, is that if X is not a vector space itself, the use of kernels enables us to operate on generic entities with essentially algebraic tools.

The kernel matrix contains sufficient information to run many classic and new linear algorithms in the embedding space, including Support Vector Machines (SVM), Fisher's Linear Discriminant (FDA), Partial Least Squares (PLS), Ridge Regression (RR), Principal Components Analysis (PCA), K-means and Spectral Clustering (SC), Canonical Correlation Analysis (CCA), Novelty Detection (ND) and many others. We refer the reader to [11, 9, 13, 3, 10, 1, 12] for more information on these methods, to [2] for a tutorial on kernel methods based on eigenvalue problems (PCA, CCA, PLS, FDA and SC), and to [16, 15] for two nice examples of the use of kernel methods in real life problems. Owing to the level maturity already achieved in these algorithmic domains, recently the focus of kernel methods research is shifting towards the design of kernels defined on general data types (such as strings, text, nodes of a graph, trees, graphs,...). Major issues in kernel design are its expressive power and its efficiency of evaluation [5, 7, 14, 8, 6].

Since by now a wide variety of kernel functions has been developed, each equivalent to a specific embedding function, the set of kernel methods has culminated into a complete toolbox to deal with real life machine learning and exploratory data analysis problems. Here we demonstrate this idea by using a variety of algorithms in combination with different text and string kernels on the articles of the Swiss constitution, which is available in 4 languages: English, French, German and Italian. What is interesting for this demonstration: the constitution is divided into several groups of articles, each group under a

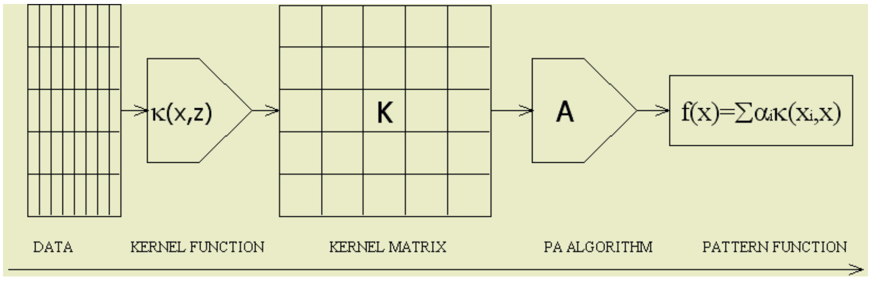


Fig. 1. A sketch of the modularity inherent in kernel-based algorithms: the data is transformed into a kernel matrix, by using a kernel function; then the pattern analysis algorithm uses this information to find the suitable relations, which are all written in the form of a linear combination of kernel functions.

different so-called 'Title' (in the English translation). All data can be found online at `www.admin.ch/ch/e/rs/c101.html`. A few articles were omitted in this case study (some because they do not have an exact equivalent in the different languages, 2 others because they are considerably different in length than the bulk of the articles), leaving a total 195 articles per language. The texts are processed by removing punctuation and stop words followed by stemming (where stop word removal and stemming are performed in a language specific way).

Ultimately, the aim of this simple case study is to exhibit how the inherent modularity of kernel methods makes them perfectly suited for fast and efficient deployment in a wide variety of tasks.

The entire matlab demo and the data used in this case-study, including scripts to remove punctuation and stop words and a stemming tool, are freely available online at `www.kernel-methods.net`, together with more free software. The pseudo-code and the detailed description of each algorithm and kernel used in this demo are described in the new book [12].

2 Pattern Algorithms

We briefly list here the algorithms that we will demonstrate. Given their large number and the space constraints of this article, it is impossible to even outline the theory behind them, so we refer the interested readers to the book [12]. All of these algorithms can work in any kernel-induced feature space, and all are amenable to statistical analysis based on Statistical Learning Theory. What we want to emphasize here is how all can be used as modules of a system, where any algorithm can be combined with any kernel, enabling practitioners to rapidly develop and test a large quantity of general purpose algorithms, and customize them by selecting the appropriate algorithms and kernels. Importantly, all algorithms (with the exception of k-means clustering) reduce to optimizing a convex function or to solving an eigenvalue problem.

Classification. One of the most classic tasks in pattern recognition is that of classification (or discrimination in the statistics literature, and categorization in text analysis). The goal is to find a function of the data that can be used to correctly assign a data item (e.g. a document) to one of a finite set of categories. A classic statistical method is Fisher's Linear Discriminant Analysis (FDA), and a classic method from machine learning is the Support Vector Machine algorithm (SVM) [3]. Both algorithms aim at finding a separating hyperplane in the embedding space, and differ in the properties of such hyperplane. In the first case (FDA) the hyperplane is chosen to maximize the proportion of the between class variance over the within class variance orthogonal to this hyperplane; in the second case (SVM) the hyperplane is chosen to maximize the margin.

Clustering. A second classic application in pattern recognition is the task of partitioning the samples in coherent groups. A common method for clustering vectorial data is K-means clustering. However K-means can be applied in a kernel induced feature space as well, making it applicable to virtually any kind of data using the kernel trick. As an alternative to K-means, we will demonstrate a more recently developed clustering technique known as spectral clustering (SC). This method is based on a cheap processing of the kernel matrix, followed by a simple eigenvalue problem.

Factor Analysis. When data is high dimensional (such as e.g. in text and bioinformatics applications), often the interesting information contained by the data can be explained by a number of underlying *factors* much smaller than this dimensionality. Depending on what is assumed to be interesting in a particular problem, different linear methods have been developed in multivariate statistics to extract these factors. The best known of these is principal component analysis (PCA), that finds a low dimensional projection of the data capturing as much of its variance as possible. Another method called canonical correlation analysis (CCA) can be used when we have two or more instantiations of the data that are all assumed to contain the relevant factors. CCA then proceeds by identifying those directions along which the data shows a large correlation between the different spaces. Both PCA and CCA can naturally be combined with kernels making it possible to identify hidden non-linear factors as well, or even factors explaining non-vectorial data such as text, trees, graphs,... For a survey on these methods based on eigenvalue problems, see [2].

3 Kernel Functions

All algorithms listed in the previous section are originally developed to be applied to vectorial data. However, for many other types of data it is possible to explicitly or implicitly construct a feature space capturing relevant information from this data. Unfortunately even when it can be expressed explicitly, often this feature space is so high dimensional that the algorithms can not be used in their original form for computational reasons. However, as pointed out above, many of these algorithms can be reformulated into a kernel version. These kernel versions directly operate on the kernel matrix instead of on the feature vectors. For many data types, methods have been devised to efficiently evaluate these kernels, avoiding the explicit construction of the feature vectors. In this way, the introduction of kernels defined for a much wider variety of data structures significantly extended the application domain of these algorithms.

In this section we briefly discuss the various kernels we will demonstrate in this case study. All kernels used here are text kernels, and we always normalized them. For a detailed description we refer the reader to [12].

Bag of Words Kernel. A text document can be represented by the words occurring in it, without considering the order in which the words appear. Of course this is a less complete representation than the texts themselves, but for many practical problems this is sufficient. Consider the complete dictionary of words occurring in all texts. Then each text document x could be represented by a *bag of words* feature vector $\phi(x)$. The entries in this vector are indexed by the words in the vocabulary, and equal to the number of times the corresponding word occurs in the given text. Then, the bag of words kernel between two texts is defined as the inner product of their bag of words vectors: $K(x, z) = \langle \phi(x), \phi(z) \rangle$. Of course the feature vectors are usually sparse (since texts are usually much smaller than the dictionary size), and some care has to be taken to efficiently implement the bag of words kernel.

Figure (2) contains an image of the bag of words kernel on all articles (of all languages)[1]. One can distinguish a block structure, corresponding to the 4 languages. In these blocks, one can see some substructure in the articles, roughly corresponding to the Titles, Chapters, Sections... the articles are arranged in. This substructure reappears in all languages to some extent.

K-mer Kernel. Another – more generally applicable – class of kernels is the class of k-mer kernels [8]. For each document a feature vector is constructed indexed by all possible length-k strings (k-mer) of the given alphabet; the value of these entries is equal to the number of times this substring occurs in the given text. The kernel between two texts is then computed in the usual way, as the inner product of their corresponding feature vectors. Note that this kernel is therefore applicable to string data, also where no words can be distinguished, such as in DNA sequences. On the other hand, its power is generally less than a bag of words kernel wherever this can be used, such as on natural language. K-mer kernels capture the order $k-1$ Markov properties of the texts, which are specific to natural languages. Therefore, even for small k they are quite powerful already in distinguishing different languages.

[1] To avoid a completely black picture except for a bright diagonal, before visualizing the diagonal is subtracted from the kernel. This is necessary because text kernels generally have a very heavy diagonal.

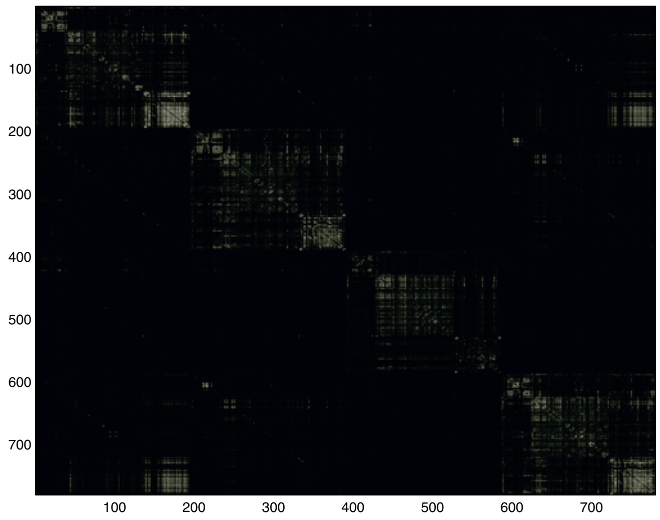


Fig. 2. A visualization of the full bag of words kernel matrix after normalization.Note that it is obvious from the figure that we have 4 distinct groups of texts, corresponding to the 4 different languages.

Note that the length of the feature vector is exponential in k, therefore a naive implementation would be prohibitively expensive for larger k. However efficient algorithms have been devised allowing the computation of this kernel for large scale problems [8].

Figures (3,4) contain the full 2-mer and 4-mer kernels. Figure (3) contains the part of the 4-mer kernel that corresponds to the English and French texts. Figure (5) above left, shows the same but now on the same articles artificially made noisy. Clearly the structure fades away.

Restricted Gappy K-mer Kernel. For noisy data, the k-mer kernel may be a bit too conservative in the sense that, even though two documents may be similar, still they don't share many k-mers. In that case, one may consider using a restricted gappy k-mer kernel. Consider feature vectors with entries corresponding to all possible k-mers again. Now, every entry is made equal to the number of k-mers up to (k+g)-mers in the text, that contain a (not necessarily contiguous) subsequence of length k equal to the k-mer of this specific entry. Here g is a parameter indicating the maximum number of gaps allowed. For details we refer the reader to [8], where an efficient way to evaluate such kernels is described. Figure (5) left below contains the restricted gappy 4-mer kernel on the same noisy texts.

Wildcard K-mer Kernels. This kernel adopts a different approach to deal with noisy texts. Now we use a feature space where each dimension corresponds

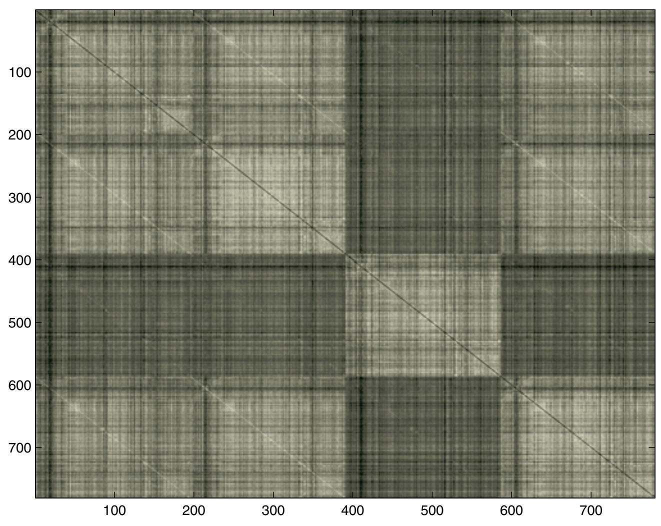


Fig. 3. The 2-mer kernel matrix after normalization. Again the cluster structure can be seen, however it is less clear than from the bag of words kernel. This is not surprising: a 2-mer kernel only takes into account 1st order Markov properties in the texts, making them probably less suitable for natural language applications. Note that the third group of texts –corresponding to the German language–, sticks out however, indicating that the 1st order Markov properties of German are significantly different from those of the other languages considered.

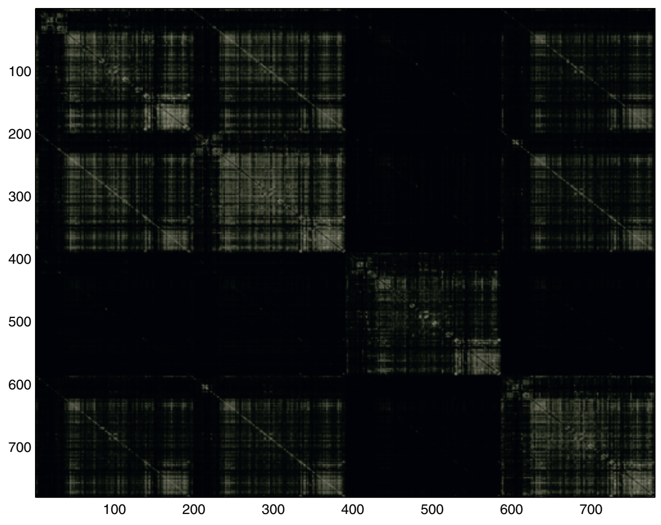


Fig. 4. The 4-mer kernel matrix after normalization. One can see that the distinction between the different languages is more clear now than for the 2-mer kernel.

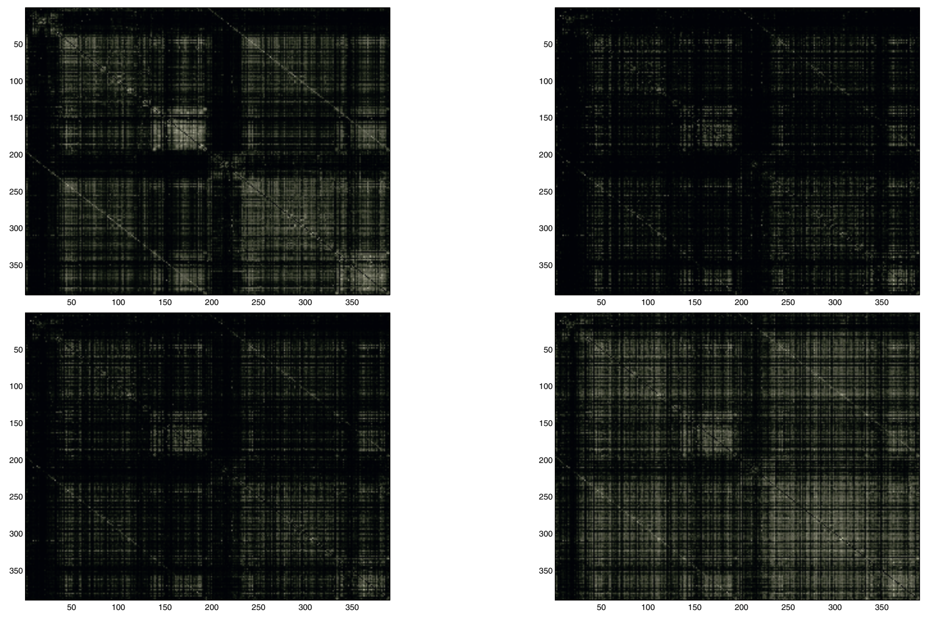


Fig. 5. Above left, the part of the normalized 4-mer kernel matrix corresponding to the English and French texts only is shown. Above right, the normalized 4-mer kernel matrix on the noisy English and French texts is depicted. One can see that the pattern has faded away a bit. The normalized restricted gappy 4-mer kernel matrix on the noisy English and French texts is shown below on the left. This kernel explicitly tries to take the noise influence into account. It is not immediately obvious from the figure, however experiments will show an improvement in performance of algorithms using this kernel over using the simple 4-mer kernel. The normalized wildcard 4-mer kernel matrix on the noisy English and French texts is shown below on the right. This kernel provides an alternative way to deal with noisy data. Also with this kernel algorithms will be shown to perform better than with the simple 4-mer kernel on the noisy data.

to a k-mer of the alphabet augmented with a wildcard. The number of wildcards in these k-mers is restricted by a parameter m. Then every feature is equal to the number of matches to this k-mer found in the text. Again, [8] describes an efficient way to evaluate such kernels. Figure (5) right below contains the wildcard 4-mer kernel on the same noisy texts.

4 A Case Study: Swiss Constitution Corpus

Thus far we have given an exposition of a wide variety of linear algorithms in machine learning that can be kernelized, and of different kernels applicable to text data. In what follows, we will show how each of these kernels can be used in the different algorithms. This inherent modularity in kernel methods is of major importance. Since for most types of data relevant kernels that can be evalu-

Table 1. Classification error rates on the noise free data, averaged over 100 randomizations with balanced 80/20 splits. The 2-mer kernel is used.

	English vs French	English vs German	English vs Italian
SVM	0.82 ± 0.05	0.03 ± 0.03	0.43 ± 0.04
FDA	5.0 ± 0.2	0 ± 0	1.2 ± 0.1

ated efficiently have been proposed in literature, the application domain is vast. Furthermore, this modularity has obvious advantages in software engineering.

4.1 Classification

Technical Notes. For FDA, we always took 1 for the regularization parameter. For the SVM we used the new-SVM formulation, and the regularization parameter ν was always chosen equal to 0.1.

Note that the ambition here is not to optimally tune the parameters: the main goal is to show how the modularity of kernel methods allows to apply a large library of algorithms to non-vectorial data; not to benchmark these algorithms.

Classifying Articles in Their Respective Language Classes

Noise Free. The first task we consider is the classification of texts into their respective language classes. The kernel we use here is the 2-mer kernel. We considered 3 binary classification problems, discriminating English texts from the texts in other languages (averaged over 100 random balanced splits in training (80%) and test sets (20%)). Error rates are in table 1.

English and French are hardest to distinguish based on the 2-mer kernel, which is probably due to many loan words present in English, recently adopted from French. Also, English and Italian are not perfectly distinguished (probably due to the same fact, and due to the fact that many English words have a Roman origin). German sticks out most clearly, which is to be expected. SVM's seem to have a better performance on this dataset.

Noisy, English versus French. Now let us consider the classification problem 'English vs French' in some greater detail. What happens if we add noise to the texts? We study this by artificially modifying the text by randomly deleting or altering 1/4th of the letters. Table 2 contains the average classification error rates for different kernels, along with the standard deviation on the estimated average, over 100 randomizations.

Note that the 4-mer kernel performs better that the 2-mer kernel. We can further improve the performance by using the restricted gappy and the wildcard 4-mer kernels.

Somewhat surprisingly FDA on the noisy data performed significantly worse with the restricted gappy as compared to the standard 4-mer kernel. However, clearly the method of choice here is SVM, which improves when using the restricted gappy or wildcard kernels.

Table 2. Classification error rates for different kernels on the noisy data, averaged over 100 randomizations with balanced 80/20 splits.

	2-mer	4-mer	gappy 4-mer	wildcard 4-mer
SVM, No noise	0.82 ± 0.05	0.42 ± 0.03	-	-
SVM With noise	2.89 ± 0.09	1.53 ± 0.07	1.28 ± 0.06	1.29 ± 0.06
FDA, No noise	5.0 ± 0.2	1.4 ± 0.1	-	-
FDA With noise	18.0 ± 0.5	8.4 ± 0.3	8.7 ± 0.3	10.9 ± 0.3

Table 3. Adjusted Rand index performances for spectral clustering and K-means clustering of the documents. The ideal clustering is clustering per language.

	bow	2-mer	4-mer
Spectral clustering	0.966 ± 0	0.437 ± 0	0.337 ± 0
K-means	0.38 ± 0.04	0.17 ± 0.03	0.26 ± 0.04

4.2 Clustering

Having shown that kernel methods allow to do classification in various ways, we will now show also clustering can be performed on data such as text in this case study. We will consider two methods: K-means clustering and spectral clustering.

Clustering the Articles in Their Language Clusters

Spectral Clustering. We cluster the articles of all languages, and check how well they are clustered into their respective language clusters. To assess the performance we use the adjusted Rand index [4], which is 1 for perfect clustering and has an expected value of 0 for random clustering. The final step consists of K-means on the eigenvectors, the clustering corresponding to the minimal K-means cost is taken over 10 starting values, chosen as described in [10].

K-means. Similarly, we perform kernel K-means on the documents. After 100 random initializations of K-means, the one with the best K-means cost is taken, and its adjusted Rand index is computed.

The results are summarized in table 3. The numbers in the table are averages over 10 runs along with the standard deviations on these averages. Note that spectral clustering (virtually) always returns the same optimal value (very small standard deviation), i.e. it is quite independent of the starting values in the K-means iterations, whereas K-means does not.

Somewhat surprisingly the 2-mer kernel performs better than the 4-mer kernel with the spectral clustering. As expected the best performance is achieved with the bow-kernel. The spectral method outperforms K-means in all cases.

Clustering the Articles into Coherent Groups

The articles in the constitution are organized into groups, called ‘Titles’. Can we use clustering to automatically categorize the articles into their Titles?

Table 4. Adjusted Rand indices for spectral clustering of the English articles into the chapters they appear in.

		bow	2-mer	4-mer
English	Spectral clustering	0.326 ± 0	0.231 ± 0	0.328 ± 0.001
	K-means	0.24 ± 0.02	0.24 ± 0.02	0.27 ± 0.02
French	Spectral clustering	0.372 ± 0	0.206 ± 0	0.340 ± 0
	K-means	0.23 ± 0.03	0.17 ± 0.01	0.30 ± 0.02
German	Spectral clustering	0.559 ± 0	0.136 ± 0	0.241 ± 0
	K-means	0.13 ± 0.02	0.12 ± 0.01	0.19 ± 0.02
Italian	Spectral clustering	0.508 ± 0.001	0.214 ± 0	0.308 ± 0
	K-means	0.26 ± 0.02	$0.0.19 \pm 0.01$	0.31 ± 0.03

Spectral Clustering and K-means. See table 4 for the adjusted Rand scores achieved on this clustering problem for the different languages, kernels and methods. The performances are much less than for clustering articles into their language classes. This is of course to be expected: now the number of samples is smaller, and the distinction between languages is an objective criterion, while the distinction between Titles in the constitution is man-made and thus subjective in nature. Still, the performance is well above what a random clustering would do.

4.3 Factor Analysis

As a last type of applications discussed in this paper, we consider two methods for doing factor analysis: principal component analysis and canonical correlation analysis. Again, even though these techniques are originally developed to analyze vectorial data, the kernel trick allows us to apply them in a kernel induced feature space on a wide variety of data types. We demonstrate the methods here on the text data of our case study.

PCA. PCA is an algorithm to project the data in a lower dimensional space such that as much of the variance as possible is captured. The first two principal components are shown in figure 6. It can be seen that in this case indeed the directions of large variance seem to visualize some interesting cluster structure in the data.

CCA. With CCA one is able to capture information that is in common between several information sources. In this case, we have the same information in different languages. Since we are in fact interested in the semantic meaning of the articles, and not in the particularities of the languages, using CCA can be a good idea. Indeed, the division of the constitution articles into groups (the 'Titles' as they are called in the constitution) has something to do with their semantic context, and not with their particularities due to the language in which they are written.

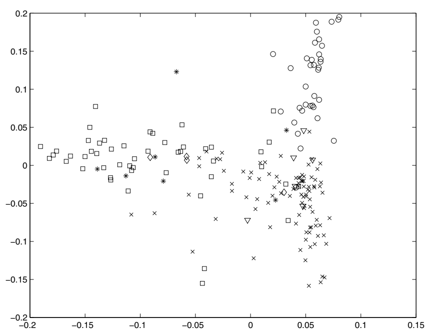


Fig. 6. First two principal components of the bag of words kernels for the English articles, as obtained by doing PCA. Articles from different chapters are represented by a different symbol.

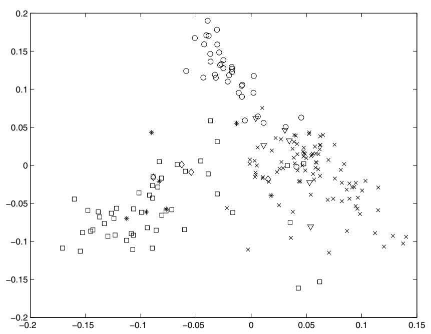


Fig. 7. First two canonical components of the bag of words kernels for the English articles, as obtained by doing CCA with the other languages. Articles from different chapters are represented by a different symbol.

We can use it here as a visualization tool: find two semantically interesting directions in the high dimensional feature space of the articles, and plot the components of the articles along these directions in the 2D plane[2]. The result can be seen in figure (7).

Apart from dimensionality reduction, CCA can also be used for cross-language text retrieval. For more information we refer the reader to the relevant literature [16].

Comparison of PCA with CCA. If we compare figure (6), where only one language is used, with figure (7), where the other languages are used to *supervise* the dimensionality reduction to some extent, we can see that the cluster structure is slightly more apparent when using CCA. We can assess this by computing the between class variance divided by the total variance (BCV/TV) in the subspaces found by PCA and CCA respectively. The larger this number, the better the class separation. The results for subspaces from 1 dimension up to 10 dimensions are shown in figure 8. Clearly CCA performs better than PCA, indicating that the different languages effectively supervise each other when selecting relevant dimensions in CCA.

5 Conclusion

We have demonstrated with a case study some of the most appealing features of kernel methods for pattern analysis: their modular design, the possibility of

[2] Note that training the regularization parameter is an issue here, and done by permutation analysis (the difference between the sum of the maximal correlations between the actual problem and a permuted version is maximized).

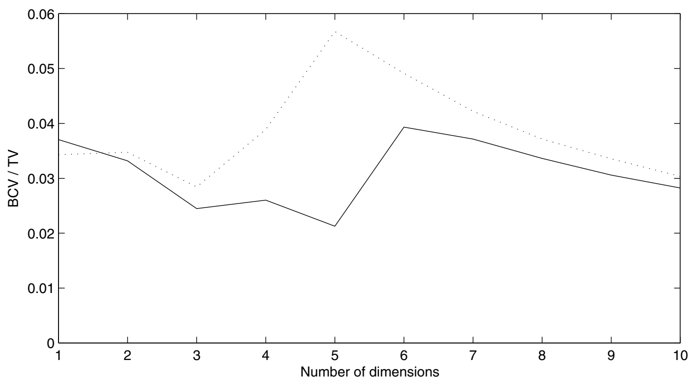


Fig. 8. Between class variance divided by total variance (BCV/TV) for PCA (full line) and CCA (dotted line) as a function of the dimension of the subspace (equivalently: the number of factors selected).

naturally using them for exploratory data analysis and rapid deployment, and their capability of operating seamlessly on non-numeric data. The theoretical details which are absent in this paper can be found in [12], and all the software and data are available at `www.kernel-methods.net`.

Acknowledgments

The authors thank John Shawe-Taylor for useful discussions and Manju Pai for contributing to part of the software development. TDB is a Research Assistant with the Fund for Scientific Research – Flanders (F.W.O.–Vlaanderen).

References

1. F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.
2. T. De Bie, N. Cristianini, and R. Rosipal. Eigenproblems in pattern recognition. In E. Bayro-Corrochano, editor, *Handbook of Computational Geometry for Pattern Recognition.* Springer-Verlag, 2004.
3. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines.* Cambridge University Press, Cambridge, U.K., 2000.
4. L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, page 193–218, 1985.
5. T. Jaakkola, M. Diekhans, and D. Haussler. Using the fisher kernel method to detect remote protein homologies. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, 1999.
6. H. Kashima, K. Tsuda, and A. Inokuchi. Kernel methods in computational biology. In B. Schoelkopf, K. Tsuda, and J.P. Vert, editors, *Handbook of Computational Geometry for Pattern Recognition.* Springer-Verlag, 2004.

7. R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the ICML*, 2002.
8. C. Leslie and R. Kuang. Fast kernels for inexact string matching. In *Conference on Learning Theory and Kernel Workshop (COLT 2003)*, 2003.
9. S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Neural Networks for Signal Processing IX*, pages 41–48. IEEE, 1999.
10. A. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.
11. B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
12. J. Shawe-Taylor and N. Cristianini. *Kernel methods for Pattern Analysis*. Cambridge University Press, Cambridge, U.K., 2004.
13. D.M.J. Tax and R.P.W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199, 1999.
14. K. Tsuda, M. Kawanabe, G. Rätsch, S. Sonnenburg, and K.-R. Müller. A new discriminative kernel from probabilistic models. *Neural Computation*, 14(10):2397–2414, 2002.
15. J.-P. Vert and M. Kanehisa. Graph-driven features extraction from microarray data using diffusion kernels and kernel cca, 2003.
16. A. Vinokourov, N. Cristianini, and J. Shawe-Taylor. Inferring a semantic representation of text via cross-language correlation analysis, 2002.

The Use of Graph Techniques for Identifying Objects and Scenes in Indoor Building Environments for Mobile Robots

Alberto Sanfeliu

Institut de Robòtica i Informàtica Industrial (UPC-CSIC)
Universitat Politècnica de Catalunya (UPC)
sanfeliu@iri.upc.es
http://www-iri.upc.es/groups/lrobots

Abstract. In this paper we present some of the common issues that appear when we try to recognize objects in indoor scenes of a building, and we describe some strategies for recognizing them by using graph techniques. These scene images are captured by the colour cameras of a mobile robot, which in the learning phase, learn the objects by taken a set of 2D images of the projective object views. Then afterwards, the robot must identify the objects once its moves through the area that has been used to learn the objects. We describe two strategies to use graph techniques for object and scene recognition, some algorithms and preliminary results.

1 Introduction

Computer vision in autonomous mobile robotics is a very well known topic that is being treated by many research groups [7]. However, the use of perception techniques to automatically learn and recognize the environment and the objects located on it is probably not so well known, although there are also a number of research work on the area of robot vision [2,5,13,14,15,19]. We will describe in this paper some of the research that we are doing in the area robot vision for mobile robots and more specifically, the one related to the graph techniques. One part of our research has been concentrated in the development of techniques to capture and process the information that surrounds a robot, taking into account that this information can be captured by diverse perception sensors (colour video cameras, stereo vision, laser telemeter, ultrasonic sensors, etc.) and the sensors related to robot movement (odometers).

We have focused our research in the development of "robust" techniques that must be as much as possible, "invariant" to illumination, colour, surface reflectance, sensor uncertainty, dead reckoning and dynamic environments. However, this wish is not always possible. We also orient our research to develop techniques to learn the perceptive world, in order to create a data base that can be used later on, by robots.

In this paper we present some results of the use of graph techniques [16] for the process of identification of objects and scenes in indoor environments for mobile robots.

In the first section we describe some of the common issues in images acquired by a robot in indoor building environments, in the second section we show two strategies used for identifying objects, in the third section we summarize two graph methods based on the segmentation-recognition strategy and in the fourth section we present several ideas to use detection-recognition methods based on graph techniques. Finally, we present some results.

2 Common Issues in Robotic Scene Images of Indoor Buildings and Their Implication in Object Recognition

In order to recognize objects in a scene, we first have to capture them and create a data base. If the objects are isolated and environment conditions do not change, there are not object appearance variations between the learning and the identification phase. However, when a robot moves around an environment, the appearance of the objects may change between the both phases due to diverse issues. Let us describe some common issues that there exist in indoor building scene images and their implication in the process of learning and recognition.

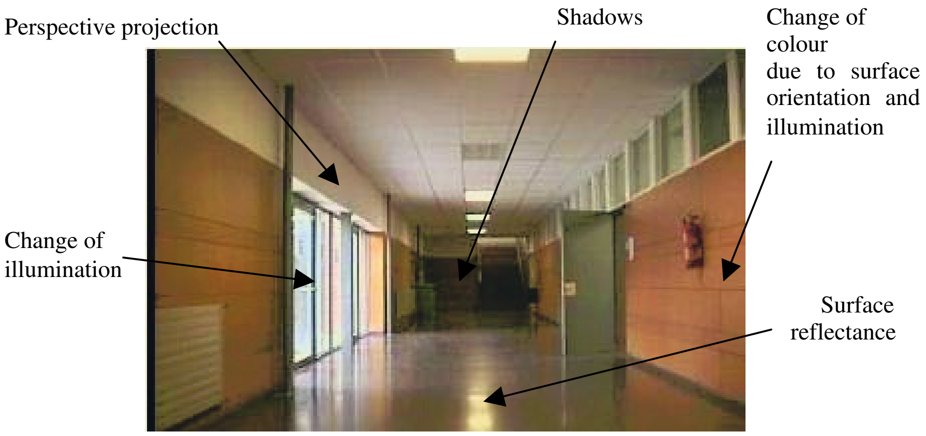


Fig. 1. Some issues on image scenes of indoor buildings

Some of the issues that produce discrepancies are the following ones (Fig. 1 and 2):

- Perspective projection due to the camera model.
- Partial occlusion due to camera point of view or due to the intersection of an obstacle between the object and the camera.
- Colour modification due to the surface orientation, surface reflectance, multiple illumination sources or sensibility of camera sensor.
- Surface reflectance
- Surface texture
- Shadows produced by other objects or by the own concavities of the object.
- Confusing background.

Some of these issues have a direct impact in the scene as far as the object recognition process is concerned. Specifically we can enumerate the following ones:

- *The separation of the object from the background*: If the features that differentiate the object from the background are sensible to the aforementioned problems, then the extraction of the object is not easy. For example, if a segmentation process is used, then the segmentation features must be invariant to colour, surface reflectance, etc. If there are shadows or the background of the object is confusing then the separation is even worse.
- *The detection of the object surface features*: The object surface colour, surface reflectance and surface texture are usually not invariant, although in some cases, these problems can be partially overcome.
- *The extraction of geometric object features*: Due that the scene is captured by means of a camera, then the perspective projection must be taken into account. This issue produce sensitive variations on the extraction of geometric features (area, centre of geometry of a surface, angles between contour lines, etc.).
- *The image view of a 2D projection of a 3D object*: A 3D object has usually multiple 2D views which depend on the orientation of the object with respect to the camera. The number of views usually depends on the number of potential object rotations and the number of concavities of the object.
- *The partial occlusion of an object*: This issue produce an important reduction of the visibility of the object which has a direct impact on the identification of the object.

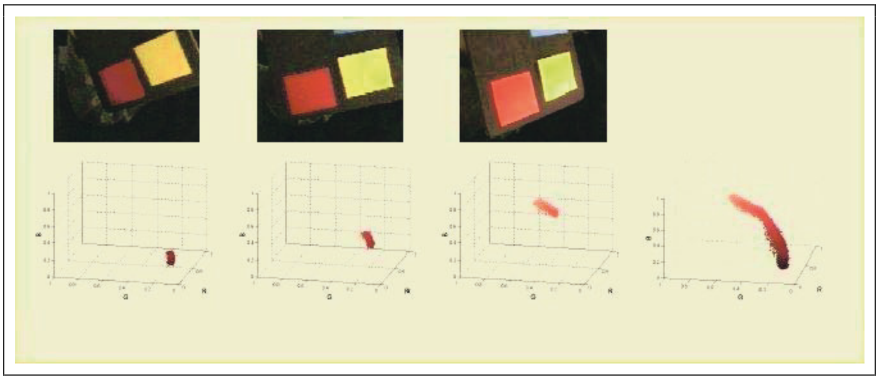


Fig. 2. Typical reflectance problems of a colour (red) planar surface: (a) a sequence of a red planar surface; (b) RGB map of the colour distribution of the sequence of the planar surface

In order to overcome some of these problems, for example, we can apply invariant techniques (colour constancy methods [9,24], projective invariants, etc.) or to fuse information of diverse sensors (colour-disparity for segmentation [1], colour-disparity-edges-motion-SSD for visual servoing [13], textons-contours-regions fors segmentation, etc.). However, these techniques are usually not enough for object recognition due to the stochastic variability of the perceptive features and to the last two commented issues: the orientation of the object and the partial occlusion. For these reasons, graph matching can be a good candidate for object learning and recognition. Moreover, if we consider attributed graphs, then the stochastic variability can be included in the nodes and arcs. If additional, it is used a distance measure to compare a graph view against object graph views of a data base, then we can also cope with the variability of the graph topology.

However, the use of graph techniques for object recognition has at least a big drawback than still have not been overcome: the time complexity of the matching process. This issue is under study.

3 Strategies for Object Recognition

There are two strategies for recognition of objects in scenes: segmentation-recognition and detection-recognition. The first one is a general approach, where it is not essential to have a priori knowledge, to extract the objects for the recognition process. It can be applied to any type of scene and the objects to identify can be partially occluded. This technique has some drawbacks in indoor images, for example, they are time consuming and very dependent on the segmentation process. The second one, detection-recognition, requires having a good knowledge of the objects to detect and moreover, the objects can not be partially occluded. The advantages are that the time complexity can be reduced and that the algorithms can be adjusted to diminish the feature extraction dependency. We will present both strategies from the point of view of the application of graph techniques.

- *Segmentation-Recognition*: Often called bottom-up strategy, it is a good approach for applying graph techniques for object recognition. In this case, the whole image can be seen as a graph and the goal is to find a sub-graph in the image graph that match one of the graphs of the object data base. We have been working in this area and we have developed several techniques. We will describe one technique based on random graphs for matching a 2D view of a scene object against a data base of 3D objects. We also will explain another one, which use oriented matroids to index 2D views of image objects.

- *Detection-Recognition*: This strategy is something similar to a top-down strategy but with some special features. In this case, the objective is to detect potential zones where there can be objects or zones of interest and then, apply a method to find the object in that zone. If the objects to identify have distinguish features then these features can be used to detect the object. This strategy has been applied successfully in object detection using non graph techniques, for example in human face detection [25]. We can also think about other techniques that are in between these two strategies, which do not require a pure segmentation process neither to have too much knowledge of the objects for recognition.

4 Segmentation-Recognition Graph Techniques for Object Recognition

We have previously described some common problems that we can find in a scene image of an indoor building, and the consequences that they produce to the objects that we have learned and we want to identify later on. Since these issues can produce big variations between the image captured in the learning phase and the image captured in the identification phase, then we need robust methods to cope with these variations.

The methodology is to segment an image, extract the graph features and then apply a graph method to identify an object against a data base of reference objects. The main issue of this methodology is to segment well the image, which is often not the case. Since there is not a good segmentation, the graph matching technique must overcome the potential variability of the extracted graph with respect to the "ideal" graph. Usually graph or sub-graph isomorphism techniques are not the most appropriate ways to identify an object due to aforementioned problems, besides a potential partial occlusion of the object to identify. It is usually required to apply distance measure methods which allow coping with the variability between the object graph and the reference one. There have been developed several well known graph techniques than can be applied to object recognition, for example [6,11,12,22,26,27, 28,29], although we only will summarize two techniques that our group have developed based on this strategy.

4.1 Matching Views of 2D Projections of 3D Objects by Using Oriented Matroids

The idea is to represent 2D views of a 3D object, by means of topological properties of the regions of the segmented image and then, to create a table with each one of the topological representations. Then the identification process is based on matching the input representation of one scene view, to the table of the topological representations of the 2D object views. In this case the graph representation of a segmented image is reduced to a list of ordered chains of symbols (denominated co-circuits), where each co-circuit is the spatial combination of regions based on two reference regions.

A topological representation is created by using the oriented matroid theory by means of encoding incidence relations and relative position of the elements of the segmented image, and by giving local and global topological information about their spatial distribution. The result is a set of co-circuits [3] of sign combinations that relates segmented regions with respect to the convex hull of two selected regions of the scene. The details of this process are explained in [22]. The set of co-circuits obtained is projective invariant, which is an important feature for the representation of the model objects. Fig. 3 shows the segmentation and process indexing of one object and Table 1 shows the resulting indexes of the object.

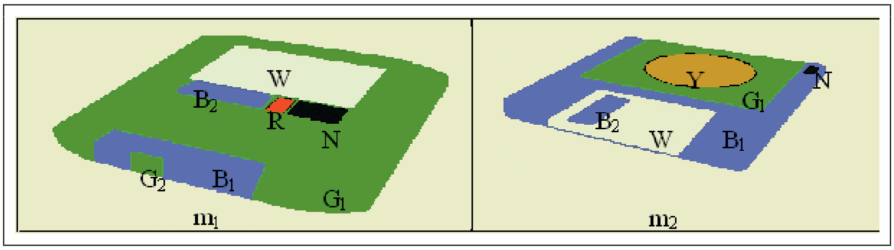


Fig. 3. Segmentation and process indexing of two objects

The result of the indexing process looks as follows:

Table 1. Index result of the process indexing of the images of Fig. 3. The first column is the baseline area from where the segmented regions are related. 0 means the region is inside the baseline area; - the region is one the left side; + the region is on the right side; and * means the region does not exist in the segmented image

	W	**R**	**Y**	**G_1**	**G_2**	**B_1**	**B_2**	**N**	**Object**
WR	0	0	*	0	0	0	-	+	m_1
WY	0	*	0	0	*	0	0	-	m_2
WG_1	0	*	*	0	*	*	*	*	m_1
WG_1	0	*	0	0	*	0	0	0	m_2
WG_2	0	0	*	0	0	+	0	0	m_1
WB_1	0	0	*	0	0	0	0	0	m_1
WB_1	0	0	*	+	+	+	0	+	m_2
WB_2	0	0	*	+	+	+	0	+	m_1
WN	0	0	*	-	-	-	-	0	m_1
WN	0	*	+	+	*	0	0	0	m_2
RG_1	*	0	*	0	*	*	*	*	m_1
…	…	…	…	…	…	…	…	…	
B_2N	+	0	*	-	-	-	0	0	m_1
B_2N	-	*	+	+	*	+	0	0	m_2

The matching process is done by comparing the set of co-circuits of the 2D projection view of the scene, to the set of co-circuits of the data base. The time complexity of the matching process is polynomial with respect to the number of segmented zones of the scene image. The reason of the reduction of the time complexity is due to two reasons: the elimination of labelling process; the comparison against a set of co-circuits which number is polynomial with respect to the number of segmented zones in the worst case.

4.2 Matching Views of 2D Projections of 3D Objects by Random Graphs

The idea is to represent 2D views of a 3D object by means of random graphs and then to obtain the model as the synthesis from the graphs that represent the 2D views of a 3D object. Once the model has been learned, the recognition process is based on applying a distance measure among the input graph (the graph that encodes the 2D view of a scene object) and the object models. The input graph is assigned to the model graph with the minimum distance measure value. Fig. 4 shows the process of learning (synthesis of the object graph views) and recognition.

Object views are often represented by graphs, and one robust representation is based on attributed graphs (AG). However, in order to synthesize AG we need a more general model representation, which is called Random Graph (RG). The generalization of these graphs is denominated General Random Graphs (GRG) which has theoretically, great representation power, but they need a lot of space to keep up with the associated data. We have defined several simplifications to the GRG to reduce the space and also to diminish the time matching complexity. Wong and You [27] proposed the First-Order Random Graphs (FORGS) with strong simplifications of the GRG, specifically they introduce three assumptions about the probabilistic independence between vertices and arcs which restrict too much the applicability of these graphs to object recognition. Later, our group introduced a new class of graphs called Function-Described Graphs (FDG) [20] to overcome some of the problems of the FORG. The FDG also considers some independence assumptions, but some useful 2° order functions are included to constrain the generalisation of the structure. Specifically a FDG includes the antagonism, occurrence and existence relations which apply to pairs of vertices and arcs. Finally, we have expanded this representation, [17,18] by means of Second-Order Random Graphs (SORG), which keep more structural and semantic information than FORGs and FDGs. These last types of representation have led to the development of synthesis techniques for model object generation (by means of 2D projections of a 3D object) and graph matching techniques for graph identification.

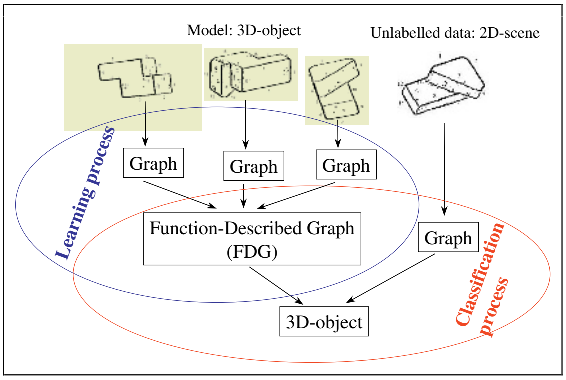


Fig. 4. Learning and classification processes in the classifiers that use only one structural representation per model

The time complexity of this method is in the worst case, exponential with respect to the number of nodes of the graph. This time complexity can be reduced pruning the number of combinations by using some ad-hoc information of the objects and the images to be applied.

5 Detection-Recognition Graph Techniques for Object Recognition

As we have commented, there are other ways to recognize objects in scenes, where graph techniques can be used. The strategy is to detect zones of potential objects and then apply classification techniques to recognize an object in that zone. The graph techniques can be used in the detection of the zones, in the classification process or at the same time, in the detection-classification. In this last case, the technique can be used, for example, as an indexing method. We will describe in this section only detection techniques, since once a zone has been detected, the methods described in the previous section can be used.

Three general detection approaches can be applied:

- *Global Search Detection*: The idea is to generate a global graph of the full image and then look for a specific sub-graph that has the potential to be a zone object. A typical technique to represent the complete image is the Voronoi diagram representation [10]. If not attributes are used, then we can apply general sub-graph matching techniques. When we have attributes, for example the colour or the area of the regions, then we can prune the potential matches using the node and arc attributes. More sophisticated techniques can also be applied, for example to grow zones using a potential field, where the function can be related to the fan in and fan out of the graph or the node and arc attributes.

- *Raster Search Detection*: The idea is to pass a window through the full image which has a basic graph structure and attributes of the zone to be located. When the match distance between the reference graph and the graph that are extract under the window limits, is higher than a threshold, then the zone is identified. One potential method to apply is a PCA approach for fast retrieval of structural patterns in attributed graphs [26]. In this case, in order to detect the nodes we can use a pre-segmentation process which outcome is a planar graph, for example using one step of the technique [25] or the technique [8] which nodes are spanning trees. Since we look for potential graph zones instead a full matching process, this algorithm can do the process in polynomial time.

- *Probabilistic Search Detection*: The idea is to probabilistically take some initial starting points where to grow a graph. This methodology has been applied successfully in diverse fields, for example in segmentation, path planning or salience detection. From the starting point of the image, we can grow the graph without restrictions, that is, looking for the neighbour nodes by using general rules, or to grow the graph imposing a graph reference model. In the last case the idea is similar to the raster search detection methodology, but in a probabilistic way.

6 Some Results

We show in this article two examples of identifying objects and images by means of graph techniques. The first one is applied to learn and recognize 3D objects by means of their 2D projection views and the second one, it is the learning and recognition of image scenes by means of oriented matroids. In the first example, the images come from a standard data base, and in the second, the images have been acquired by the colour camera of the robot. In both examples we used the segmentation-recognition strategy, where the segmentation was based on the colour of the image pixels using the method described in [8]. Moreover, in Fig. 8, we present the set of images that we are using and the segmentation results.

For the first example we used a set of objects extracted from the database COIL-100 from Columbia University. We did the study with 100 isolated objects, where each one is represented by 72 views (one view each 5 degrees). The test set was composed by 36 views per object (taken at the angles 0, 10, 20 and so on), whereas the reference set was composed by the 36 remaining views (taken at the angles 5, 15, 25 and so on).

The learning and recognition process was as follows: (1) perform colour segmentation of each individual object view image; (2) create an adjacency graph for each one of the segmented regions of each object view; and (3) transform the adjacency graph in an attributed graph (AG) using the hue feature as the attribute for each node graph. The learning process was based on 36 views of each object and for each object, we synthesise four random graphs, the first of one grouping the views from 0º to 90º, the second one grouping from 95º to 180º and so on. We used four different techniques for the representation of random graphs: AG (Attributed Graph), FORG (First Order Random Graph), FDG (Function Described Graph) and SORG (Second Order Random Graph). The learning techniques (synthesis of graphs) are described in [18]. For the recognition process we used the distance measures explained in [18,20].

Fig. 4 shows 20 objects at angle 100º and their segmented images with the adjacency graphs. FORGs, FDGs and SORGs were synthesised automatically using the AGs in the reference set that represent the same object. The method of incremental synthesis, in which the FDGs are updated while new AGs are sequentially presented, was applied. We made 6 different experiments in which the number of random graphs, FORGs, FDGs and SORGs, that represents each 3D-object varied. The best result appears when the SORG and FDG representations were used, although the best is the SORG representation. Fig. 5 shows the ratio of recognition success of the 100 objects using different object representation and distance measures. This figure also shows the result of describing individually each object view by means of an AG and then comparing each input AG against the rest of the prototype AG.

For the second example, we used two set of examples: (1) 10 different reference images of an indoor building, and from each one, three images were taking at different position and orientation by the colour camera of a mobile robot; and (2) a sequence of several hundred of images acquired by the robot. Figure 6.b shows an image taken from three different views, their segmented images and the learning process. Fig. 8 shows two images of the image sequence.

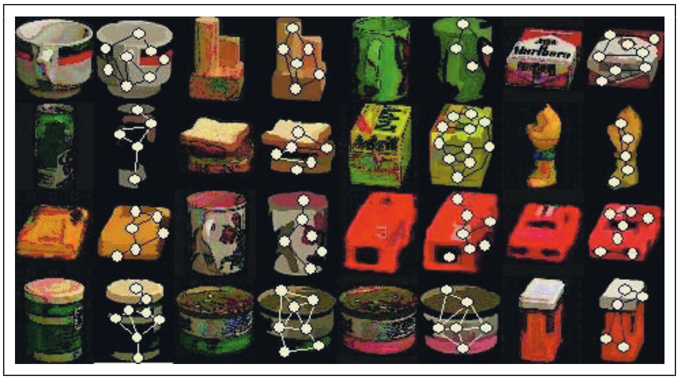


Fig. 5. Some objects at angle 100 and the segmented images with the AGs

The learning and recognition process was the following one: (1) perform colour segmentation of each image scene; (2) extract the co-circuits of each image; and (3) construct a data base joining the co-circuits. We applied a distance measure between co-circuits to identify the image. See [21] for details. For the images of Fig.7, 74% of the images where well recognized and for a sequence of images of Fig.8, 100% of the images were well classified.

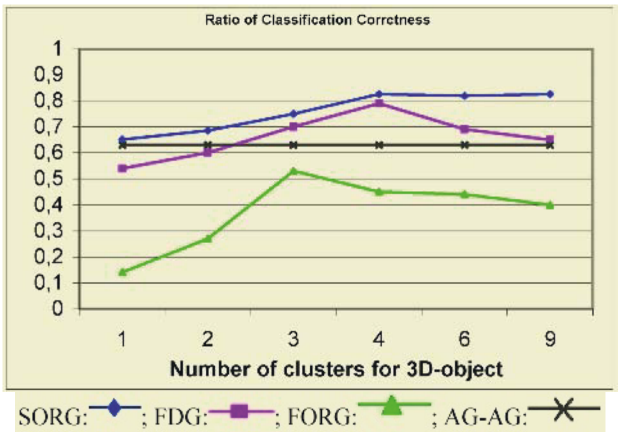


Fig. 6. Ratio of recognition correctness of the objects using SORG, FDG, FORG and AG-AG

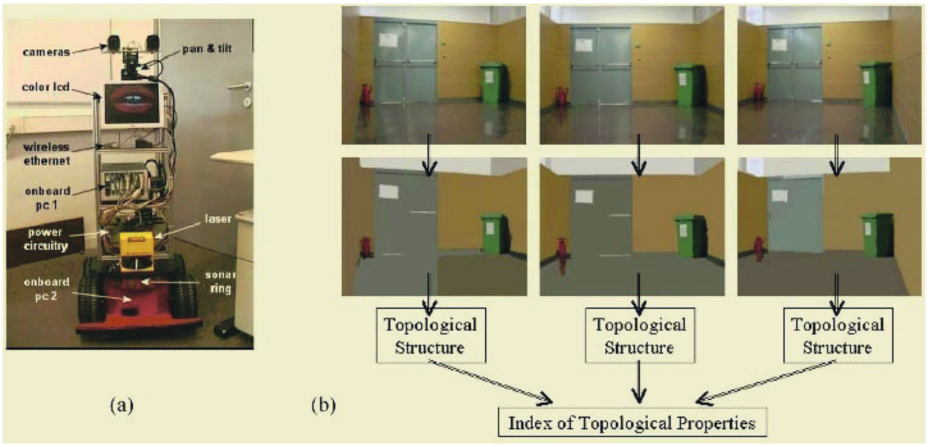


Fig. 7. (a) ANNA mobile robot; (b) learning process using three different views

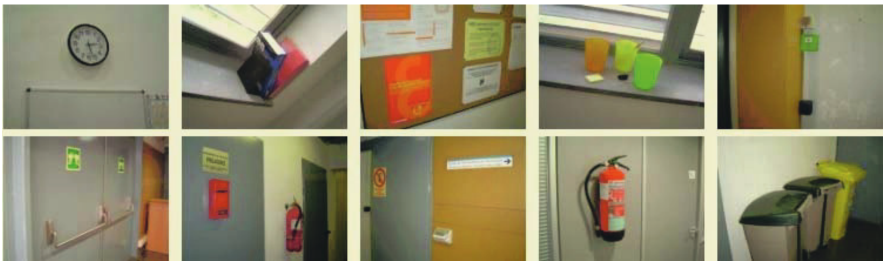


Fig. 8. Some images taken from the mobile robot called Marco

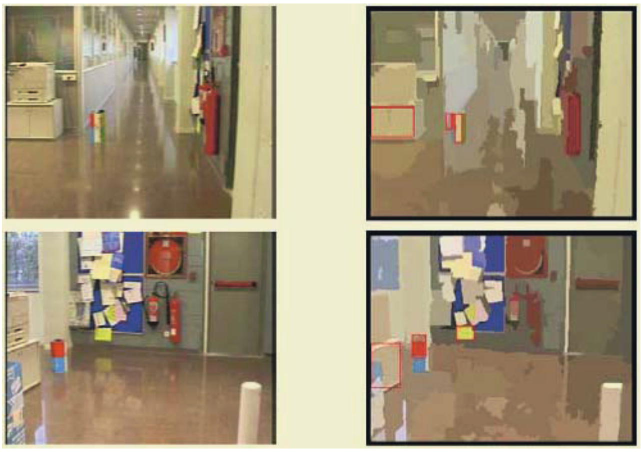


Fig. 9. Segmentation results of indoor buildings

7 Conclusions

In this paper we present some common issues that we find in robotics when a robot must use computer vision techniques for identifying objects and scenes in indoor buildings. We also explain several strategies used to locate and identify objects and some graph techniques applied in the identification process. We are at present testing these techniques in several sequences of indoor building images in order to see the robustness of them.

References

1. J. Andrade and A. Sanfeliu: Integration of perceptual grouping and depth. Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, September 3-7, IEEE Computer Society, Vol. 1, (2000), pp. 295-298.
2. S. Belongie and J. Malik: Matching with shape contexts. Proceedings IEEE Workshop on Image and Video Libraries, June 12, (2000), pp. 20-26.
3. A. Björner, M.L. Vergnas, B. Sturmfels, N. White, G.M.: Oriented matroids. Volume 43 of Encyclopedia of Mathematics and its Applications. Cambridge University Press, (1993).
4. H. Bunke: Recent developments in graph matching. Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, September 3-7, IEEE Computer Society, Vol. 1, (2000).
5. C. Chen and A. Kak: Robot vision system for recognizing objects in low-order polynomial time. IEEE Trans. on System, Man and Cybernetics, 18(6), Nov. (1989), pp. 1535-1536.
6. W.J. Christmas, J. Kittler and M. Petrou: Structural matching in computer vision using probabilistic relaxation. IEEE Transactions on PAMI, vol. 17, (1995), pp. 749-764.
7. G.N. DeSouza and A.C. Kak: Vision for mobile robot navigation, a survey. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 24, No. 2, Feb. (2002).
8. P.F. Felzenswalb and D.P. Huttenlocher: Image Segmentation Using Local Variation. Proc. of the IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition, (1998), pp. 98-104.

9. G. Finlayson and S. Hordley: Colour by correlation: a simple, unifying framework for colour constancy. IEEE Trans. on PAMI, 23(11): 1209-1221, Nov. (2001).
10. I. G. Gowda, D. G. Kirkpatrick, D. T. Lee and A. Naamad: Dynamic Voronoi diagrams, IEEE Trans. on Information Theory, Vol. IT-29, No.5, Sept. (1983).
11. B.Huet & E.R.Hancock: Relational object recognition from large structural libraries. Pattern Recognition, Vol. 35, (2002), pp: 1895-1915.
12. X.Jiang, A.Münger and H. Bunke: On median graphs: Properties, algorithms and applications. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 23, No. 10, (2001), pp: 1144-1151.
13. D. Kragié and H. I. Christensen: Cue integration for visual servoing. IEEE Trans. on Robotics and Automation, Vol. 17, No. 1, Feb. (2001).
14. J. Malik, S. Belongie, J. Shi and T. Leung: Textons, contours and regions: cue integration in image segmentation. Proceedings of the Seventh IEEE International Conference on Computer Vision, Vol.2, Sept. (1999), pp.918-925.
15. S. Nayar, S. Nene and H. Murase: Subspace methods for robot vision. IEEE trans. on Robotics and Automation Vol. 12 No. 5, (1996), pp. 750-758.
16. A. Sanfeliu, R. Alquezar, J. Andrade, J. Climent, F. Serratosa and J. Verges: Graph-based representations and techniques for image processing and image analysis. Pattern Recognition, Vol. 35, (2002), pp. 639-650.
17. A. Sanfeliu and F. Serratosa: Learning and recognising 3D models represented by múltiple views by means of methods based on random graphs. IEEE Int. Conference on Image Processing, Barcelona 2003 (ICIP 2003), Sept. (2003).
18. A. Sanfeliu, F. Serratosa and R. Alquezar: Second-order graphs for modelling sets of attributed graphs and their application to object learning and recognition. Int. J. Pattern Recognition and Artificial Intelligence, Vol. 18, No.3, (in press), (2004).
19. K.Sengupta & K.L.Boyer: Organizing large structural model bases. IEEE Trans. on PAMI, Vol. 17, No. 4, (1995), pp: 321-331.
20. F. Serratosa, R. Alquezar and A. Sanfeliu: Function-described graphs for modelling objects represented by sets of attributed graphs. Pattern Recognition, Vol. 36, (2003), pp. 781-798.
21. F. Serratosa, T. Grau and A. Sanfeliu: Distance between 2D-scenes base on oriented matroid theory. Proc. of the International Conference on Pattern Recognition, ICPR'04, Cambridge, August 23-26, (2004).
22. E. Staffetti, A. Grau, F. Serratosa and A. Sanfeliu: Shape representation and indexing based on region connection calculus and oriented matroid theory. In Discrete Geometry for Computer Imagery, Lecture Notes in Computer Science, Springer-Verlag, Napoles, (2003).
23. S. Umeyama: An eigendecomposition approach to weighet graph matching problems. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 10, No.5, Sept. (1988).
24. J. Vergés-LLahí, A. Sanfeliu: Colour constancy algorithm based on object function minimization. Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA), Puerto Andrax, Mallorca, 4-6 June. Lecture Notes in Computer Science, (2003).
25. P. Viola and M. Jones: Rapid object detection using a boosted cascade of simple features. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 8-14 dec. (2001), pp. 511-518.
26. L. Xu and I. King: A PCA approach for fast retrieval of structural patterns in attributed graphs. IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, Vol. 31, No. 5, Oct. (2001).
27. A.K.C. Wong and M. You: Entropy and distance of random graphs with application to structural pattern recognition. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 7, (1985), pp. 599-609.
28. E.K. Wong: Model matching in robot vision by sub-graph isomorphism. Pattern Recognition, Vol. 25, (1994), pp: 287-304.
29. H. Zhang and H. Yan: Graphic matching based on costrained Voronoi diagrams, Proceedings of the Fifth International Symposium on Sygnal Processing and its Applications, Brisbane, 22-25 August, (1999).

Graphical-Based Learning Environments for Pattern Recognition

Franco Scarselli[1], Ah Chung Tsoi[3], Marco Gori[1], and Markus Hagenbuchner[2]

[•] Dipartimento di Ingegneria dell'Informazione
University of Siena, Siena, Italy
[•] Faculty of Informatics
University of Wollonong, Wollongong, Australia
[•] Executive Director, Mathematics, Informatics & Comunication–Science
Australian Research Council, Camberra, Australia

Abstract. In this paper, we present a new neural network model, called graph neural network model, which is a generalization of two existing approaches, viz., the graph focused approach, and the node focused approach. The graph focused approach considers the mapping from a graph structure to a real vector, in which the mapping is independent of the particular node involved; while the node focused approach considers the mapping from a graph structure to a real vector, in which the mapping depends on the properties of the node involved. It is shown that the graph neural network model maintains some of the characteristics of the graph focused models and the node focused models respectively. A supervised learning algorithm is derived to estimate the parameters of the graph neural network model. Some experimental results are shown to validate the proposed learning algorithm, and demonstrate the generalization capability of the proposed model.

1 Introduction

In several applications, the data can be naturally represented by graph structures. The simplest kind of graph structures is a sequence, but, in many application domains, the information is organized in more complex graph structures such as trees, acyclic graphs, or cyclic graphs. In machine learning, the structured data is often associated with the goal of either supervised or unsupervised learning from examples, a function $\boldsymbol{h}$ which maps a graph G and one of its nodes n to a vector of reals[1]: $\boldsymbol{h}(G,n) \in \mathcal{R}^m$.

In general, applications to a graphical domain can be divided into two classes: called *graph focused* and *node focused* applications, respectively.

In graph focused applications, $\boldsymbol{h}$ is independent of the node n and implements a classifier or a regressor on a graph structured dataset. For example, an image can be represented by a Region Adjacency Graph (RAG) where the nodes

[•] Note that in classification problems, the mapping is to a set of integers $\mathcal{I}^m$, while in regression problems, the mapping is to a set of reals $\mathcal{R}^m$. Here for simplicity of exposition, we will denote only the regression case.

denote homogeneous regions of the image and the arcs represent their adjacency relationship (see Fig. 1(a)). In this case, $\boldsymbol{h}(G)$ may be used to classify the image into different classes, e.g., castles, cars, people, and so on.

In node focused applications, $\boldsymbol{h}$ depends on n, so that the classification (or the regression) relates to each node. Object localization is an example of this class of applications. It consists of finding whether an image contains a given object or not, and, if so, detect its position. This problem can be solved by a function $\boldsymbol{h}$ which classifies the nodes of the RAG according to whether the corresponding region belongs to the object or not. For example, in Fig. 1(a), the output of $\boldsymbol{h}$ might be 1 for the black nodes, which correspond to the castle, and 0 otherwise. Another example comes from web page classification. The web can be represented by a graph where nodes stand for pages and edges represent the hyperlinks (see Fig. 1(b)). The web connectivity can be exploited, along with page contents, for several purposes, e.g. classifying the pages into a set of topics.

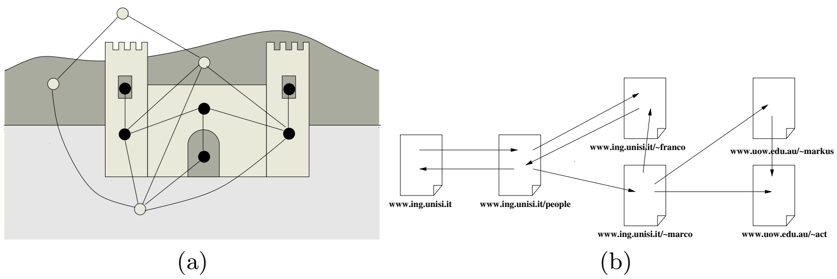


Fig. 1. Some applications where the information is represented by graphs: (a) an image; and (b) a subset of the web.

Most applications cope with graph structured data using a preprocessing phase which maps the graph structured information to a simpler representation, e.g. vectors of reals. However, important information, e.g., the topological dependency of information on node n may be lost during the preprocessing stage and the final result may depend, in an unpredictable manner, on the details of the preprocessing algorithm. More recently, there are various approaches [3, 1] attempting to preserve the graph structured nature of the data, for as long as required, before processing the data. In other words, these approaches attempt to avoid the preprocessing step of "squashing" the graph structured data into a vector of reals first, and then deal with the preprocessed data using a list based data processing technique, rather than paying special attention to the underlying graph structured nature of the data. In these recent approaches, the idea is to encode the underlying graph structured data using the topological relationship among the nodes of the graph. In other words, these recent approaches attempt to incorporate the graph structured information in the data processing step. In the graph focused approaches [3, 11, 5] this is done using *recursive neural networks* and in the node focused approaches [1, 8, 12], this is done commonly by using *random walk* techniques.

In this paper, we present a new neural network model which is suitable for both graph and node focused applications. This new model unifies these two existing models into a common framework. We will call this new neural network model a *graph neural network* (GNN). It will be shown that GNN is an extension of both recursive neural networks and random walk models and that it retains their characteristics.

The model extends recursive neural networks since it can process a more general class of graphs including cyclic, directed and undirected graphs, and to deal with node focused applications without any preprocessing steps. The approach extends random walk theory by the introduction of a learning algorithm and by enlarging the class of processes that can be modeled.

The structure of this paper is as follows: The notation used in this paper as well as preliminary materials are described in Section 2. Then, the concept of a graph neural network model, together with a learning algorithm for the parameter estimation of the model are presented in Section 3. Furthermore, some experimental results are presented in Section 4, and some conclusions are drawn in Section 5.

2 Notation and Preliminaries

In the following, $\| \cdot \|_1$ denotes norm 1, i.e. for a vector $\boldsymbol{V} = [v_1, .., v_k]$, $\|\boldsymbol{V}\|_1 = \sum_{i=1}^{k} |v_i|$, for a matrix $\boldsymbol{M} = \{m_{i,j}\}$, $\|\boldsymbol{M}\|_1 = \max_j \sum_i |m_{i,j}|$. A graph G is a pair $(\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{n_1, \ldots, n_r\}$ is a set of nodes and $\mathcal{E} = \{e_1, \ldots, e_p\}$ a set of edges. The set of children and parents of n are denoted by $\mathrm{ch}[n]$ and $\mathrm{pa}[n]$, respectively. The set $\mathrm{ne}[n]$ stands for the nodes connected to n by an arc: for directed graphs, we have $\mathrm{ne}[n] = \mathrm{pa}[n] \cup \mathrm{ch}[n]$, and for undirected graphs, $\mathrm{ne}[n] = \mathrm{pa}[n] = \mathrm{ch}[n]$ holds. Similarly, the set of arcs entering and emerging from node n are represented by $\mathrm{to}[n]$ and $\mathrm{from}[n]$, respectively, while $\mathrm{co}[n]$ represents their union. Nodes and edges may have labels, which we assume to be represented by real vectors. The labels attached to node n and edge (n_1, n_2) will be represented by $L_n \in \mathcal{R}^{l_N}$ and $L_{(n_1,n_2)} \in \mathcal{R}^{l_E}$ respectively. Given a set of integers $\mathcal{S}$ and a set of vectors $\boldsymbol{y}_i, i \in \mathcal{S}$, $\boldsymbol{y}_{\mathcal{S}}$ denotes the vector obtained by stacking together the $\boldsymbol{y}_i$. Thus, for example, $L_{\mathrm{ch}[n]}$ stands for the vector containing the labels of all the children of n.

Remark 1. Labels usually include features of objects related to nodes and features of the relationships between the objects. For example, in the case of a RAG (Fig. 1(a)), node labels may represent properties of the regions (e.g., area, perimeter, average color intensity), and edge labels the relative position of the regions (e.g., distance between baricenters and the angle between the momentums). Similarly, in the example shown in Fig. 1(b), node and edge labels can include a representation of the text contained in the documents and in the anchor texts, respectively. ∎

No assumption is made on the nature of the arcs, directed and undirected edges are both permitted. However, when different kinds of edges coexist in the same dataset, it is necessary to distinguish among them. Such a goal can be easily reached by attaching a proper label to each edge. Thus, in this case, different kinds of arcs turn out to be just arcs with different labels.

The purpose of the proposed method is to learn by examples a function $\boldsymbol{h} : \mathcal{G} \times \mathcal{N} \to \mathcal{R}^m$, where $\mathcal{G}$ is any set of graphs and $\mathcal{N}$ is the set of their nodes. Thus, a learning data set is a set of three tuples $\mathcal{L} = \{(G_i, n_i, \boldsymbol{t}_i) |\, G_i = (\mathcal{N}_i, \mathcal{E}_i) \in \mathcal{G}, n_i \in \mathcal{N}, \boldsymbol{t}_i \in \mathcal{R}^m, 1 \le i \le m\}$. A three tuple $(G_i, n_i, \boldsymbol{t}_i)$ denotes the fact that the desired target for node n_i (of graph G_i) is $\boldsymbol{t}_i$.

Remark 2. The learning data set may contain any number of graphs. In the limit it is possible both from a theoretical and a practical point of view that the whole dataset comprises of a single graph. The dataset consists of nodes with their associated data and the learning problem is well defined provided that there are reasonable numbers of nodes both in the learning data set and in the testing data set respectively. The problem of classifying web pages is a straightforward example of the limiting case. The web is represented by one single graph, the learning data set consists of some pages whose desired classification is known, whereas the classification of other pages on the web should be obtained by generalization. ■

Finally, our approach is based on fixed point theory and contraction mappings [7]. Here, we use the following simple fixed point theorem.

Theorem 1. *If $g : \mathrm{R}^d \to \mathrm{R}^d$ is differentiable and there exists $0 \le e < 1$ such that $\left\| \frac{\partial g}{\partial x}(x) \right\|_1 \le e$ where $\frac{\partial g}{\partial x}$ is the Jacobian matrix of g, then g is a contraction function. Thus, the following system*

$$x = g(x)$$

has one and only one solution x^*. Moreover, the sequence

$$x(t) = g(x(t-1))$$

converges exponentially to x^* for any $x(0)$. ■

3 A New Neural Network Model

The intuitive idea underlining the proposed approach is that graph nodes represent objects or concepts and edges represent their relationships. Thus, we can attach to each node n a vector $\boldsymbol{x}_n \in \mathcal{R}^s$, called *state*, which collects a representation of the object denoted by n [2]. In order to define $\boldsymbol{x}_n$, we observe that the related nodes are connected by edges. Thus, $\boldsymbol{x}_n$ is naturally specified using the information contained in the neighborhood of n, which includes the label of n, the labels of the edges which are connected to n, and the states and the labels of the nodes on the neighborhood of n, respectively (see Figure 2).

[2] More precisely, $\boldsymbol{x}_n$ should collect all the information which is relevant for deciding the output $\boldsymbol{h}(G, n)$ in correspondence of n.

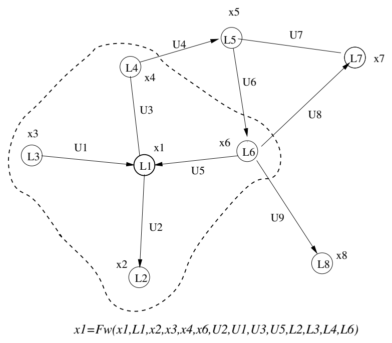


Fig. 2. The state $\boldsymbol{x}.$ depends on the neighborhood information.

More precisely, let F_w be a parametric function that expresses the dependence of a node on its neighborhood. The states $\boldsymbol{x}_n$ are defined as the solution of the following system of equations

$$\boldsymbol{x}_n = \boldsymbol{F}_w(\boldsymbol{x}_n, L_n, \boldsymbol{x}_{\mathrm{ne}[n]}, L_{\mathrm{co}[n]}, L_{\mathrm{ne}[n]}),\ 1 \le n \le r \tag{1}$$

where L_n, $L_{\mathrm{co}[n]}$, $\boldsymbol{x}_{\mathrm{ne}[n]}$, $L_{\mathrm{ne}[n]}$ are the label of n, the labels of its edges, the states and the labels of the nodes in the neighborhood of n respectively.

Remark 3. Definition (1) is customized for undirected graphs. When dealing with directed graphs, $L_{\mathrm{co}[n]}$ should be replaced by $L_{\mathrm{from}[n]}$, $L_{\mathrm{to}[n]}$ and similarly, $\boldsymbol{x}_{\mathrm{ne}[n]}$, $L_{\mathrm{ne}[n]}$ by $\boldsymbol{x}_{\mathrm{ch}[n]}$, $\boldsymbol{x}_{\mathrm{pa}[n]}$, and $L_{\mathrm{ch}[n]}$ $L_{\mathrm{pa}[n]}$, respectively. In the following sections, in order to keep the notations simple, we maintain this customization. However, unless explicitly stated, all the results hold also for directed graphs and mixed undirected and directed graphs. ■

Remark 4. Equation (1) should be considered only an example of the possible dependences of a node on its neighborhood. More generally, x_n could be computed from a subset of the parameters in (1) or, on the other hand, the neighborhood could include nodes which are k edges far from n. ■

For each node n, an output vector $\boldsymbol{o}_n \in \mathcal{R}^m$ is also defined which depends on the state $\boldsymbol{x}_n$ and label L_n. The dependence is described by a parametric function O_w

$$\boldsymbol{o}_n = \boldsymbol{O}_w(\boldsymbol{x}_n, L_n),\ 1 \le n \le r. \tag{2}$$

Notice that, in order to ensure that $\boldsymbol{x}_n$ is correctly defined, system (1) must have a unique solution. In general, the number and the existence of solutions depend on $\boldsymbol{F}_w$. Here, we assume that $\boldsymbol{F}_w$ is appropriately designed so that the solution is unique. More precisely, let $\boldsymbol{X}$ and $\boldsymbol{L}$ respectively be the vectors constructed by stacking all the states and all the labels. Then, Equations (1) can be written as:

$$\boldsymbol{X} = \boldsymbol{\Phi}_{\boldsymbol{w},\boldsymbol{L}}(\boldsymbol{X}) \tag{3}$$

where $\boldsymbol{\Phi}_{\boldsymbol{w},\boldsymbol{L}}$ consists of r instances of $\boldsymbol{F}_w$, and $\boldsymbol{w}$ is the set of parameters. The key choice adopted in the proposed approach consists of designing $\boldsymbol{F}_w$ such that it will be a contraction mapping and $\boldsymbol{\Phi}_{\boldsymbol{w},\boldsymbol{L}}$ will satisfy the hypothesis of Theorem 1, i.e. there exists $0 \leq e < 1$ such that $\|\frac{\partial \boldsymbol{\Phi}_{\boldsymbol{w},\boldsymbol{L}}}{\partial \boldsymbol{X}}(\boldsymbol{X})\| \leq e$ for any $\boldsymbol{w}, \boldsymbol{L}, \boldsymbol{X}$.

In fact, function $\boldsymbol{F}_w$ and $\boldsymbol{O}_w$ will be implemented by particular models of static neural networks. Thus, Equations (1) and (2) specify a new theoretical model suitable for node focused applications. In fact, (1) and (2) define a method to attach an output o_n to each node of a graph, i.e. a parametric function $\boldsymbol{f}_w(G,n) = \boldsymbol{o}_n$ which operates on graphs.

The corresponding learning problem consists of adapting the parameters $\boldsymbol{w}$ of $\boldsymbol{O}_w$ and $\boldsymbol{F}_w$ so that f_w approximates the data in the learning data set. In practice, the learning problem may be implemented by the minimization of a quadratic error function

$$e_w = \sum_{i=1}^{r} (\boldsymbol{t}_i - \boldsymbol{f}_w(G_i, n_i))^2 \,. \tag{4}$$

Finally, since the number of inputs of $\boldsymbol{F}_w$ is not fixed, but depends on the number of neighbors of each node, the implementation of $\boldsymbol{F}_w$ may be difficult, particularly when the degree of node connectivity undergoes large changes. For this reason, it may be useful to replace Equations (1) with

$$\boldsymbol{x}_n = \sum_{\ell \in \mathrm{ne}[n]} \boldsymbol{H}_w(\boldsymbol{x}_n, L_n, L_{(n,\ell)}, L_\ell) \tag{5}$$

The intuitive idea underlining eq. (5) consists of computing the state $\boldsymbol{x}_n$ by the summing a set of "contributions". Each contribution is generated considering only one node in the neighborhood of n. Definition eq. (5) is less general than (1), but the implementation of $\boldsymbol{H}_w$ is easier since $\boldsymbol{H}_w$ has a fixed number of parameters.

In order to implement the model formally defined by Equations (1) and (2), the following items must be provided:

- A method to solve (1);
- A learning algorithm to adapt $\boldsymbol{F}_w$ and $\boldsymbol{O}_w$ by examples from the training data set[3];
- An implementation of $\boldsymbol{F}_w$ and $\boldsymbol{O}_w$ for which $\boldsymbol{\Phi}_{\boldsymbol{w},\boldsymbol{L}}$ is a contraction mapping.

These aspects will be considered in turn in the following subsections.

3.1 Computing the States

Theorem 1 does not only provide a sufficient condition for the existence of the solution of equation (1), but it also suggest how to compute its fixed point. In fact, for any initial set of states the following dynamical system

[3] In other words, the parameters $\boldsymbol{w}$ are estimated from the training data set.

$$\boldsymbol{x}_n(t) = \boldsymbol{F}_w(\boldsymbol{x}_n(t-1), L_n, x_{\mathrm{ne}[n]}(t-1), L_{\mathrm{co}[n]}, L_{\mathrm{ne}[n]}), \tag{6}$$

where $\boldsymbol{x}(t)$ denotes the t-th iterate of $\boldsymbol{x}$, converges exponentially fast to the solution of system (1).

Notice that system (6) can be interpreted as the representation of a network consists of units which compute $\boldsymbol{F}_w$ and $\boldsymbol{O}_w$. Such a network will be called an *encoding network*, following a similar terminology used for the recursive neural network model [3]. In order to build the encoding network, each node of the graph can be replaced by a unit computing the function $\boldsymbol{F}_w$ (see Figure 3). Each unit stores the current state $\boldsymbol{x}_n(t)$ of the corresponding node n, and, when activated, it calculates the state $\boldsymbol{x}_n(t+1)$ using the labels and the states stored in its neighborhood. The simultaneous and repeated activation of the units produces the behavior described by system (6). In the encoding network, the output for node n is produced by another unit which implements $\boldsymbol{O}_w$.

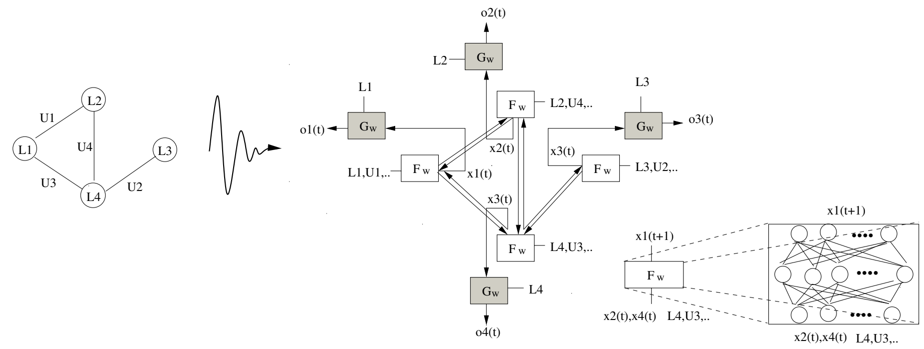


Fig. 3. A graph and its corresponding encoding network.

When $\boldsymbol{F}_w$ and $\boldsymbol{O}_w$ are implemented by static neural networks, the encoding network is a large recurrent neural network where the connections between the neurons can be divided into internal and external connections respectively. The internal connectivity is determined by the neural network architecture used to implement the unit. The neural architecture which have been suggested for realizing this type of problems in the literature for solving a graph focused problem include multilayer perceptrons [3, 11], cascade correlation, and self organizing maps [5, 6]. For node focused problems, e.g., in web page classifications, as far as we are aware, there is only one application of such a concept using a linear model [12]. The external connectivity mimics the graph connections. Moreover, the weights of such a recurrent neural network are shared, since the same parameters w are common to all the units.

3.2 A Learning Algorithm

Without loss of generality let us assume that the learning data set contains one single graph. This is a general case, as when we have many graphs, it is possible to transform this into a single graph by grouping them into one single non–connected graph. The learning algorithm we propose consists of two phases:

(a) the states $\boldsymbol{x}(t)$ are repeatedly updated, using Eq. (6) until they reach a stable point at time T;
(b) the gradient $\frac{\partial e_w(T)}{\partial \boldsymbol{w}}$ is computed and the weights $\boldsymbol{w}$ are updated according to a gradient descent strategy.

These two phases are repeated until a given stopping criterion is reached A similar approach, based on a stabilizing and a learning phase, was already proposed for training a random walk process in [2]. Thus, while phase (a) moves the system to the stable point, phase (b) adapts the weights to change the outputs towards the desired target. It is worth noting that the gradient $\frac{\partial e_w(T)}{\partial \boldsymbol{w}}$ depends only on the error at time T, when the system is supposed to be stable. In fact, the output of our model depends on function $\boldsymbol{O}_w$ and on the stable point which is determined by $\boldsymbol{F}_w$. In order to obtain the desired outputs, it is necessary to change the fixed point along with $\boldsymbol{O}_w$. The proposed algorithm can be interpreted as a gradient descend whose goal consists of moving the fixed point to a new position where the function $\boldsymbol{O}_w$ can produce the desired output more readily. For this reason, only the error at time T is to be considered.

The Gradient Computation. The gradient could be computed using a backpropagation through time algorithm [4]. In this case, the encoding network is unfolded from time T back to an initial time t_0. The unfolding produces a layered network (see Figure 4). Each layer corresponds to a time instance and contains a copy of all the units $\boldsymbol{F}_w$ of the encoding network. The units of two consecutive layers are connected following the encoding network connectivity (i.e. the graph connectivity). The last layer corresponding to time T includes also the unit $\boldsymbol{O}_w$ and allows to compute the output of the network. Backpropagation through time consists of carrying out a common backpropagation on the unfolded network in order to compute the gradient of the error at time T with respect to all the instances of $\boldsymbol{F}_w$ and $\boldsymbol{O}_w$. Then, $\frac{\partial e_w(T)}{\partial \boldsymbol{w}}$ is obtained summing the gradients of all instances.

However, backpropagation through time requires to store the states of every instance of the units. When the graphs and $T - t_0$ are large, the memory required may be considerable[4]. On the other hand, in our case, a more efficient approach is possible. Since the system has reached a stable point, we can assume that $\boldsymbol{x}(t) = \boldsymbol{x}(t_0)$ for any $t \geq t_0$. Thus, the states of units remain constant for each instant, and backpropagation through time can be carried out storing only $\boldsymbol{x}(t_0)$. More details on gradient computation are available in [9].

[4] Internet applications, where the graph may represent a portion of the web, are a straightforward example of cases when the amount of required storage may have a very important role.

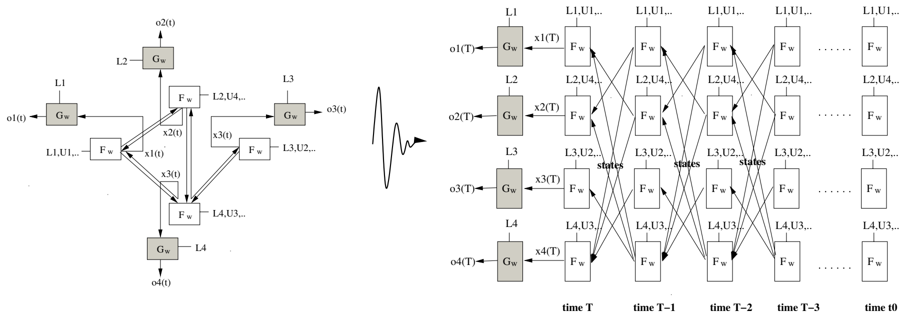


Fig. 4. A graph and its encoding network to illustrate the backpropagation through time concept.

The Learning Algorithm. The learning algorithm is summarized in Table 1. It consists of a main procedure and two functions called *Forward* and *Backward* respectively. The function *Forward* takes in input the current set of parameters $\boldsymbol{w}$ and the current state $\boldsymbol{X}$ and iterates the system equations. The iteration is stopped when $\|\boldsymbol{X}(t+1)-\boldsymbol{X}(t)\|$ is less than a given threshold ϵ_f. The function *Backward* computes the gradient using a time window $[T, t_0]$ such that $\|\frac{\partial e_w(T)}{\partial \boldsymbol{X}(t_0)} - \frac{\partial e_w(T)}{\partial \boldsymbol{X}(t_0-1)}\| \leq \epsilon_b$.

Table 1. The learning algorithm.

```
/* Main procedure */
Learn($\boldsymbol{F}_w, \boldsymbol{O}_w, \mathcal{L}$)
initialize $\boldsymbol{w}$, $\boldsymbol{X}$;
$\boldsymbol{X}$:=Forward($\boldsymbol{X}, \boldsymbol{w}$);
repeat
    $\frac{\partial e_w}{\partial \boldsymbol{w}}$:=Backward($\boldsymbol{X}, \boldsymbol{w}$);
    $\boldsymbol{w}$:=$\boldsymbol{w} - \lambda \cdot \frac{\partial e_w}{\partial \boldsymbol{w}}$;
    $\boldsymbol{X}$:=Forward($\boldsymbol{X}, \boldsymbol{w}$);
until the stopping criterion
        is achieved;
return $\boldsymbol{w}$;
end

/* Move to a stable point */
Forward($\boldsymbol{X}, \boldsymbol{w}$)
$\boldsymbol{X}(0)$:=$\boldsymbol{X}$, $t = 0$;
repeat
    Compute $\boldsymbol{X}(t+1)$ ;
        from $\boldsymbol{X}(t)$;
    $t$:=$t+1$;
until $\|\boldsymbol{X}(t+1)-\boldsymbol{X}(t)\| \leq \epsilon_f$;
return $\boldsymbol{X}(t+1)$;
end

/* Compute the gradient */
Backward($\boldsymbol{X}$,$\boldsymbol{w}$)
Assume $\boldsymbol{X}(t) = \boldsymbol{X}$ for each $t$;
Find a window $[T, t_0]$ s.t.
    $\|\frac{\partial e_w(T)}{\partial \boldsymbol{X}(t_0)} - \frac{\partial e_w(T)}{\partial \boldsymbol{X}(t_0-1)}\| \leq \epsilon_b$;
Compute $\frac{\partial e_w(T)}{\partial \boldsymbol{w}}$ by
    backpropagation through time
    on the window $[T, t_0]$;
end
```

The main procedure calls the functions Forward and Backward and updates the weights until the output reaches a desired accuracy or some other stopping criterion is achieved. In Table 1, the weights are updated according to a simple gradient descent strategy with a fixed learning rate λ. However, other strategies are also possible, e.g. based on an adaptive learning rate, as long as the adaptive learning rate decreases faster than a constant rate. Moreover, while the initialization of parameters w depends on the particular implementation of F_w and O_w, in theory, X in the main procedure can be initialized to any value. In practice, it is simplest to set this to be $X = 0$.

Finally, the values ϵ_f and ϵ_b are design parameters. It can be proved that if $\boldsymbol{\Phi}_{\boldsymbol{w},\boldsymbol{L}}$ is a contraction mapping, then $\|\boldsymbol{X}(t+1)-\boldsymbol{X}(t)\|$ and $\|\frac{\partial e_w(T)}{\partial \boldsymbol{X}(t_0)}-\frac{\partial e_w(T)}{\partial \boldsymbol{X}(t_0-1)}\|$ converge exponentially to 0, when t and $t-t_0$ increase, respectively. Thus, it is possible to set ϵ_f and ϵ_b to very small values without effecting heavily the performance of the algorithm.

3.3 Comparing Our Approach with Recursive Neural Networks and Random Walks

Recursive neural networks are a special case of the model described in (6), where

- the input graph is directed and acyclic;
- the inputs of $\boldsymbol{F}_w$ are limited to $\boldsymbol{L}_n$ and $\boldsymbol{x}_{\mathrm{ch}[n]}(t-1)$;
- the graph should contain a node s called supersource from which all the other nodes can be reached;
- the recursive network is the output $\boldsymbol{o}_s$ computed the supersource.

Note that the above constraints on the processed graphs and on the inputs of $\boldsymbol{F}_w$ exclude any sort of cyclic dependence of a state on itself. Thus, in the recursive model, the encoding networks are feedforward networks.

This assumption simplifies the computation of the node states. In fact, the states can be computed following a predefined direction, i.e. from the leaf nodes to the supersource node of the graph. First, the states of the leaf nodes are calculated, then the states of their parents are computed and so on up to the supersource node. For the supersource node, the recursive neural network computes also an output, which is returned as the result of the graph computation.

Moreover, the above assumptions allow to train recursive neural networks by applying a common backpropagation procedure on the encoding network [3, 11]. This solution is not viable for GNNs, since the presence of cyclic dependencies among the states transforms the encoding network into a dynamic system. For this reason, it has been necessary to assume that the function $\boldsymbol{\Phi}_{\boldsymbol{w},\boldsymbol{L}}$ is a contraction map and to propose a new learning algorithm. However, it must be pointed out that the learning algorithm adopted for GNNs is an extension of the one used for recursive neural networks and that the two algorithms behave in the same way on acylic graphs.

On the other hand, in a random walk model, $\boldsymbol{F}_w$ is a linear function. In a simple case, the states x_n associated with nodes are real values and satisfy

$$x_n(t+1) = \sum_{i \in \mathrm{pa}[n]} w_{n,i} x_i(t) \qquad (7)$$

where $w_{n,i} \in \mathcal{R}$ and $w_{n,i} \geq 0$ hold for each n, i. The $w_{n,i}$ are normalized so that $\sum_{i \in ch[n]} w_{i,n} = 1$. In fact, Eq. (7) can represent a random walker who is traveling on the graph. The value $w_{n,i}$ represents the probability that the walker, when he/she is on node n, decides to go to node i. The state x_n stands for the probability that the walker is on node n in the steady state. When all the x_n are stacked into a vector $\boldsymbol{X}$, Eq. (7) becomes

$$\boldsymbol{X}(t+1) = \boldsymbol{W}\boldsymbol{X}(t) \tag{8}$$

where $\boldsymbol{W} = \{w_{n,i}\}$ and $w_{n,i}$ is as defined in Eq. (7) if $i \in \mathrm{pa}[n]$ and $w_{n,i} = 0$ otherwise. It is easily verified that $\|\boldsymbol{W}\|_1 = 1$. Markov chain theory suggests that if there exists t such that all the elements of the matrix $\boldsymbol{W}^t$ are non–null, then Eq. (7) is a contraction mapping [10].

Thus random walks on graphs are an instance of our model, since they implement a linear version of it. The set of processed graphs include cyclic graphs, but these graphs are usually unlabeled. Moreover, random walk theory does not provide a learning algorithm. In our development described in this paper, we have proposed a learning algorithm which allows the estimation of the set of parameters from training samples. Thus, our model extends the work on random walk models by providing the possibility of learning the parameters from training samples.

3.4 Implementing $\boldsymbol{F_w}$ and $\boldsymbol{O_w}$

The implementation of $\boldsymbol{O}_w$ does not need to fulfill any particular constraints. In our experiments, $\boldsymbol{O}_w$ will be simply implemented by a feedforward neural network (a multilayer perceptron). On the other hand, $\boldsymbol{F}_w$ plays a crucial role in the proposed model, since its implementation determines the number and the existence of the solution of Equation (1).

The key choice adopted in our approach consists of designing $\boldsymbol{F}_w$ such that $\boldsymbol{\Phi}_{w,L}$ is a contraction mapping. Let $\delta_{n,i,u,j}$ denote the element of the Jacobian $\frac{\partial \boldsymbol{\Phi}_{w,L}(X)}{\partial \boldsymbol{X}}$ of $\boldsymbol{\Phi}_{w,L}$ whose row corresponds to j–th component of node u and whose column corresponds to i–th component of node n. By Theorem 1, $\boldsymbol{\Phi}_{w,L}$ is a contraction mapping provided that it is differentiable and

$$\left| \sum_{n,i} \delta_{n,i,u,j} \right| \leq e \tag{9}$$

holds for some real number $0 \leq e < 1$. Inequality (9) can be used to design the $\boldsymbol{F}_w$ in (1) or the $\boldsymbol{H}_w$ in (5) such that $\boldsymbol{\Phi}_{w,L}$ is a contraction mapping.

In this paper, two implementations of $\boldsymbol{\Phi}_{,L}$ are suggested:

(a) $\boldsymbol{H}_w$ is realized by a linear system whose parameters are determined by a neural network; the model is such that Eq. (9) holds for any set of parameters w.

(b) $\boldsymbol{F}_w$ is realized by a common feedforward neural network: the cost function adopted in the learning procedure includes a penalty term that keeps the parameters w in the region where Eq. (9) is fulfilled.

Implementation of $\boldsymbol{H}_w$ by a Linear Function. In the first implementation, $\boldsymbol{H}_w$ is a linear function

$$\boldsymbol{x}_n = \boldsymbol{E}_n + \sum_{r \in \mathrm{ne}[n]} \boldsymbol{W}_{n,r} \boldsymbol{x}_r$$

similar to that used in random walks. But, the state attached to the nodes are vector of reals instead of simple reals and the parameters are not statically defined, but they are computed by two feedforward neural networks $\mathcal{N}_E$ and $\mathcal{N}_W$. The neural network decides the parameters $\boldsymbol{E}_n \in \mathcal{R}^s$ and $\boldsymbol{W}_{n,r} \in \mathcal{R}^{s \times s}$ on the basis of the labels attached to nodes n, r and the arc (r, n). More formally, let $\boldsymbol{f}_E : \mathcal{R}^{l_N} \to \mathcal{R}^{nr}$ and $\boldsymbol{f}_W : \mathcal{R}^{l_E} \to \mathcal{R}^{n^2 r^2}$ be the functions implemented by $\mathcal{N}_E$ and $\mathcal{N}_W$, respectively. Then, we can define

$$\begin{aligned} \boldsymbol{E}_n &= \boldsymbol{f}_E(L_n) \\ \boldsymbol{W}_{n,r} &= \mathrm{Resize}_{s \times s} \left(\frac{e}{s|\mathrm{ne}[r]|} \boldsymbol{f}_W(L_{(n,r)}) \right) \end{aligned}$$

where $\mathrm{Resize}_{s \times s}(\cdot)$ denotes the operator that rearranges the components of a $s^2 \times 1$ vector into a $s \times s$ matrix and $|\mathrm{ne}[r]|$ represents the cardinality of $\mathrm{ne}[r]$.

In this case, inequality (9) holds provided that the output of $\boldsymbol{f}_W$ is in the range $[-1, 1]$, which can be achieved by using a sigmoidal activation function in the output layer of $\mathcal{N}_W$. In fact, $\delta_{n,i,u,j} = (\boldsymbol{W}_{n,u})_{i,j}$ and, as a consequence,

$$\left| \sum_{n,i} \delta_{n,i,u,j} \right| \le \sum_{n,i} |\boldsymbol{W}_{n,u}|_{i,j} \le \sum_{n \in \mathrm{ne}[u], i} \frac{e}{s|\mathrm{ne}[u]|} = 1 \, .$$

Implementation of $\boldsymbol{F}_w$ by a Neural Network. Let us suppose that $\boldsymbol{F}_w$ is realized by a layered feedforward neural network with logistic sigmoid activation functions. In this case, (9) holds only for some values of the parameters $\boldsymbol{w}$. In fact, $\delta_{r,j,n,i}$ is small for small values of the network parameters, but it may become large for large values, e.g. when the hidden–to–output weights are large. In order to ensure (9) is fulfilled, a penalty term can be added to the error function which becomes

$$e_w = \sum_{k=1}^{m} (t_k - f_w(G_k, n_k))^2 + \beta \sum_{n,i} L \left(\sum_{r,j} \delta_{r,j,n,i} \right)$$

where β is a predefined parameter balancing the importance of the error on patterns and the penalty term, and $L(y)$ is $(y - e)^2$ if $|y| > e$ and 0, otherwise. Note that the same reasoning can be applied also to the case when $\boldsymbol{H}_w$ instead of $\boldsymbol{F}_w$ is implemented by a layered neural network.

4 Experimental Results

In the paper, we present some preliminary results obtained using the linear implementation of $\boldsymbol{H}_w$. More experiments, including some obtained by directly implementing $\boldsymbol{H}_w$ with a neural network, are in [9]. The linear implementation of GNN has been verified on the subgraph recognition problem and the web page ranking problem.

The subgraph recognition problem consists of identifying the presence of a subgraph in a larger graph. In our experiments, we used random graphs. The graphs have integer labels in the range $[0, 10]$ and these labels have been added a normal distribute noise having mean of 0 and a variance 0.25. Different dimensions for the graphs and the subgraphs have been evaluated in different experiments.

Tables 2 shows the results of the experiments. Each column is related to a different set of experiments. The notation $s - g$ in the header of the column defines the number of nodes s of the subgraph to be identified and the total number of nodes g in the graphs that contain the subgraph. For any pair $s - g$ the experiment has been carried out 5 times with different subgraphs. In each experiment the dataset contained 450 graphs, equally distributed between the training dataset, the testing dataset, and a validation set.

The results are interesting. In fact, it must be pointed out that that the classic subgraph recognition algorithms cannot evaluate situations when there are corruptions in the graph labels. On the other hand, as may be observed in our results, our proposed method can handle such situation quite easily. Moreover, to verify the capability of the method in learning the graph connectivity, the results have been compared with those achieved by a common three layered (one hidden) neural network (FNN) which takes in as its input only the label of a node n. It is observed that GNNs clearly outperforms this latter approach.

Table 2. The results of the subgraph recognition problem.

	$3-6$	$5-6$	$3-10$	$5-10$	$7-10$	$3-14$	$5-14$	$7-14$	*Average*
GNN on testset with noise	91.62	93.05	86.41	78.70	86.94	86.71	78.56	79.81	85.22
FNN on testset with noise	71.67	87.22	69.39	58.17	74.16	72.86	67.34	55.93	69.59
GNN on trainset with noise	92.28	93.85	86.96	79.64	87.97	86.87	80.56	80.99	86.14
FNN on traintset with noise	70.85	87.08	69.83	57.71	74.23	73.09	67.43	55.85	69.51
GNN on testset no noise	94.22	93.03	89.95	84.88	90.24	89.75	83.49	80.11	88.21
GNN on trainset no noise	94.77	93.58	91.06	85.81	90.86	90.41	83.97	80.17	88.83
FNN on trainset no noise	73.48	88.23	69.96	66.45	78.74	71.68	65.49	58.89	71.62

In a second experiment, the goal was to simulate a web page ranking algorithm. For this experiment a random graph containing 1000 nodes has been generated. To each node, a label $[a, b]$ has been attached where a, b are binary values. The label represents whether the page belongs to two given topics. If the page belongs to both topics, then, $[a, b] = [1, 1]$; if it belongs to only one topic, then $[a, b] = [1, 0]$, or $[a, b] = [0, 1]$ and if it does not belong to either topics then $[a, b] = [0, 0]$. The GNN was trained in order to produce the following output o_i

$$o_i = \begin{cases} 2 * PR_i & \text{if } a = 0, b = 1 \; or \; a = 1, b = 0 \\ PR_i & \text{otherwise} \end{cases}$$

where PR_i stands for the Google's PageRank [1] of page n. Thus this experiment simulates the situation when a user wishes to see pages which belong to one topic and not the other and have their page ranks raised to twice that of the PageRank as given by method described in [1]. Such wishes are encountered often in the construction of web portals.

The two function $\boldsymbol{F}_w$ and $\boldsymbol{f}_E$ were implemented by three layer neural networks (one hidden layer) with linear output function. For the output function $\boldsymbol{O}_w$, two implementations have been evaluated: a three layer neural network and a two layer neural network. Figure 5 (a) and (b) show the output of the two layer network and the output of the three layer network, respectively. The plots display the desired rank (the continuous line) w.r.t. the rank computed by GNN (the dots). The pages, sorted by the desired rank, are displayed on horizontal axes, the ranks on the vertical ones. The plots show that the three layered network achieves better results and approximate well the desired function.

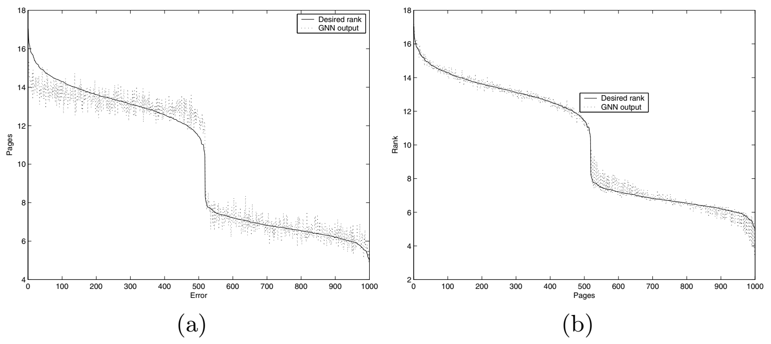


Fig. 5. The output of the model using a two layer network (a) and a three layer network (b) to implement $\boldsymbol{O}_w$.

5 Conclusions

In this paper, we have presented a unified approach to considering both a graph focused approach and a node focused approach to graph structured data. We have discussed the properties of the new neural model (GNN) and we have further provided a learning algorithm which can estimate the parameters. The preliminary experimental results confirms the viability of the approach.

Future research directions include a wide experimentation of GNNs, both to validate them on real life applications and to test different implementations of the functions $\boldsymbol{F}_w$ and $\boldsymbol{O}_w$. At the same time, a number of theoretical questions are still open, including an analysis of the approximation capability of GNNs and more general sufficient conditions to guarantee the existence and the uniqueness of the solution of Eq. (1).

Acknowledgements

The first, third and fourth authors acknowledge the financial support from the Australian Research Council through a Linkage International Exchange scheme which makes the research reported in this paper possible. In addition, the first and fourth authors acknowledge the financial support from the Australian Research Council through a discovery project grant.

References

1. Sergey Brin and Lawrence Page. The anatomy of a large–scale hypertextual Web search engine. In *Proceedings of the 7th World Wide Web Conference*, April 1998.
2. M. Diligenti, M. Gori, and M. Maggini. A learning algorithm for web page scoring systems. In *Proceedings of International Joint Conference on Artificial Intelligence*, 2003.
3. P. Frasconi, M. Gori, and A. Sperduti. A general framework for adaptive processing of data structures. *IEEE Transactions on Neural Networks*, 9(5):768–786, September 1998.
4. C. Lee Giles and Marco Gori, editors. *Adaptive Processing of Sequences and Data Structures, International Summer School on Neural Networks, "E.R. Caianiello", Vietri sul Mare, Salerno, Italy, September 6-13, 1997, Tutorial Lectures*, volume 1387 of *Lecture Notes in Computer Science*. Springer, 1998.
5. M. Hagenbuchner, A. Sperduti, and A.C. Tsoi. A self-organizing map for adaptive processing of structured data. *IEEE Transactions on Neural Networks*, 2003.
6. M. Hagenbuchner, A.C. Tsoi, and A. Sperduti. A supervised self-organising map for structured data. In N.Allinson, H.Yin, L.Allinson, and J.Slack, editors, *WSOM 2001 - Advances in Self-Organising Maps*, pages 21–28. Springer, June 2001.
7. Mohamed A. Khamsi. *An Introduction to Metric Spaces and Fixed Point Theory.* John Wiley & Sons Inc, 2001.
8. J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
9. F. Scarselli, A. C. Tsoi, M. Gori, and M. Hegenbuchner. A new neural network approach to graph processing. Technical Report DII /04, Dipartimento di Ingeneria dell'Inforazione, University of Siena, Siena, Italy, 2004.
10. E. Seneta. *Non–negative matrices and Markov chains.* Springer Verlag, 1981. Chapter 4, pages 112–158.
11. A. Sperduti and A. Starita. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*, 8:429–459, 1997.
12. A. C. Tsoi, G. Morini, F. Scarselli, Hagenbuchner, and M. M., Maggini. Adaptive ranking of web pages. In *Proceedings of the 12th WWW Conference*, Budapest, Hungary, May 2003.

Spectral Analysis of Complex Laplacian Matrices

Richard C. Wilson and Edwin R. Hancock

Department of Computer Science
University of York, York Y01 5DD, UK

Abstract. This paper explores how to extend the spectral analysis of graphs to the case where the nodes and edges are attributed. To do this we introduce a complex Hermitian variant of the Laplacian matrix. Our spectral representation is based on the eigendecomposition of the resulting Hermitian property matrix. The eigenvalues of the matrix are real while the eigenvectors are complex. We show how to use symmetric polynomials to construct permutation invariants from the elements of the resulting complex spectral matrix. We construct pattern vectors from the resulting invariants, and use them to embed the graphs in a low dimensional pattern space using a number of well-known techniques including principal components analysis, linear discriminant analysis and multidimensional scaling.

1 Introduction

Spectral graph theory is concerned with understanding how the structural properties of graphs can be characterised using the eigenvectors of the adjacency matrix or the closely related Laplacian matrix (the degree matrix minus the adjacency matrix). There is a good introductory text on the subject by Biggs [5], and comprehensive reviews of recent progress in the field can be found in the research monograph of Chung [1], and the survey paper of Mohar [4], Although spectral methods have been extensively used to address the segmentation, or grouping [8] and correspondence matching [2] problems, there has been less work on using spectral characteristics to perform pattern analysis on sets of graphs and trees. Recently, however, there has been some work aimed at filling this gap in the literature. First, it has been shown how eigenvalues can be used to index shock trees [7]. Second, adjacency matrix eigenvectors can be used to construct simple structural attributes for graphs [3].

However, exsiting spectral methods are confined to the case of graphs with weighted nodes and edges, and do not easily extend to the case of attributed graphs. To overcome this problem in this paper, we explore the use of a richer property matrix representation which can be used with attributed graphs. Conventional spectral methods make use of the eigenvalues of the Laplacian matrix (i.e. the degree matrix minus the weight matrix). This allows only very limited information concerning the properties of the graph to be encoded. Our property matrix, on the other hand, allows more information concerning graphs to be encoded by allowing complex entries, rather than the purely real entries in the conventional Laplacian. To compute this matrix we multiply the off-diagonal elements of the Laplacian (i.e. the negative edge weights) by a complex number that encodes the edge attributes. The node attributes are encoded in the diagonal elements.

The property matrix is Hermitian, and hence it has real eigenvalues and complex eigenvectors. To characterise the properties of the graphs, we construct permutation invariants by applying symmetric polynomials to the real and imaginary components of the complex eigenvectors. The invariants are used as the components of pattern vectors for the shock graphs. Sets of shock graphs can be visualised and clustered by applying simple pattern analysis techniques to the pattern vectors. Here we investigate the use of principal components analysis, linear discriminant analysis and multidimensional scaling. We demonstrate the utility of the new method in the clustering of line-patterns and shock graphs.

2 Representation

Consider the undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ with node-set $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$, edge-set $\mathcal{E} = \{e_1, e_2, \ldots, e_m\} \subset \mathcal{V} \times \mathcal{V}$ and weight function $\mathcal{W} : \mathcal{E} :\rightarrow [0, 1]$. The adjacency matrix $\mathbf{A}$ for the graph $\mathcal{G}$ is the $n \times n$ symmetric matrix with elements

$$A_{ab} = \begin{cases} 1 & \text{if } (v_a, v_b) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}$$

In other words, the matrix represents the edge structure of the graph. Clearly if the graph is undirected, the matrix $\mathbf{A}$ is symmetric. The corresponding weighted adjacency matrix is defined to be

$$A_{ab} = \begin{cases} \mathcal{W}(v_a, v_b) & \text{if } (v_a, v_b) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}$$

The Laplacian of the graph is given by $\mathbf{L} = \mathbf{D} - \mathbf{A}$. where $D_{a,b} = \sum_{b=1}^{n} A_{ab}$ is the diagonal node degree matrix whose elements are the number of edges which exit the node. The Laplacian is more suitable for spectral analysis than the adjacency matrix since it is positive semi-definite.

Here we would like to extend the Laplacian to weighted and attributed graphs. The attributes or weights may be unary (i.e. assigned to the nodes) or binary (i.e. assigned to the edges) in nature. To accommodate such structures, we need to augment the representation to accommodate measurement vectors on the nodes and edges. We doe this be encoding the weights and attributes using complex numbers.

A Hermitian matrix $\mathbf{H}$ is a square matrix with complex elements that remains unchanged under the operations of transposition and complex conjugation of the elements (denoted by the dagger operator $\dagger$), i.e. $\mathbf{H}^\dagger = \mathbf{H}$. Hermitian matrices can be viewed as the counterpart of the symmetric matrix for complex numbers. Each off-diagonal element is a complex number which has two components, and can therefore represent a 2-dimensional measurement vector. The on-diagonal elements are necessarily real quantities, so the node measurements are limited to a single quantity.

There are some constraints on how the measurements must be represented in order to produce a positive semi-definite Hermitian matrix. Let $\{x_1, x_2, \ldots, x_n\}$ be a set of unary measurements for the node-set $\mathcal{V}$. Further, let $\{y_{1,2}, y_{1,3}, \ldots, y_{n,n}\}$ be the set of binary measurements associated with the edges of the graph. Each edge then has a pair of measurements $(\mathcal{W}_{a,b}, y_{a,b})$ associated with it. There are a number of ways in which the complex number $H_{a,b}$ could represent this information, for example with the real part as $\mathcal{W}$ and the imaginary part as y. However, the off-diagonal elements of $\mathbf{H}$ are chosen to be $H_{a,b} = -\mathcal{W}_{a,b}e^{iy_{ab}}$. In other words, the connection weights are encoded by the magnitude of the complex number $H_{a,b}$ and the binary measurement by its phase. By using this encoding, the magnitude of the numbers is the same as in the original symmetric matrix.

The measurements must satisfy the conditions $-\pi \leq y_{a,b} < \pi$ and $y_{a,b} = -y_{b,a}$ to produce a Hermitian matrix. To ensure a positive definite matrix, we require $H_{aa} > -\sum_{b\neq a} |H_{ab}|$. This condition is satisfied if $H_{aa} = x_a + \sum_{b\neq a} \mathcal{W}_{a,b}$ where $x_a \geq 0$. When defined in this way the matrix is the weighted Laplacian for the graph.

For a Hermitian matrix there is an orthogonal complete basis set of eigenvectors and eigenvalues obeying the eigenvalue equation $\mathbf{He} = \lambda\mathbf{e}$. In the Hermitian case, the eigenvalues λ are real and the components of the eigenvectors $\mathbf{e}$ are complex. There is a potential ambiguity in the eigenvectors, in that any multiple of an eigenvector is also a solution, i.e. $\mathbf{H}\alpha\mathbf{e} = \lambda\alpha\mathbf{e}$. In the real case, we choose α such that $\mathbf{e}$ is of unit length. In the complex case, α itself may be complex, and needs to determined by two constraints. We set the vector length to $|\mathbf{e}_i| = 1$ and in addition we impose $\arg \sum_{i=1}^{n} \mathbf{e}_i = 0$, which specifies both real and imaginary parts.

When the eigenvectors are constructed in this way the spectral matrix is found by performing the eigenvector expansion $\mathbf{H} = \sum_{i=1}^{n} \lambda_i \mathbf{e}_i \mathbf{e}_i^{\dagger}$, where λ_i and $\mathbf{e}_i$ are the n eigenvectors and eigenvalues of the Hermitian matrix $\mathbf{H}$. We construct the complex spectral matrix for the graph $\mathcal{G}$ using the eigenvectors as columns, i.e. $\mathbf{\Phi} = \left(\sqrt{\lambda_1}\mathbf{e}_1, \sqrt{\lambda_2}\mathbf{e}_2, \ldots \sqrt{\lambda_n}\mathbf{e}_n\right)$. The matrix $\mathbf{\Phi}$ is a complete representation of the graph in the sense that we can use it to reconstruct the original Hermitian property matrix using the relation $\mathbf{H} = \mathbf{\Phi}\mathbf{\Phi}^{\dagger}$.

3 Node Permutations and Invariants

The topology of a graph is invariant under permutations of the node labels. However, the adjacency matrix, and hence the Laplacian matrix, is modified by the node order since the rows and columns are indexed by the node order. If we relabel the nodes, the Laplacian matrix undergoes a permutation of both rows and columns, and the corresponding spectral matrix undergoes a permutation of columns only. In previous work, we showed how the spectral matrix can be characterised in a permutation invariant way using sets of symmetric polynomials. If the vector $\boldsymbol{\phi}_i = (\phi_{1,i}, \phi_{2,i}, ..., \phi_{i,n})^T$ represents a column of Φ, i.e. a spectral mode, then the elementary symmetric polynomials for the mode are given by

$$
\begin{aligned}
S_1(\boldsymbol{\phi}_i) &= \sum_{j=1}^{n} \phi_{j,i} \\
S_2(\boldsymbol{\phi}_i) &= \sum_{j=1}^{n} \sum_{k=i+1}^{n} \phi_{j,i}\phi_{k,i} \\
&\vdots \\
S_r(\boldsymbol{\phi}_i) &= \sum_{j_1<j_2<\ldots<j_r} \phi_{j_1,i}\phi_{j_2,i}\ldots\phi_{j_r,i} \\
&\vdots \\
S_n(\boldsymbol{\phi}_i) &= \prod_{j=1}^{n} \phi_{j,i}
\end{aligned}
$$

Since the components of the eigenvectors are complex numbers, and therefore each $\boldsymbol{\phi}_i$ is complex. the symmetric polynomials must be evaluated with complex arithmetic and also evaluate to complex numbers. Each S_r therefore has both real and complex components. The real and complex components of the symmetric polynomials are interleaved stacked to form a feature vector $\mathbf{B}_k$ for the graph.

In order to accommodate graph of different sizes, we need to be able to compare representations of different sizes. This is achieved by expanding the representation. Consider two graphs of size m and n, $m < n$. If we add $n - m$ nodes with no connections to the first graph, we obtain two graphs of the same size. The edit cost in terms of edge insertions and deletions between these two graphs is identical to the original pair. The effect on the spectral representation is merely to add trailing zeros to each eigenvector and also additional zero eigenmodes. As a consequence, the first m elementary symmetric polynomials are unchanged, and the subsequent $n - m$ are zero. The new representation in the symmetric polynomials S can therefore be easily calculated.

4 Graph Embedding Methods

We explore three different methods for embedding the graph feature vectors in a pattern space, namely principal components analysis (PCA), multidimensional scaling (MDS) and linear discriminant analysis (LDA). In this paper we are concerned with the set of graphs $\mathcal{G}_1, \mathcal{G}_2, .., \mathcal{G}_k, ..., \mathcal{G}_N$. The kth graph is denoted by $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$ and the associated vector of symmetric polynomials is denoted by $\mathbf{B}_k$.

4.1 Principal Components Analysis

Principal component analysis commences by constructing the matrix $\mathbf{S} = [\mathbf{B}_1 |\mathbf{B}_2| \ldots |\mathbf{B}_k| \ldots |\mathbf{B}_N]$. with the graph feature vectors as columns. Next, we compute the covariance matrix for the elements of the feature vectors by taking the matrix product $\mathbf{C} = \mathbf{S}\mathbf{S}^T$.. We extract the principal components directions by performing the eigendecomposition $\mathbf{C} = \sum_{i=1}^{N} l_i \mathbf{u}_i \mathbf{u}_i^T$ on the covariance matrix $\mathbf{C}$, where the l_i are the eigenvalues and the $\mathbf{u}_i$ are the eigenvectors. We use the first s leading eigenvectors (2 or 3 in practice for visualisation purposes) to represent the graphs extracted from the images. The co-ordinate system of the eigenspace is spanned by the s orthogonal vectors $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, .., \mathbf{u}_s)$. The individual graphs are represented by the long vectors $\mathbf{B}_k, k = 1, 2, \ldots, N$ can be projected onto this eigenspace using the formula $\mathbf{x}_k = \mathbf{U}^T \mathbf{B}_k$.. Hence each graph G_k is represented by an s-component vector $\mathbf{x}_k$ in the eigenspace.

4.2 Multidimensional Scaling

Multidimensional scaling(MDS) is a procedure which allows data specified in terms of a matrix of pairwise distances to be embedded in a Euclidean space. Here we intend to use the method to embed the graphs extracted from different viewpoints in a low dimensional space. To commence we require pairwise distances between graphs. We do this by computing the $L2$ norms between the spectral pattern vectors for the graphs, weighted by the variance of each feature. For the graphs indexed i_1 and i_2, the distance is $d_{i_1,i_2} = (\mathbf{B}_{i_1} - \mathbf{B}_{i_2})^T \Sigma_D^{-1} (\mathbf{B}_{i_1} - \mathbf{B}_{i_2})$ where Σ_D is a diagonal matrix with the feature variances on the diagonal. The pairwise similarities d_{i_1,i_2} are used as the elements of an $N \times N$ dissimilarity matrix $\mathbf{S}$.

In this paper, we use the classical multidimensional scaling method to embed the graphs in a Euclidean space using the matrix of pairwise dissimilarities $\mathbf{S}$. The first step of MDS is to calculate a matrix $\mathbf{T}$ whose element with row r and column c is given by $T_{rc} = -\frac{1}{2}[d_{rc}^2 - \hat{d}_{r.}^2 - \hat{d}_{.c}^2 + \hat{d}_{..}^2]$, where $\hat{d}_{r.} = \frac{1}{N} \sum_{c=1}^{N} d_{rc}$ is the average dissimilarity value over the r^{th} row, $\hat{d}_{.c}$ is the similarly defined average value over the c^{th} column and $\hat{d}_{..} = \frac{1}{N^2} \sum_{r=1}^{N} \sum_{c=1}^{N} d_{r,c}$ is the average similarity value over all rows and columns of the similarity matrix $\mathbf{T}$.

We subject the matrix $\mathbf{T}$ to an eigenvector analysis to obtain a matrix of embedding co-ordinates $\mathbf{X}$. If the rank of $\mathbf{T}$ is $k, k \leq N$, then we will have k non-zero eigenvalues. We arrange these k non-zero eigenvalues in descending order, i.e. $l_1 \geq l_2 \geq \ldots \geq l_k > 0$. The corresponding ordered eigenvectors are denoted by $\mathbf{u}_i$ where l_i is the ith eigenvalue. The embedding co-ordinate system for the graphs obtained from different views is $\mathbf{X} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_s]$, where $\mathbf{f}_i = \sqrt{l_i} \mathbf{u}_i$ are the scaled eigenvectors. For the graph indexed i, the embedded vector of co-ordinates is $\mathbf{x}_i = (X_{i,1}, X_{i,2}, ..., X_{i,s})^T$..

4.3 Linear Discriminant Analysis

Linear discriminant analysis is closely connected to PCA. We commence by constructing separate data matrices $S_1, S_2, \ldots$ for each class. These may be used to compute the corresponding class covariance matrices $C_i = S_i S_i^T$. The average class covariance matrix $C = \frac{1}{n} \sum_{i=1}^{n} C_i$ is the found. This matrix is used as a sphering transform by computing the eigendecomposition $C = U \Lambda U^T$ and using the transform $S' = \Lambda^{-\frac{1}{2}} U^T S$. Standard PCA is then applied to the resulting data matrix S'. The purpose of this technique is to find a linear projection which describes the class differences rather than the overall variance of the data. Since we have a limited number of data samples, we use only the first twenty dimensions of the sphering transform.

5 Line Patterns

This first experimental example involves a database of the letters A-Z with rotations in 5 degree increments and hence 72 examples per character. Here we have used the method of Huet and Hancock [6] to compute pairwise attributes from the relative angles and lengths of the line-segments defining the characters. In the left-hand panel of Figure 1 shows the 11 closest retreivals of the letter "V" based on the Euclidean distance of the spectral feature vectors. The middlt panel of Figure 1 shows the result of performing MDS on the matrix of distances for the "A"s, "E"s and "Z"s in the database. The characters form well defined clusters.

To take this study on synthetic data one step further, we have performed a classification experiment. We have generated 100 graphs of 25 nodes each. For each graph the edge-structure is randomly generated. Associated with each edge is a weight randomly and uniformly drawn from the interval $[0, 1]$. We have investigated the effect of adding random noise to the edge-weights. The weight noise is drawn from a Gaussian distribution of zero mean and known standard deviation.

We have investigated the effect of this noise on three different vector representations of the attributed graphs. The first of these is a vector with the first four polynomial features as components. The second is a vector whose components are the bin-contents of the normalised edge-weight histogram. Here the edge weights are allocated to 8 uniformly spaced bins. The final vector has the leading 4 eigenvalues of the Laplacian matrix as components.

We have computed the distances between the feature vectors for the uncorrupted and noise corrupted graphs. To compute a classification error rate, we have recorded the fraction of times that the uncorrupted graphs do not have the smallest distance to the corresponding noise corrupted graph. The right-hand panel of Figure 1 shows the error-rate as a function of the edge-weight noise standard deviation. The main feature to note from this plot are that the lowest error rate is returned by the polynomial features and the highest error rate results from the use of the edge-weight histogram.

6 Shock Graphs

The second example of the use of the complex property matrix representation is furnished by shock trees, which are an abstraction of the skeleton structure of 2D or 3D shape silhouettes. The skeleton is the locus of the centre of the bitangent circle to the object boundary, and is hence related to the medial axis transform (which seeks points which are equidistant from pairs of points on the

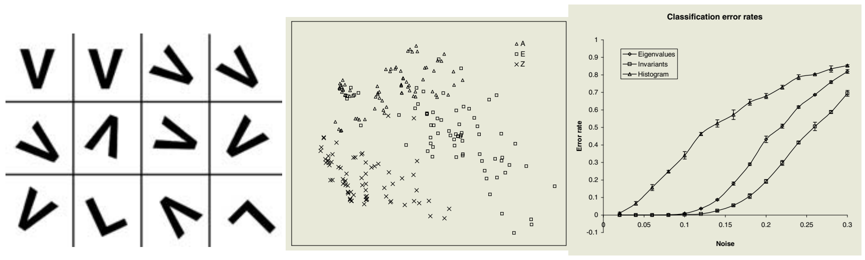


Fig. 1. Retreival results (left), MDS (middle) and classification error rate (right) for character data.

shape boundary). In practice searching for the skeleton corresponds to finding ridges in the distance transform of the object boundary. The medial axis has a natural tree structure which makes it suitable for the tree representation of shapes.

The edges of the tree represent the existence of a connecting skeletal branch between pairs of junctions. The nodes of the tree are characterised using the radius $r(a)$ of the smallest bitangent circle from the junction to the boundary. Hence, for the node a, $x_a = r(a)$. The edges are characterised by two measurements. For the edge (a, b) the first of these, $y_{a,b}$ is the angle between the nodes a and b, i.e. $y_{a,b} = \theta(a, b)$. Since most skeleton branches are relatively straight, this is an approximation to the angle of the corresponding skeletal branch. Furthermore, since $-\pi \leq \theta(a, b) < \pi$ and $\theta(a, b) = -\theta(b, a)$, the measurement is already suitable for use in the Hermitian Laplacian matrix.

In order to compute edge weights, we note that the importance of a section of the skeleton may be determined by the rate of change of boundary length with skeleton length [9], which we denote by dl/ds. This quantity is related to the rate of change of the bitangent circle radius along the skeleton, i.e. dr/ds, by the formula $\frac{dl}{ds} = \sqrt{1 - \left(\frac{dr}{ds}\right)^2}$. The edge weight $\mathcal{W}_{a,b}$ is given by the average value of dl/ds along the relevant skeletal branch.

Our experiments are performed using a database of 42 binary shapes. Each binary shape is extracted from a 2D view of a 3D object. There are 3 classes in the database, and for each object there are a number of views acquired from different viewing directions and a number of different examples of the class . We extract the skeleton from each binary shape and attribute the resulting tree in the manner outlined in Section 4.

We commence by showing some results for the three shapes shown in Figure 1. The objects studied are a hand, some puppies and some cars. The dog and car shapes consist of a number of different objects and different views of each object. The hand category contains different hand configurations. We apply the three embedding strategies outlined in Section 5 to the vectors of permutation invariants. We commence in the left-hand panel of Figure 2 by showing the result of applying MDS procedure to the three shape categories. The 'hand' shapes form a compact cluster in the MDS space. There are other local clusters consisting of three or four members of the remaining two classes. This reflects the fact that while the hand shapes have very similar shock graphs, the remaining two categories have rather variable shock graphs because of the different objects.

The middle panel of Figure 2 shows the result of using PCA. Here the distributions of shapes are much less compact. While a distinct cluster of hand shapes still occurs, they are generally more dispersed over the feature space. There are some distinct clusters of the car shape, but the distributions overlap more in the PCA projection when compared to the MDS space.

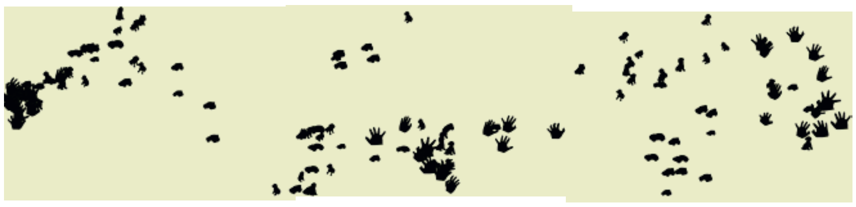


Fig. 2. MDS (left), PCA (centre) and LDA (right) applied to the shock graphs.

In the LDA projection, we introduce information about the class designations of the shape trees. The right-hand panel of Figure 2 shows the result of the LDA procedure on the dataset. The result is a much better class separation than the PCA or MDS methods.

Based on the analysis of the different embedding methods, it appears that LDA gives the best results. One of the motivations for the work presented here was the potential ambiguities that are encountered when using the spectral features of trees. To demonstrate the effect of using attributed trees rather than simply weighting the edges, we have compared the LDA projections using both types of data. Figure 3 illustrates the result of this comparison. The right-hand plot shows the result obtained using the symmetric polynomials from the eigenvectors of the Laplacian matrix $L = D - W$, associated with the edge weight matrix. The left-hand plot shows the result of using the using the Hermitian property matrix. The Hermitian property matrix for the attributed trees clearly produces a better class separation than the Laplacian matrix for the weighted trees.

7 Conclusions

In this paper we have described a complex property matrix that can be used to encode the structure of attributed graphs. We have shown how to construct permutation invariants from the complex components of the eigenvectors of the Hermitian property matrix. The invariants are used as the components of a real-valued pattern vector, which can be embedded in a pattern space, suitable for clustering the graphs. There are clearly a number of ways in which the work presented in this paper can be developed. For instance, since the representation

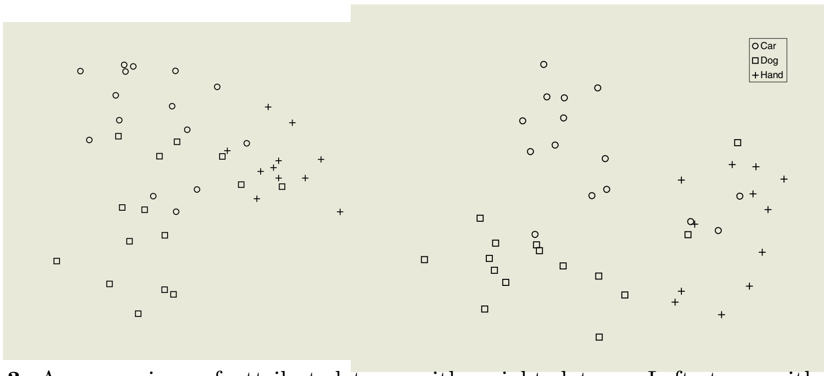


Fig. 3. A comparison of attributed trees with weighted trees. Left: trees with edge weights based on boundary lengths. Right: Attributed trees with additional edge angle information.

based on the symmetric polynomials is complete, they may provide the means by which a generative model of variations in graph structure can be developed. This model could be learned in the space spanned by the permutation invariants, and the mean graph and its modes of variation reconstructed by inverting the system of equations associated with the symmetric polynomials.

References

1. F.R.K. Chung. *Spectral Graph Theory.* American Mathmatical Society Ed., CBMS series 92, 1997.
2. S. Umeyama. An eigen decomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:695–703, 1988.
3. B. Luo, R. C. Wilson and E. R. Hancock, "Spectral Embedding of Graphs", *Pattern Recognition* 36(10), pp. 2213–2230, 2003
4. B.Mohar,"Laplace Eigenvalues of Graphs - A survey", *Discrete Mathematics*, **109**, pp. 171-183, 1992.
5. N. Biggs, "Algebraic Graph Theory", CUP.
6. B. Huet and E. R Hancock, "Line Pattern Retrieval Using Relational Histograms", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**, pp. 1363-1370, 1999.
7. K. Siddiqi, A. Shokoufandeh, S. J. Dickinson and S. W. Zucker "Shock Graphs and Shape Matching", International Journal of Computer Vision, 35(1), pp13–32, 1999.
8. J. Shi and J. Malik, "Normalized Cuts and Image Segmentation" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8) pp888–905, 2000
9. A. Torsello and E. R. Hancock, "A skeletal measure of 2D shape similarity", *Lecture notes on Computer Science*, 2059, pp594–605, 2001

Decision Tree Structures for Graph Database Filtering

Christophe Irniger and Horst Bunke

Department of Computer Science, University of Bern
Neubrückstrasse 10, CH-3012 Bern, Switzerland
{bunke,irniger}@iam.unibe.ch

Abstract. In structural pattern recognition it is often required to match an unknown sample against a database of candidate patterns in order to find the most similar prototype. If the patterns are represented using graphs, the sample's graph is matched against a database of model graphs and the pattern recognition problem is turned into a graph matching problem. Graph matching is a powerful yet computationally expensive procedure. If the unknown sample is matched against a whole database of prototypes, the size of the database is introduced as an additional factor into the overall complexity of the matching process. To reduce the influence of that factor an approach based on machine learning techniques is proposed in this paper. The graphs are represented using feature vectors. Based on these vectors a decision tree is built to index the database. The decision tree allows at runtime to eliminate a number of graphs from the database as possible matching candidates. Experimental results are reported demonstrating the efficiency of the proposed filtering procedure. The work presented in this paper extends previous studies from the case of graph-isomorphism to the problem of subgraph-isomorphism.

1 Introduction

Graphs play an important role in structural pattern recognition. Besides comparing two given patterns, it is often required to match an input pattern against a database of known patterns or sub-patterns. If graphs are used to represent structural data, the task of matching patterns is turned into a problem of graph matching. Graph matching is used in a variety of applications, for example document processing [1], image analysis [2, 3], biometric identification [4] and video analysis [5]. Despite being a computationally expensive approach, graph matching is attractive for pattern recognition problems since graphs are a universal representation formalism. If databases of model graphs are used, an additional factor proportional to the size of the database is introduced in the overall complexity of the matching process. A variety of mechanisms have been proposed to reduce the complexity of graph matching when large databases are involved [6–10]. In this paper we propose an approach based on machine learning techniques.

The presented approach proposes to characterize graphs by features which can efficiently be extracted (e.g. the number of nodes or edges in a graph or

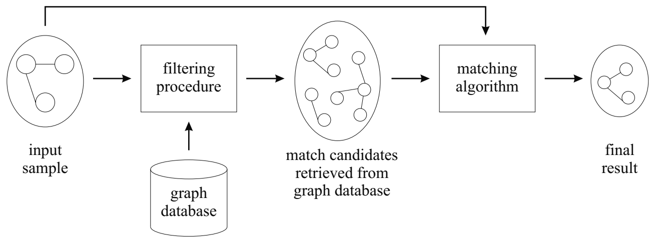


Fig. 1. Illustration of the filtering procedure in the database matching process.

the number of nodes or edges with a certain label). These features are used to perform a filtering on the database. A filtering procedure is a method which performs a quick and inexpensive reduction of the initial graph database size with respect to a given input graph. The aim of database filtering is to reduce the number of graph candidates in the database that need to undergo an expensive, full fledged graph matching process. A graphical illustration is shown in Figure 1.

The presented work extends previous studies on graph matching performance and graph database filtering (see [11, 12]). In contrast with the work reported in [11], which was restricted to the case of graph isomorphism, the present paper deals with the problem of subgraph-isomorphism. Database filtering in conjunction with subgraph-isomorphism search was also studied in [13]. However, the decision trees used in [13] were identical to the decision trees used in [11, 12] for graph-isomorphism, and the case of subgraph-isomorphism was dealt with by means of an extended decision tree traversal procedure. By contrast, a generalized decision tree induction procedure is proposed in the present paper. In the next section, we briefly introduce terminology and graph features used in this study. Then, we show how the concept of graph representation by means of vectors can be combined with the decision tree filtering approach. Experimental results will be presented in Section 4, and conclusions drawn in Section 5.

2 Terminology and Graph Features

In this work, structural data or patterns are represented as graphs. A graph is defined as a four-tuple $g = (V, E, \alpha, \beta)$, where V denotes a finite set of nodes, $E \subseteq V \times V$ is a finite set of edges, $\alpha : V \to L_V$ is a node labelling function, and $\beta : E \to L_E$ is an edge labelling function. L_V and L_E are finite or infinite sets of node and edge labels, respectively. In this work only directed graphs are considered. However, the same ideas can be applied to undirected graphs as well. A subgraph $g_s = (V_s, E_s, \alpha_s, \beta_s)$ of a graph g is a subset of its nodes and edges, such that $V_s \subseteq V$, $E_s = E \cap (V_s \times V_s)$, $\alpha_s(v) = \alpha(v)$ and $\beta_s(e) = \beta(e)$. Two graphs g and g' are isomorphic to each other if there exists a bijective mapping u from the nodes of g to the nodes of g', such that the structure of the edges as well as all node and edge labels are preserved under u. Similarly, an isomorphism between a graph g and a subgraph g'_s of a graph g' is called a subgraph-isomorphism from g to g'.

In this study, feature vectors are used to represent graphs. Since the vectors are used for the purpose of database filtering, the features must meet the following two requirements. First, their extraction from a sample graph must be fast (compared to performing a graph matching) and second, they should possess a high degree of saliency (they should be able to differentiate between as many graphs as possible). Fast extraction of the features from sample graphs directly influences the filtering performance whereas high saliency ensures a precise candidate retrieval (which means that fewer graphs will be subject to the full-fledged graph matching procedure after filtering). In this work, the following features are used:

1. the total number of vertices in the graph
2. the total number of vertices with a given label in the graph
3. the total number of incoming (outgoing) edges per vertex label in the graph
4. the total number of vertices with a given number of incoming (outgoing) edges in the graph
5. the total number of vertices with a given label and a given number of incoming (outgoing) edges in the graph

Previous studies have shown that the selection of these features is very effective for database filtering. In [11] it can be seen that when searching for graph-isomorphism candidates, these features are completely sufficient to distinguish between all graphs in the database. Hence in this study, the same features have been used.

In case of graph-isomorphism [11], a necessary condition for two graphs g and g' being isomorphic is that they have identical feature values. Hence, given the feature vector $f = (f_1, \ldots, f_m)$ extracted from g and $f' = (f'_1, \ldots, f'_m)$ extracted from g', g and g' can immediately be ruled out to be isomorphic if a feature j is discovered such that $f_j \neq f'_j$. However, for subgraph-isomorphism, which is considered in this paper, the relation is not that simple. Looking at the list of features presented above, we can see that there are two basic types of features:

- features not containing edge information (features 1 and 2)
- features containing edge information (features 3 to 5)

Extending the feature vector comparison from graph-isomorphism to subgraph-isomorphism is straightforward for features not containing edge information. In order for a graph g' possibly being isomorphic to a subgraph of a graph g, the value of such a feature f'_j must be smaller than, or equal to, the feature value f_j in g. Therefore, the relation $f'_j = f_j$ for graph-isomorphism needs to be replaced by $f'_j \leq f_j$ for subgraph-isomorphism. (Note that this is only a necessary, but not a sufficient condition for g' being a subgraph of g.) The difficulty for features 3 to 5 (features containing edge information) is that in subgraph-isomorphism, nodes of lower vertex degrees in the subgraph may be mapped onto nodes with a higher degree in the supergraph. (This is contrary to graph-isomorphism, where nodes can only be mapped onto nodes of equal degree.) A simple example would be a star-graph as the (original) graph and the same graph with its center node removed as the subgraph. In the original graph, all nodes except the center node are of degree 1 (the center node is of degree $n - 1$, where n is the number of nodes in the graph). In the subgraph, since the center node has been removed, the surrounding nodes are now of degree 0. Hence, it can happen that a subgraph has a larger number of nodes with a certain degree than its supergraph. In order to overcome this problem, the features used in our approach must be able to associate the nodes of degree 0 in the subgraph with the nodes of degree 1 in the original graph. In order to properly compare these feature values, one must not only consider the value for the current node's degree but also include the values for the nodes of a lower degree. This can be done by summing the feature vector's values according to node degree order. Applying this technique allows us to use the same features for both graph-isomorphism and subgraph-isomorphism (but the way the features are handled by the subgraph isomorphism filtering procedure is different from filtering in case of graph isomorphism).

3 Decision Trees

In this section, we will show how the concept of feature-vector comparison for graph-/subgraph-isomorphism can be used in combination with decision trees. The general idea is that in a preprocessing step a decision tree is built to classify graphs according to their feature vectors. Depending on the given matching task, two types of decision trees can be induced. In order to filter the database for graph-isomorphism candidates, a graph-isomorphism tree is used and analogously for subgraph-isomorphism filtering, a subgraph-isomorphism tree is induced. The tree induction algorithm ensures that out of the entire feature set only the most salient features are used for filtering. At runtime, all features needed are extracted from the sample graph and then, the decision tree is traversed to retrieve suitable graph candidates from the database. This approach minimizes the number of features to be tested in order to eliminate the maximum number of non-candidates from the database. The decision tree procedure itself is analogous to standard decision tree methods (see [14], for example) with the difference that, whereas ordinary decision tree methods try to generalize from a training set of objects, the approach presented here tries to 'overfit' the data in the sense that all leaf nodes in the tree include just a single graph. In general, the smaller the number of graphs in a leaf node is, the smaller is the number of full-fledged graph matchings that need to be computed.

In this section a brief explanation will be given on how to induce decision trees useful for subgraph-isomorphism filtering. For the purpose of completeness we also describe how to induce trees for graph-isomorphism. Then, we will focus on the problem of using the trees to create two filter types, one for graph-isomorphism and one for subgraph-isomorphism.

3.1 Graph-Isomorphism Decision Tree Induction

As mentioned in Section 2, a necessary condition for two graphs g and g' being isomorphic is that they have identical feature vectors. The decision tree induction algorithm, which is derived from [14], classifies the graphs in the database based on their feature values. As an initializing step, the tree's root node is constructed and it is assigned the entire graph set of the database. Then, each available feature is tested and its suitability is evaluated according to a given split criterion (see [14]). Amongst all features, the best one is chosen and the current root's graph set is split into subsets according to the best feature. For each feature value, a son node is created and the node is assigned the subset of graphs that correspond to that feature value. The induction procedure is recursively continued with the son nodes until one of the following termination conditions holds: a) the graph set in a node contains only one graph; b) no features are left to divide a subset; c) the features left cannot distinguish the remaining graphs in the set. Cases b) and c) correspond to the situation where, later in the decision traversal phase, multiple graphs are returned by the filtering procedure, while case a) reflects the 'ideal' situation where only one candidate graph remains to undergo the full-fledged graph matching procedure.

3.2 Subgraph-Isomorphism Decision Tree Induction

Before extending the approach presented above to the problem of filtering for subgraph-isomorphism candidates, one must first define the search task to be considered. There are two possible search scenarios:

– supergraph-search: the input sample is considered to be isomorphic to subgraphs of the graphs in the database.
– subgraph-search: the database contains graphs possibly isomorphic to a subgraph of the input sample.

In the following explanations, we will focus on the second scenario where the input sample is a supergraph and the database graphs are subgraphs (subgraph-search). Note that the adaption of this traversal to the task of supergraph-search is straightforward.

The decision tree structure as presented in the graph-isomorphism case needs two major adaptations before it can be used for subgraph-filtering purposes. The first adaption concerns the assignment of graph-subsets to son nodes. In the isomorphism case, the father node's graph set is split into disjoint subsets according to the best feature's values. For subgraph-isomorphism trees however, these subsets are not disjoint anymore. Consider the case where a feature occurs n times in the input sample. In that case, all graphs in the database where the same feature occurs $n' < n$ times are possible subgraph-isomorphism candidates and need to be assigned to the son-node representing feature value n. As an additional consequence, the son-node with the maximal feature value at any level in the tree is assigned the entire graph set of its father (hence the depth of the decision tree is only limited by the number of features evaluated). Depending on the graph type, the number of possible features is in general quite large. In case a son node containing its father's graph set is unlikely to be reached during traversal, it should be induced after other nodes that are more likely to be

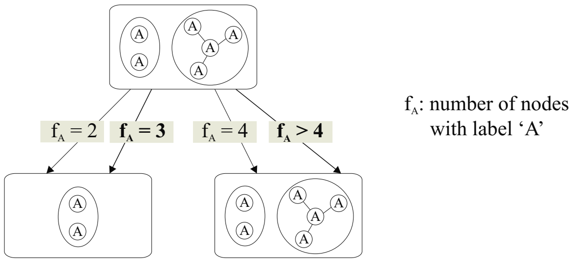


Fig. 2. Example of a decision tree split.

reached. Hence, while constructing the tree, the nodes ready for expansion must be ordered so that these candidates are not (or very late in the process) expanded. An intuitively plausible ordering is given by the probability with which a node is likely to be reached during tree traversal. This probability is given by the number of candidates only appearing in the node under consideration compared to the number of candidates in the father node.

The second adaption is caused by the fact that feature values not occurring in the graph database may occur in the input sample and must therefore be considered during tree construction. This can be done by introducing additional edges from the father to the appropriate son node (the node representing the next smaller feature value) for each non-occurring value of the considered feature.

Figure 2 illustrates the described modifications, namely overlapping graph sets in the son nodes as well as feature values not occurring in database graphs. In the father node, there are two graphs with two and four nodes with label 'A', respectively. Hence, the decision tree consists of two son nodes (and their corresponding edges), one for each feature value $f_A = 2$ and $f_A = 4$. Furthermore, since a sample graph may contain 3 nodes with label 'A', an additional edge with label $f_A = 3$ must be introduced, also pointing to the node for graphs with $f_A = 2$. Naturally, all samples where $f_A > 4$ must be directed to the node where $f_A = 4$ (branch to the very right) and all samples where $f_A < 2$ are not possible supergraphs to the graphs in the database (therefore no branch is provided in the tree for this case). Consider the case where, at runtime, the decision tree is traversed for a sample graph with $f_A = 5$. While traversing the tree, the leaf with $f_A > 4$ is reached. Hence, the input sample is a possible supergraph to both graphs in the illustration which makes sense since the sample consists of at least 5 nodes with label 'A'.

3.3 Decision Tree Traversal

Decision trees induced as described above can be used to retrieve possible graph-isomorphism or subgraph-isomorphism candidates of a given sample graph in the following way. First, the same features that were extracted from the database graphs and used to induce the decision tree are extracted from the input graph. Then, the traversal algorithm follows the tree branch whose feature values are equal to the values extracted from the sample graph. There are only two possible outcomes of the decision tree traversal procedure. The first outcome is that a leaf node is reached. In this case, the graphs associated with the leaf node are possible matches to the input graph. Each of these graphs must then be tested against the input graph for (sub-)graph-isomorphism using a conventional algorithm as described in [15–17], for example. The second outcome is that no leaf node is reached in which cases there are no graphs in the database that can be (sub-)graph-isomorphic to the input graph.

4 Experimental Results

To demonstrate the efficiency of the approach, we tested it on several different types of graphs described below (see [18]).

– Random Graphs: The random graph database consists of connected graphs with different node and edge label alphabet size.
– Bounded Valence Graphs (regular, irregular): This databases consists of graphs with a fixed valence per node (regular bounded valence graphs) or a fixed valence over the entire graph (irregular bounded valence graphs).
– Meshes / Hyper-Cuboids (regular, irregular): The database consists of (Hyper-) Cuboids of varying dimension $n = 2$ (meshes), $n = 3$ (cuboids), $n = 4, 5$ (hypercuboids).

For each graph type and parameter setting a database of 1,000 graphs was created. During creation of the database, it was made sure that each graph was isomorphic only to itself.

The primary objective of the proposed filtering method is to reduce the number of candidates which have to undergo a full-fledged isomorphism or subgraph-isomorphism matching. Hence, the quality of the approach can be expressed by measuring the number of graphs that are assigned to the leaf nodes in the decision tree. (We will also refer to this number as the *cluster size* of the decision tree.) The cluster size determines the number of full-fledged matchings that need to be executed. The other important measure is the average number of nodes visited during traversal. This value directly affects traversal and therefore filter time.

Decision trees suitable for graph-isomorphism retrieval can be fully induced with no problems. Decision trees suitable for subgraph-isomorphism on the other hand can grow, due to their special structure, quite large and therefore need to be limited in size. To control the subgraph-isomorphism decision tree growth, the trees were limited in size to the same number of nodes as the corresponding graph-isomorphism trees (from hundreds up to several thousand nodes).

In order to measure the average cluster size, graphs were randomly picked from the database and were then used as an input sample. Before extracting the features, the sample graph's size was increased and the additional nodes were assigned labels not occurring in the database graphs (this was to ensure that no additional subgraph-isomorphisms were introduced). Then the feature vector

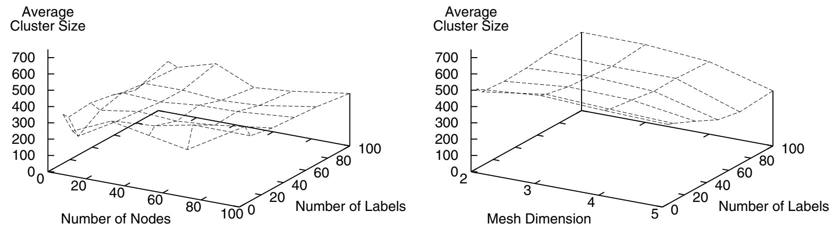


Fig. 3. Cluster size for random and mesh graphs.

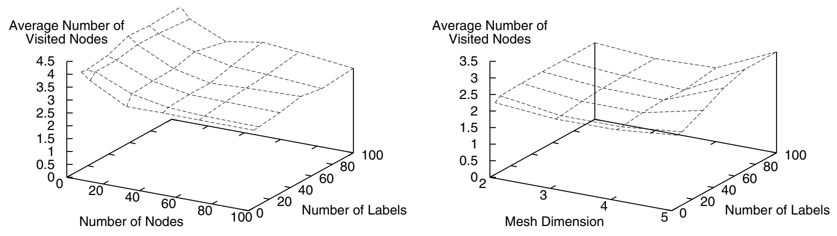


Fig. 4. Average number of visited nodes for random and mesh graphs.

was extracted and the decision tree traversed. In order to get an average value for the cluster size, the procedure was repeated 1,000 times for each database.

Figure 3 shows the cluster size for random and mesh graphs, respectively. The results for bounded valence graphs are similar to the values obtained for random graphs and therefore not depicted. In [11] it has been shown that for graph-isomorphism, the cluster size can be reduced down to only one graph remaining after filtering. For subgraph-isomorphism, naturally, the reduction factor is not that high. However, as is shown in Figure 3 the approach on average reduces the database size to about 300 graphs for random graphs. This means that the initial database size is reduced by about 70%. Due to the much more regular structure of mesh graphs, the filtering effect is not quite that high. However, still approximately 500 graphs can be eliminated from the database which is equal to a reduction factor of 50%.

The second parameter influencing the filter's performance is the average number of nodes visited during tree traversal. Figure 4 shows that for both graph types, random as well as mesh graphs, less than 4 nodes are visited during traversal (again the results obtained for bounded valence graphs are similar to the results of random graphs.) It can be concluded from Figure 4 that the size of the database can be very effectively reduced through a small number of tests. The computation time of these tests is in fact negligible.

5 Conclusions

In this paper an approach to graph database filtering using machine learning techniques has been presented. The method is based on a decision tree data structure. It reduces the number of candidate graphs in a database to be tested for graph- and subgraph-isomorphism. Depending on the considered graph matching task (graph isomorphism or subgraph isomorphism), special types of decision trees are built that classify the graphs using given features. At runtime, possible matching candidates are retrieved from the database by traversing the decision tree. This paper presented two decision tree induction methods. One method is able to induce trees suitable for graph-isomorphism candidate retrieval, while the other is designed to induce trees suitable for subgraph-isomorphism candidate retrieval.

Considering the complexity of the proposed method we notice that the method is divided into two stages: a) tree induction and b) tree traversal. Tree induction is considered to be an off-line step, hence its complexity is not of primary interest. Tree traversal, however, is the main objective concerning complexity. It is only dependent upon the number of visited nodes during tree traversal. The overall complexity of filtering-based graph matching is determined by the decision tree traversal complexity, the number of final matchings to be performed (this number is identical to the cluster size associated with the leaf nodes in the decision tree) and the complexity of computing the final matchings, which is depending upon graph type and graph matching algorithm. In this paper, a number of experiments investigating cluster size and number of visited nodes have been conducted. The results indicate that the cluster size can be significantly reduced by few tests, resulting in a small number of nodes to be visited as well as a small number of final matchings to be performed.

The main contribution of the present paper is an extension of the method proposed in [11] from graph-isomorphism to the problem of subgraph-isomorphism by generalization of the decision tree's structure. Future work is planned extending the approach to error tolerant, edit distance based graph matching methods and determining its efficiency on real world data.

Acknowledgment

This research was supported by the Swiss National Science Foundation (Nr. 2100-066700). The authors thank the Foundation for the support.

References

1. Llados, J., Marti, E., Villanueva, J.: Symbol recognition by error-tolerant subgraph matching between region adjacency graphs. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. Volume 23 – 10. (2001) 1137 – 1143
2. Torsella, A., Hancock, E.: Learning stuctural variations in shock trees. In: Proc. of the Joint IAPR International Workshops SSPR and SPR. (2002) 113 – 122
3. Luo, B., Hancock, E.: Structural graph matching using the em algorithm and singular value decomposition. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. Volume 23 – 10. (2001) 1120 – 1136
4. Lumini, A., Maio, D., Maltoni, D.: Inexact graph matching for fingerprint classification. Machine Graphics and Vision, Special Issue on Graph Transformations in Pattern Generation and CAD **8** (1999) 231 – 248

5. Chen, H., Lin, H., Liu, T.: Multi-object tracking using dynamical graph matching. In: Proc. of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. (2001) 210 – 217
6. Guigno, R., Sasha, D.: Graphgrep: A fast and universal method for querying graphs. In: Proceeding of the International Conference in Pattern Recognition (ICPR). (2002) 112 – 115
7. Wang, J.l., Sasha, D., Guigno, R.: Algorithmics and applications of tree and graph searching. In: Proceeding of the ACM Symposium on Principles of Database Systems (PODS). (2002)
8. Messmer, B., Bunke, H.: A new algorithm for error–tolerant subgraph isomorphism detection. In: IEEE Trans. Pattern Analysis and Machine Intelligence. Volume 20. (1998) 493 – 505
9. Shapiro, L., Haralick, R.: Organization of relational models for scene analysis. In: IEEE Trans. Pattern Analysis and Machine Intelligence. Volume 3. (1982) 595 – 602
10. Sengupta, K., Boyer, K.: Organizing large structural modelbases. In: IEEE Trans. Pattern Analysis and Machine Intelligence. Volume 17. (1995)
11. Irniger, C., Bunke, H.: Graph matching: Filtering large databases of graphs using decision trees. In Jolion, J.M., Kropatsch, W., Vento, M., eds.: Graph-based Representations in Pattern Recognition, Cuen (2001) 239 – 249
12. Irniger, C., Bunke, H.: Theoretical analysis and experimental comparison of graph matching algorithms for database filtering. In Hancock, E., Vento, M., eds.: Graph-based Representations in Pattern Recognition, Springer Verlag Berlin Heidelberg (2003) 118 – 129
13. Irniger, C., Bunke, H.: Graph database filtering using decision trees. In: Proceedings of the International Conference in Pattern Recognition (ICPR). (2004)
14. Quinlan, J.: C4.5: Programs for Machine Learning. Document Analysis Systems II. Morgan Kaufmann Publishers (1993)
15. Ullmann, J.: An algorithm for subgraph isomorphism. In: JACM. Volume 23. (1976) 31 – 42
16. Cordella, L., Foggia, P., Sansone, C., Vento, M.: An improved algorithm for matching large graphs. In Jolion, J.M., Kropatsch, W., Vento, M., eds.: Graph-based Representations in Pattern Recognition, Cuen (2001) 149 – 159
17. McKay, B.: Practical graph isomorphism. In: Congressus Numerantium. Volume 30. (1981) 45 – 87
18. Foggia, P., Sansone, C., Vento, M.: A database of graphs for isomorphism and subgraph isomorphism. In Jolion, J.M., Kropatsch, W., Vento, M., eds.: Graph-based Representations in Pattern Recognition, Cuen (2001) 176 – 188

A Significant Improvement of Softassign with Diffusion Kernels

Miguel Angel Lozano and Francisco Escolano

Robot Vision Group
Departamento de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante, Spain
{malozano,sco}.dccia.ua.es
http://rvg.ua.es

Abstract. In this paper we propose a simple way of significantly improving the performance of the Softassign graph-matching algorithm of Gold and Rangarajan. Exploiting recent theoretical results in spectral graph theory we use diffusion kernels to transform a matching problem between unweighted graphs into a matching between weighted ones in which the weights rely on the entropies of the probability distributions associated to the vertices after kernel computation. In our experiments, we report that weighting the original quadratic cost function results in a notable improvement of the matching performance, even in medium and high noise conditions.

1 Introduction

Energy-minimization approaches to graph matching [4][5][8] rely on transforming the discrete search space into a continuous one and then exploiting optimization techniques to find a, typically approximate, solution. One of the first algorithms, Softassign, the well-known graduated assignment method introduced by Gold and Rangarajan [4], optimizes a quadratic cost function through a low-order computational complexity process which updates the assignment variables encoding the matching proposals. However, it has been reported that the performance of the algorithm decays significantly at mid and high levels of structural corruption, and also that such a decay can be attenuated by optimizing an alternative non-quadratic energy function [5]. In this paper we report comparable results by weighting the quadratic cost function properly. This is due to the fact that we transform the original matching problem between two non-attributed graphs into a matching problem between attributed ones and then these attributes are used to weight the original cost function. The practical effect of this weighting is that it yields a good characterization of the local structure, which in turn helps to choose the proper attractor in a context of high ambiguity.

We address the key point of extracting good attributes for the nodes of the non-attributed graphs by exploiting recent theoretical results in spectral graph theory [1]: the definition of diffusion kernels on graphs [6] and their generalization to other families of kernels [13]. These latter works have transferred to the discrete domain of graphs the concept of a *kernel*, originally defined in the vector domain (see [3] for a survey on kernels for structures like strings, trees and graphs). Kernels, are key concepts in the context of statistical learning theory[2][12][7] which capture the structure of a domain by defining a similarity measure between two input elements in the domain. Such a similarity measure relies on the inner product of the results of mapping both inputs to a, usually higher dimensional, Hilbert space. Due to the so-called *kernel trick* such a mapping is implicitly defined once the kernel is specified, and the benefit of such a transformation consists on transforming non-linear relations between the inputs in the original domain into linear relations after the mapping. For instance, in the context of support-vector machines (in general we can talk about kernel machines), the task of classifying two non-linearly separable inputs is accomplished by using a suitable kernel to map them to another space in which these inputs are linearly separable (it works in the well-known two-spirals example).

When applied to graphs, kernels provide a similarity measure between the vertices of the same graph. In the case of diffusion kernels, such a similarity can be seen as the sum of probabilities of all paths connecting such vertices, and it is computed from the matrix exponentiation of the Laplacian of the adjacency matrix (section 2). As the Laplacian encodes information about the local structure of the graph, the global structure emerges in the kernel. However, we do not use directly the probabilities of connecting paths because they may change very easily when the graph is edited or corrupted, and, consequently, they are not useful for finding corresponding vertices. What we do is to is to compute a characteristic measure of the distribution of probabilities associated to paths emanating from a given vertex, the entropy of such a distribution, and use it as attribute for that vertex. The entropy of the probabilities associated to connecting paths is more stable and allows us to find correct matches (section 3). In section 4 we present the *kernelized* version of the quadratic cost function and its implications in the Softassign process. Our results are showed in section 5 and in 6 we present our conclusions and future work.

2 Diffusion Kernels on Graphs

Given a undirected and unweighted graph $G = (V, E)$ with vertex-set V of size m, and edge-set $E = \{(i,j)|(i,j) \in V \times V, i \neq j\}$, its respective adjacency matrix is defined as usual:

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

and the diagonal degree matrix is defined by

$$D_{ij} = \begin{cases} \sum_{j=1}^{n} A_{ij} & \text{if } i = j \\ 0 & \text{otherwise} . \end{cases}$$

Then, the Laplacian of G is defined as $L = D - A$, that is,

$$L_{ij} = \begin{cases} -1 & \text{if } (i,j) \in E \\ D_{ii} & \text{if } i = j \\ 0 & \text{otherwise .} \end{cases}$$

Following [6] the associated diffusion kernel K is the result of the matrix exponentation

$$K = e^{-\beta L} = \lim_{n \to \infty} \left(1 - \frac{\beta L}{n}\right)^n , \tag{1}$$

and after solving the latter limit we obtain

$$e^{-\beta L} = I_m + L + \frac{1}{2!}L^2 + \frac{1}{3!}L^3 + \ldots , \tag{2}$$

where I_m is the $m \times m$ identity matrix. Moreover, $e^{-\beta L}$ is the solution of the heat equation [1]

$$\frac{d}{d\beta}K_\beta = -LK_\beta. \tag{3}$$

As L is symmetric, the solution $K = e^{-\beta L}$, the Gram matrix, satisfies the positive semi-definiteness condition for kernels. Although in this paper we will focus on diffusion kernels, this framework is generalized in [13] where a family of graph kernels is proposed in the context of regularization.

3 Diffusion Kernels and Node Entropy

On behalf of the so-called *kernel trick* the $m \times m$ matrix K defines a real-valued function between pairs of vertices, and K_{ij} can be interpreted as the inner product of the mappings of both vertices to a Hilbert space [12]. This means that such a inner product encodes the similarity between pairs of vertices in a possibly high-dimensional space. But, from the point of view of discrete structures what is interesting of such similarity is that as L encodes the local structure of V in G, the global structure emerges in K.

More precisely, and due to the fact that K is the solution of the heat equation, the diffusion kernel K is the version for discrete spaces of the Gaussian kernel for $\mathbb{R}^m$ with variance $\sigma^2 = 2\beta$, that is, the value of K_{ij} decays exponentially with the *distance* between i and j. But, how to apply this idea to a graph? From the point of view of random fields, the diffusion kernel K relies on the covariance matrix of a stochastic process in which each vertex has attached a random variable of zero mean an variance σ^2 and each variable sends a small fraction of its value to its neighbors. In this regard, K_{ij} can be interpreted as the amount of substance accumulated at vertex j after a given amount of time after injecting the substance at i and let it diffuse through the edges of the graph. The more *distant* are i and j the less amount we have.

In terms of random walks, K_{ij} can be regarded as the sum of probabilities that a *lazy* random walk takes each path from i to j [6]. A lazy random walk over the undirected graph G and with parameter β is a stochastic process which will take each of the edges emanating from i with a fixed probability β and will remain in i with probability $1 - \beta D_{ii}$, being $\beta \leq 1/(max_i D_{ii})$. From this point of view, the final value of K_{ij} depends on the edge distribution and branching process between i and j. If j is an isolated node, then $K_{ij} = 0 \ \forall i \neq j$ and $K_{jj} = 1$. Moreover, as each row i of K satisfies that $0 \leq K_{ij} \leq 1 \ \forall j$ and $\sum_{j=1}^{m} K_{ij} = 1$, then we can consider each row as a probability distribution associated to vertex i. This allows us to build a proper attribute for each vertex in terms of the shape of the corresponding distribution. In our initial experiments we have found that as edit operations or noise addition on the graph will give a different kernel in terms of the number of nodes and edges, and obviously in terms of the diffusion process, building attributes in the properties of the distributions yields more stability that building such attributes in individual values of K_{ij}. This is why we retain as attribute for node i the entropy of the distribution

$$H_i^K = -\sum_{j=1}^{m} K_{ij} \log K_{ij} \ . \tag{4}$$

As we will see in the following sections, although this attribute does not provide a good discrimination between vertices it is very helpful in the continuation process in which Softassign relies. In fact, the kernel approach is closely related to the use of distance matrices in matching and tests for isomorphism [11], and, more recently, to the use of powers of the adjacency matrix [14].

In order to clarify the concept of kernel and node entropy, in Fig. 1 we show two graphs in which the smaller one X is a subgraph of the other, Y. We show the kernels of both of them and the distribution of the vertex 1 of Y.

4 Kernelizing Softassign

Given two graphs $G_X = (V_X, E_X)$, with nodes $a \in V_X$ and edges $(a,b) \in E_X$, and $G_Y = (V_Y, E_Y)$, with nodes $i \in V_Y$ and edges $(i,j) \in E_Y$, their adjacency matrices X and Y are defined by

$$X_{ab} = \begin{cases} 1 & \text{if } (a,b) \in E_X \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad Y_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E_Y \\ 0 & \text{otherwise} \ . \end{cases}$$

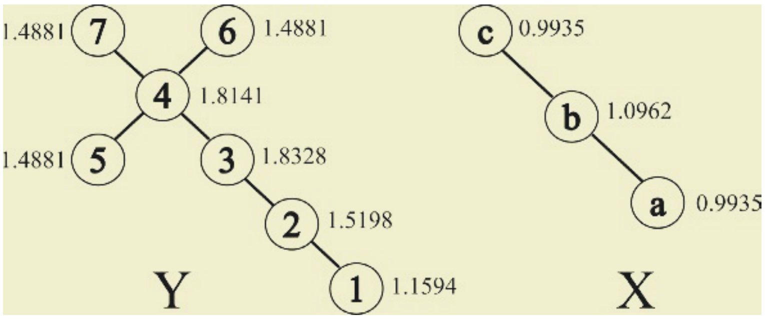
A feasible solution to the graph matching problem between G_X and G_Y is encoded by a matrix M of size $m \times n$, being $m = |V_X|$ and $n = |V_Y|$, with binary variables

$$M_{ai} = \begin{cases} 1 & \text{if } a \in V_X \text{ matches } i \in V_Y \\ 0 & \text{otherwise} \end{cases}$$

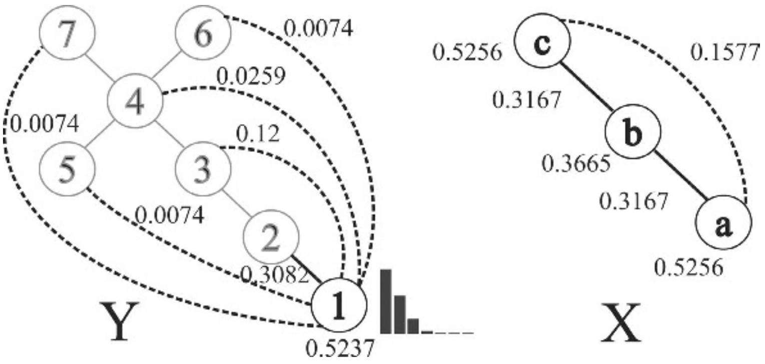
satisfying the constraints defined respectively over the rows and columns of M

$$\sum_{i=1}^{m+1} M_{ai} = 1, \forall a \quad \text{and} \quad \sum_{a=1}^{n+1} M_{ai} = 1, \forall i \ , \tag{5}$$

where equality comes from introducing slack variables for registering outliers.



(a)



(b)

$$K_Y = \begin{bmatrix} .5237 & .3082 & .1200 & .0259 & .0074 & .0074 & .0074 \\ .3082 & .3356 & .2140 & .0645 & .0259 & .0259 & .0259 \\ .1200 & .2140 & .2800 & .1369 & .0830 & .0830 & .0830 \\ .0259 & .0645 & .1369 & .1906 & .1940 & .1940 & .1940 \\ .0074 & .0259 & .0830 & .1940 & .4752 & .1073 & .1073 \\ .0074 & .0259 & .0830 & .1940 & .1073 & .4752 & .1073 \\ .0074 & .0259 & .0830 & .1940 & .1073 & .1073 & .4752 \end{bmatrix} \qquad K_X = \begin{bmatrix} .5256 & .3167 & .1577 \\ .3167 & .3665 & .3167 \\ .1577 & .3167 & .5256 \end{bmatrix}$$

(c) (d)

Fig. 1. Illustrating graph kernels and entropy. Example graphs X and Y where nodes are labelled with their entropies (a). Kernel values and distribution for vertex 1 of graph Y, and kernel values for all vertices in graph X (b). Kernel K_Y (c) and kernel K_X (d).

Following the Gold and Rangarajan formulation we are interested in finding the feasible solution M that minimizes the following cost function,

$$F(M) = -\frac{1}{2}\sum_{a=1}^{m}\sum_{i=1}^{n}\sum_{b=1}^{m}\sum_{j=1}^{n} M_{ai}M_{bj}C_{aibj} \ , \tag{6}$$

where typically $C_{aibj} = X_{ab}Y_{ij}$, that is, when $a \in V_X$ matches $i \in V_Y$, it is desirable that nodes b adjacent to a (with $X_{ab} \neq 0$) and nodes j adjacent to

i (with $Y_{ij} \neq 0$) also match, that is $M_{ai} = M_{bj} = 1$. This is the well known rectangle rule (in maximization terms we want to obtain as more rectangles as possible). Furthermore, considering the entropies defined in the previous section a simple way of *kernelizing* the latter energy function is to redefine C_{aibj} as

$$C^K_{aibj} = X_{ab}Y_{ij} \exp -[(H^{K_X}_a - H^{K_Y}_i)^2 + (H^{K_X}_b - H^{K_Y}_j)^2] \ , \tag{7}$$

where the entropies H^{K_X} and H^{K_Y} are associated to the kernels

$$K_X = e^{-\frac{\beta}{m}L_X} \text{ and } K_Y = e^{-\frac{\beta}{n}L_Y} \ ,$$

that is, we normalize the decays by the number of nodes in each graph in order to make both diffusion processes, and consequently both kernels, comparable. This normalization is useful in big graphs, where it contributes to avoid the tendency of the diffusion process towards uniform distributions, but makes no sense in small graphs. But, normalization apart, the latter definition of C^K_{aibj} ensures that $C^K_{aibj} \leq C_{aibj}$, and the equality is only verified when nodes a and i have similar entropies, and the same for nodes b and j. In practice, this weights the rectangles in such a way that rectangles with compatible entropies in their opposite vertices are preferred, and otherwise they are underweighted.

Paying now attention to the deterministic annealing process implemented by Softassign, the assignment variables are updated by

$$M_{ai} = \exp\left[-\frac{1}{T}\frac{\partial F}{\partial M_{ai}}\right] = \exp\left[\frac{1}{2T}\sum_{i=a}^{m} M_{bj}C^K_{aibj}\right] \ ,$$

where T is the temperature control parameter. Then, these assignments feed a Sinkhorn process [10], which iteratively normalizes rows and columns. After this process we obtain a doubly stochastic matrix, decrease T and a new iteration begins. The final doubly stochastic matrix is transformed into a permutation matrix by a proper clean-up process.

To see intuitively the difference between the classical Softassign and the kernelized one, in Fig. 2 we show the evolution of both algorithms for the two example graphs showed in Fig. 1. The classical Softassign prefers clearly the assignment $(b, 4)$ which is consistent with the cardinality heuristic(notable vertices in X prefer notable vertices in Y. However, a and c can be assigned either with 3, 5, 6 or 7 (ambiguity). On the other hand, in the kernelized case, the assignment $(b, 2)$ is clearly preferred and a and c may be assigned either to 1 or 3. The cardinality heuristic is inhibited in favor of a structural compatibility heuristic.

5 Experiments

We have performed several matching experiments with graphs of 50 nodes, and considering two levels of edge density: 25% and 50%. These levels of edge density are relatively high because we want to study the performance of the kernelized Softassign which it is assumed to have more problems in this context, because

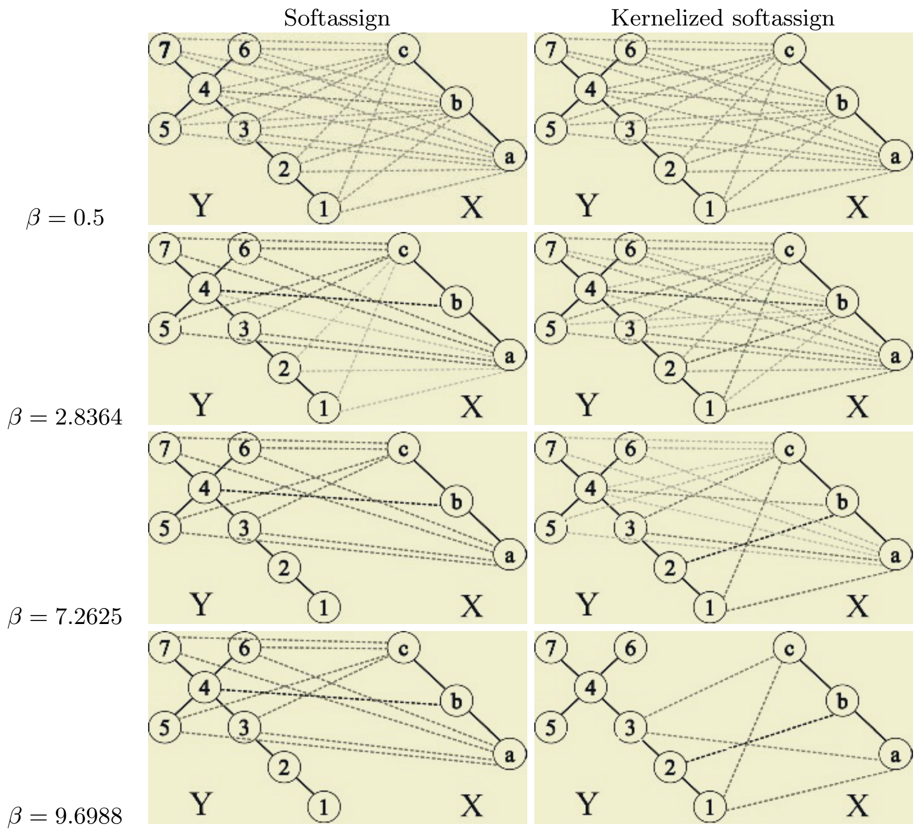
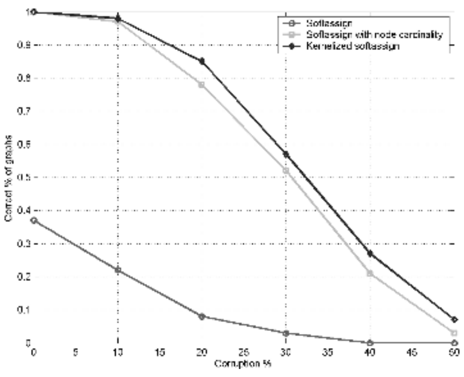


Fig. 2. Evolution of the algorithm for a simple matching problem. Matching matrices for many values of β for the classical Softassign and the kernelized version.

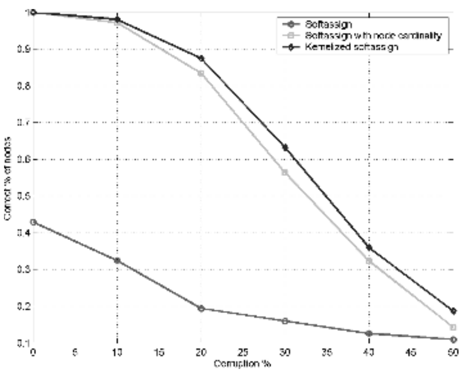
the difussion processes tend to generate uniform distributions. In all cases we use the classical initialization of Softassign. Each point corresponds to the averaged result for 100 graphs randomly generated. We have considered different noise levels: from 0% (isomorphism) to 50%. We have registered both the fraction of complete graphs successfully matched and the fraction of nodes successfully matched. In all cases the kernelized version outperforms significantly the classical one. Moreover, the kernelized version is also better than an attributed one with

$$C_{aibj} = X_{ab} Y_{ij} \exp \left[- \frac{\left| \sum_{b=1}^{m} X_{ab} - \sum_{j=1}^{n} Y_{ij} \right|}{\min(m,n)} \right] ,$$

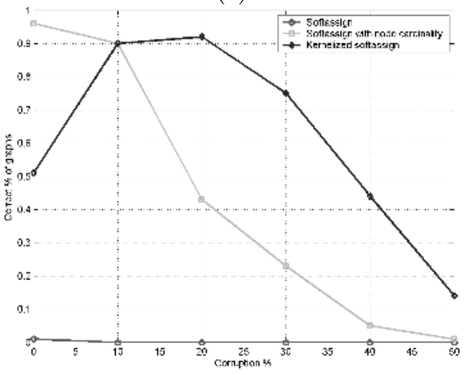
that is, a Softassign version relying on node cardinality.



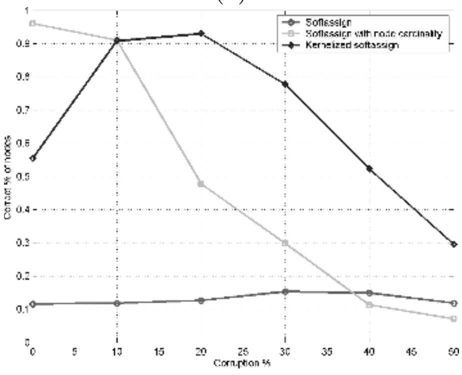
(a)



(b)



(c)



(d)

Fig. 3. Matching results. Graphs (a) and nodes (b) successfully matched with an edge density of 25%. Graphs (c) and nodes (d) successfully matched with an edge density of 50% (b).

6 Conclusions and Future Work

In this paper we have introduced a simple way of improving the performance of the Softassign graph-matching algorithm through the kernelization of the classical quadratic cost function. Our experimental results indicate that such an improvement is significant even in medium and high noise levels. Current and future work in this context includes the kernelization of other energy minimization and state-space algorithms, the formalization of the edit distance in terms of kernels, and the comparison with other approaches relying on node-neighborhood attributes.

Acknowledgements

This work was partially supported by grant $TIC2002-02792$ funded by *Ministerio de Ciencia y Tecnología* of the Spanish Government and by *FEDER*.

References

1. Chung, F.R.K.: Spectral Graph Theory. Conference Board of the Mathematical Sciences (CBMS) **92**. American Mathematical Society (1997)
2. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines Cambridge University Press (2000)
3. Gärtner: A Survey of Kernels for Structured Data. ACM SIGKDD Explorations Newsletter **5**(1) (2003) 49–58
4. Gold, S., Rangarajan, A.: A Graduated Assignment Algorithm for Graph Matching. IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (4) (1996) 377-388
5. Finch, A.M., Wilson, R.C., Hancock, E.: An Energy Function and Continuous Edit Process for Graph Matching. Neural Computation, **10** (7) (1998) 1873-1894
6. Kondor, R.I., Lafferty, J.: Diffusion Kernels on Graphs and other Discrete Input Spaces. In: Sammut, C., and Hoffmann, A. G. (eds) Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002). Morgan Kaufmann (2002) 315–322
7. Müller, K.-R., Mika, S., Räshc, Tsuda, K., Schölkopf, B.: An Introduction to Kernel-based Learning Algorithms. IEEE Transactions on Neural Networks, **12**(2) (2001) 181–201.
8. Pelillo, M.: Replicator Equations, Maximal Cliques, and Graph Isomorphism. Neural Computation **11** (1999) 1933–1955
9. Robles-Kelly, A., Hancock, E.: Graph Matching Using Spectral Seriation. In: Rangaraja, A., Figueiredo, M., and Zerubia, J. (eds) Energy Minimization Methods in Computer Vision and Pattern Recognition, Proceedings of the 4th International Workshop, EMMCVPR 2003. Lecture Notes in Computer Science. Springer. Vol **2683** (2003) 517–532
10. R. Sinkhorn.: A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices. Annals of Mathematical Statistics **35** (1964) 876–879
11. Schmidt, D.C., Druffel, L.E.: A Fast Backtracking Algorithm to Test Direct Graphs for Isomorphism Using Distance Matrices. Journal of the ACM **23** (3) (1976) 433-445
12. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press (2002).
13. Smola, A., Kondor, R.I.: Kernels and Regularization on Graphs. In: Schölkopf,B., and Warmuth, M. K. (eds) Computational Learning Theory and Kernel Machines, 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003. Lecture Notes in Computer Science. Springer. Vol. **2777** (2003) 144–158
14. DePiero, F.W., Trivedi, M., Serbin, S.: Graph Matching Using a Direct Classification of Node Attendance. Pattern Recognition, Vol. **29**(6) (1996) 1031–1048
15. Ozer, B., Wolf, W., Akansu, A.N.: A Graph Based Object Description for Information Retrieval in Digital Image and Video Libraries. In: Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries (1999) 79–83

Structural Perceptrons for Attributed Graphs

Brijnesh J. Jain and Fritz Wysotzki

Dept. of Electrical Engineering and Computer Science
Technical University Berlin, Germany

Abstract. We propose a structural perceptron for supervised and unsupervised learning on data represented in terms of attributed graphs. We integrate structural perceptrons into a multi-layer perceptron and competitive learning network to provide examples of supervised and unsupervised neural learning machines which are suited to process graphs. In first experiments the proposed algorithms were successfully applied to function regression, classification, and clustering.

1 Introduction

It is common practice to represent data in terms of vectors of an Euclidean space, because the Euclidean space provides powerful analytical techniques for data analysis usually not available in other representations. Such a representation, however, is too limited for many relevant application areas including domains such as computer vision, bioinformatics, chemistry, or text mining. A more versatile and expressive tool for representing structured data are, for example, attributed graphs.

Despite its applicability and importance, learning on graphs is still widely unexplored. Current research focus on (1) the problem of embedding graphs into vector spaces to access the whole plethora of analytical tools (e.g. [10]); (2) algorithms on pairwise proximity data [6]; (3) kernel methods for structures [2]; and (4) adaptive processing of graphs (e.g. [1, 11, 12]). In the structural pattern recognition literature, adaptive processing of graphs is mainly concerned with devising clustering algorithms (e.g. [1, 4, 9]). In the neural networks community, connectionist models have been proposed for supervised learning on the rather restrictive class of directed (ordered) acyclic graphs [12].

In this contribution, we suggest a structural perceptron for adaptive processing of graphs within a supervised and unsupervised setting. To facilitate adaptive processing, we associate a structural perceptron with an attributed weight graph and replace the concept of an inner product of vectors by the *Schur-Hadamard* (SH) *inner product* of graphs. Despite its name, the SH inner product is not an inner product, but shares some useful properties of an inner product to extend supervised and unsupervised neural learning machines for attributed graphs. Finally, (un)supervised training of networks composed of structural units is then based on minimizing a suitable error function as a function of adjustable weights.

The rest of this paper is organized as follows: Section 2 introduces the SH inner product. Section 3 describes structural neural learning machines. In Section 4 we present first experiments. Finally, Section 5 concludes.

2 The Schur-Hadamard Inner Product

This section provides basic notions and introduces the SH inner product.

Terminology. Let $\mathcal{S}$ be a set. By $\mathcal{S}^{[2]}$ we denote the set of all ordered tuples $(i,j) \in \mathcal{S}^2$ with $i \neq j$. The set of all $n \times m$-matrices $A = (\boldsymbol{a}_{ij})$ with entries $\boldsymbol{a}_{ij}$ from a set $\mathcal{S}$ is denoted by $\mathcal{M}_{n \times m}(\mathcal{S})$.

Let $\mathcal{A}$ be an inner product space over $\mathbb{R}$, for example $\mathcal{A} = \mathbb{R}^m$. An *attributed graph* is a tuple $X = (V, \mu)$ consisting of a finite set $V \neq \emptyset$ and a function $\mu : V^2 \rightarrow \mathcal{A}$. The elements of V are the *vertices* of the graph X and the pairs $(i,j) \in V^{[2]}$ with $\mu(i,j) \neq \mathbf{0}$ are its edges. The function μ is the *attribute function* of X. By $\mathcal{G}_{\mathcal{A}}$ we denote the set of attributed graphs with attributes from $\mathcal{A}$. The vertex set of a graph X is referred to as $V(X)$, its edge set as $E(X)$, and its attribute function as μ_X. Let X be an attributed graph of order $|X| = |V(X)| = n$. The *(attributed) adjacency matrix* of X is a matrix $A(X) = (\boldsymbol{x}_{ij}) \in \mathcal{M}_{n \times n}(\mathcal{A})$ with entries $\boldsymbol{x}_{ij} = \mu_X(i,j)$.

A *permutation* acting on X is a bijection $\pi : V(X) \rightarrow V(X)$ from $V(X)$ onto itself. The image graph of a permutation π acting on X is denoted by X^π. The set $\mathcal{S}_X$ of all permutations acting on X is called the *symmetric group* of X. Note that in general $A(X) \neq A(X^\pi)$.

The Schur-Hadamard Inner Product. First we define the inner product of attributed matrices. Let $A, B \in \mathcal{M}_{n \times n}(\mathcal{A})$ be matrices with $A = (\boldsymbol{a}_{ij})$ and $B = (\boldsymbol{b}_{ij})$. Addition of matrices and scalar multiplication are defined componentwise. The inner product $\langle\, , \rangle$ defined on $\mathcal{A}$ induces an inner product on $\mathcal{M}_{n \times n}(\mathcal{A})$ by

$$\big\langle A, B \big\rangle = \sum_{i=1}^{n} \sum_{j=i}^{n} \langle \boldsymbol{a}_{ij}, \boldsymbol{b}_{ij} \rangle \,.$$

To define a formal addition, scalar multiplication, and the SH inner product of graphs we use the following technical convention: Let $X, Y \in \mathcal{G}_{\mathcal{A}}$. Suppose that $n = \max\{|X|, |Y|\}$ and $m = \min\{|X|, |Y|\}$. We implicitly assume that both graphs are of the same order n by inserting $n - m$ isolated nodes into the smaller of both graphs, each labeled with $\mathbf{0}$. Then $X + Y = A(X) + A(Y)$ and $\lambda X = \lambda A(X)$ for all $\lambda \in \mathbb{R}$. The SH inner product is of the form

$$\sigma : \mathcal{G}_{\mathcal{A}} \times \mathcal{G}_{\mathcal{A}} \rightarrow \mathbb{R}, \quad \big(X, Y\big) \mapsto \max_{\pi \in S_X} \big\langle A(X^\pi), A(Y) \big\rangle \,.$$

A permutation $\pi \in \mathcal{S}_X$ with $\sigma\big(X, Y\big) = \big\langle A(X^\pi), A(Y) \big\rangle$ is called *embedding* from X into Y. By $\mathcal{I}(X, Y)$ we denote the set of all embeddings from X into Y.

In [7] it is shown that the SH inner product is symmetric and positive, but not bilinear. Hence, it is not an inner product. But the Cauchy-Schwarz inequality holds giving rise to an Euclidean metric $\delta(.)$ induced by the SH inner product

$$\delta : \mathcal{G}_{\mathcal{A}} \times \mathcal{G}_{\mathcal{A}} \rightarrow \mathbb{R}, \quad \big(X, Y\big) \mapsto \sqrt{\sigma(X,X) - 2\sigma(X,Y) + \sigma(Y,Y)} \,.$$

We conclude this section with two important remarks: (1) Determining the SH inner product is NP complete, since it generalizes the maximum common subgraph problem. (2) There is no restriction on the order of the graphs. Aligning two graphs to the same size is a pure technical trick and is not required in a practical implementation when computing the SH inner product.

3 Structural Perceptrons

This section first introduces structural perceptrons for attributed graphs. Next we provide an example of a supervised and of an unsupervised neural learning machine. Finally, we discuss some limitations of this approach.

The Model. Let $X, W \in \mathcal{G}_{\mathcal{A}}$ be attributed graphs with adjacency matrix $A(X) = (\boldsymbol{x}_{ij})$ and $A(W) = (\boldsymbol{w}_{ij})$. A structural perceptron is of the form:

$$\begin{aligned} a &= \sigma(X, W) + b \\ y &= g(a) \end{aligned}$$

where X denotes a data graph, W is the weight matrix of the perceptron, b is the bias, $\sigma(X, W)$ the SH inner product of X and W, a is the activation, $g(.)$ is the non-linear activation function, and y is the output.

Supervised Learning with Multi-layer Perceptrons. Assume that we are given a training sample $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$ consisting of a set of data graphs $\mathcal{X} = \{X_1, \ldots, X_M\} \subseteq \mathcal{G}_{\mathcal{A}}$ together with corresponding output values $\mathcal{Y} = \{\hat{\boldsymbol{y}}_{\boldsymbol{1}}, \ldots, \hat{\boldsymbol{y}}_{\boldsymbol{M}}\} \subseteq \mathbb{R}^m$. The problem is to estimate an unknown relation

$$f : \mathcal{G}_{\mathcal{A}} \to \mathbb{R}^m$$

given the sample $\mathcal{Z}$ and a set $\mathcal{H}$ of functions $h : \mathcal{G}_{\mathcal{A}} \to \mathbb{R}^m$. Here we are concerned with functions $h \in \mathcal{H}$ implemented by MLPs for attributed graphs. The functions of $\mathcal{H}$ are of the form $h(X, \mathcal{W}) = h_L(\boldsymbol{y}_{\boldsymbol{L}\text{-}\boldsymbol{1}}, \mathcal{W}_L)$ with

$$\boldsymbol{y}_{\boldsymbol{1}} = h_1(X, \mathcal{W}_l) \tag{1}$$

$$\boldsymbol{y}_{\boldsymbol{l}} = h_l(\boldsymbol{y}_{\boldsymbol{l}\text{-}\boldsymbol{1}}, \mathcal{W}_l) \tag{2}$$

for all layers $l \in \{1, \ldots, L\}$ where X is a data graph, $\boldsymbol{y}_{\boldsymbol{l}}$ is a vector representing the output of the function h_l, and $\mathcal{W}_l$ is the set of adjustable weights (including biases) of h_l. The learning task considered here is based on minimizing a suitable error function with respect to the weights and biases by gradient descent using the back-propagation algorithm.

We commence with supervised learning of a single structural perceptron using the error-back-propagation algorithm. Since $L = m = 1$, we may drop dispensable indices. For any data graph $X \in \mathcal{X}$ let $E = E(y)$ be a differentiable error function of the network output variables. The derivative of E with respect to some weight $w_{ij}^k \in \mathcal{W}$ is of the form

$$\frac{\partial E}{\partial w_{ij}^k} = \frac{\partial E}{\partial a} \frac{\partial a}{\partial w_{ij}^k} = \delta \frac{\partial a}{\partial w_{ij}^k}$$

where w_{ij}^k is the k^{th} component of the vector $\boldsymbol{w}_{ij} \in \mathcal{A}$ and

$$\delta = \frac{\partial E}{\partial a} = g'(a)\frac{\partial E}{\partial y} \tag{3}$$

is often referred to as the *error*. Since g and E are known we substitute appropriate expressions for $g'(a)$ and $\partial E/\partial y$ to evaluate (3). To evaluate $\partial a/\partial w_{ij}^k$ we choose an embedding $\pi \in \mathcal{I}(X, W)$ and set

$$\frac{\partial a}{\partial w_{ij}^k} = \frac{\partial}{\partial w_{ij}^k}\Big\{\sigma(X,W) + b\Big\} = \frac{\partial}{\partial w_{ij}^k}\Big\{\langle A(X^\pi), A(W)\rangle + b\Big\} = \pi\left(x_{ij}^k\right)$$

where $\pi(x_{ij}^k) = x_{rs}^k$ with $r = i^\pi$ and $s = j^\pi$. Putting all together we obtain

$$\frac{\partial E}{\partial w_{ij}^k} = \delta \cdot \pi\left(x_{ij}^k\right) .$$

Since the bias b is not involved in the computation of the SH inner product, the derivative $\partial E/\partial b = \delta$ is of the same form as in the case of vectorial perceptrons.

Given a gradient of $\nabla E(\mathcal{W})$ we adjust the weights w_{ij}^k according to the rule

$$w_{ij}^k \leftarrow w_{ij}^k + \eta\delta \cdot \pi\left(x_{ij}^k\right)$$

where δ is the error, $\pi \in \mathcal{I}(X, W)$ is the embedding chosen to evaluate the derivatives $\partial E/\partial w_{ij}^k$, and $\pi(x_{ij}^k)$ is the permuted input.

Once we know how a structural perceptron operates, it is straightforward to assemble several structural units with conventional processing units to a structural MLP. For sake of presentation, we restrict to the simple case that structural perceptrons may occur only in the first hidden layer.

A structural single-layer network implementing a function $\boldsymbol{y} = h(X, \mathcal{W})$ with $\boldsymbol{y} \in \mathbb{R}^m$ is composed of m structural output units, each associated with an attributed weight graph. A structural multi-layer network with $L \geq 2$ layers is composed of a structural single-layer network implementing the function $\boldsymbol{y_1} = h(X, \mathcal{W}_1)$ and a conventional vectorial network with $L-1$ layers representing the function $\boldsymbol{y} = h_L(\boldsymbol{y_{L\text{-}1}}, \mathcal{W}_L)$ such that (2) holds for all $l \in \{2, \ldots, L\}$. Since $\boldsymbol{y_1} = h_1(X, \mathcal{W}_1)$ is a real valued vector, it is straightforward to link the modules $h_1(X, \mathcal{W}_1)$ and $h_2(\boldsymbol{y_1}, \mathcal{W}_2)$ implementing the first and second layer, resp., of the structural MLP. The forward and backward passes, and the error correcting rule for layers $l \geq 2$ follow the same procedure as for vectorial MLPs.

Structural Competitive Learning. Simple competitive learning (CL) is well suited to cluster or categorize unlabeled data points where the competitive network discovers statistically salient features by itself from the correlations of the data points. Competitive learning is closely related to *Vector Quantization*, *Adaptive Resonance Theory*, or *Self-Organizing Maps*. Here, we are primarily concerned with CL as an elementary building block of unsupervised neural learning machines from a conceptual rather than practical point of view.

Clustering M attributed graphs $\mathcal{X} = \{X_1, \ldots, X_M\} \subseteq \mathcal{G}_\mathcal{A}$ amounts to partition the feature space $\mathcal{G}_\mathcal{A}$ such that the average distortion of data graphs to their cluster centers $\mathcal{Y} = \{Y_1, \ldots, Y_K\} \subseteq \mathcal{G}_{\mathcal{A}}$ is minimized. The average distortion to be minimized is of the form

$$E(M, \mathcal{Y}, \mathcal{X}) = \frac{1}{N} \sum_{j=1}^{K} \sum_{i=1}^{N} m_{ij} \delta(X_i, Y_j) \tag{4}$$

where $\delta(.)$ measures the structural distortion induced by the representation Y_{i^*} of data graph X_i. Here we assume that the structural distortion $\delta(.)$ is the Euclidean distance induced by the SH inner product. Then competitive learning proceeds as follows to minimize E in online mode[1]:

1. Initialize $\mathcal{Y} = \{Y_\bullet, \ldots Y_K\}$.
2. **Repeat**
   (a) Randomly select a data graph $X \in \mathcal{X}$.
   (b) Find Y_{i^*} with $i^* = \arg\min_i \delta(X, Y_i)$ and $\pi \in \mathcal{I}(X, Y_{i^*})$.
   (c) Adjust Y_{i^*} by using the update formula $Y_{i^*} \leftarrow \eta X^\pi + (1 - \eta) Y_{i^*}$.
   (d) Decrease learning rate η.
3. **until** no noticeable changes in $\mathcal{Y}$ are observed.
4. Output cluster centers $\mathcal{Y} = \{Y_\bullet, \ldots Y_K\}$.

The essential parts of the algorithm are step (2b) and (2c). Step (2b) selects the model Y_{i^*} closest to the current data graph X with respect to the Euclidean distance. In step (2c) the competitive learning rule adjusts model Y_{i^*} to move it closer to the current data graph X. This makes the winner more likely to win on X in the future. Indeed, it has been shown in [7], that any graph isomorphic to $\eta X^\pi + (1-\eta) Y$ is a weighted mean of X and Y in the sense of Jiang et al. [8].

Discussion. Besides the well known limitations neural learning machines in the domain of feature vectors, additional problems arise when dealing with graphs.

Elusiveness of adaptive processing: The gradients ∇E of both error functions as a function of adjustable parameters are not well defined, because they both depend on the the particular choice of an embedding. Consequently, there may be several directions of steepest descent. Hence adjusting the parameters may move the algorithm in a wrong direction. Since there is no canonical embedding the algorithm may have a tendency to zigzag its way through the weight space without gaining substantial improvements. Moreover, the non-uniqueness of ∇E makes a theoretical analysis of convergence properties difficult.

Computational inefficiency: As opposed to learning on vectorial data, neural learning machines in the domain of attributed graphs are computationally inefficient. The exponential computational effort results from the NP completeness of determining the SH inner product. Solving a large number of NP-complete problems to train a neural network may hinder its practical use. Thus we may resort to approximate solutions of the SH inner product. Approximate solutions, however, increase the number of potential directions to mislead the algorithm. In the case of MLPs, using approximate solutions of the SH inner product might

[1] Minimizing (4) in batch mode corresponds exactly to the K-means algorithm [7].

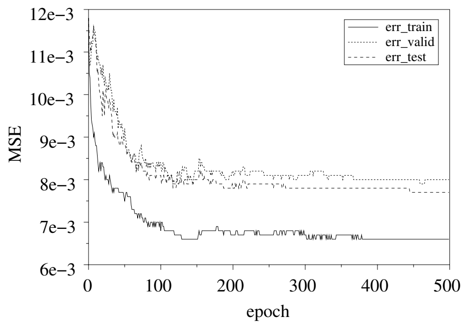


Fig. 1. The evolving MSE of the function regression problem. Shown are the training (solid line), validation (dotted line), and test error (dashed line).

have a similar effect as training with noise. The approximations will *smear out* data graphs and reduce over-fitting.

Optimal MLP *architecture:* Due to missing analytical tools, the most natural choice of classifiers in the area of structural pattern recognition are K-nearest neighbor (KNN) classifiers. The performance of KNN classifiers is heavily dependent on the choice of K and the similarity measure. Structural MLPs sweep this problem under a big rug of determining the optimal number of structural and vectorial hidden units and the order of weights graphs in the first hidden layer.

4 Experiments

This section serves to illustrate that adaptive processing of structures using structural perceptrons can be successfully applied to simple function regression, classification, and clustering tasks on attributed graphs. In all of our experiments we used an approach proposed by [7] to approximate the SH inner product.

Function Regression

In our first experiment we tested a structural MLP on a function regression problem. The data set was generated by sampling the function

$$f : \mathcal{W}_{\mathbb{R}}^{20} \to [0,1], \quad X \mapsto \frac{\sum_{(i,j)\in E(X)} |\mu_X(i,j)|}{|X|\big(|X|-1\big)}$$

where $\mathcal{W}_{\mathbb{R}}^{20}$ denotes the set of all random weighted graphs X of order $|X| \leq 20$ having weights drawn from a $N(0,1)$ normal distribution. The function $f(X) \in [0,1]$ measures the weighted edge density of a graph X.

To compile the data set, we generated 1250 graphs from a uniform distribution over the set $\mathcal{W}_{\mathbb{R}}^{20}$. The data set was divided into a training, validation, and test set composed of 500, 250, and 500 weighted graphs, respectively.

We used a two-layer perceptron consisting of 10 structural units with *tanh*-activation function and one linear output unit. Each weight graph in the first

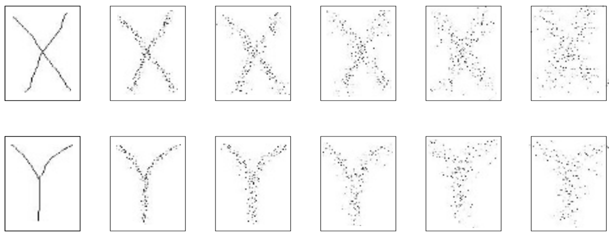


Fig. 2. Images of handwritten characters $'X'$ and $'Y'$. The first column shows the model images. Column 2-6 are samples of corrupted images with increasing noise level $\sigma = 2, 4, 6, 8, 10$. For sake of presentation no rotation is shown.

hidden layer was of order 3. We set the initial learning rate η and the momentum term α to 0.1. We trained the network using the standard mean sum-of-squares error (`MSE`) function.

Figure 1 plots the training, validation and test error against the number of passes through the training set. All three error rates have converged by maintaining a small oscillation around $E_{train} = 0.0066$, $E_{valid} = 0.008$, and $E_{test} = 0.0077$. Since $\eta \to 0$, the oscillations are due to the randomness of the approximate solution of the `SH` inner product. From the plot we see that randomness of approximations is similar to training with noise and therefore prevents over-fitting of the training data.

Classification: Synthetic Characters
In this experiment we investigated the capabilities of a single structural perceptron (`SP`) to deal with both types of errors occuring in graph based representations, structural variations and noisy attributes.

We used synthetic data to emulate rotation, translation, and scaling invariant handwriting recognition of characters as it typically occurs in pen technology of small hand-held devices. We have drawn two handwritten characters models $'X'$ and $'Y'$. The contours of each image were expressed as a set of points in the 2D plane. For each model we generated corrupted data characters as follows: First we randomly rotated the model image. Then to each point we added $N(0, \sigma)$ noise with standard deviation $\sigma = 2, 4, 6, 8, 10$. Each point had 10% probability to be deleted. Figure 2 shows the models and a sample of corrupted data images. Each point set was transformed to a fully connected attributed graph. Vertices represent the points and edges an abstract line between the corresponding points. The attributes are from $\mathcal{A} = [0, 1]^3 \subseteq \mathbb{R}^3$ and contain normalized distance values including their statistical spread and location parameters. Table 1 summarizes the structural variation of the data set revealing a strong variation, in particular at a high noise level.

We trained a `SP` with logistic sigmoid activation function. The weight graph W was of order $|W| = 20$. The learning rate was initially set to 0.1 and the momentum term to 0.2. The network was optimized using the cross-entropy error function.

Table 1. Structural variations of the character data set. Shown are the mean, variance, and the maximal difference $\Delta = \max - \min$ of vertices for each noise level σ. For each entry x/y the numbers x and y refer to character $'X'$ and $'Y'$, respectively.

	2.0	4.0	6.0	8.0	10.0
mean	16.8/20.5	21.7/25.9	27.2/32.9	31.6/38.2	36.0/43.3
var	0.98/1.07	3.70/4.10	4.84/6.35	6.75/7.41	7.43/9.40
Δ	6.0/6.0	11.0/12.0	12.0/16.0	17.0/14.0	15.0/19.0

Table 2. Classification results: (a) synthetic characters and (b) arm postures. To (a): Shown are the percentage test error rates of the SP and SV classifiers for varying noise level σ. To (b): Shown are the number of misclassified samples by NN. and NN. for each class and the total percentage error rate E. The numbers in parenthesis indicate the number of images of the corresponding class.

(a)

σ	2	4	6	8	10
SP	0	0	1	2	2
SV.	0	0	6	8	29
SV.	0	0	4	6	23
SV.	0	0	5	4	29
SV.	0	0	5	4	20

(b)

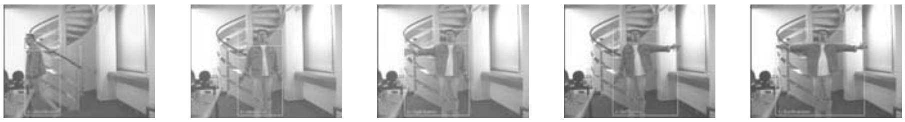
	$P.$	$P.$	$P.$	$P.$	$P.$	E
	(37)	(48)	48)	(48)	(48)	[%]
NN.	3	15	44	40	0	44.5
NN.	3	0	0	1	0	1.7

We compared the SP classifier with four support vector (SV) classifiers for proximity data: (SV_1) the pairwise proximities classifier proposed by [5], (SV_2) the same classifier with RBF kernel, (SV_3) support vector learning using the SH inner product as a non-positive semi-definite 'kernel', and (SV_4) support vector learning with RBF kernel on the Euclidean distance induced by the SH inner product. To provide a fair comparison we used the same setup as in [3]. We sampled 50 training examples of each character and performed 10-fold cross validation to estimate the generalization error.

The results are given in Table 2(a). The proposed SP algorithm performed more robust to noise and structural variation than the SV classifiers. As expected, the performance of all classifiers decreased with increasing noise level though the recognition rate, in particular for the SP classifier, is very good even for highly corrupted characters.

Clustering: Sensing People

In our last experiment we applied the CL algorithm to learn the class structure of arm postures as shown in Figure 3. Five different classes of 229 postures are considered: (P_0) Unknown, (P_1) NoArms, (P_2) RightArm, (P_3) LeftArm, and (P_4) BothArms, each referring to the lifted arms of a person. Each image was obtained by automatically localizing a person in video data from a camera. The localized person was enclosed in a bounding box. Position of body parts, like head and hands of the person are identified by skin color. We transformed



(a) $P.$ (b) $P.$ (c) $P.$ (d) $P.$ (e) $P.$

Fig. 3. Sample images of arm postures.

each image to a fully connected attributed graph. The vertices represent the left or right corner of the box, or a body part. We used the 1-c coding scheme with $c = 3$ to transform the discrete attributes into a numerical vector $\boldsymbol{a} \in \{0, 1\}^3$. Edge attributes are the distance between the corresponding components in the image.

We randomly selected five perturbed patterns of each class to initialize the models[2]. After 450 iterations the average distortion E has almost converged. Small oscillations of E are due to the approximative nature of the proposed algorithm and the non-uniqueness of the weighted mean. Table 2 shows the percentage error rate of the nearest neighbor classifiers using the models after initialization ($\texttt{NN}_0$) and after clustering ($\texttt{NN}_1$). The results indicate that structural `CL` is able to find the class structure of the data set.

5 Conclusion

In this paper we considered the problem of constructing supervised and unsupervised neural learning machines when data is given in terms of attributed graphs. We proposed a structural perceptron for adaptive processing of graphs. The key concepts to facilitate learning on graphs are adjustable attributed graphs and the `SH` inner product of graphs mimicking inner products of vector spaces. In first experiments we applied structural perceptrons to solve supervised and unsupervised learning problems. The main problems, however, with structural perceptrons are the elusiveness of adaptive processing and the high computational complexity. Hence, for practical application the problem of analytical and computational intractability inherent in the `SH` inner product will be of increasing importance. Possible directions of future work include application of structural perceptrons to other supervised and unsupervised neural learning architectures, experimental validation on practical problems, investigations on tricks of the trade to accelerate convergence, and theoretical analysis on statistical learning theory, representational capabilities and convergence properties.

References

1. H. Bunke, P. Foggia, C. Guidobaldi, and M. Vento. Graph clustering using the weighted minimum common supergraph. In E. Hancock and M. Vento, editors, *Graph Based Representations in Pattern Recognition. 4th IAPR International Workshop, GbRPR 2003*, LNCS 2726, pages 235–246. Springer-Verlag, 2003.

[2] The initialization scheme is due to the fact that simple `CL` performs poor even for feature vectors, if the initial models are not chosen carefully.

2. T. Gärtner. A survey of kernels for structured data. *SIGKKD Explorations*, 5(1):49–58, 2003.
3. P. Geibel, B. Jain, and F. Wysotzki. SVM learning with the SH inner product. In M. Verleysen, editor, *Proc. of the 12th European Symposium on Artificial Neural Networks, ESANN'04*. D-Facto, Brussels, 2004. *Accepted for publication.*
4. S. Günter and H. Bunke. Self-organizing map for clustering in the graph domain. *Pattern Recognition Letters*, 23:401–417, 2002.
5. R. Herbrich. *Learning Kernel Classifiers*. The MIT Press, Cambridge, MA, 2002.
6. T. Hofmann and J. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(1):1–14, 1997.
7. B. Jain and F. Wysotzki. Central clustering in the domain of graphs. *Machine Learning Journal*, 2004. Accepted for publication.
8. X. Jiang, A. Münger, and H. Bunke. On median graphs: properties, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1144–1151, 2001.
9. M. Lozano and F. Escolano. Em algorithm for clustering an ensemble of graphs with comb matching. In A. Rangarajan, M. Figueiredo, and J. Zerubia, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition, EMMCVPR 2003*, LNCS 2683, pages 52–67. Springer-Verlag, 2003.
10. B. Luo, R. Wilson, and E. Hancock. Spectral embedding of graphs. *Pattern Recognition*, 36(10):2213–2223, 2003.
11. M. V. P. Foggia, R. Genna. Symbolic vs connectionist learning: an experimental comparison in a structured domain. *IEEE Transaction on Knowledge and Data Engineering*, 13(2):176–195, 2001.
12. A. Sperduti and A. Starita. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*, 8(3):714–735, 1997.

Eigenspace Method by Autoassociative Networks for Object Recognition

Takamasa Yokoi, Wataru Ohyama,
Tetsushi Wakabayashi, and Fumitaka Kimura

Faculty of Engineering, Mie University 1515 Kamihama, Tsu 514-8507, Japan
yokoi@hi.info.mie-u.ac.jp

Abstract. This paper studies on a new eignespace method which employs autoassociative networks for object recognition. Five layered autoassociative network is available to obtain a manifold on the minimum square error hypersurface which approximates a distribution of learning sample. Recognition experiments were performed to show that the manifold of rotating object is obtained by learning and the objects, such as a mouse and a stapler, are correctly recognized by the autoassociative networks. It is also shown that the accuracy of approximating closed manifold and the accuracy of recognition are improved by emploing multiple autoassociative networks each of which is trained by a partition of the learning sample.The property and the advantage of the five layered autoassociative network are demonstrated by a comparative study with the nearest neighbor method and the eigenspace method.

1 Introduction

The object recognition techniques play essential roles in widely raging applications from inspection and classification of industrial parts to vision system for autonomous mobile robot.

One of the typical object recognition techniques is the nearest neighbor method which compares an input image with multiple template images of each object captured a priori for variety of positions and illuminating condition. A drawback of the nearest neighbor method is that the required computation time and storage increases with the number of template images which increases rapidly depending on the degree of freedom of the variations.

To solve the problem Murase et al.[1] proposed a parametric eigenspace method which compresses an input image onto the eigenspace. The method parametrically represents a sequence of continuously varying images as a manifold in the eigenspace spanned by a small number of pricipal components of the variation to reduce the computation time and storage. However, since the eigenspace is derived by the K-L transformation, which is linear, the dimensionality of the eignespace tends to be higher than the intrinsic dimensionality of the variation. While the parametric representation is suitable for variations related to explicit parameters such as rotation, it is impossible to represent nonparametric or hidden parametric variations.

This paper studies on a new eignespace method which employs autoassociative networks for object recognition. The autoassociative network is an artificial neural network having the same number of neurons in input and output layers, and the less in the middle (hidden) layers. The network is trained using the input vector itself as the desired output by the back propagation method [2]. This training leads to organize a dimension reduction network between the input layer and the middle layer, and a restoration network between the middle layer and the output layer (Fig.1). The autoassociative neural network was applied to image compression and dimension reduction [3–5]. The five layered autoassociative network performs dimension reduction by nonlinear mapping and the input pattern is mapped onto a manifold on minimum square error hypersurface which approximates the distribution of the learning sample. The square error between the input and the output of the network stands for squared distance between the input and the manifold, by which the minimum distance classification is easily performed for the input object [6].

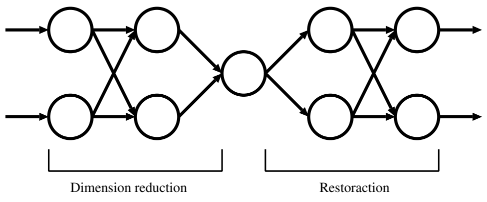


Fig. 1. Autoassociative neural network

Recognition experiments were performed to show that the manifold of rotating object is obtained by learning and the objects, such as a mouse and a stapler, are correctly recognized by the autoassociative networks. It is also shown that the accuracy of approximating closed manifold and the accuracy of recognition are improved by emploing multiple autoassociative networks each of which is trained by a partition of the learning sample. The property and the advantage of the five layered autoassociative network are demonstrated by a comparative study with the nearest neighbor method and the eigenspace method.

Section 2 outlines the learning and the classification by means of the autoassociative network, and Sect.3 describes the learning of the manifold yielded by a rotating object. Section 4 describes the experiments and the results of the object recognition, and Sect.5 summarizes and concludes this work.

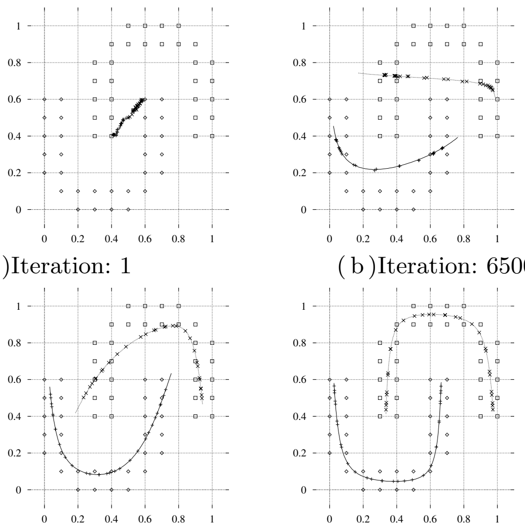
2 Autoassociative Neural Networks

The proposed method organizes a set of networks each of which are trained independently for each class using the feature vector of the class. As a result, the squared error between an input and the output is generally minimized by the network of the class to which the input pattern belongs. This property enables us to classify an unknown input pattern: The unknown pattern is fed to all networks, and is classified to the class with minimum squared error.

In contrast with the pattern recognition using the mutual associative networks, each autoassociative network is organized independently for each class, and the training load of the networks can be distributed to handle large scale pattern recognition problems. The first and the second layers of the five layered networks perform nonlinear encoding operation, and the fourth and the fifth layers the nonlinear decoding operation. The five layered networks studied in this paper have the same number of neurons in their second and fourth layers, and thus symmetric encoding and decoding networks.

Figure 2 shows an example of learning process of the five layered autoassociative networks for nonlinearly distributed two dimensional patterns. The axes of graphs represent the feature value (x_1,y_1). The number of neurons in the input layer (first layer) to the output layer (fifth layer) is 2, 2, 1, 2, 2 respectively. The input signal is samples of two dimensional feature vector from each class, and the same signal is given as the teaching signal (desired output), and the learning process is repeated independently for each class. Two curved line segments in this figure represent the trace of the output which are obtained by sweeping the output of hidden layers from 0 to 1. Squares on the line segments represent the projections of the samples. These line segments converge to the medial axes of the distributions, which play the same role as the principal axes of the principal component analysis.

Figure 3 shows how to classify an input pattern by the converged autoassociative networks. An input pattern (x_1, x_2) is given to the network of each


(a)Iteration: 1 (b)Iteration: 6500

(c)Iteration: 13000 (d)Iteration: 4000000

Fig. 2. Learning process of five layered autoassociative neural networks

class. The output (y_1, y_2) and (z_1, z_2) are the coordinates of Y, and Z on the projection line respectively, and the squared distance are given by $\|Y - X\|^2$, and $\|Z - X\|^2$ respectively. The input pattern X is classified to the class with the minimum distance, i.e. the class which minimizes the square error of the input and the output of the network, e.g. the left lower class in this figure. The output u and v from the hidden layers take the value from 0 to 1.

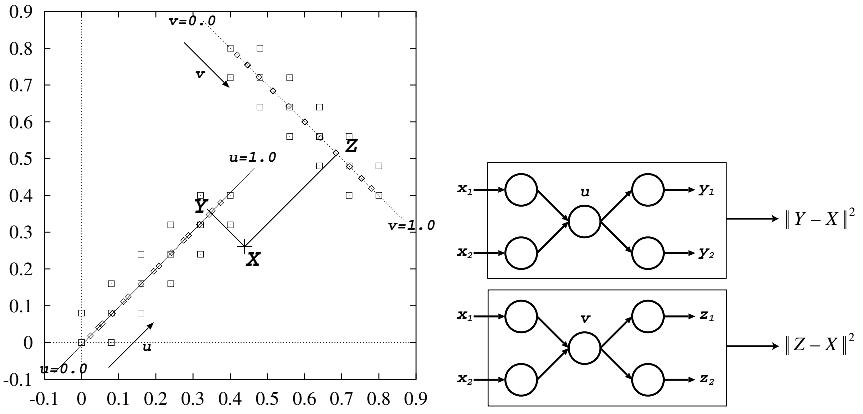


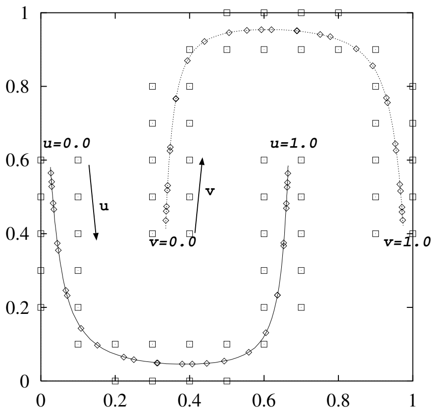
Fig. 3. Discrimination of unknown pattern by three layered autoassociative neural networks

Figure 4 shows example of dimension reduction and classification by five layered autoassociative networks for nonlinearly distributed patterns. This figure shows that the samples are projected to curved hypersurfaces (curved lines in two dimensional case) by five layered networks. An input pattern is classified to the class with the minimum squared error, i.e. the class with the nearest curves. While the nonlinearly distributed pattern can not be completely separated by the three layered networks, all samples can be separated by the five layered networks which can perform nonlinear projections (Fig. 4).

3 Learning of the Eigenspace by the Autoassociative Network

This section describes how the five layered autoassociative network learns the image sequence of an object continuously changing its appearence.

The learning sample of each object consists of 36 gray scale images of size 400×400 which are captured at every 10 degree of rotation on vertical axis. We used a computer controlled turn table to acquire the learning sample. Figure 5 shows the example of the learning sample. The input of the autoassociaive network is 100 dimensional feature vector composed of the average brightness in 10×10 blocks of the image. The feature vector is input to the autoassociaive



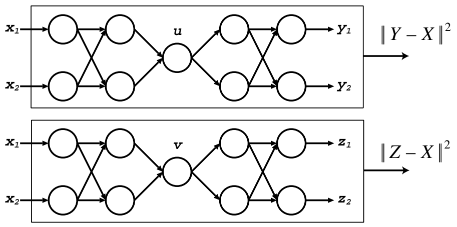


Fig. 4. Dimension reduction and discrimination by five layered autoassociative neural networks

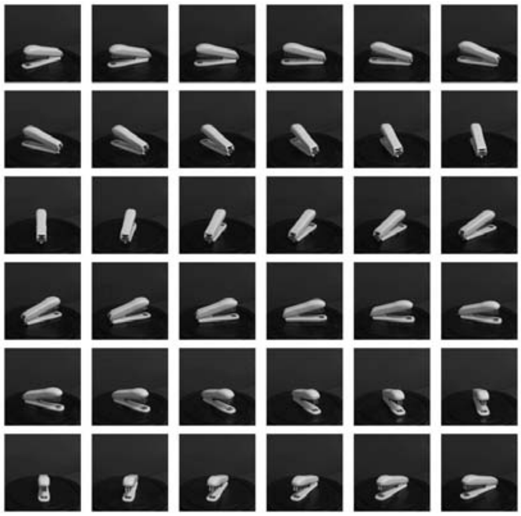


Fig. 5. Example of learning sample of a rotating object

network as an input signal and a desired output. The number of neurons of 5 layered network is 100,4,1,4,100 respectively. The network of this organization approximates the distribution of the learning sample in the feature space by four pieces of one dimensional curves.

Figures 5 and 6 show an example of rotating object and the distribution of the feature vectors. Figure 6 shows the distribution projected and visualized in the three demensional eigenspace obtained by the principal component analysis of the 100 dimensional feature distribution. The x, y, and z axis respectivly stands for the principal components with largest three variances (eignvalues) of

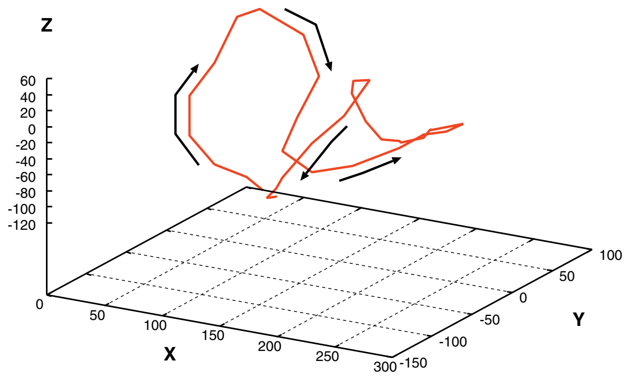


Fig. 6. Distribution of the learning sample in the eigenspace

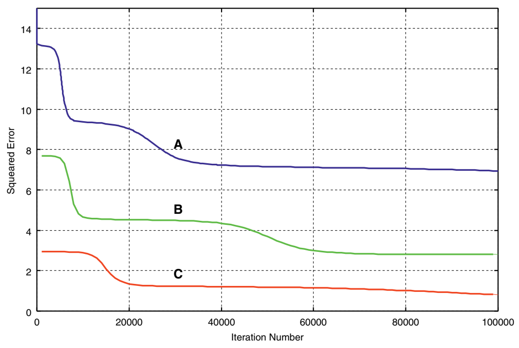
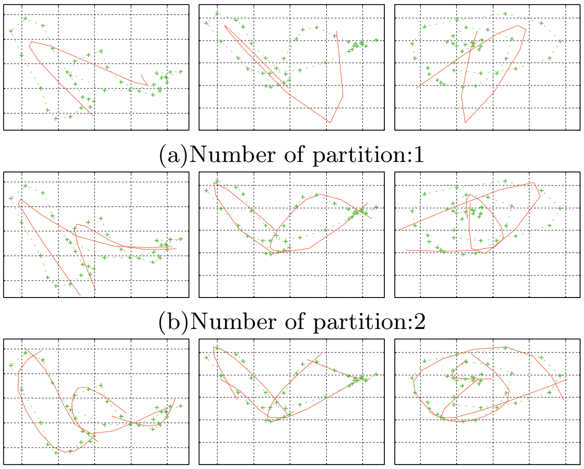


Fig. 7. The relationship between the squared error and the times of learning

the distribution. The feature vector of the rotating object draws a trajectory in the feature space. In order to impove the accuracy and the efficiency of approximating the closed manifold, the learning sample is divided into several groups and fed to separate networks.

Figure 7 shows the relationship between the number of learning and sum of the squeared error. The curve A, B and C in the figure shows the squared error when the learning sample is divided into 1, 2 and 4 groupes, respectively. This figure shows that the more the number of groupes is the less the squared error is.

Figure 8 shows the distribution of the learning sample and the approximating curves obtained by the autoassociative networks. Figure 8(a), (b) and (c) is the result for the case where the learning sample is devided into 1, 2 and 4 groupes, respectively. Each figure shows the three dimensional eignspace represented in the same way as in Fig.6 and its projection to x–y, y–z and z–x planes for visual clarity's sake. The dotted line in the figure is the manifold on which the learning sample is distributed, and the solid line is the approximating curves of the manifold which is learned by the autoassociative networks. Each approximating


(a)Number of partition:1

(b)Number of partition:2

(c)Number of partition:4

Fig. 8. The approximating curves obtained by the autoassociative networks

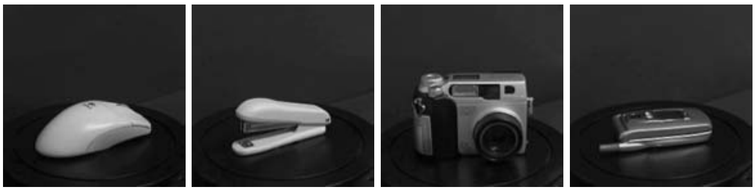

(a)Mouse (b)Stapler (c)Camera (d)Mobilephone

Fig. 9. Four classes of objects used in the recognition experiment

curve is the output of the network when the output of the middle hidden unit is swept from 0 to 1 as described in Sect.2. This figure shows that the more the number of groupes the more accurately the distribution is approximated.

4 Object Recognition by the Autoassociative Networks

To demonstrate the feasibility of the object recognition by the autoassociative networks, recognition experiment of four classes of objects shown in Fig.9 was performed.

Each sample for learning and test consists of 36 gray scale images per class, and were acquired in the same way as described in Sect.3. The learning sample and the test sample is different in the rotating angle by 5 degrees.

Recognition experiments were performed under the following five conditions:

1. Five layered autoassociative network (single segment)
2. Five layered autoassociative network (two segments)
3. Five layered autoassociative network (four segments)
4. Nearest neighbor method
5. Nearest neighbor method in eigenspace

Table 1. The recognition rate of each method

Method	Recognition rate(%)
Five layered autoassociative network (single segment)	97.92
Five layered autoassociative network (two segments)	98.69
Five layered autoassociative network (four segment)	100.00
Nearest neighbor method	100.00
Nearest neighbor method in eigenspace	100.00

Table 2. The result of recognition experiment with smaller learning samples

Method	Number of learning samples		
	36	12	4
Five layered autoassociative networks(four segments)	100.00%	100.00%	100.00%
Nearest neighbor method	100.00%	100.00%	93.06%
Nearest neghbor method in eigenspace	100.00%	99.31%	94.44%

Where "single segment" is a synonym of division into one group, and so on. Table 1 shows the recognition rate of each method.

mmThe five layered autoassociative networks with more segments achieved higher recognition rate. This result is justified by the fact that the closed manifold is more accurately approximated when separate networks are trained by divided learning samples. The nearest neghbor methods achieved 100% recognition rate both in original feature sapce and eigenspace.

Table 2 shows the result of recognition experiment when smaller learning samples were used.

The autoassociative networks keep high recognition rates for smaller sample case while the nearest neghbou methods deteriorate the performance.

5 Concluding Remarks

This paper studied on a new eignespace method which employs autoassociative networks for object recognition. The results of recognition experiments for four classes of objects demonstrated that:

1. the five layered autoassociative networks achieve higher recognition rate when separate networks are trained by divided learning samples because the closed manifold is more accurately approximated, and
2. they outperform the nearest neighbor methods in original space and the eigenspace with smaller learning samples.

Remaining future research topics are shown as follows,

1. Learning and recognition of object images which change with higher degree of freedom,
2. recognition of more classes of objects, and
3. pose estimation of the object

References

1. Murase,H.,Nayar,S.:Visual Learning and Recognition of 3D Objects from Appearance.International Journal of Computer Vision,**14-1**(1995)5–24
2. Hassoun,M.:Fundamentals of Artificial Neural Networks.MIT Press,(1995)
3. Cottrell,G.,Munro,P.,Zipser,D.:Image compression by back-propagation: An example of extensional programing. Models of Cognition: A Review of Cognitive Science,**1**(1989)208–240
4. Cottrell,G.,Munro,P.,Zipser,D.:Learning internal representations from gray-scale images:An example of extensional programing.Ninth Annual Conference of the Cognitive Science Society,(1987)462–473
5. DeMers,D.,Cottell,G.:Non-linear dimensionality reduction. Advances in Neural Information Processing Systems 5,(1992) 550–587
6. Kimura,F.,Inoue,S.,Wakabayashi,T.,Tsuruoka,S.,Miyake,Y.:Handwritten Numeral Recognition using Autoassociative Neural Networks. Proc. 14th International Conference on Pattern Recognition,**1**(1998)166–171

A Novel Constraint-Based Approach to Online Graphics Recognition

Luo Yan, Guanglin Huang, Liu Yin, and Liu Wenyin

Department of Computer Science, City University of Hong Kong
83, Tat Chee Avenue, Kowloon Tong, Kowloon, Hong Kong SAR, China
{luoyan,hwanggl}@cs.cityu.edu.hk, {liuyin,csliuwy}@cityu.edu.hk

Abstract. Online graphics recognition has become the key problem for pen-based user interface on small screen devices, such as PDA and Tablet PC. In this paper, a novel constraint-based approach to online graphics recognition is proposed. The key idea of our approach is that when the user is drawing a graphic object, the system can extract the constraints between primitives and basic shapes from the object and use these constraints to retrieve similar graphic objects from the database at run time. The user can then choose the standard object from the ranked list of results to replace his sketches before he finishes drawing all strokes of the object. For this purpose, we summarize three types of primitives and several types of basic shapes as the basic components of a graphic object. We also define a set of constraints between primitives and basic shapes to represent their structural relations. The algorithms for online constraint extraction and graphics recognition are also presented. Experimental results show that our approach is efficient for online graphics recognition and effective for improving the user's productivity.

1 Introduction

Recently pen-based devices such as PDA and Tablet PC have become more and more common to the general public. In these devices, graphics is an important and useful means for users to store information, express thought, and sketch designs. Many systems were developed to facilitate users to draw graphics, such as Microsoft Visio, SmartDraw, and AutoCAD. In these systems, the user is asked to draw graphics by selecting the particular type of graphic object from lots of toolbar buttons or menu items. This task is very time-consuming and inconvenient, especially when the number of predefined graphic objects in the system is very large. The most convenient and natural way for human beings to draw graphics should be using a pen to draw sketches, just like drawing on a real sheet of paper. However, the sketches drawn in this way are not standard and clear in appearance, not compact in representation and storage, and not easy for machines to understand and process. It is necessary to recognize and convert the sketches to the regular and standard graphic objects that the user intends to draw. Moreover, it is even better if we can do recognition while the user is sketching since the recognized parts can provide immediate and useful feedback to the user so that he can realize errors or inappropriateness earlier and therefore draw the graphics more perfectly. In many cases, recognizing graphic objects early can also significantly save the user's input strokes and time. Hence, online graphics recognition has become the key problem for pen-based user interface on these small screen devices. Moreover, online graphics recognition can be also viewed as a query and retrieval problem. The user's input strokes can be viewed as a query and the system retrieves the similar graphic objects from a number of predefined standard graphic objects. Although the aims of retrieval and recognition are different, the underlying technology is common in that a matching procedure is needed to compare the input pattern with each known pattern. Therefore, the techniques for retrieving online graphics are also within the scope of online graphics recognition. In the following, we will not distinguish retrieval from recognition. The readers should bear in mind the common points and differences between them.

Compared with offline graphics recognition, online graphics recognition has some special characteristics. First, the input graphic object for online graphics recognition is usually incomplete, since our goal is to recognize the user's sketches before he finishes the whole graphic object, which can provide an immediate and useful feedback to the user. This characteristic implies online graphics recognition has to recognize the user intended object based on partial information in many cases. Second, the strokes in the same graphic object can be drawn in different orders by different users. Hence, the incomplete user's input of the same graphic object can be very different for online graphics recognition. That means it is not easy to apply the traditional matching methods for offline graphics recognition to online graphics recognition, since there can be many different kinds of incomplete graphic objects for the same complete one and it is difficult to match all of them to the complete one. Third, online graphics recognition needs more efficiency than offline graphics recognition. The system has to provide the immediate feedback to the user at run time; otherwise, it will be tedious and time-consuming instead of saving the user's input strokes and time. Hence, the efficiency of online graphics recognition is very important for a good user interface.

Many research works have been done on such online graphics recognition. Zeleznik et al. [1] have invented an interface to input 3D sketchy shapes by recognizing the predefined patterns of some 2D graphic objects. Jorge's group [2][3] have implemented an online graphics recognition tool that can recognize several classes of simple shapes based on global area calculation, which can hardly distinguish ambiguous shapes such as pentagon and hexagon and therefore cannot achieve high recognition precision generally. SILK [4] is an informal sketching tool that combines many of the benefits of paper-based sketching with the merits of current electronic tools. JavaSketchIt [5] is another system for this purpose, which can generate a Java interface from hand-drawn geometric shapes. SKETCHIT [6] is a system that can transform a single sketch of a mechanical device into multiple families of new designs. LADDER [7] is a language to describe how sketched diagrams in a domain are drawn, displayed, and edited, and used for online graphics recognition. The recognition approach is still not adequate for a real software tool that can be used for inputting most classes of diagrams. Hence, in order to provide the capability to input more complex diagrams, it is necessary to extend the online graphics recognition approach to handle more complex and composite shapes, as done in SmartSketchpad [8], which can efficiently and effectively input composite graphic objects by sketching only a few constituent strokes.

2 Our Approach and Contribution

In this paper, we propose a novel constraint-based approach to online graphics recognition. The key idea of our approach is that when the user is drawing a graphic object, the system can extract the constraints between primitives and basic shapes from the object and use these constraints to retrieve or recognize similar standard graphic objects from the database at run time. The user can then choose the standard object from the ranked list of results to replace his sketches before he finishes drawing all strokes of the object.

Our contribution includes, 1) we summarized three types of primitives and several types of basic shapes; 2) we defined a set of constraints between primitives and basic shapes to represent their structural relations; 3) we developed an algorithm for online constraint extraction from the user's input graphic object, which is incomplete in many cases; 4) we developed another algorithm for online graphics recognition based on the constraints of the user's input graphic object; 5) we proposed an algorithm for calculating the similarity between the user's input graphic object and the candidate graphic objects for displaying the recognized results in a ranked list.

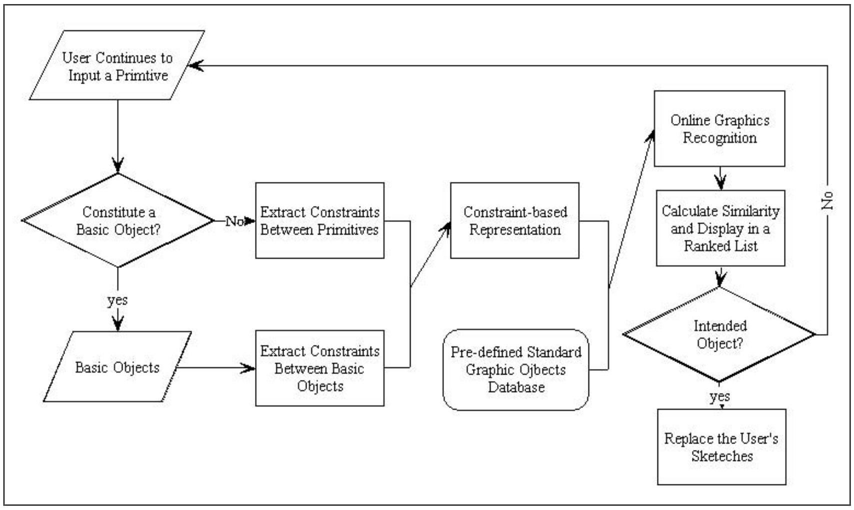

User Continues to Input a Primtive
Constitute a Basic Object?
No
yes
Basic Objects
Extract Constraints Between Primitives
Extract Constraints Between Basic Objects
Constraint-based Representation
Pre-defined Standard Graphic Ojbects Database
Online Graphics Recognition
Calculate Similarity and Display in a Ranked List
Intended Object?
No
yes
Replace the User's Sketeches

Fig. 1. The flowchart of our approach

Figure 1 is the flowchart of our approach. The user begins his sketches by drawing some basic strokes (or primitives). The system starts to extract the constraints between these primitives and uses the extracted constraints to recognize the similar standard graphic objects in the database. By using our proposed similarity calculation algorithm the system can then calculate the similarity between the user's input graphic objects and the candidate graphic objects, and display the recognized results in a ranked list. If the user's intended graphic object is displayed in the list, he can just choose this standard object to replace that incomplete sketches he has just drawn. The system applies these procedures, such as constraint extraction, graphics recognition, and similarity calculation, at the same time as the user is drawing the sketches. Hence it can facilitate the user to draw graphics by significantly saving the user's input strokes and time.

In the following of this paper, we first propose our constraint-based approach to describe the user's input graphic object in Section 3. Then, algorithms for constraint extraction and graphics recognition are discussed in Section 4 and 5, respectively. Finally, experimental results and concluding remarks are presented.

3 Constraint-Based Representation of Graphic Objects

As we discussed above, our approach focuses on the relative spatial relations between primitives and basic shapes. Hence, we use constraints to represent the user's input graphic object in our approach. Constraint, or geometric constraint, is not a new concept, which has been widely used in CAD systems (e.g., [9]). However, in many CAD systems (e.g., [9][10][11]), the constraints are defined, extracted, and specified by professional and experienced users. In our approach, we defined a set of constraints to describe the spatial relations between primitives and basic shapes. The system can extract constraints while the user is drawing the sketches and uses these constraints to recognize similar standard graphic objects in the database at run time. Thus, our definition of constraints should be broad enough to support a wide range of graphic objects, while remaining narrow enough to be comprehensible.

First of all, we define three types of primitives: Line, Circle, and Arc. As shown in Figure 2, P_1 and P_2 are two endpoints of a Line. We can assume P_1 is the start-point and P_2 is the end-point such that we can define the direction of a Line is from P_1 to P_2. For a Circle primitive, it also has two attributes, C (center-point) and R (radius). In the definition of an Arc, we use P_1 and P_2 to represent the start-point and end-point of an Arc since the user usually pays more attention to the start-point and end-point than the center-point. That means the user does not care about the curving of an Arc but the position of an Arc. However, the direction of the bow of an Arc is very important for the user to distinguish different graphic objects. Hence, if we define a positive direction from P_1 to P_2, like X-axis, then we can define the Direction of the bow of an Arc.

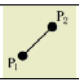
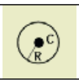
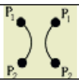
		
Line	**Circle**	**Arc**

Fig. 2. Primitives

Then we define the constraints between the above primitives. We analyzed more than 300 types of graphic objects to summarize the constraints. Since we only use three parameters (i.e., P_1, P_2, and Direction) to define an Arc primitive, we can image an Arc as a Line plus a Direction. Hence, we can just analyze the constraints between Line primitives and apply these constraints to Arc primitives by simply adding a Direction parameter. Therefore, we first define four constraints between Line primitives and Arc Primitives, including Connection, Intersection, Parallelism, and Perpendicularity. For a Circle primitive, we regard it as a basic shape, which is discussed in the following section, and define the constraints between basic shapes and primitives to describe their spatial relations. Here, for easily understanding, we only use Line primitives to describe the four constraints between Line and Arc primitives. For the cases including Arc primitives, only one additional parameter, Direction, is required.

(1) Connection

Connection is a constraint to describe that two primitives share the same end-point, just like they are connected at one end. Figure 3 illustrates this constraint.

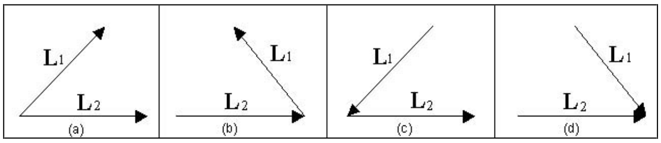


Fig. 3. Connection

From the above figure, we can see that there are only four cases between two primitives that are connected with each other, since one Line or one Arc has two end-points. We use a parameter *type* to represent this information and use another parameter *angle* to store the angle between the two primitives.

$$angle = \cos(\alpha) = L_1 \bullet L_2 / (|L_1||L_2|)$$

In this definition, the parameter *angle* itself is not sufficient to fully specify the spatial relationship of two intersected lines since the angle has a direction. Thus, we use another parameter *direction* to describe this information. Consider $L_1(x_1,y_1,0)$ and $L_2(x_2,y_2,0)$, which are 2D vectors in 3D space, and their cross product

$$L = L_1 \times L_2 = \left(\begin{vmatrix} y_1 & 0 \\ y_2 & 0 \end{vmatrix}, \begin{vmatrix} 0 & x_1 \\ 0 & x_2 \end{vmatrix}, \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} \right) = (0,0,L_z) \quad L_z = \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} = x_1 y_2 - x_2 y_1$$

L is perpendicular to the plane formed by L_1 and L_2, and its direction complies with the *Right Hand Rule*. Thus we can determine the direction by the sign of L_z. In addition, we use the parameter *length* to describe the relative length of L_2 to L_1 (*length* = $|L_2|/|L_1|$).

(2) Intersection

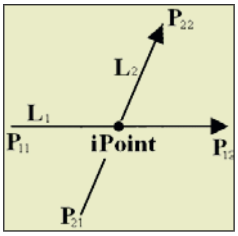


Fig. 4. Intersection

Intersection is a constraint to describe that two primitives are intersected with each other, which means they share the common point on the primitives.

In Figure 4, two Line primitives are intersected with each other at iPoint. We define four parameters to describe this constraint. The first two parameters describe the relative position of iPoint on two Line primitives as follows.

$$\text{iP}_1 = \frac{\text{iPoint} - P_{12}}{P_{11} - P_{12}} \qquad \text{iP}_2 = \frac{\text{iPoint} - P_{22}}{P_{21} - P_{22}}$$

We use other two parameters, *angle* and *length*, to describe the angle between two primitives and relative length of them just like Connection constraint.

(3) Parallelism

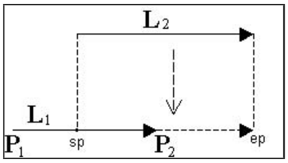


Fig. 5. Parallelism

Similar to Intersection, we also use four parameters to describe Parallelism geometric constraint. The first one is *distance* = $D(L_1,L_2)/|L_1|$, in which $D(L_1,L_2)$ denotes the real distance between line L_1 and L_2. The second one, *direction*, is used to describe whether L_2 is on the left or right to L_1 and the computing method is similar to the definition in Connection constraint. Moreover, we use two other parameters to specify their relative position and length. In Figure 5, L_1 and L_2 are parallel to each other; *sp* and *ep* are the projections of the endpoints of L_2 on L_1. We set:

$$\text{start-point} = \frac{\text{sp} - L_1.P_2}{L_1.P_1 - L_1.P_2} \quad \text{end-point} = \frac{\text{ep} - L_1.P_2}{L_1.P_1 - L_1.P_2}$$

(4) Perpendicularity

For the Perpendicularity relationship in which two primitives are connected or intersected, we can use Connection or Intersection to represent it, respectively. Here, we only define the Perpendicularity between two primitives when they are not connected or intersected:

- Length $= |L_2| / |L_1|$
- Per-point is the perpendicular point of L_2 on L_1
- Start-point $= |\text{per-point}, L_2.P_1| / |L_2|$
- End-point $= |\text{per-point}, L_2.P_2| / |L_2|$

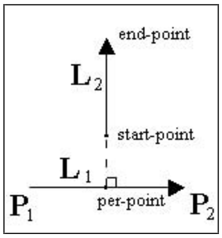


Fig. 6. Perpendicularity

When we calculate start-point and end-point, we set a sign to the value of them. We set it positive if the point is on the left-hand side of L_1 and negative on the right-hand side. The computing method is similar to computing direction in Connection constraint. In Figure 6, the values of start-point and end-point are both positive.

Some primitives can constitute a very common and basic shape, which is often used by users in many complex graphic objects. Especially, the user usually divides the whole sketch into some basic shapes when drawing a complex sketch. Therefore, we also summarized some basic shapes to represent the user's input graphic object at a higher level, as illustrated in Figure 7.

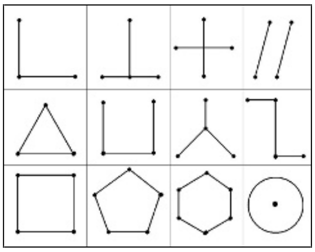


Fig. 7. Some basic shapes

For these basic shapes, we also define a set of constraints to describe the structural relations between them. For instance, to the closed shapes, such as Rectangle and Circle, we defined the Inner/Outer constraint to describe whether other primitives or basic shapes are inside or outside them, because, in many cases, the user pays more attention to the Inner/Outer relations between shapes than the precise position or orientation of these shapes. For other non-closed shapes, we also defined other constraints (e.g., relative position and orientation) to describe the structural relations between these basic shapes and other primitives.

4 Online Constraint Extraction

In this section, we discuss our developed algorithm for online constraint extraction, which means that our approach extracts the constraints between the primitives and basic shapes while the user is drawing sketches. This algorithm is developed based on our previous work for offline graphics recognition [12]. We divide the procedure of recognizing user's drawing sketches into three stages.

1. The user begins his sketches with simple primitives, which do not constitute any basic shapes. However, the simple primitives do contain useful information about the user's intention, e.g., they can be a part of a standard graphic object. Hence, our algorithm extracts the constraints between the primitives as the representation of user's input at this stage and uses these constraints to retrieve the standard graphic objects that contain the similar part.
2. When the user continues to draw sketches, there are enough primitives to constitute a basic shape. At this stage, our approach uses the constraints between the primitives to recognize them as a basic shape and provides a useful and immediate feedback to the user. The user can accept the feedback or adjust his sketches at this stage. Once the user accepts his current sketches as a basic shape, his sketches are replaced by the standard basic shape and he can go on with his sketches. The system will then extract the constraints between the newly drawn primitives until a new basic shape is recognized.
3. As the user goes on with his sketches, the constraints between the basic shapes should also be extracted since they contain much useful information for recognition. Hence, at the third stage, the system extracts the constraints between basic shapes and constructs a hierarchical constraint-based structure for recognition.

For the detail of the online constraint extraction algorithm, see the Case-based Knowledge Acquisition Algorithm (CKAA) [12].

5 Online Graphics Recognition

The constraints extracted by the above algorithm are stored in a syntactical tree. We use this tree to retrieve or recognize the similar standard graphic objects. We search all the predefined graphic objects in the database for those that contain the similar constraints, i.e., contain the similar graphic object to user's input. However, we cannot use the matching method for recognition since the user's input is usually incomplete. Therefore, we propose a new scheme, which is like a reasoning method, for recognizing graphic objects based on the constraints. When we test one standard graphic object for whether it contains the similar graphic object to the user's input or not, we first hypothesize that one stroke of the standard graphic object is in the user's input. Using the constraints extracted from the user's input, we can calculate the specification of another primitive or basic shape based on the hypothesis stroke. Then we search the standard graphic object to see whether it contains this stroke. If the stroke is found, we continue tracing other constraints until all strokes are found in the standard graphic object, which means, this standard graphic object contains the similar graphic object to the user's input. Otherwise, we select another stroke to repeat this hypothesizing/testing procedure. The algorithm presented below deals with ideal situations. In practice, the tolerance should be considered and the matching measure should be defined, which are discussed in our previous work [12]. The detail of the online graphics recognition algorithm is shown in Algorithm 1.

When the result is output, the similarity between the user's input graphic object and the standard graphic object is calculated from two aspects. The first is the similarity between primitives, which is calculated according to the difference of length, angle and position between the two primitives. The second is the similarity of constraints, which is calculated by the percentage of exact matched primitives in the standard graphic object. According to the similarity of the standard graphic objects, we select top 10 objects in the database and return them in a ranked list to the user.

```
Algorithm 1: Online Graphics Recognition
Input:      SC:  the set of constraints from the user's input graphic object
            DB:  the database consists of standard graphic objects
            TL:  the tolerances, e.g., length and number tolerance
Variables:  CT:  the temporary constructed tree for reasoning procedure
            SM:  the set of marks to indicate primitives that have been tested
Output:     RR:  the recognition result, which type the graphic object is
 1.  Select a standard graphic object SG from DB. If all standard graphic objects have
     been searched, then stop (failure)
 2.  Set CT empty and initialize SM
 3.  Select the next primitive P from SG, which has not been marked in SM. Add it into
     CT as the root, and mark it in SM to indicate this primitive has been tested. If all
     primitives have been marked in SM, goto step 1.
 4.  Select the next constraint C from SC. If all constraints have been traced then stop
     (success) and output the current SG as RR
 5.  Calculate the new primitive or basic shape P' using P and C
 6.  Search for a P'' in SG, which is similar to P' using the tolerances in TL.
 7.  If P'' is found then set it as a child of P in CT and mark in SM to indicate P'' has
     been used and goto Step 4
 8.  If P'' is not found and the number of missing primitives exceeds the tolerance then
     goto Step 2. Otherwise, goto Step 4
```

6 Experimental Results

We have implemented a prototype system and done several experiments based on a database consisting of 345 standard graphic objects, some of which are illustrated in Figure 8. The user is asked to draw graphic objects and the system provides immediate recognition results, from which the user can select his intended standard graphic object. The average recognition accuracy is 90.5% since the user's input can be very different. We also record the number of strokes that have been saved for drawing an object. In our experiments, the number of one standard object's strokes ranges from 1 to 14 and the average is 10.32. The average number of saved strokes is 2.78, nearly 27%. We also evaluate the response time of our approach. The average response time to user's

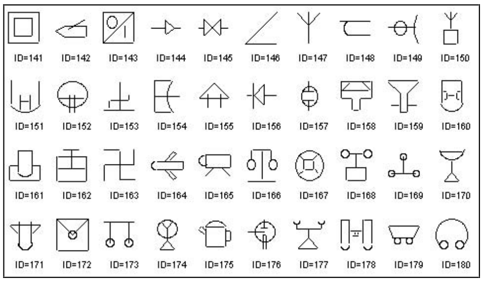


Fig. 8. Some standard graphic objects

input is within 100ms, which is efficient enough to give real-time response for a database consisting of several hundreds of graphic objects. From the experimental results, we can see that our approach is effective for online graphics recognition and saving the user's input strokes and time.

7 Conclusion and Future Work

In this paper, we proposed a novel constraint-based approach to online graphics recognition, with which the system can extract the constraints between primitives and basic shapes from the user's input and use these constraints to recognize similar standard graphic objects. Several constraints are defined and two algorithms are developed. Experimental results show that our approach is efficient for online graphics recognition and effective for saving the user's input strokes and time. However, some aspects of our approach can be improved. More types of primitives, basic shapes, and constraints can be added into our approach in the future to support more complex and various graphic objects. Two algorithms for online constraint extraction and graphics recognition can be also revised to improve the recognition accuracy and save the user's input stroke and time. We also plan to provide more graphic objects from various domains to do experiments to test our system.

Acknowledgement

The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong SAR, China [Project No. CityU 1073/02E].

References

1. R.C. Zeleznik, K.P. Herndon, and J.F. Hughes, "KETCH: An Interface for Sketching 3D Scenes", *Proc. of SIGGRAPH*, New Orleans, pp.163-170, 1996.
2. M.J. Fonseca and J.A. Jorge, "Using Fuzzy Logic to Recognize Geometric Shapes Interactively", *Proc. of the 9th IEEE Conf. on Fuzzy Systems*, Vol.1, pp.291-296, 2000.
3. M.J. Fonseca, C. Pimentel, J.A. Jorge, "An Online Scribble Recognizer for Calligraphic Interfaces", *Proc. of AAAI Spring Symposium Series – Sketch Understanding*, 2002.
4. J.M. Landay and B.A. Myers, "Sketching Interfaces: Toward More Human Interface Design", *IEEE Computer*, Vol. 34, No. 3, pp. 56-64, 2001.
5. A. Caetano, N. Goulart, M.J. Fonseca, and J.A. Jorge, "JavaSketchIt: Issues in Sketching the Look of User Interfaces", *Proc. AAAI'02 Spring Symposium – Sketch Understanding*.
6. C. Calhoun, T.F. Stahovich, T. Kurtoglu, L.M. Kara, "Recognizing Multi-Stroke Symbols", *Proc. of AAAI Sprint Symposium Series – Sketch Understanding*, 2002.
7. T. Hammond and R. Davis, "Ladder: A Language to Describe Drawing, Display, and Editing in Sketch Recognition", *Proc. of IJCAI'03*, 2003.
8. W. Liu, X. Jin, and Z. Sun, "Sketch-Based User Interface for Inputting Graphic Objects on Small Screen Devices", *Lecture Notes in Computer Science* 2390, pp.67-80, 2002.

9. J.K. Lee and K. Kim, “Geometric Reasoning for Knowledge-based Parametric Design using Graph Representation"”, *Computer-Aided Design*, Vol.28, No.10, pp.831-841, 1996.
10. S. Ait-Aoudia, B. Hamid, A. Moussaoui, T. Saadi, “Solving Geometric Constraints by a Graph-Constructive Approach”, *Proc. of ICIV’1999*, pp.250-255, 1999.
11. I. Fudos, C.M. Hoffmann, “A Graph-Constructive Approach to Solving Systems of Geometric Constraints”, *ACM Trans. on Graphics*, Vol.16, No.2, pp.179-216, 1997.
12. Y. Luo and W. Liu, “A Case-based Interactive Approach to Graphics Recognition in Engineering Drawings”, *Proc. of GREC’2003,* pp.170-181, 2003.

Extraction of Skeletal Shape Features Using a Visual Attention Operator

Roman M. Palenichka, Rokia Missaoui, and Marek B. Zaremba

Dept. of Computer Science and Engineering
Université du Québec, Gatineau, Québec, Canada
{palenich,missaoui,zaremba}@uqo.ca

Abstract. The goal of the shape extraction method presented in this paper was to obtain a concise, robust, and invariant description of planar object shapes for object detection and identification purposes. The solution of this problem was chosen in the form of a piecewise-linear skeleton representation of local shapes in a limited number of salient object locations. A visual attention operator, which can measure the saliency level of image fragments, selects a set of most salient object locations for concise shape description. The proposed operator, called image relevance function, is a multi-scale non-linear matched filter, which takes local maxima at centers of locations of the objects of interest. This attention operator allows a simple extraction of vertices for the skeletal shape description by local maxima analysis.

1 Introduction

In a variety of image analysis tasks related to fast object detection and identification (verification), the main concern is adequate and concise representation of the object shape [1, 2]. The approach based on shape skeletons is, in the context of this application, efficient since it can represent in a very concise manner the topology of an object with several connected parts and shape details [3-6]. Such a description permits a complete morphological reconstruction of the planar shape provided local scale values (i.e., diameter values) are available in each skeleton point.

The classical skeletonization algorithms such as those based on an iterative (morphological) thinning and distance transformation provide the skeletal shape description but they are not robust to various shape distortions and noise [3, 4]. These methods are usually limited to process only binary images. Some multi-scale algorithmic generalizations to gray-scale images and three-dimensional (volume) images are also proposed [5-7]. Their performance strongly depends on the knowledge of some additional parameters, which are sensitive to distortions and irregularities. Complete skeletal shape is usually redundant to describe shape in the majority of object detection applications [1, 2]. Consequently, such a representation creates difficulties when comparing skeletal shapes, especially in the case of present noise and shape distortions.

More recently, several methods were developed to describe skeletal shapes in a piecewise-linear manner by skeleton vertices and their interconnections in the form of straight-line segments [8-10]. This is a concise representation of skeletal shapes without using classical skeletonization algorithms. For example, a statistical method of principal curves was used to extract directly the skeletal description of point sets [8, 9]. The algorithms for drawing principal curves using piecewise-linear approximation are, in their initial form, limited to simple curves or manifolds, where, for example, no intersections are allowed. Another kind of piecewise-linear skeletonization algorithms are based on unsupervised neural network methods, such as those based on self-organizing maps [10]. The shape skeleton can be obtained from a data-driven minimal spanning tree topology of a self-organizing map. The method is quite robust against sparse shapes and distortions but limited to process binary images and it deteriorates significantly if the segmented object contains components of various local sizes.

In this paper, we suggest a novel approach to skeletal shape description of gray-scale images based on the determination of salient object locations (i.e., interest points) and skeletal shape description relatively to the extracted locations. The whole object shape is described in terms of such local skeletal shapes and their relative positions and connectivity patterns. At the same time, this approach is an adaptation of the skeletal shape description for the case of object detection (localization) and identification in gray-scale images, without using an explicit image binarization. The development of the shape extraction method has the following objectives.

- Concise skeletal shape description by feature extraction in a selected number of salient object locations only.
- A simple distance (e.g., Euclidean distance) between shape feature vectors can compare two different shapes without computationally costly shape alignments.
- The shape features have to be invariant to geometrical transformations such as translation, scaling, and rotation.
- The method can process gray-scale images and have to be robust against noise and some local occlusions provided they do not occlude salient locations.

Salient object locations can be determined by the attention focusing approach, which was initially proposed to perform time-efficient search for objects of interest [11-13]. The underlying idea consists in focusing attention on the most salient image fragments or objects of interest, which are stable to intensity changes and shape geometrical transformations and can capture well the overall object shape. This is a biologically inspired approach that models basic elements of the visual perception and fast visual search in humans and animals. Given salient fragments, a complex object shape can be represented in terms of local shapes of these fragments and their relative positions on the image plane.

The determination of salient locations and shape feature extraction are both implemented by one visual attention operator called *image relevance function* (IRF) [14]. This operator is a multi-scale non-linear matched filter, which measures the saliency level of image locations and takes local maxima at the centers of locations of the objects of interest. The feature vector in each salient location includes local planar shape features and geometry features (i.e., parameters of affine transformations) such as relative position, local scale (size), and local orientation.

2 Morphological Modeling of Skeletal Shape Features

The proposed IRF method provides a description skeletal object shape in the form a set of most salient object locations each of them being described as a shape feature vector. The salient locations can be connected to each other provided connectivity conditions between two locations are fulfilled. The connected salient locations describe the whole (global) shape of a connected object. An image may contain many such connected objects each of them described by the local skeletal shape at salient locations. Moreover, each salient location can contain intensity (color) and texture features related to that location for object identification purposes using both local shape features and local intensity (texture) features.

In the IRF framework, planar shape features are separated and are independent from intensity features. Such a separation has a certain advantage over the integrated shape features extraction (e.g., features based on differentiation with Gaussian smoothing [11, 15]) because of the achieved invariance to transformation of translation, scaling and rotation and some intensity changes and lighting conditions. Additionally, a few intensity and texture features can be used for object intensity description to represent intensity variations as a texture, especially in the case of large scales (sizes of object regions).

For the purpose of multi-scale image analysis, a formal definition of a scale system is used [14]: a structuring element at scale n of a *uniform scales system* is formed by the morphological dilation (denoted by $\oplus$) by S_0, $S_n = S_{n-1} \oplus S_0$, n=1, 2,...,M-1, where M is the total number of scales and the structuring element S_0 defines the minimal scale and object resolution. The structuring elements have the same shape such as the disk shape (see examples in Fig. 1). The above is a morphological definition of scales, which is different from the notion of scales in the scale-space filtering [11].

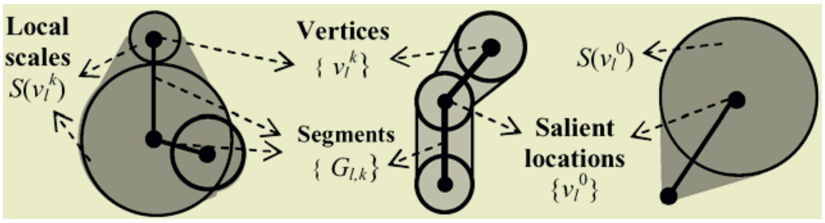


Fig. 1. Examples of local skeletal shape features.

We have proposed a piecewise-linear local skeletal description of planar shapes related to salient object locations. This skeletal shape representation is an economical approach to shape description. An object local shape is related to a particular salient location v_l^0 and the local scale value at that location, $S(v_l^0)$ (see Fig. 1). Given K vertices and K scale values associated with each vertex, the local planar shape (as a support region U) of an object of interest located at v_l^0 is formed by the dilation operations of skeleton straight-line segments $\{G_{l,k}\}$ with size-variable structuring elements, $\{S(G_{l,k})\}$:

$$U(v_l^0) = \bigcup_{k=1}^{K} G_{l,k} \oplus S(G_{l,k}) = \bigcup_{k=1}^{K} \bigcup_{v_l^m \in G_{l,k}} S(v_l^m), \tag{2.1}$$

where $\oplus$ denotes the morphological dilation, $S(v_l^m)$ is a structuring element with variable size (e.g., diameter r_m) as a function of point $v_l^m \in G_{l,k}$., and K is the maximal topological order of the skeleton vertices. The value of diameter r_m is a linear combination of the scale sizes r_0 and r_k at terminal vertices v_l^0 and v_l^k of segment $G_{l,k}$. Equation (2.1) represents a method of *scale-interpolated dilation* in the piecewise-linear modeling of skeletal shapes and

$$r_m = \frac{d(v_l^k, v_l^m)}{d(v_l^k, v_l^0)} \cdot r_0 + \frac{d(v_l^0, v_l^m)}{d(v_l^k, v_l^0)} \cdot r_k, \tag{2.2}$$

where $d(.,.)$ is the Euclidean distance between two skeleton vertices on the image plane.

The whole planar shape of a multi-scale object of interest is formed by pair-wise concatenations of the local shapes at L vertices $\{v_l^0, l=1,...,L\}$ if the connectivity between the corresponding vertices can be established. For each l, the local skeleton vertices $\{v_l^k, k=1,...,K\}$ can be considered as shape details at that vertex, i.e., respective salient location.

This model of planar local shape is associated with an intensity model of image fragment of size $2r_l$ centered at v_l^0. The intensity modeling involves two dominant intensity levels with an additive noise model, which can also represent a textured intensity, in order to descrobe image intensity locally and concisely [14].

3 Determination of Salient Locations Using Image Relevance Function

Each salient object location is associated with its own salient fragment centered at a particular local maximum of the IRF. The IRF is defined generically as an image operator, which takes local maximal values at centers of salient image fragments and can be used to describe objects of interest in the salient locations. At certain conditions, the IRF maximums are positioned on object medial axes or at the centers of its parts, which are relevant to shape description (see Fig. 2). In order to address the aforementioned problems of skeletal shape extraction for object detection it is suggested to apply an improved version of a model-based IRF described in the context of object detection [14]. Localization of salient image fragments is based on a fast computation of the multi-scale IRF and determination of its local maxima. The positions of local maximum values of the multi-scale IRF coincide with location points of the salient image objects in a region of interest A:

$$(i_f, j_f)_l = \max_{(i,j)\in A} \max_k \{\Phi[(g(i,j), S_k], (i,j) \notin \Gamma_l\}, \tag{3.1}$$

where $g(i,j)$ is the input gray-scale image, $\Phi[g(i,j),S_k]$ is a non-linear matched filter at kth scale, and $(i_f,j_f)_l$ are two coordinates of lth maximum. The region $\Gamma_l \subset A$ corresponds to the masking region, which excludes determined maximum points from further analysis.

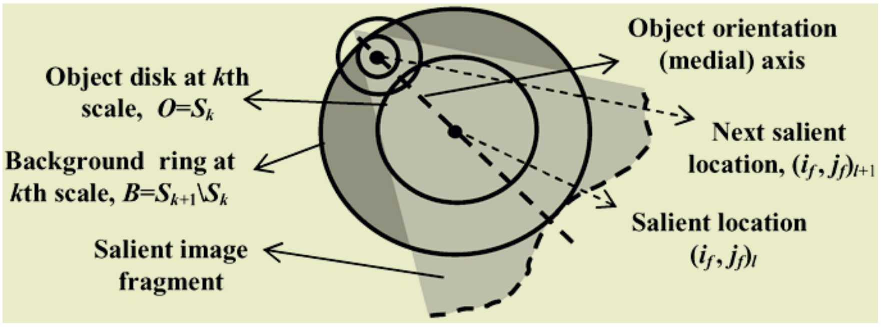

Object orientation (medial) axis
Object disk at kth scale, $O=S_k$
Next salient location, $(i_f, j_f)_{l+1}$
Background ring at kth scale, $B=S_{k+1}\backslash S_k$
Salient location $(i_f, j_f)_l$
Salient image fragment

Fig. 2. Illustration of the relevance function computation for the single-scale case.

Four saliency conditions are considered in the design of $\Phi(g(i,j),S_k)$: 1) significant local contrast; 2) local homogeneity of object intensity; 3) specific object intensity range; 4) specific range of object sizes and shape of the scales $\{S_k\}$. The first condition is described by the absolute value for the local object-to-background contrast. The local homogeneity condition means that the intensity variance is relatively small in the object region. The intensity range means specific values for the object intensity in order to distinguish it from the background or other objects. Since the measures for contrast, homogeneity and intensity range involve object disk regions and background ring regions of a particular range, the IRF will take implicitly into account shape and scale constraints (condition) of the objects. Taken these conditions, the IRF can be computed in point (i,j), at scale S_k as follows:

$$\Phi[g(i,j),S_k] = c^2(i,j,S_k) - \alpha \cdot d^2(i,j,S_k) - \beta \cdot e^2(i,j,S_k), \qquad (3.2)$$

where $c(i,j,S_k)$ is an estimate for the local contrast, $d(i,j,S_k)$ is an estimate for intensity deviation in the object region, $e(i,j,S_k)$ is the object intensity shift, α and β are constraint coefficients which control the contributions of the two constraints to the overall value of IRF. An estimate of the optimal value of α and β in the sense of the maximum likelihood decision can be computed assuming some distributions (e.g., Gaussian functions with different parameters) for the three variables in Eq.(3.2) under the condition of object presence in point (i,j). The constraint coefficients are inversely proportional to the variances of two constraints in the case of Gaussian distributions: $\alpha = \sigma_c^2 / \sigma_d^2$ and $\beta = \sigma_c^2 / \sigma_e^2$. For example, the contrast estimate $c(i,j,S_k)$ is the intensity difference,

$$c(i,j,S_k) = f_1(i,j,S_k) - f_0(i,j,Q_k), \qquad (3.3)$$

where $Q_k = S_{k+1}/S_k$, is the background estimation region at scale k, i.e., a ring around the disk S_k. $f_1(i,j,S_k)$ and $f_0(i,j,Q_k)$ are the mean values of $g(i,j)$ in regions S_k and Q_k, respectively (see Fig. 2). The mean square deviation was used for the estimation of $d(i,j,S_k)$ in Eq.(3.2). The object intensity shift is measured as a deviation of the mean intensity value $f_1(i,j,S_k)$ from an object intensity of reference.

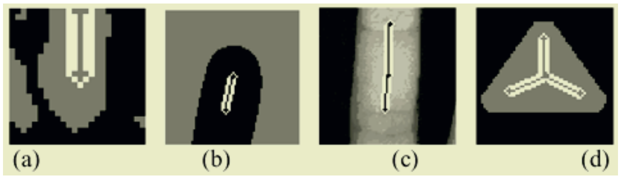


Fig. 3. Examples of local shape features (piecewise-linear skeletons) extracted at salient image locations.

4 Extraction of Skeletal Shape Features

A so-called *saliency hypothesis* is tested first in each local maximum point before the shape feature extraction. It consists of comparisons of local contrast and local homogeneity with saliency thresholds [14].

The extraction of skeletal shape features uses mostly intermediate results of IRF computation, (Eq. 3.2), and is computationally insignificant as compared to the IRF calculation. The invariance parameters for the considered geometrical transformations (translation, scaling, and rotation) are computed with respect to the current local maximum of the IRF. The first parameter is the absolute position of the lth salient location, v_l^0, consisting of two coordinates $(i_f,j_f)_l$. The next two parameters, local scale and local orientation, which are related to point $(i_f,j_f)_l$, are estimated using intermediate results of the IRF calculation (see Fig. 2). The local scale is determined by the contrast maximization,

$$\rho(i_f, j_f) = \arg\max_k \left\{ c^2(i_f, j_f, S_k) - \alpha \cdot d^2(i_f, j_f, S_k) \right\} \qquad (4.1)$$

where the variables and the constant coefficient α have the same meanings as in Eq.(3.1) and Eq.(3.2). Object orientation can be estimated in a simple way since the next maximum point $(i_f,j_f)_{l+1}$ in the current region of attention with respect to the focus of attention $(i_f,j_f)_l$ provides the orientation vector (see Fig. 2).

The proposed IRF approach provides at the same time a simple method to determine vertices for the piecewise-linear skeletal representation of object local shapes in salient locations. This can be done by analysing consecutive K maximums of IRF next to a given salient location v_l^0. Such a procedure determines K local skeleton vertices $\{v_l^k, k=1,\ldots,K\}$, which all are connected to vertex v_l^0 according to the morphological model in Eq. (2.1). Given a neighborhood region $B(v_l^0)$ around vertex v_l^0, the algorithm for the local shape feature extraction is as follows if starting from k=1.

Step 1. Determine location v_l^k of the *k*th local maximum of IRF in the masked neighborhood region $B(v_l^0)$, non-including previous (k-1) IRF maxima.

Step 2. Test the saliency hypothesis with respect to *k*th local maximum point. If the testing outcome is positive then go to *Step* 3, otherwise go to *Step* 4.

Step 3. Determine relative scale and relative orientation associated with v_l^k. Attach these values to the shape vector at positions $2k$ and $(2k+1)$. If $k<K$ then increment $k=k+1$, and mask the neighborhood of vertex v_1^k and go to Step 1, otherwise go to *Step* 4.

Step 4. Attach two coordinates of the vertex v_l^0, its local scale and orientation at the first four positions of the local shape vector. If $k<K$ then the remaining $2\cdot(K-k)$ shape features are set to zero.

The accuracy of shape features determined by this algorithm depends on the correspondence of processed images to the underlying model. Examples of detected salient image fragments in real images with superimposed skeleton fragments are shown in Fig. 3. For the purpose of object verification, intensity and texture features can also be attached to the shape feature vector. They may include object mean intensity, color intensity components, local contrast, and local object variance. The texture features can be used as well in order to describe concisely intensity fluctuations for large scales (object sizes) depending on the application.

Determined salient locations, $\{v_l^0, l=1,...,L\}$, with extracted shape features can be connected to each other if the connectivity between them can be established. This will provide a complete and two-level skeletal shape description of the objects of interest. We were looking for the connectivity in the form of a *spanning tree* (*forest*) constructed in an optimal way. The connectivity algorithm is based on the Markov random chain models of vertex connectivity and finding optimal connectivity pattern between all the salient location vertices $\{v_l^0, l=1,...,L\}$ by the maximization of joint probability $P_c(v_1^0,\ldots,v_L^0)$ of vertex connectivity:

$$P_c(v_1^0,...,v_L^0) = \prod_{v_s^0, v_s^0, v_t^0} P_c(v_q^0 \wedge v_s^0 / v_s^0 \wedge v_t^0) \quad \Rightarrow \quad \max_{v_s^0, v_s^0, v_t^0}, \tag{4.2}$$

where $P_c(v_q^0 \wedge v_s^0 / v_s^0 \wedge v_t^0)$ is the probability that skeleton vertices v_q^0 and v_s^0 are connected with each other provided the precedent vertex v_t^0 in the spanning tree is connected to v_s^0.

The probabilistic connectivity framework was chosen because it provides an optimal solution in the case of gray-scale images, when object shape is sparse with noise presence and possible local occlusions. At certain model assumptions (or particular likelihood functions) this problem can be reduced to the problem of finding minimum spanning tree for a set of extracted vertices, $\{v_l^0, l=1,...,L\}$. The likelihood function, $P_c(v_q^0 \wedge v_s^0, a_{q,s}, d_{q,s} / v_s^0 \wedge v_t^0)$, which is involved in the probability model in Eq. (4.2), uses intensity values $\{a_{q,s}\}$ and distances $\{d_{q,s}\}$ between the vertices as the connectivity variables. The complete description of this algorithm is out of scope of this paper and some details for global skeletal shape extraction can be found in Ref. [10, 16].

Table 1. Measured accuracy of shape feature extraction (error given in pixel resolution) versus contrast-to-noise ratio.

Contrast-to-noise ratio	2	4	8	16	32
Localization error	1.4	1.1	0.7	0	0
Scale error	3.2	2.9	0.9	0.1	0
Orientation error	2.4	1.2	0.3	0.4	0

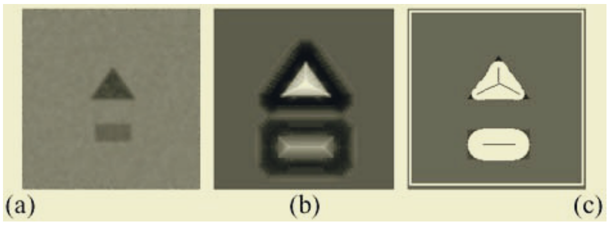


Fig. 4. Illustrative example of IRF calculation (b) and local shape extraction (c) using six most salient IRF maximum points as applied to initial synthetic image in (a).

5 Experimental Results

The first kind of experiments was the performance testing of the IRF approach to shape feature extraction based on synthetic images with known values of the shape features. The position (two coordinates), scale, and orientation values of the salient locations determined by the proposed IRF have been measured and compared with the reference values to determine the accuracy. Image noise has been imitated in synthetic images in order to calculate the accuracy as a function of ratio of the object-to-background contrast and noise magnitude (standard deviation). An example of used synthetic image objects with known shape features and added noise is shown in Fig. 4. The results of accuracy testing are given in Table 1. The error in feature values was measured in pixel resolution relatively to the correct feature values. In particular, the scale error was measured in pixels as the deviation of the scale diameter. The orientation error was measured in term of the displacement of the second most salient maximum of IRF with respect to its correct position. Analysis of these data shows good accuracy and robustness of the proposed approach to feature extraction.

The objective of the second kind of experiments was the visual evaluation of the IRF performance in extracting skeletal shape in application to biometrical and medical imaging. One example of using IRF approach to detect fingers and determine their shape for the purpose of a biometrical identification from a hand image is shown in Fig. 5. This is an example of vertex extraction by IRF local maxima and establishing piecewise-linear connectivity between the extracted vertices. A detail analysis of finger geometry and texture of the finger skin have to be performed in each salient location. The skeletal shape was extracted directly from the grey-scale image in Fig. 5a without the image binarization.

The proposed IRF method was also compared with the skeletonization method using self-organizing maps by applying both methods to the same test image [10]. An example of obtained results by the two methods is shown in Fig. 6. The method of piecewise-linear skeletonization using self-organizing maps performed worse even

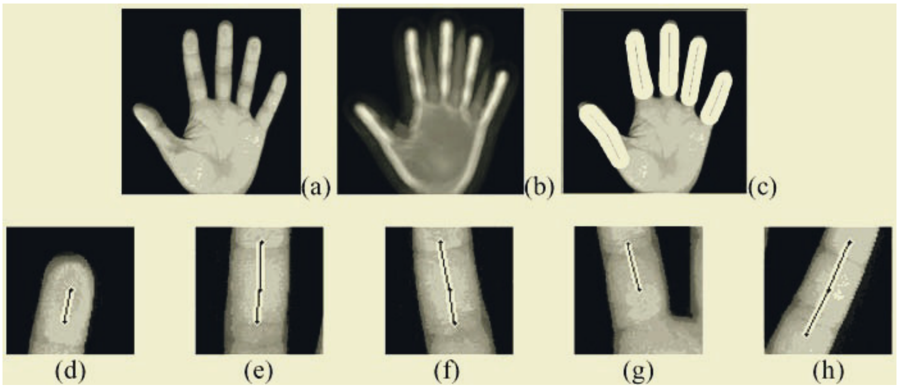


Fig. 5. Results of skeletal shape extraction of fingers obtained directly from the gray-scale image of a hand: (a) – initial image; (b) – IRF calculation; (c) – skeletal shape of fingers. Examples of local shapes at salient locations are shown below.

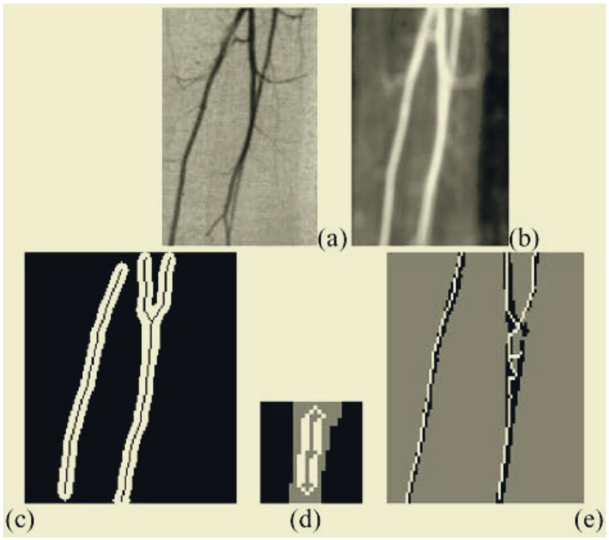


Fig. 6. An illustration to skeletal shape extraction in digital angiography: input image, (a); image of IRF, (b); skeletal shape of main blood vessels in the selected region of interest, (c); most salient object fragment, (d); result of skeletonization using method of self-organizing maps [10], (e).

when applied to the binary version of the input image and gave visible imprecision such as jaggedness of lines.

6 Conclusions

A method for the extraction of skeletal shape features using a visual attention operator was developed. It is based on the determination of salient object locations by local maxima analysis of the introduced multi-scale IRF. The same IRF approach was applied to extract a piecewise-linear skeletal shape at determined salient locations. The proposed concise description of local shapes has the following advantages in the context of object detection and shape verification. The shape comparison does not require computationally complex alignments because two different shapes can be compared by a simple distance measure (e.g., Euclidean distance). The IRF approach provides a robust shape extraction directly from gray-scale images, in the presence of noise and under some local distortions. The obtained shape features can easily become invariant with respect to translation, scaling, and rotation by a normalization relatively to geometrical parameters for a current location.

Acknowledgments

We are grateful to VRQ (Valorisation Recherche Québec) and Canadian Heritage for their financial support to CoRIMedia (Consortium de Recherche en Image et Multimedia).

References

1. M. D. Wheeler and K. Ikeuchi, "Sensor modeling, probabilistic hypothesis generation, and robust localization for object recognition", *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. 17, No. 3, pp. 252-265, 1995.
2. V. Conception and H. Wechsler, "Detection and localization of objects in time-varying imagery using attention, representation and memory pyramids", *Pattern Recognition*, Vol. 29, No. 9, pp. 1543-1557, 1996.
3. N. Blum and R.N. Nagel, "Shape description using weighted symmetric axis features", *Pattern Recognition*, Vol. 10, pp. 167-180, 1978.
4. Y. S. Chen, and Y.T. Yu, "Thinning approaches for noisy digital patterns". *Pattern Recognition*, Lol. 29, No. 11, pp. 1847-1862, 1996.
5. G. Borgefors, "Distance transformation in digital images", *Vision, Graphics, and Image Processing,* Vol. 34, pp. 344-371, 1986.
6. G. Borgefors, G. Ramella, G. Sanniti di Baja, and S. Svenson, "On the multi-scale representation of 2D and 3D shapes", *Graphical Models and Image Processing,* Vol. 61, pp. 44-62, 1999.
7. C. Archelli and G. Ramella, "Sketching a grey-tone pattern from its distance transform," *Pattern Recognition,* Vol. 29, No. 12, pp. 2033-2045, 1996.
8. T. Hastie, and W. Stuetzle, "Principal curves", *Journal of the American Statistical Association*, Vol. 84 (406), pp. 502-516, 1989.
9. B. Kegl, *et al.*, "Learning and design of principal curves", *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. 22, No. 3, pp. 281-297, 2000.
10. R. Singh, V. Cherkassky, and N. Papanikopoulos, "Self-organizing maps for the skeletonization of sparse shapes", *IEEE Trans. on Neural Networks*, Vol. 11, No. 1, pp. 241-248, 2000.
11. T. Lindeberg, "Detecting salient blob-like image structures and their scale with a scale-space primal sketch: a method for focus of attention", *Int. Journal of Computer Vision*, Vol. 11, pp. 283-318, 1993.

12. L. Itti, C. Koch, and E. Niebur, “A model of saliency-based visual attention for rapid scene analysis”, *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. 20, No. 11, pp. 1254-1259, 1998.
13. H. D. Tagare, K. Toyama, and J.G. Wang, “A maximum-likelihood strategy for directing attention during visual search”, *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. 23, No. 5, pp. 490-500, 2001.
14. R. M. Palenichka, “A visual attention operator based on morphological models of images and maximum likelihood decision”, *Proc. Int. Workshop SSPR 2002, LNCS* 2396, pp. 310-319, 2002.
15. J. J. Koenderink and A. J. van Doorm, “Representation of local geometry in the visual system”, *Biological cybernetics,* Vol. 55, pp. 367-375, 1987.
16. M. B. Zaremba and R. M. Palenichka, “Probabilistic morphological modeling of hydrographic networks from satellite imagery using self-organizing maps,” *Control & Cybernetics*, Vol. 31, No. 2, pp. 343-370, 2002.

Computing the Cyclic Edit Distance for Pattern Classification by Ranking Edit Paths[⋆]

Víctor M. Jiménez, Andrés Marzal, Vicente Palazón, and Guillermo Peris

DLSI, Universitat Jaume I, 12071 Castellón, Spain
{vjimenez,amarzal,palazon,peris}@uji.es

Abstract. The cyclic edit distance between two strings A and B of lengths m and n is the minimum edit distance between A and every cyclic shift of B. This can be applied, for instance, in classification tasks where strings represent the contour of objects. Bunke and Bühler proposed an algorithm that approximates the cyclic edit distance in time $O(mn)$. In this paper we show how to apply a technique for ranking the K shortest paths to an edit graph underlying the Bunke and Bühler algorithm to obtain the exact solution. This technique, combined with pruning rules, leads to an efficient and exact procedure for nearest-neighbour classification based on cyclic edit distances. Experimental results show that the proposed method can be used to classify handwritten digits using the exact cyclic edit distance with only a small increase in computing time with respect to the original Bunke and Bühler algorithm.

Keywords: Cyclic strings, cyclic edit distance, string matching, Bunke and Bühler algorithm, handwritten text recognition, OCR, K shortest paths.

1 Introduction

Measuring dissimilarities between strings is a fundamental problem in pattern recognition [1]. The most widely used measure of dissimilarity between two strings is the *edit distance* (ED), also known as the *weighted Levensthein distance*, which is defined as the weight of the best sequence of edit operations (insertions, substitutions and deletions of symbols) needed to transform one string into the other [2].

There are many applications where the objects are better modelled by *cyclic strings*, which are strings whose last symbol is considered to be followed by the first symbol. For instance, contours of objects can be appropriately represented by cyclic chain-codes [3, 4] (see Fig. 1). The dissimilarity between cyclic strings can be measured by means of the *cyclic edit distance* (CED), which is defined as the weight of the best sequence of edit operations needed to transform any cyclic shift of one string into any cyclic shift of the other. A trivial way to obtain

[⋆] This work has been supported by the *Generalitat Valenciana* under grant CTIDIA/2002/209 and by the Spanish *Ministerio de Ciencia y Tecnología* and FEDER under grant TIC2002-02684.

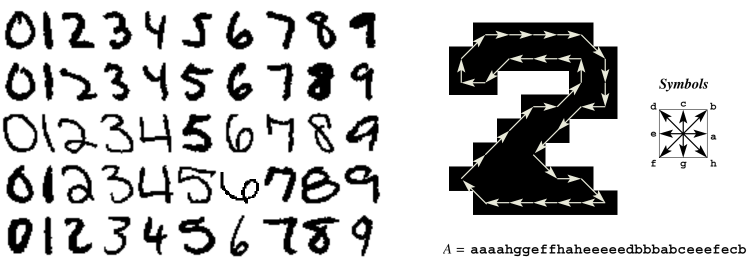


Fig. 1. Example digits from the NIST Special Database 19 and a string representing the contour of a digit with an 8-directional chain-code.

the cyclic edit distance in $\mathcal{O}(mn^2)$ time consists in computing the edit distance between one string and all the possible cyclic shifts of the other. Maes [5] proposed a divide and conquer algorithm that reduces the time cost to $\mathcal{O}(mn \log n)$. Marzal, Barrachina, and Peris [6, 7] reformulated this method as a branch and bound algorithm and proposed bounding functions that produce a significant speeding up of Maes' algorithm while maintaining its worst-case complexity.

In applications where the running-time of the algorithm is a major concern (for instance, in classification systems in which the CED between every string to be classified and a large number of labelled samples is computed), alternative approximate methods that run faster than the exact methods can be used. A well-known approximate method to compute the CED is the Bunke and Bühler algorithm (BBA), which runs in $\mathcal{O}(mn)$ time [3]. Mollineda, Vidal, and Casacuberta have proposed other approximate solutions based on the BBA that require a training stage [8, 9].

In this paper, we present a new *exact* method to compute the CED. Our proposal is based on the BBA, combined with an efficient technique for finding the K shortest paths in graphs [10], which is adapted to this problem. In classification tasks, this method can be combined with pruning rules to abort the computation of distances with values above the best distance found so far. In this way, according to experimental results reported in this paper, the value of K needed to find the exact solution is quite low in practice and the total running time is only slightly greater than the time needed by the BBA to find an approximate solution.

2 Notation and Problem Formulation

Let Σ be a set of symbols and let $\Sigma^\star$ be the set of all finite strings over Σ. Let a, b denote symbols in Σ and let λ denote the empty string. Throughout this paper, we consider that $A = a_1 a_2 \dots a_m$ and $B = b_1 b_2 \dots b_n$ are strings in $\Sigma^\star$ of length m and n, respectively.

Edit Distance. An *edit sequence* is a sequence $E = e_1 e_2 \dots e_p$ in which e_i, for $1 \leq i \leq p$, is one of four possible edit operations: (i) deletion of a symbol a, denoted $a \to \lambda$; (ii) insertion of a symbol b, denoted $\lambda \to b$; (iii) substitution of a symbol a by a symbol b, denoted $a \to b$; and (iv) matching (substitution of a symbol a by itself), denoted $a \to a$. Let $\gamma(E) = \sum_{i=1}^{p} \gamma(e_i)$ be the weight of the edit sequence E, with $\gamma(e_i)$ being the weight of the edit operation e_i. The *edit distance* $ED(A, B)$ is defined as the minimum value of $\gamma(E)$ for any edit sequence E that transforms A into B.

Cyclic Edit Distance. Let $\sigma(A) = a_2 a_3 \dots a_m a_1$ denote a *cyclic shift* of A, and let $\sigma^j(A) = a_{j+1} a_{j+2} \dots a_m a_1 \dots a_j$ denote the composition of j cyclic shifts. In many applications (for instance, in classification tasks where strings represent contours of objects) it makes sense to consider that the strings A and $\sigma^j(A)$, for any $j \in \mathbb{N}$, are equivalent. The equivalence class $[A] = \{\sigma^j(A) : j \in \mathbb{N}\}$ is called a *cyclic string.* The *cyclic edit distance* $CED([A], [B])$ is a measure of dissimilarity between the classes that the strings A and B represent, and is defined as $CED([A], [B]) = \min_{i,j \in \mathbb{N}} ED(\sigma^i(A), \sigma^j(B))$, which is the same as $CED([A], [B]) = \min_{1 \le j \le n} ED(A, \sigma^j(B))$ [5].

In this paper, we are interested in computing $CED([A], [B])$ for any given pair of strings A and B. In the next section we review how a refinement of the Bunke and Bühler algorithm, which approximates the value of $CED([A], [B])$ in time $O(mn)$, can be seen as an algorithm for finding a shortest s-t path in a graph. Then in Sect. 4 we will see how an algorithm for finding the K shortest s-t paths can be adapted to compute the exact value of $CED([A], [B])$.

3 Approximating the Cyclic Edit Distance with the Bunke and Bühler Algorithm

The computation of the edit distance $ED(A, B)$ using the Wagner and Fischer algorithm [2] can be formulated in terms of finding the shortest path between a pair of nodes in a graph G_A^B (the so-called *edit graph*). The nodes of G_A^B are all the pairs (i, j) for $0 \le i \le m$ and $0 \le j \le n$. There are (at most) three incoming edges for each node (i, j) (see Fig. 2a): (i) coming from $(i-1, j)$, if $i > 0$, with weight $\gamma(a_i \to \lambda)$; (ii) from $(i, j-1)$, if $j > 0$, with weight $\gamma(\lambda \to b_j)$; and (iii) from $(i-1, j-1)$, if $i > 0$ and $j > 0$, with weight $\gamma(a_i \to b_j)$. The edit distance $ED(A, B)$ is the weight of the shortest path between nodes $s = (0, 0)$ and $t = (m, n)$. The edit graph is acyclic and has $O(mn)$ edges; therefore, the shortest s-t path can be found in $O(mn)$ time by following any topological order of nodes [11].

In order to compute the cyclic edit distance $CED([A], [B])$, we can consider the edit graph G_A^{BB} associated to $ED(A, BB)$, the edit distance between A and B concatenated with itself. In this graph, the shortest path from the node $s = (0, j)$ to the node $t = (m, n + j)$, for every $j = 1, 2, \dots, n$, represents the best edit sequence that transforms A into $\sigma^j(B)$, whose weight is $ED(A, \sigma^j(B))$. The minimum of these n weights is $CED([A], [B])$. This value can be computed in time $O(mn^2)$ by just running a shortest s-t path algorithm for each of these n s-t pairs.

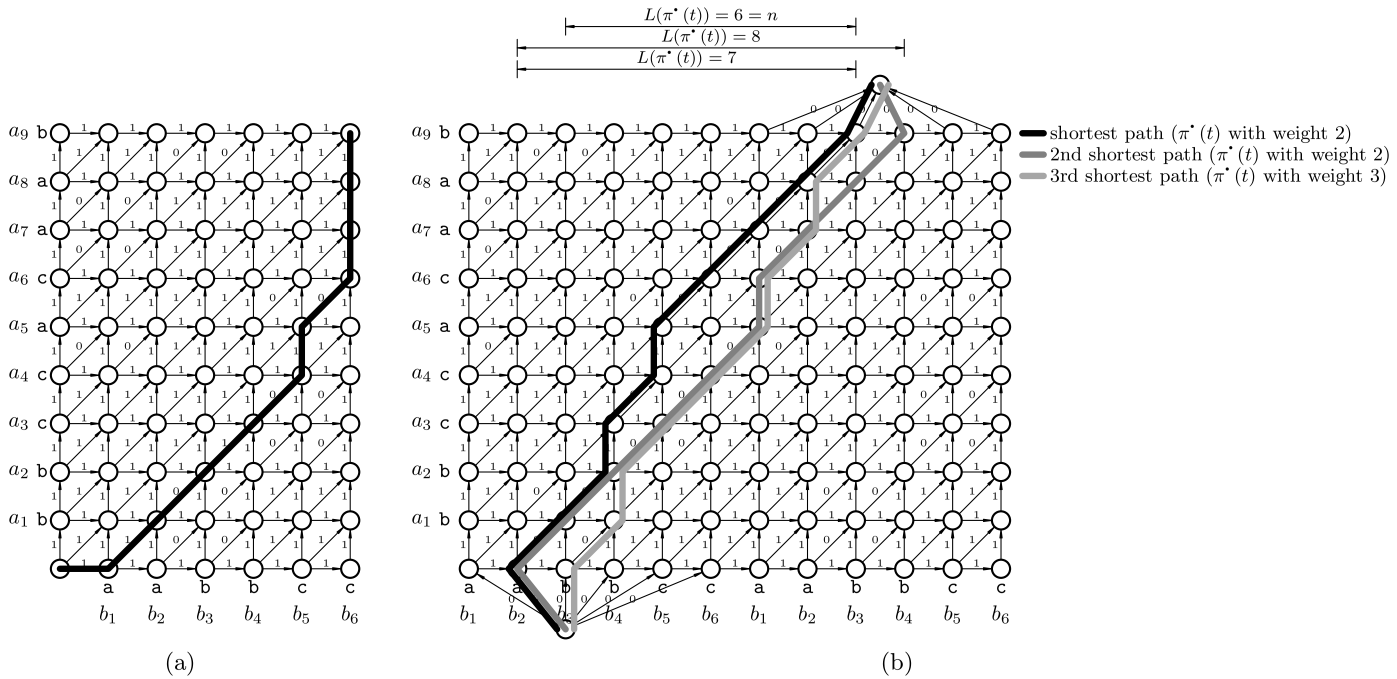

$L(\pi^{\bullet}(t)) = 6 = n$
$L(\pi^{\bullet}(t)) = 8$
$L(\pi^{\bullet}(t)) = 7$
shortest path ($\pi^{\bullet}(t)$ with weight 2)
2nd shortest path ($\pi^{\bullet}(t)$ with weight 2)
3rd shortest path ($\pi^{\bullet}(t)$ with weight 3)
a_9 b
a_8 a
a_7 a
a_6 c
a_5 a
a_4 c
a_3 c
a_2 b
a_1 b
a a b b c c
b_1 b_2 b_3 b_4 b_5 b_6
(a)
(b)

Fig. 2. Let $A =$bbccacaab and $B =$aabbcc represent two cyclic strings. (a) Edit graph G_A^B. (b) Edit graph $\bar{G}_A^{BB}$. In this example, matchings have a weight of 0 and any other edit operation has a weight of 1. The Wagner and Fischer algorithm finds $ED(A,B) = 7$, the weight of the shortest s-t path in G_A^B. The Bunke and Bühler algorithm obtains 2 as an approximate value of $CED([A],[B])$: the weight of the shortest s-t path in $\bar{G}_A^{BB}$. The exact value of $CED([A],[B])$ is 3: the weight of $\pi^{\bullet}(t)$, the shortest s-t path π in $\bar{G}_A^{BB}$ such that $L(\pi) = 6 = n$, computed using the method described in Sect. 4.

An approximate value (a lower bound) of $\mathit{CED}([A],[B])$ can be found more efficiently, in time $O(mn)$, by finding, in the edit graph G_A^{BB}, the shortest path starting at any node in $S = \{(0,j) : 1 \leq j \leq n\}$ and finishing at any node in $T = \{(m, n{+}j) : 1 \leq j \leq n\}$. This is equivalent to finding the shortest path from s to t in the graph obtained from G_A^{BB} by removing the nodes $\{(i,0) : 0 \leq i \leq m\}$ and the edges departing from them, and adding edges of weight 0 from an extra node s to every node in S and from every node in T to an extra node t. Let $\bar{G}_A^{BB}$ denote the resulting graph (see Fig. 2b). Again, this is a shortest s-t path problem in an acyclic graph with $O(mn)$ edges and can be solved in $O(mn)$ time. More precisely, it takes twice the time required to compute $\mathit{ED}(A,B)$. We call this method the Bunke and Bühler Algorithm (BBA) since a similar proposal to estimate a lower bound of $\mathit{CED}([A],[B])$ was originally made by Bunke and Bühler in [3]. The suboptimality of this method is due to the fact that the optimal path that it finds could start going from s to $(0,j)$ and finish going from (m,j') to t, with $j' \neq n+j$, while the path corresponding to $\mathit{CED}([A],[B])$ should verify $j' = n+j$. In the next section, we will see how an algorithm to enumerate the K shortest s-t paths in a weighted graph, the so-called *Recursive Enumeration Algorithm* (REA) [10], can be adapted to find the path with the minimum weight verifying $j' = n+j$. The weight of such a path is the exact value of $\mathit{CED}([A],[B])$.

4 Computing the Cyclic Edit Distance by Ranking Paths in the Bunke and Bühler Edit Graph

Let V be the set of nodes and let E be the set of edges in $\bar{G}_A^{BB}$. Given a path π and a node v, let $\pi \cdot v$ denote the path formed by π followed by v. For any path π in $\bar{G}_A^{BB}$ that starts going from s to $(0,j)$ and ends at (i,j'), as well as for any path π in $\bar{G}_A^{BB}$ that starts going from s to $(0,j)$ and ends by going from (m,j') to t, let us define $L(\pi) = j' - j$ (see Fig. 2b). In order to compute the exact value of $\mathit{CED}([A],[B])$, we are interested in finding the path π from s to t with the minimum weight among those verifying $L(\pi) = n$. This can be done by enumerating, by ascending weight value, the paths from s to t until the first path verifying $L(\pi) = n$ is found, as follows [10]:

```
A.1 Compute $\pi^1(v)$, the shortest path from $s$ to $v$, for all $v \in V$ and set $k \leftarrow 1$.
A.2 While $L(\pi^k(t)) \neq n$ do:
    A.2.1 Set $k \leftarrow k+1$ and compute $\pi^k(t)$ by calling $\mathit{NextPath}(t,k)$.
```

For $k > 1$, and once $\pi^1(v)$, $\pi^2(v)$,..., $\pi^{k-1}(v)$ are available, $\mathit{NextPath}(v,k)$ computes $\pi^k(v)$ as follows:

```
B.1 If $k = 2$, then initialise a set of candidates to the next shortest path from $s$
        to $v$, $C[v] \leftarrow \{\pi^1(u) \cdot v : (u,v) \in E$ and $\pi^1(v) \neq \pi^1(u) \cdot v\}$.
```

```
B.2 If $v = s$, then $\pi^k(v)$ does not exist; else
    B.2.a Let $u$ and $k'$ be the node and index such that $\pi^{k-1}(v) = \pi^{k'}(u) \cdot v$. If
          $\pi^{k'+1}(u)$ has not already been computed, then compute it by calling
          *NextPath*$(u, k'+1)$.
    B.2.b If $\pi^{k'+1}(u)$ exists, then insert $\pi^{k'+1}(u) \cdot v$ in $C[v]$.
    B.2.c If $C[v] = \emptyset$, then $\pi^k(v)$ does not exist.
    B.2.d If $C[v] \neq \emptyset$, then extract the path $\pi$ with minimum weight from $C[v]$
          and let $\pi^k(v) \leftarrow \pi$.
```

Proof of correctness of this method to compute the K shortest s-t paths in a weighted graph can be found in [10]. In this particular application to compute the CED, the algorithm runs in $O(mn + K(m+n))$ time: each of the K shortest paths is computed by recursively visiting, at most, the nodes of the previous shortest s-t path [10], and each s-t path in $\bar{G}_A^{BB}$ has $O(m+n)$ nodes.

The algorithm can be speeded up in this application by taking into account that we are not interested in the K shortest paths, but only in the first s-t path π that satisfies the restriction $L(\pi) = n$. Therefore, the partial paths that do not lead to a new s-t path with a different value of L can be discarded. This can be done by simply replacing Step B.2.d by:

```
B.2.d If $C[v] \neq \emptyset$, then extract the path $\pi$ with minimum weight from $C[v]$. If
      $L(\pi) \neq L(\pi^j(v))$, for all $j = 1, 2, \ldots, k-1$, then let $\pi^k(v) \leftarrow \pi$; else
    B.2.d.1 Let $u$ and $k'$ be the node and index such that $\pi = \pi^{k'}(u) \cdot v$.
            If $\pi^{k'+1}(u)$ has not already been computed, then compute it by
            calling *NextPath*$(u, k'+1)$.
    B.2.d.2 Goto B.2.b
```

With this modification, $\pi^k(v)$ is the path from s to v with minimum weight such that $L(\pi^k(v))$ is different from $L(\pi^j(v))$ for all $j \in \{1, 2, \ldots, k-1\}$.

5 Pruning the Search Space in Classification Tasks

The method described in Sect. 4 can be further speeded up in nearest-neighbour classification, where we have N labelled samples, B^1, B^2, $\ldots$, B^N, and we want to compute $\min_{1 \leq i \leq N} CED([A], [B^i])$ in order to classify A. Let us assume that we have already computed $d_{j-1} = \min_{1 \leq i < j} CED([A], [B^i])$ and that we are going to compute $d_j = \min\{d_{j-1}, CED([A], [B^j])\}$. The computation of $CED([A], [B^j])$ can be aborted as soon as we know that its value cannot be lower than d_{j-1}, according to these rules:

1. The computation can be avoided if $m > n$ and $(m-n) \min_{a \in \Sigma} \gamma(a \to \lambda) \geq d_{j-1}$, or $n > m$ and $(n-m) \min_{b \in \Sigma} \gamma(\lambda \to b) \geq d_{j-1}$. This rule is based on the fact that at least $|m-n|$ insertions or deletions must be performed to transform one string into the other.
2. Taking into account that the edit weights are non-negative, the execution of Step A.1 can be aborted, for any $i \in \{1, 2, \ldots, m\}$, if the weight of $\pi^1((i,j))$ is greater than or equal to d_{j-1} for all $j \in \{1, 2, \ldots, 2n\}$.

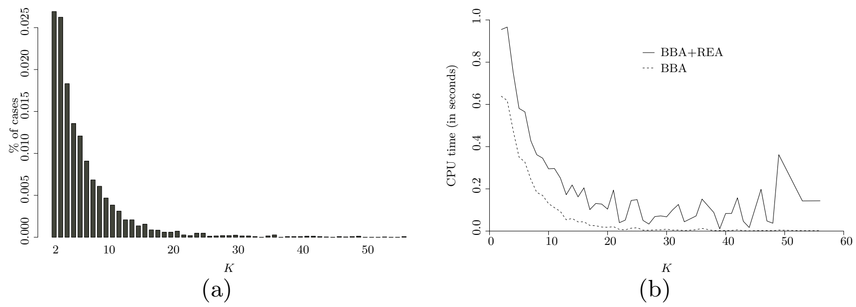


Fig. 3. (a) Percentage of cases for which K shortest s-t paths with different value of L have to be computed, for $K > 1$. (b) Total CPU time invested in those cases.

3. The ranking of s-t paths by Step A.2 can be stopped as soon as we reach a value of k such that the weight of $\pi^k(t)$ is greater than or equal to d_{j-1}.

None of these rules modify the worst-case computational complexity of the algorithm and, in practice, they entail a significant reduction in running time. They can also be extended to deal with the N nearest-neighbours classification rule.

6 Experimental Results

In order to assess the behaviour of the algorithm in practice, we performed experiments on a handwritten digits recognition task. A test set containing 500 digit images (5 instances of each digit by 10 writers) randomly selected from the hsf_4 set in the NIST Special Database 19 [12], was used (see Fig. 1). Each test digit was compared to 5 000 labelled instances from the sets hsf_{0,1,2,3} (5 instances of each digit by 25 writers from each set) in order to perform a nearest-neighbour classification. All the images were clipped, scaled into a 32×32 pixels matrix and binarised, and their outer contours were represented by 4-directional chain-codes. The average length of the resulting cyclic strings is 125. The edit distances were then computed assuming unit weight for insertions, deletions and substitutions of symbols, and zero weight for matchings.

The classification error rate is 8.6% using the (non-cyclic) edit distance, 3.8% using the approximate cyclic edit distance obtained with the BBA, and 3.2% using the exact cyclic edit distance. This confirms previous results showing that the classification using the exact cyclic edit distance performs better than the approximate method [9].

In principle, 2 500 000 cyclic edit distances had to be computed in order to classify the 500 test digits. The method proposed in this paper only required the computation of the K shortest s-t paths with different value of L, for K greater than 1, in 0.15% of the cases. Figure 3a shows a histogram with the percentage of cases for each value of K. It can be seen that computing the exact CED never required the computation of more than 60 shortest paths (the average value of K, when $K > 1$ shortest paths had to be computed, was 6.53).

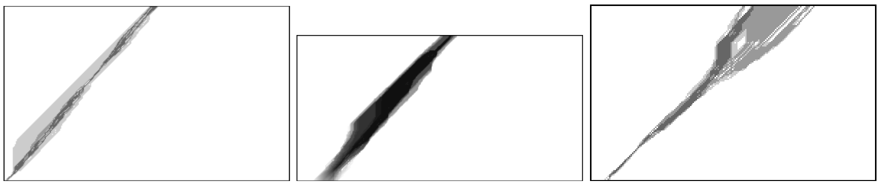


Fig. 4. Edit graph $\bar{G}_A^{BB}$, colouring the region where the REA searches for alternative paths, for 3 cases in which the shortest path is not the exact solution. A darker colour represents a higher number of computed paths.

The total time required to classify the 500 test digits was 763.89 seconds on a 2.4GHz Pentium 4 running under Linux 2.4 (the algorithms were implemented in C). The execution of the BBA accounts for 754.09 seconds. Only 9.80 seconds (1.28% of the total running time) were devoted to computing alternative paths with the REA. Figure 3b shows, for the cases in which $K > 1$ shortest s-t paths have been computed, the total running time and the running time of the BBA. It can be observed that, for the largest values of K, the execution time of the REA is significantly greater than the time due to the BBA but, thanks to the pruning rules given in Sect. 5, such values are required in a very small percentage of cases, and they hardly affect the total running time of the classification procedure.

In practice, the efficiency of the REA not only depends on the number of computed paths, but also on the number of internal nodes in which alternative paths must be computed. Figure 4 shows these nodes for three different cases and illustrates that only a small region of the graph needs to be visited when looking for alternative paths.

7 Conclusions

The algorithm proposed by Bunke and Bühler [3] computes very efficiently an approximate value of the cyclic edit distance between two cyclic strings. In this paper, we have shown how a K shortest paths algorithm [10] can be adapted to this problem and applied to an edit graph underlying the Bunke and Bühler algorithm in order to find the first shortest path satisfying a particular restriction. The weight of this path is the exact cyclic edit distance. In classification tasks, this method can be combined with pruning rules to abort the computation of distances with values above the best distance found so far. Experimental results with a handwritten digit classification system show that the proposed method can serve to reduce the error rate and only entails a very small increase in computing time with respect to the approximate method.

References

1. D. Sankoff, J. Kruskal, eds.: Time Warps, String Edits, and Macromolecules: the Theory and Practice of Sequence Comparison. Addison-Wesley, Reading, MA (1983)

2. R.A. Wagner, M.J. Fischer: The string-to-string correction problem. Journal of the ACM **21** (1974) 168–173
3. H. Bunke, H. Bühler: Applications of approximate string matching to 2D shape recognition. Pattern Recognition **26** (1993) 1797–1812
4. D. Zhang, G. Lu: Review of shape representation and description techniques. Pattern Recognition **37** (2004) 1–19
5. M. Maes: On a cyclic string-to-string correction problem. Information Processing Letters **35** (1990) 73–78
6. A. Marzal, S. Barrachina: Speeding up the computation of the edit distance for cyclic strings. Int. Conf. on Pattern Recognition (2000) 271–280
7. G. Peris, A. Marzal: Fast cyclic edit distance computation with weighted edit costs in classification. Int. Conf. on Pattern Recognition (2002) 184–187
8. R. A. Mollineda, E. Vidal, F. Casacuberta: Efficient techniques for a very accurate measurement of dissimilarities between cyclic patterns. Lecture Notes in Computer Science **1876** (2000) 337–346
9. A. Marzal, R. Mollineda, G. Peris, E. Vidal: Cyclic string matching: efficient exact and approximate algorithms. In D. Chen, X. Cheng, eds.: Pattern Recognition and String Matching. Kluwer Academic (2002) 477–497
10. V. M. Jiménez, A. Marzal: Computing the K shortest paths: a new algorithm and an experimental comparison. Lecture Notes in Computer Science **1668** (1999) 15–29
11. Cormen, T., Leiserson, C., Rivest, R.: Introduction to Algorithms. The MIT Press, Cambridge, MA (1990)
12. P. J. Grother: NIST Special Database 19: Handprinted forms and characters database. Technical report, National Institute of Standards and Technology (1995)

Contour Segments from Spline Interpolation

Niklas Ludtke and Richard C. Wilson

• Beckman Institute, University of Illinois, USA
• Dept. of Computer Science University of York, UK

Abstract. An important function of perceptual grouping is the restoration of contours. Edge maps produced by low level edge detectors are invariably noisy and inconsistent. It it the aim of perceptual grouping to refine these edge segments by imposing consistency based on considerations about real object outlines. In this paper we describe a method for grouping edge segments into perceptually salient contours using splines. The two important ingredients of our method are firstly the use of probability distributions for possible orientation structure in the image, and secondly the use of Kellman-Shipley relatability to find perceptually meaningful structure. The spline parameters are adjusted to optimise their probabilities in terms of image structure and bending. Consistent structure is then identified using both perceptual criteria and similarity to contour structure in the image.

1 Introduction

An important function of perceptual grouping is the restoration of contours. Edge maps produced by low level edge detectors are invariably noisy and inconsistent. It it the aim of perceptual grouping to refine these edge contours by imposing consistency based on considerations about real object outlines. To overcome local distortions in machine vision, numerous authors (e.g. Sha'ashua and Ullman[1]; Sarkar and Boyer[2]; Elder and Zucker[3]; Guy and Medioni[4]) have therefore proposed incorporating contextual relations among local features by combining responses of neighbouring feature detectors into a globalised and consequently more robust processing.

Sha'ashua and Ullman[1] defined a measure of perceptual saliency of a curve, based on geometric properties. The saliency measure increases monotonically with the length of the evaluated curve and decreases with its total squared curvature. Additionally, the degree of fragmentation, expressed in terms of the number of gaps and total gap length, is penalised. A relaxation procedure is then performed to maximise the saliency measure.

Guy and Medioni[4] devised an algorithm for contour grouping, based on the Gestalt principles of co-curvilinearity and proximity. A convolution is performed on the edge map using a special mask called *extension field*, a vector field encoding the likelihood and orientation of possible continuations from an edge segment at its centre to all other points in the image. The direction of the extension field at a point (x, y) equals the tangent angle of the most likely curve connecting (x, y) with the edge segment at the centre of the extension field. The magnitude of the vector field in (x, y) is the likelihood of the existence of the connecting curve.

Elder and Zucker[3] address the problem of computing closed bounding contours. A multi-scale edge detection algorithm yields information about edge position and tangent orientation from which a sparsely connected tangent graph is constructed. Each node is assigned with the tangent information and, according to a Bayesian model of tangent linking, each arc is labelled with the likelihood that the corresponding tangent pair forms a contiguous component of the same contour. Each node is connected to only a small number of neighbours (usually six), according to the most likely pairings. The goal of closure grouping is then to find the *maximum likelihood cycles* for every tangent in the graph. Thus the grouping task is reduced to a shortest path problem.

Such perceptual organisation is very much in the spirit of Shipley and Kellman's psychophysically motivated theory of visual interpolation [5]. Although their criterion of edge *relatability* provides a useful test for co-curvilinearity, the geometrical configurations of detected edge segments are in practice often not as precise as required, due to noise. Instead, edge segments would often be erroneously dismissed as "unrelatable" , particularly on straight contours. Thus, orientation estimates will often have to be revised, in order to yield "relatable" and more accurate tangent configurations. The revision will be based on mutual consistency, as well as on the quality of agreement between the resultant curve segment and the Gabor transform of the given image.

2 Distributions of Orientations and Spline Interpolation

We commence with a description of the orientation structure of the image in terms of a mixture of von-Mises distributions[6, 7].

$$p(\theta) = \sum_i \frac{P_i}{2\pi I_0(\kappa_i)} \exp\left[\kappa_i \cos(\theta - \theta_i)\right] \tag{1}$$

This mixture model of von-Mises distributions represents multiple local orientations θ and their certainties through the widths κ_i. Typically there will be one or two orientation components, i.e. $i = \{0, 1\}$. We therefore encode both mutiple directions and uncertainty about each of those directions.

Within this framework, mutual consistency of contours can be expressed in terms of a spline likelihood, comprising the joint density of the orientation pdfs and an additional bending constraint. We have information about the positions and possible tangent directions of edge structure in the image, and therefore we use splines of the quadratic Hermite-type. Their parameters are fully determined by the positions and tangent orientations of the end points. As a consequence of the probabilistic model of orientation, tangent angles are governed by probability densities, and the uncertainty of tangent orientations is transformed into the system's uncertainty regarding the connecting spline. An important feature of the perceptual grouping framework presented in this paper is that the degree of uncertainty in the orientation determines the "inertia" of a local tangent estimate, i.e., how easily an initial orientation measurement (given by a mode in the corresponding pdf) can be modified during consistency optimisation. In this role of certainty lies a conceptual difference to other grouping schemes. Usually, the coarseness of the initial local orientation measurements is expressed in terms of the likelihood of potential continuations at the grouping level, for example, characterised by a "support function" in relaxation labelling [8, 9] or by orientation "votes" propagated through an "extension field" [4]. The *initial* certainty of the local measurement, however, is not modelled.

3 Quadratic Splines

After low-level processing[6, 7] the positions of the control points, the corresponding distributions of tangent orientations are known, and points with multiple orientations are identified as such. Therefore, *piecewise quadratic spline interpolation* provides a very straightforward means of connecting such control points.

In the grouping framework presented, the constraint of C^2 and C^1 continuity at the control points will not be imposed. Instead, left and right limits of tangent orientation are introduced, whereby smooth and polygonal curves can be represented equally well [10]. As a result, the algorithm is capable of representing tangent discontinuities suggested by feature associations, in addition to the locally detected points with multiple orientations.
Let $t \in [0, 1]$ be the spline parameter, and let $\mathbf{s}(t)$ denote a position on the spline, i.e., $\mathbf{s}(t) = [x(t), y(t)]^T$. Then the quadratic spline is defined as:

$$\mathbf{s}(t) = \mathbf{a}t^2 + \mathbf{b}t + \mathbf{c}\,, \qquad \text{with } \mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^2\,. \tag{2}$$

Here $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$ are the vector-valued spline coefficients. These quantities are not geometrically meaningful in this form, so we write the spline in terms of the positions of the endpoints $\mathbf{r}_1, \mathbf{r}_2$ and the tangent directions at the endpoints; θ_1, θ_2.

$$\mathbf{s}(t) = (-t^2 + 1)\,\mathbf{r}_1 \;+\; t^2\,\mathbf{r}_2 + (-t^2 + t) \cdot 2d\frac{\sin(\phi - \theta_2)}{\sin(\theta_1 - \theta_2)} \begin{pmatrix} \cos\theta_1 \\ \sin\theta_1 \end{pmatrix} \tag{3}$$

Here d and ϕ are the length and angle of the vector connecting the endpoints, $\mathbf{r}_2 - \mathbf{r}_1$. It is important to note that the final term becomes singular for $\theta_1 = \theta_2$. Since the splines do not contain inflexions, equality of θ_1 and θ_2 is only possible if $\theta_1 = \theta_2 = \phi$. The spline therefore approaches the straight line $\mathbf{s}(t) = \mathbf{r}_1 + t(\mathbf{r}_2 - \mathbf{r}_1)$.

4 Optimisation of Spline Parameters

The essential point of our method is that the local orientation is represented by a probability distribution, and therefore the spline parameters θ_1 and θ_2 are not fixed quantities, and may be varied in order to obtain more consistent splines. The final splines may be relatable even if the initial configuration $(\bar{\theta}_1, \bar{\theta}_2)$ is not.

By virtue of (3), each pair $(p(\theta_1), p(\theta_2))$ of two locally extracted orientation densities implies a density $p(\mathbf{s}|\,\theta_1, \theta_2)$ in the the spline, thus describing a "bundle" of possible quadratic splines passing through the fixed end points. There is no need to actually compute $p(\mathbf{s}|\,\theta_1, \theta_2)$. Instead, the optimisation is performed with respect to the tangent angles θ_1 and θ_2, and the corresponding optimal $\mathbf{s}$ is calculated afterwards.

In order to enforce smoothness of contours, it is necessary to impose a shape constraint on the connecting spline bundle that penalises a high degree of bending. The new tangent angles are then found by means of a maximum likelihood estimation procedure, which results in a trade-off between closeness to initial local measurements and smoothness constraint. The final decision about the relatability of a pair of key points is made after this optimisation.

4.1 The Spline Likelihood Function

In general terms, the total likelihood of a pair of tangent angles $(\theta_1,\ \theta_2)$ is given by the product of the joint density of that pair, obtained from (1), and a probability density that depends on the degree of bending of the corresponding spline:

$$\mathcal{L}(\theta_1, \theta_2) = p_{\text{pop}}(\theta_1, \theta_2)\, p_{\text{bend}}(\theta_1, \theta_2)\,. \tag{4}$$

The quantity that describes the bending of the spline is the magnitude of the second derivative with respect to the parameter t, denoted by $\ddot{s}(t)$. For a quadratic spline $\ddot{s}$ is a constant determined by the boundary conditions. The "bending probability", which acts as a penalty term in the likelihood function (4), can be defined as a Gaussian distribution in the scale invariant quantity $\ddot{s}/d$:

$$p_{\text{bend}}(\theta_1, \theta_2) = p(\ddot{s}(\theta_1, \theta_2)) = \frac{1}{\sqrt{2\pi}\sigma_b}\, \exp\left(\frac{\ddot{s}^{\,2}(\theta_1, \theta_2)}{2d^2\sigma_b^2}\right) \tag{5}$$

The variance σ_b^2 determines how strongly bending is penalised and is to be optimised together with θ_1 and θ_2. In order to find $\ddot{s}(\theta_1, \theta_2)$, equation (3) is differentiated twice with respect to t, yielding the square of its magnitude as:

$$\ddot{s}^2(\theta_1, \theta_2) = 4d^2 - 16d^2\, \tfrac{\sin(\phi-\theta_2)\cos(\phi-\theta_1)}{\sin(\theta_1-\theta_2)} + 16d^2 \tfrac{\sin^2(\phi-\theta_2)}{\sin^2(\theta_1-\theta_2)}$$

The density relating to the image orientation may be expressed as the product of the densities for each endpoint, since the local orientations are treated as independent random variables:

$$\begin{aligned} p_{\text{pop}}(\theta_1, \theta_2|\, i, j) &= p(\theta_1|\, i)\, p(\theta_2|\, j) \\ &= \frac{1}{4\pi^2 I_0(\kappa_i)\, I_0(\kappa_j)}\; e^{\kappa_i \cos(\theta_1 - \bar{\theta}_i) + \kappa_j \cos(\theta_2 - \bar{\theta}_j)} \end{aligned} \tag{6}$$

In most cases both points are on an edge, and consequently $i = 1$ and $j = 1$. Only in corner points or junctions several possible associations (i, j) have to be considered. Inserting (5) and (6) in (4) the spline log-likelihood becomes:

$$\ln \mathcal{L}(\theta_1, \theta_2;\, i, j) = -\ln[4\pi^2 I_0(\kappa_i)\, I_0(\kappa_j)] + \kappa_i \cos(\theta_1 - \bar{\theta}_i) + \kappa_j \cos(\theta_2 - \bar{\theta}_j)$$
$$-\tfrac{1}{2}\ln[2\pi] - \ln \sigma_b - \frac{\ddot{s}^{\,2}(\theta_1, \theta_2)}{2d^2\sigma_b^2} \tag{7}$$

The log-likelyhood may then be solved using a standard optimisation method. For the results presented in this paper, we used a simple gradient ascent approach.

It is important to note that in the log-likelihood function (7) the concentration parameters κ_i and κ_j of the pdfs act as weights of angular modifications during the optimisation process. If a concentration parameter is large, any deviation from the initial orientation $\bar{\theta}$ will result in a sharp decrease of the likelihood function unless the overall curvature is substantially reduced simultaneously. In other words, the concentration parameters (and thus the certainties, which are monotonic functions thereof) determine the “inertia” of orientation estimates, i.e., their “flexibility to compromise for the sake of mutual consistency”. Herein lies the essential difference to other grouping methods, where measurement of certainty is not an integral part of local feature extraction.

4.2 Detection of Control Points

To implement such a spline scheme, we must begin by locating the necessary control points. These control points are located at points of significant edge response in the image. Given a set of key points, the task is then to decide which points can be connected by splines. Here the idea is to eliminate only very unlikely configurations, leaving more plausible arrangements to be decided on after the splines have been reconfigured. For each key point, only a limited number of its nearest neighbours are considered for grouping, reflecting the Gestalt law of *proximity.* A weak relatability criterion is then applied to remove connections which are inconsistent. After optimisation of the spline parameters, the resulting splines are checked to establish whether they correspond to real edge structure in the image. Those that have no support are discarded. Furthermore, Shipley-Kellman relatability is checked with regard to the splines neighbours. Those that are inconsistent are discarded.

5 Experiments

Figure 1 shows an image of part of the sculpture of Paolina Borghese by Antonio Canova (1757-1822) and the different steps of contour extraction, from Gabor responses to the tangent elements extracted at key points and the result of spline interpolation between them. Some parts of the hair region contain very narrow features which are more suitable for processing with a line detector. The method has difficulties in correctly relating segments which are parallel and very close to each other.

The same image has been used by Iverson and Zucker (1995) to demonstrate the performance of their “logical/linear operators”. Referring to earlier work by

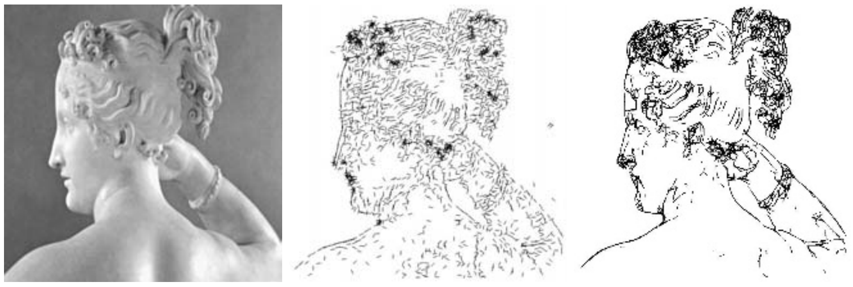


Fig. 1. Stages of contour extraction using a photograph of the sculpture "Paolina" (512×512 pixels, from the archive of the Vision group of Pietro Perona at Caltech).

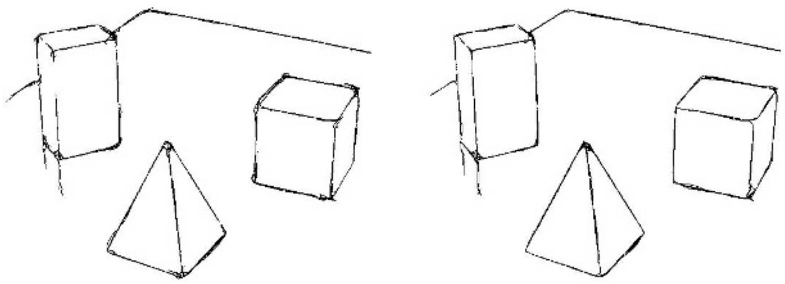
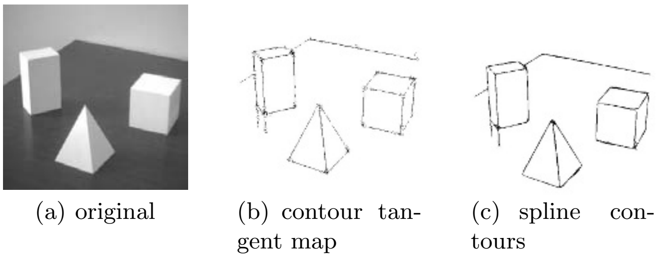


Fig. 2. The effect of tangent optimisation. In (a) the "relatability" criterion of Shipley and Kellman (1991) is applied directly to the tangent orientations given by the modes of the corresponding mixture densities. Figure (b) shows the same process after tangent optimisation.

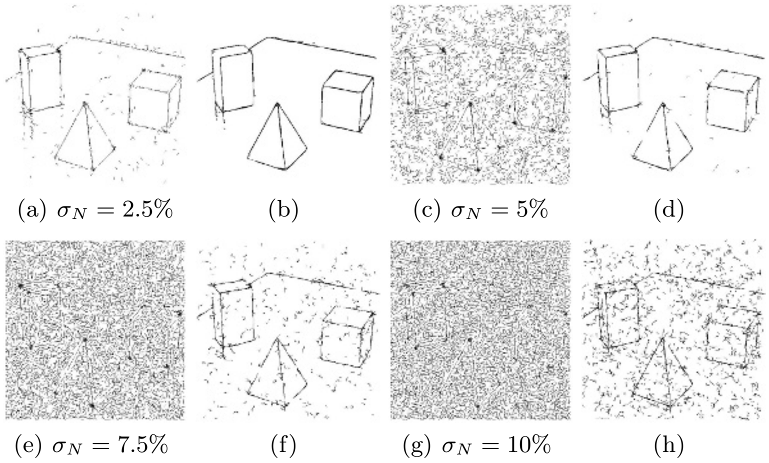
Koendrink and co-workers (1982), the authors point out the perceptual significance of bifurcations and line terminations in regions, such as the folds around the neck, which provide vital information about three-dimensional structure. They also demonstrate that the Canny detector [13], like any other essentially linear edge operator, is not capable of correctly representing bifurcations and tends to smooth out tangent discontinuities in corner points and T-junctions. Since feature extraction with probabilistic population coding explicitly represents points with multiple orientations and orientation discontinuities, the spline interpolation algorithm can accurately capture most of the essential discontinuities and bifurcations (Fig. 1 (f)).

Figure 2 illustrates the effect of the tangent optimisation algorithm on the spline contour representation. When applied directly to the tangent orientations given by the modes of the corresponding (mixture) densities, the "relatability" criterion of Shipley and Kellman (1991) rejects a number of tangent configurations, and consequently a lot of contour segments are not detected. Also, many



(a) original (b) contour tangent map (c) spline contours

Fig. 3. Original image with the tangent map and spline contours.



(a) $\sigma_N = 2.5\%$ (b) (c) $\sigma_N = 5\%$ (d)

(e) $\sigma_N = 7.5\%$ (f) (g) $\sigma_N = 10\%$ (h)

Fig. 4. An image with different amounts of additive Gaussian noise.

splines tend to differ from the actual contours, since inexact tangent angles tend to create curved rather than straight splines. Through optimisation, a significant number of tangent angles can be adjusted, in order to yield "relatable" configurations, many of which prove to be consistent with the intensity gradient in the image. As a result, a more complete and accurate contour representation is obtained.

In another experiment, the performance of the algorithm in the presence of additive Gaussian has been investigated. Figure 3 shows the original image with the extracted spline contours. Figure 4 shows the result of feature detection and subsequent perceptual grouping for moderate noise. For a moderate noise level ($\sigma_N < 5\%$, SNR < 26 dB) there are only few false positives in the spline representation, since most erroneous key points form only isolated splines that can easily be identified and removed. Above a noise value of about 10% (SNR $= 20$ dB), the density of false positive key points reaches a level where spurious splines begin to form erroneous contour segments of considerable length which could only be eliminated by perceptual organisation of higher order. At this stage curvature

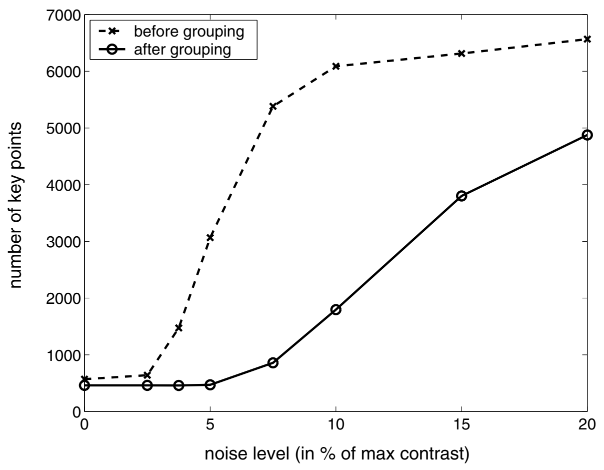


Fig. 5. The number of key points detected in Figure 4 (a) as a function of the noise level (standard deviation in % of maximum contrast) before (dashed curve), and after perceptual organisation (solid curve).

consistency would be a vital constraint, since the noise-induced contour segments exhibit frequent, sudden changes in the sign of curvature, which rarely occur in natural object boundaries and folds. The erroneously *discarded* key points (false negatives) are small in number but, of course, much more obvious, since they lead to gaps in the contour representation.

The number of detected features as a function of the noise level is an indicator for the efficiency of the feature detection in the presence of noise, since the number of additional key points compared to the case without noise approximately equals the number of false positives. Figure 5 shows a plot of this relation for the image in Figure 4 (a). Though more and more spurious key points appear with increasing noise level, most of them do not fulfill the relatability criterion, and even after angular optimisation a potential spline connection often lacks consistency with the filter responses. Thus many false positives can be identified and rejected.

References

1. Sha'ashua, A., Ullman, S.: Structural saliency: The detection of globally salient structures using a locally connected network. In: Proceedings of the 2nd ICCV. (1988) 321–327 Also Weizmann Institute of Science Report CS88-18, (October 1988).
2. Sarkar, S., Boyer, K.: Computing Perceptual Organisation in Computer Vision. World Scientific (1994)
3. Elder, J., Zucker, S.: Computing contour closure. In: Proceedings of the European Conference on Computer Vision. Volume Lecture Notes in Computer Science 1064., Springer Verlag (1996) 399–

4. Guy, G., Medioni, G.: Inferring global perceptual contours from local features. Int. J. Comp. Vis. **20** (1996) 113–133
5. Kellman, P., Shipley, T.: A theory of visual interpolation in object perception. Cognitive Psychology **23** (1991) 141–221
6. Lüdtke, N., Wilson, R., Hancock, E.: Population codes for orientation estimation. In: Proceedings of the 15th International Conference on Pattern Recognition. (2000)
7. Lüdtke, N., Wilson, R., Hancock, E.: Probabilistic population coding of multiple edge orientation. In: Proceedings of the International Conference on Image Processing, IEEE (2002) to appear.
8. Parent, P., Zucker, S.: Trace inference, curvature consistency, and curve detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **11** (1989) 823–839
9. Hancock, E., Kittler, J.: Discrete relaxation. Pattern Recognition **23** (1990) 711–733
10. Gavrila, D.: Hermite deformable contours. Technical Report CS-TR-3610, Center for Automation Research, University of Maryland (1996) also ICPR'96.
11. Iverson, L., Zucker, S.: Logical/linear operators for image curves. IEEE Transactions on Pattern Recognition and Machine Intelligence **17** (1995) 982–996
12. Koendrink, J., van Doorn, A.: The shape of smooth objects and the way contours end. Perception **11** (1982) 129–137
13. Canny, J.: A computational approach to edge detection. IEEE Transact. on Patt. Rec. and Machine Intell. **8** (1986) 679–700

Steady State Random Walks for Path Estimation

Antonio Robles-Kelly and Edwin R. Hancock

Department of Computer Science
University of York,York YO1 5DD, UK
{arobkell,erh}@minster.cs.york.ac.uk

Abstract. This paper describes a graph-spectral method for path estimation. Our aim is to find a maximum probability path through a lattice of pixel sites. We characterise the path recovery problem using a site transition matrix. A graph-spectral analysis of the transition matrix reveals how the maximum probability path can be located using an eigenvector of the associated normalised affinity matrix. We demonstrate the utility of the resulting method on the problem of recovering surface height from a field of surface normals.

1 Introduction

The recovery of maximum probability paths through a pixel lattice is one that arises throughout computer vision. This problem involves computing transition probabilities or costs associated with sites, and then searching for the maximum probability or minimum cost path. Of course, the underlying optimisation problem has exponential complexity, and hence exhaustive search is not a valid option. It is for this reason that optimisation methods such as dynamic programming [1], simulated annealing [2] and bayesian techniques [3] have been used to provide practical solutions to the problem. However, in this paper we aim to take a different approach and adopt a graph-spectral approach to the problem.

The idea underpinning graph-spectral methods is to abstract the problem in hand using a weighted graph. Here the nodes represent these basic image entities, and the weighted edges represent affinity relations between the entities. By computing the eigenvalues and eigenvectors of the weight matrix, it is possible to find groups or clusters of tokens. The graph-spectral method is in fact one of energy minimisation since the eigenvectors can be shown to be minimisers of a quadratic form. In fact, graph-spectral methods have recently proved highly effective in image processing and computer vision. Perhaps the best known method is that of Shi and Malik [4] which has shown how to locate image regions by recursively bisecting a weighted graph that represents the affinity of pairs of pixels. The method is based on the normalised cut. This is a measure of the relative weight of the edges connecting the two parts of the partition (the cut) to the weight assigned to the edges within the two parts of the bisection (the association). A relaxed solution to the bisection problem is found by locating the eigenvector associated with the second smallest eigenvalue of the Laplacian matrix (the degree matrix minus the affinity weight matrix).

Although it is convenient to work with the Laplacian, since it is positive and semi-definite, grouping and segmentation can also be performed using an edge-weight or affinity matrix. For instance, both Sarkar and Boyer [5] and Perona and Freeman [6] have developed matrix factorisation methods for line-segment grouping that use eigenvectors of an affinity matrix rather than the associated Laplacian. The Sarkar and Boyer [5] method can be understood as maximising the association (i.e. the total edge weight) of the clusters.

The methods described above all share the feature of using the eigenvectors of a Laplacian or an affinity matrix to define groups or clusters or objects. However, graph-spectral methods can also be used for path analysis tasks on graphs. For instance, it is well known that the path length distribution can be computed from the spectrum of eigenvalues of the adjacency matrix [7]. Ideas from spectral-graph theory have also been used to analyse the behaviour of random walks on graphs [8–10]. The observation underpinning this work is that random walks on a weighted graph can be represented as Markov chains in which the transition probabilities are computed from the normalised edge weights. The problem investigated is to compute the transition probability between pairs of pixel sites after a large number of time steps have elapsed. This study has lead to a number of interesting findings. Of direct relevance to this paper is the fact that the steady state random walk on the graph is characterised by the leading eigenvector of the normalised edge-weight matrix. In addition, there are important relationships between the eigenvectors of the edge-weight matrix and other quantities related to random walks. These include the access time for a node (i.e. the expected number of time steps that must have elapsed before the node is visited) and the mixing rate (i.e. the rate at which the random walk converges to its steady state). The relationship between the leading eigenvector of the edge weight matrix and the steady state random walk has been exploited in a number areas including routeing theory and information retrieval [11, 12].

The advantage of graph-spectral methods is that they can be used to find approximate or relaxed solutions without the need for parallel iterative updates at the pixel site level. The method also obviates the need for complex search algorithms. However, although they have been applied to region segmentation and grouping problems, graph-spectral methods have not been applied to curve detection problems of the sort that arise in the determination of the optimal integration path.

2 Graph Spectral Analysis

To cast the curve estimation problem in a graph-spectral setting we adopt an abstraction where the sites to be traversed are represented by a node-set V, the connectivity relations by an edge-set E and the edges have a weight function $W : E \to [0, 1]$. Here we aim to use the weight matrix W to define a Markov chain and to use the steady state random walk associated to this chain to find a path across the graph $G = (V, E)$. The elements of the weight matrix are computed using the energy or cost associated with the transitions between sites on the pixel lattice.

Suppose that $\mathcal{E}_{i,j}$ is the energy associated with the transitions between the sites with node-labels i and j, then the weight associated with the transition is $W_{i,j} = \exp[-\beta \mathcal{E}_{i,j}]$. Unfortunately, when computed in this way, the weight matrix W cannot be used directly as the transition probability matrix for the Markov chain since its rows do not sum to unity. To normalise the rows of the matrix we compute the degree of each node $deg(i) = \sum_{j=1}^{|V|} W(i,j)$. With the diagonal degree matrix $D = diag(deg(1), deg(2), ..., deg(|V|))$ at hand, the transition probability matrix is given by $P = D^{-1}W$. The elements of the transition matrix are hence given by $P_{i,j} = \frac{1}{deg(i)} W_{i,j}$. It is interesting to note that the transition matrix P is a row stochastic matrix. Moreover, it is related to the normalised symmetric positive definite matrix $\hat{W} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} = D^{\frac{1}{2}} P D^{-\frac{1}{2}}$,and as a result, we can write $P = D^{-\frac{1}{2}} \hat{W} D^{\frac{1}{2}}$. It is worth noting in passing that the matrix $\hat{W}$ is related to the normalised Laplacian $L = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} = I - \hat{W}$.

Our aim is to use the steady state random walk on the graph G as an estimate of the maximum probability path across the graph G. The walk commences at the pixel j_1 and proceeds via the sequence of pixel sites $\Gamma = \{j_1, j_2, j_3, ...\}$. If the random walk can be represented by a Markov chain with transition matrix P, then the probability of visiting the pixel sites in the sequence above is

$$P_\Gamma = P(j_1) \prod_{l \in \Gamma} P_{j_{l+1}, j_l} = \prod_{l \in \Gamma} \frac{W_{j_{l+1}, j_l}}{deg(l)}$$

Substituting for the path energy, we have that

$$P_\Gamma = \frac{\exp\left[-\beta \sum_{l \in \Gamma} \mathcal{E}_l\right]}{\prod_{l \in \Gamma} deg(l)} = \frac{1}{Z_\Gamma} \exp[-\mathcal{E}_\Gamma]$$

where $\mathcal{E}_\Gamma = \beta \sum_{l \in \Gamma} \mathcal{E}_l$ and $Z_\Gamma = \prod_{l \in \Gamma} deg(l)$. Hence, the integration path is a Markov chain with energy function $\mathcal{E}_\Gamma$ and partition function Z_Γ. Further, let $Q_t(i)$ be the probability of visiting the pixel site indexed i after t-steps of the random walk and let $Q_t = (Q_t(1), Q_t(2), ...)^T$ be the vector whose components are the probabilities of visiting the sites at time t. After t time steps we have that $Q_t = P^t Q_0$. If $\hat{W}^t$ is the result of multiplying the symmetric positive definite matrix $\hat{W}$ by itself t times, then $P^t = D^{-\frac{1}{2}} \hat{W}^t D^{\frac{1}{2}}$. To develop a spectral method for locating the steady state random walk, we turn to the spectral decomposition of the normalised affinity matrix $\hat{W} = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} = \sum_{i=1}^{N} \lambda_i \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T$ where the λ_i are the eigenvalues of $\hat{W}$ and the $\boldsymbol{\phi}_i$ are the corresponding eigenvectors. By constructing the matrix $\Phi = (\boldsymbol{\phi}_1 | \boldsymbol{\phi}_2 | .... | \boldsymbol{\phi}_N)$ with the eigenvectors of $\hat{W}$ as columns and the matrix $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_N)$ with the eigenvalues as diagonal elements, we can write the spectral decomposition in the more compact form $\hat{W} = \Phi \Lambda \Phi^T$. Since, the eigenvectors of $\hat{W}$ are orthonormal, i.e. $\Phi \Phi^T = I$, we have that $\hat{W}^t = \Phi \Lambda^t \Phi^T$. Substituting the spectral expansion of the matrix $\hat{W}$ into the expression for the state-vector of the random walk at time step t, we find

$$Q_t = D^{-\frac{1}{2}}\Phi\Lambda^t\Phi^T D^{\frac{1}{2}} Q_0 = \left\{\sum_{i=1}^{|V|} \lambda_i^t D^{-\frac{1}{2}} \phi_i \phi_i^T D^{\frac{1}{2}}\right\} Q_o$$

The leading eigenvalue of $\hat{W}$ is unity, i.e. $\lambda_* = 1$. Furthermore, from spectral-graph theory [9], we know that, provided that the graph G is not a bipartite graph, then the smallest eigenvalue $\lambda_{|V|}$ is greater than -1. As a result, when the Markov chain approaches its steady state, i.e. $t \to \infty$, then all but the first term in the above series become negligible. Hence, the steady state random walk is given by $Q_s = \lim_{t\to\infty} Q_t = D^{\frac{1}{2}}\phi_*\phi_*^T D^{-\frac{1}{2}} Q_0$, where ϕ_* is the leading eigenvector of the normalised affinity matrix $\hat{W}$. This establishes that the leading eigenvector of the normalised affinity matrix $\hat{W}$ determines the steady state of the random walk. It is also important to note that the equilibrium equation for the Markov process is $Q_s = PQ_s$, where Q_s is the vector of steady-state site visitation probabilities. Hence, since the leading eigenvalue of P is unity, then it follows that Q_s is the leading eigenvector of P. For a more complete proof of this result see the book by Varga [13] or the review of Lovasz [8].

We aim to visit the pixel sites on the lattice in the order of their steady-state state probabilities. Suppose that the initial state vector for the sites is uniform, i.e. $Q_0 = (\frac{1}{|V|}, \ldots, \frac{1}{|V|})^T$. As a result the steady-state probability of visiting the pixel site i is

$$Q_s(i) = \frac{1}{|V|}\sum_{j=1}^{|V|}\sqrt{\frac{deg(j)}{deg(i)}}\phi_*(i)\phi_*(j) = \frac{1}{|V|}\frac{\phi_*(i)}{\sqrt{deg(i)}}\sum_{j=1}^{|V|}\sqrt{deg(j)}\phi_*(j)$$

Since the summation appearing above is the same for all pixel sites, the probability rank order is determined by the quantity $\hat{\phi}_*(i) = \frac{\phi_*(i)}{\sqrt{deg(i)}}$.

3 Curvature Dependant Weights

The application vehicle used in this paper is the identification of an integration path that can be used to reconstruct a surface from a field of surface normals. The surface integration problem arises in shape-from-shading and shape-from-texture. Our aim is to reconstruct the height function for the surface S from a planar field of surface normals, under the assumption that the image of the surface is formed under orthographic projection. To realise this goal, we require an integration path. This path must traverse or connect the sites of the pixel lattice. By traversing the path, the relative surface height function can be reconstructed. This is done using the trapezium rule to increment the height using the distance travelled on the path and the known slant and tilt angles of the surface normals at different locations on the image plane. In the work reported here the path is one that optimises a graph-spectral criterion that penalises high curvature. To this end, we require a means of gauging the affinity of pixels based on an image plane approximation to the surface curvature. The path must minimise the change in surface normal direction or sectional curvature across the field of surface normals. Suppose that $\boldsymbol{N}_i$ is the surface normal at the point indexed i on the pixel lattice. We note that if the path between the locations i and j can be approximated by a circle of radius R on the surface, then the approximate sectional curvature is $|\hat{\kappa}_{i,j}| = \frac{1}{R_{i,j}}$. If the line connecting the pixel sites on the image plane is of length $s_{i,j}$, then the change in direction of the radius vector of the circle is $\theta_{i,j} = \arccos \boldsymbol{N}_i \cdot \boldsymbol{N}_j$, and as a result $\cos\theta_{i,j} = \boldsymbol{N}_i \cdot \boldsymbol{N}_j$. If the angle $\theta_{i,j}$ is small, then we can make the Maclaurin approximation $\cos\theta_{i,j} \simeq 1 - \frac{\theta_{i,j}^2}{2} = \boldsymbol{N}_i \cdot \boldsymbol{N}_j$. Moreover, the small angle approximation to the radius of curvature of the circle is $R_{i,j} = \frac{s_{i,j}}{\theta_{i,j}}$ and hence

$$\hat{\kappa}_{i,j}^2 = \frac{2(1 - \boldsymbol{N}_i \cdot \boldsymbol{N}_j)}{s_{i,j}^2} \tag{1}$$

To compute the elements of the transition probability matrix we associate to the pair of pixels a cost or energy that is equal to the square of the product of the distance between the sites and sectional curvature of the connecting path. Hence, the transition weight matrix has elements

$$W_{i,j} = \exp\left[-\beta\hat{\kappa}_{i,j}^2 l_{i,j}^2\right] = \exp\left[-2\beta(1 - \boldsymbol{N}_i \cdot \boldsymbol{N}_j)\right] \tag{2}$$

With this definition of the weight matrix, we can also view the recovery of the graph-spectral integration path as one of energy minimisation. The leading eigenvector of the matrix $\hat{W}$ satisfies the condition $\boldsymbol{\phi}_* = \arg\max_{\Phi} \boldsymbol{\phi}^T \hat{W} \phi = \arg\max_{\Phi} \boldsymbol{\phi}^T D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \phi$ We can make the relationship to the raw field of surface normals more explicit by introducing the matrix $F = (\boldsymbol{N}_1|\boldsymbol{N}_2|...|\boldsymbol{N}_{|V|})$ with the surface normals as columns. When the constant β is small, then making use of the Maclaurin expansion of the exponential weighting function we can write $W = \boldsymbol{e}\boldsymbol{e}^T - \beta(\boldsymbol{e}\boldsymbol{e}^T - F^T F)$ where $\boldsymbol{e} = (1, 1, ..., 1)$ is an all-ones vector of length $|V|$. Using this approximation it is a straightforward matter to show that the path is the one that satisfies the condition

$$\boldsymbol{\phi}_* = \arg\max_{\Phi} \boldsymbol{\phi}^T F^T F \phi = \arg\min_{\Phi} \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \boldsymbol{N}_i \cdot \boldsymbol{N}_j \phi(i)\phi(j)$$

Hence, the integration path will minimise the change in surface normal direction.

Our aim is to use the sequence of pixel sites given by the rank-order of the eigenvector coefficients to define a serial ordering for the sites on the pixel lattice. If we visit the sites of the pixel lattice in the order defined by the magnitudes of the coefficients of the leading eigenvector of the normalised affinity matrix, then the path is the steady state of the Markov chain. In this paper, we aim to exploit this property to locate a connected path on the sites of the pixel lattice, and to use this path for surface integration and height recovery. Unfortunately, the path followed by the steady state random walk is not edge-connected. Hence, we need a means of placing the pixel sites in an order in which neighbourhood connectivity constraints are preserved using the elements of the leading eigenvector ϕ_*.

To do this we commence from the pixel site associated with the largest component of ϕ_*, i.e. the largest site probability. We then sort the elements of the scaled leading eigenvector such that they are both in the decreasing magnitude order of the coefficients of the eigenvector, and satisfy edge connectivity constraints on the graph. The procedure is a recursive one that proceeds as follows. At each iteration, we maintain a list of pixel sites visited. At iteration k let the list of pixel sites be denoted by $\mathcal{L}_k$. Initially, $\mathcal{L}_0 = j_0$ where $j_0 = \arg\max_j \phi_*(j)$, i.e. j_0 is the component of ϕ_* with the largest magnitude. Next, we search through the set of first neighbours $\mathcal{N}_{j_0} = \{k|(j_0, k) \in E\}$ of j_o to find the pixel site associated with the largest remaining component of ϕ_*. The second element in the list is $j_1 = \arg\max_{l \in \mathcal{N}_{j_0}} \phi_*(l)$. The pixel site index j_1 is appended to the list of pixel sites visited and the result is $\mathcal{L}_1$. In the kth (general) step of the algorithm we are at the pixel site indexed j_k and the list of pixel sites visited by the path so far is $\mathcal{L}_k$. We search through those first-neighbours of j_k that have not already been traversed by the path. The set of pixel sites is $C_k = \{l|l \in \mathcal{N}_{j_k} \wedge l \notin \mathcal{L}_k\}$. The next site to be appended to the path list is therefore $j_{k+1} = \arg\max_{l \in C_k} \phi_*(l)$. This process is repeated until no further moves can be made. This occurs when $C_k = \emptyset$ and we denote the index of the termination of the path by T. The serial ordering of the pixel sites that results from this edge-based sorting is the integration path $\Gamma = \mathcal{L}_T$.

Our surface height recovery algorithm proceeds along the sequence of pixel-sites defined by the order of the coefficients As we move from pixel-site to pixel-site defined by this path we increment the surface-height function. The trigonometry of the height incrementation process is as follows. At step n of the algorithm we make a transition from the pixel with path-index j_{n-1} to the pixel with path-index j_n. The distance between the pixel-centres associated with this transition is d_n. This distance together with the surface normals $\boldsymbol{N}_{j_n} = [N_{j_n}(x), N_{j_n}(y), N_{j_n}(z)]^T$ and $\boldsymbol{N}_{j_{n-1}} = [N_{j_{n-1}}(x), N_{j_{n-1}}(y), N_{j_{n-1}}(z)]^T$ at the two pixel-sites may be used to compute the change in surface height associated with the transition. The height increment is given by

$$h_n = \frac{d_n}{2}\left\{\frac{N_{j_n}(x)}{N_{j_n}(y)} + \frac{N_{j_{n-1}}(x)}{N_{j_{n-1}}(y)}\right\} \tag{3}$$

If the height-function is initialised by setting $z_{j_0} = 0$, then the centre-height for the pixel with path-index j_n is $z_{j_{n+1}} = z_{j_n} + h_n$.

4 Experiments

We commence with some experiments on synthetic data. The aim here is to determine the accuracy of the surface reconstruction method. To this end, we have generated synthetic surfaces. From the surfaces, we have computed the field of surface normal directions. We have then applied the graph-spectral method to the field of surface normals to recover an estimate of the surface height.

In Figure 1, we show the results obtained for a series of different surfaces. In the top row we show the original synthetic surface. The second row shows the

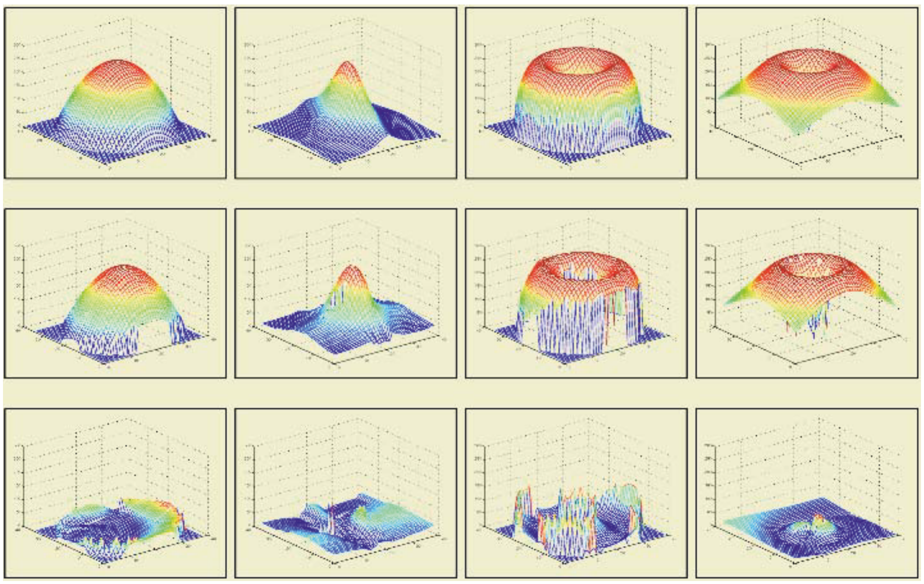


Fig. 1. Top row: Artificially generated data; Second row: Reconstructed surface; Bottom row: Error plot.

surface reconstructed from the field of surface normals. The bottom row shows the absolute error between the ground-truth and reconstructed surface height. From left-to-right, the surfaces studied are a dome, a sharp ridge, a torus and a volcano. In all four cases the surface reconstructions are qualitatively good. For the dome the height errors are greater at the edges of the surface where the slope is largest. In the case of the ridge, there are errors at the crest. For the volcano, there are some problems with the recovery of the correct depth of the “caldera”, i.e. the depression in the centre. For the reconstructed surfaces, the mean-squared errors are 5.6% for the dome, 10.8% for the ridge, 7.8% for the torus and 4.7% for the volcano. Hence, the method seems to have greater difficulty for surfaces containing sharp creases.

We have repeated these experiments under conditions of controlled noise. To do this we have added random measurement errors to the surface height. The measurement errors have been sampled from a Gaussian distribution with zero mean and variance $\sigma = 1$. In Figure 2, we show the result of reconstructing the surfaces shown in Figure 1 when noise has been added. In the left-hand column of the figure we show the field of surface normals for the noise-free surface. In the second column, we show the field of surface normals for the noise-corrupted surface. In the third column, we show the reconstructed height-function obtained from the noisy surface normals. The fourth, i.e. rightmost, column shows the difference between the height of the surface reconstructed from the noisy surface normals and the ground-truth height function. In the case of all four surfaces, the gross structure is maintained. However, the recovered height is clearly noisy.

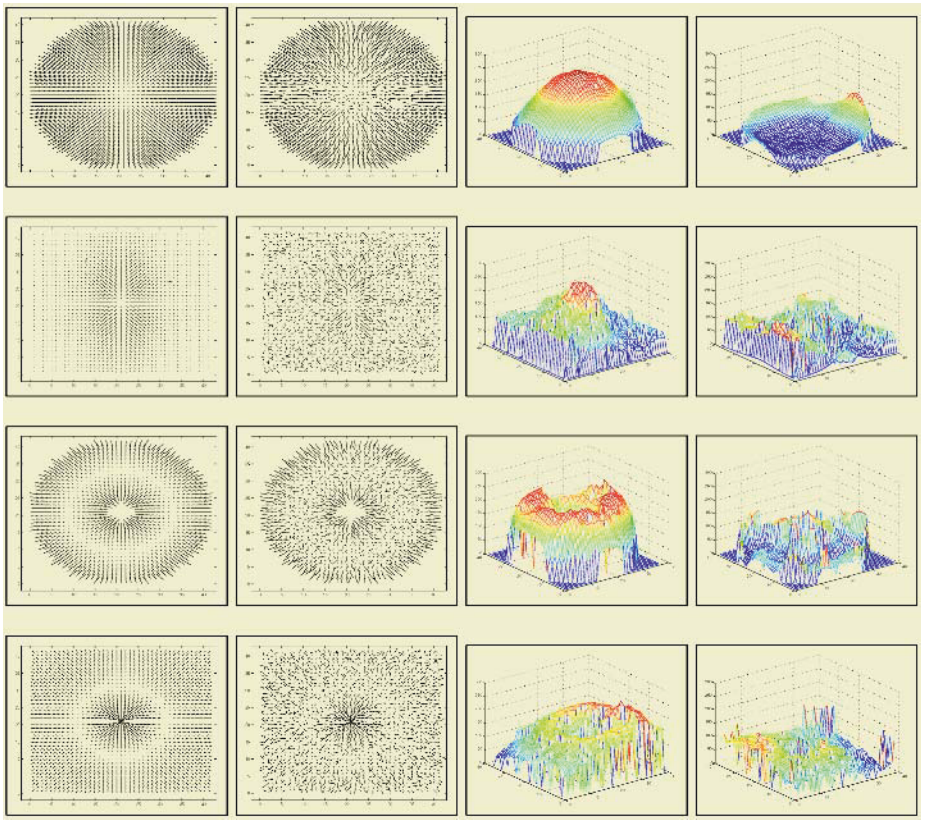


Fig. 2. Left-hand column: Needle-map without added noise; Second Column: Needle-map with Gaussian noise added; Third column: Reconstructed surface; Fourth column: Error plot.

The height difference plots are relatively unstructured. These are important observations. They mean that our graph-spectral method simply transfers errors in surface normal direction into errors in height, without producing structural noise artefacts.

We have also applied our surface recovery method to needle-maps extracted from real-world data using the shape-from-shading algorithm of Worthington and Hancock [14]. In the columns of Figure 3 we show, from left-to-right, the raw image, two views of the reconstructed surface and the integration path. In each case the integration path seems to follow the height contours on the surface, and both the overall geometry and the surface detail of the objects is well reproduced.

5 Conclusions

In this paper, we have demonstrated how steady state random walks can be used for path estimation on pixel lattices. We have illustrated the utility of the

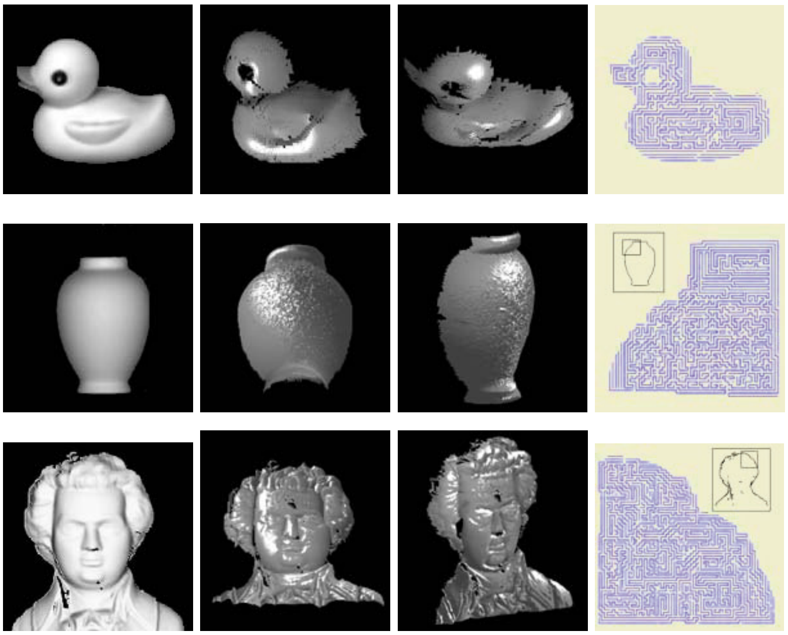


Fig. 3. Results on real-world imagery.

method for purposes of surface integration from fields of surface normals. Our future plans are to develop a more sophisticated model. In this paper, we have sought the path that is the steady state random walk of a Markov chain on a graph. This is a type of diffusion process. A more principled approach may be to pose the recovery of the integration path as the solution of a stochastic differential equation. It may also be interesting to investigate whether the idea of recovering a path using graph-spectral a methods can be applied to other 2D curve enhancement problems.

References

1. L. J. Hubert and R. G. Golledge. Matrix reorganization and dynamic programming: Applications to paired comparisons and unidimensional seriation. *Psychometrika*, 46:429–441, 1981.
2. R. Burkard, V. Deinko, and G. Woeginger. The traveling salesman and the pq-tree. *Mathematics of Operations Research*, 23(3):613–623, 1998.
3. A.L. Yuille, M. Ferraro, and T. Zhang. Image warping for shape recovery and recognition. *Computer Vision and Image Understanding*, 72(3):351–359, 1998.
4. J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
5. S. Sarkar and K. L. Boyer. Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. *Computer Vision and Image Understanding*, 71(1):110–136, 1998.

6. P. Perona and W. T. Freeman. Factorization approach to grouping. In *Proc. ECCV*, pages 655–670, 1998.
7. N. L. Biggs. *Algebraic Graph Theory.* Cambridge University Press, 1993.
8. L. Lovász. Random walks on graphs: a survey. *Bolyai Society Mathematical Studies*, 2(2):1–46, 1993.
9. Fan R. K. Chung. *Spectral Graph Theory.* American Mathematical Society, 1997.
10. D. Cvetković, M. Doob, and H. Sachs. *Spectra of Graphs:Theory and Application.* Academic Press, 1980.
11. Y. Azar, A. Fiat, A. R. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *ACM Symposium on Theory of Computing*, pages 619–626, 2000.
12. J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. ACM-SIAM symposium on discrete algorithms*, pages 668–677, 1998.
13. R. S. Varga. *Matrix Iterative Analysis.* Springer, second edition, 2000.
14. P. L. Worthington and E. R. Hancock. New constraints on data-closeness and needle map consistency for shape-from-shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1250–1267, 1999.

Learning of Form Models from Exemplars

Petra Perner and Silke Jähnichen

Institute of Computer Vision and Applied Computer Sciences IBaI
Körnerstr. 10, 04107 Leipzig
eibaiperner@aol.com
www.ibai-institut.de

Abstract. Model-based image recognition requires a general model of the object that should be detected. In many applications such models are not known a priori, but have to be learnt from examples. In this paper we describe our procedure for the acquisition and learning of general contour models. We developed a modified Procrustes algorithm for alignment and similarity calculation of shapes. Based on the calculated pair-wise similarity we learn groups of shapes. For each group we calculated prototypes. The set of prototypes will be used as models for the detection of object instances in new images.

1 Introduction

One of the most commonly encountered problems in image analysis is the recognition of objects in an image. This can be done by a model-based object recognition method. Such a method works as follows: A shape model is applied to the image and matched according to the pixel points of the image. If the considered pixels have an appearance that is similar to the model points, then the result of the matching process will be a score equal to one for identity and less then one for similarity. The basis for such a method is a good model of the object to be recognized and a good similarity measure. We will consider in this paper the generation of the models from exemplars.

The model can be generated synthetically or from the original image. We are considering the process where the model or a set of models should be learnt from a set of instances elicited from a set of real images. Generally we are attempting to solve the following problems: Create a set of m shape instances from real images, each is represented by a set of arbitrary boundary points. Align these shapes and calculate the pair-wise similarity, partition them into a set of clusters and, for each shape cluster, compute a prototype. The set of prototypes will be used as models for the detection of object instances in new images.

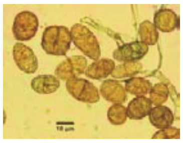
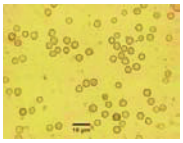
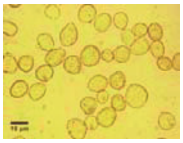
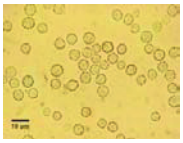
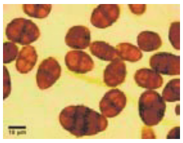
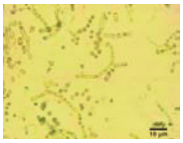
In Section 2 of this paper we give the basic notion and we briefly describe the material used for this study. The acquisition of shape instances is described in Section 3. The alignment of shapes and the computation of the pair-wise similarity is described in Section 4. Clustering and prototype calculation is presented in Section 5 and Section 6. Finally we give results in Section 7. The methods described in this paper are implemented in a program named *CACM Version 1.0* that assists the user in the acquisition of 2-D shapes and learns groups of shape models and their prototypes. An outlook to our research is given in Section 8.

2 Basic Notions and Material Used for this Study

Model-based object recognition can be done based on the object model or based on the contour model. We are considering the contour of an object *S* but not the appearance of the object inside the contour. Therefore we want to elicit from the real image the shape $S_c = \{(x_{ci}, y_{ci})\}\ i = 1 \ldots n_c$ represented by a set of n_c boundary points (x_c, y_c).

The material we used for our study are fungal strains that are naturally 3-D objects, but which are acquired in a 2-D image. These objects have a great variance in the appearance of the shape of the object because of their nature and the imaging constraints. Six fungal strains representing species with different spore types were used for the study (Table 1). A database of images from the spores of these species was produced.

Table 1. Images of Six Different Fungi Strains

		
Alternaria Alternata	Aspergillus Niger	Rhizopus Stolonifer
		
Scopularioupsis Brevicaulis	Ulocladium Botrytis	Wallenia Sebi

3 Acquisition of Shape Cases

We obtain the set of boundary pixels by implementing into our program a function that allows the user to mark the contour S_C of an object *S* by moving the mouse cursor of the computer or by moving an electronic pen over a digitizer tablet (Figure 1). Notice that the sampled points are not required to be landmark co-ordinates [1] or curvature extrema. The user starts labelling an object *S* at an arbitrary pixel s_{start} of its contour. After having traced the complete object the labelling ends at a pixel s_j in the 8-neighbourhood of s_{start}. To obtain the complete set S_C of all boundary pixels we need to ensure that the contour is closed, which means s_j is a direct neighbour of s_{start}. Therefore we insert missing boundary pixels using the Bresenham [2] procedure.

As a result of the labelling process we obtain set S_C with an amount of n_C ordered, connected points that describe the boundary of object *S* with the highest possible accuracy as far as is possible with this kind of labelling procedure.

Having labelled the contour S_C of the object *S* its boundary pixels are still defined by their absolute position in the 2-D matrix of the original image. In order to describe and compare the shapes of objects it is useful to specify a common co-ordinate system that is invariant for translation. Therefore we transform the set S_c of boundary points to the origin $x = 0$ and $y = 0$.

A following approximation of the contour might reduce this set of pixels to a sufficiently large set of pixels that will speed up the succeeding computation time of the alignment and clustering process. The numbers of pixels in this set will be influenced by the chosen order of the polygon and the allowed approximation error. Our approach to the polygonal approximation is based on the area/length ratio according to Wall and Daniellson [3].

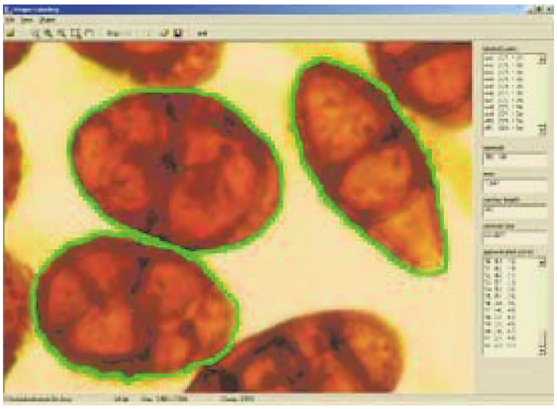


Fig. 1. Labeled and Approximated Shape with Co-ordinates

4 Shape Alignment and Similarity Calculation

4.1 Theory of Procrustes Alignment

The aim of the alignment process is to compare the shapes of two objects in order to define a measure of similarity between them. Consider two shape instances P and O defined by the point-sets $p_i \in R^2, \quad i = 1, 2, ..., n_C$ and $o_j \in R^2, \quad j = 1, 2, ..., n_K$ respectively. The basic task of aligning two shapes consists of transforming one of them (say P) so that it fits in some optimal way the other one (say O). Generally the shape instance $P = \{ (x_i^P , y_i^P) \}$ with $i = 1 ... n_c$ is said to be aligned to the shape instance $O = \{ (x_j^O , y_j^O) \}$ with $j = 1 ... n_k$ if a distance function $d(P, O)$ between the two shapes cannot be decreased by applying to P a transformation Θ. The differences between various alignment approaches is the group of allowed transformations (similarity, rigid, affinity) on one side and the definition of the distance function on the other side.

In our application we use for the Procrustes distance [4] [5], a least-squares type distance function. The alignment of shapes is limited to similarity transformation in order to eliminate differences in rotation and scale of the two shapes P and O.

After computing a similarity transformation between *P* and *O* the Procrustes distance is defined by

$$d(P,O) = \sum_{i=1}^{N_c} \left\| \frac{(p_i - \mu_P)}{\sigma_P} - R(\theta)\frac{(o_i - \mu_O)}{\sigma_O} \right\|^2 \qquad (1)$$

where θ is the rotation matrix, μ_P and μ_O are the centroids of the object *P* and *O* respectively and σ_P and σ_O are the sums of squared distances of each point-set from the centroids.

In its basic form, the Procrustes alignment centers and scales each set of points so that the sum of squared distances of all points in each point-set is unity. Then it is possible to compute a similarity transformation based on these centered pre-shapes. Finally the Procrustes average shape and Procrustes residuals can be evaluated.

4.2 Our Approach to Shape Alignment

As described in Section 3 we are considering a set of shape instances where differences in translation were already eliminated. To compare the shapes of two instances we still have to eliminate differences in rotation and scale. As a measure of similarity we use the Procrustes distance between all points of *P* and their correspondences in *O*.

As can be seen from equation (1) the Procrustes distance requires the knowledge of point correspondences between the shapes *P* and *O*. Therefore we are confronted with the following problems:

1. In our application we use an approximation of the manually labelled set of contour points instead of a predefined number of landmark coordinates. Therefore we cannot guarantee that all shape instances are defined by an identical number of contour points. We can only assume to have nearly the same amount of contour points regardless of which size or shape an object has.
2. The point correspondences between the two shapes of the instance *P* and *O* are completely unknown.
3. As a result of the above-mentioned facts we do not have information about point outliers either.

The Softassign Procrustes Matching algorithm [7] solves the point correspondence problem using deterministic annealing. This algorithm works robust with respect to outlier identification and noise, but is it also a computationally-expensive procedure. Belongie et al. [8] found correspondences between points on the basis of the shape context descriptor.

We are solving the problem of unknown point correspondences by applying an iterative robust point matching algorithm. The outline of our approach to shape alignment is as follows:

For every pair of points $(p_i, o_j) \in P \times O$ we calculate the similarity transformation ψ_{ij} that aligns these two points $\{p_i, o_j\}$. The transformation ψ_{ij} is applied to all points in *P* to obtain the transformed shape instance *P'*, which is defined by the point-set $p'_i \in R^2$, $i = 1, 2, \ldots, n_C$. For every point p'_i we define the nearest neighbour $NN(p'_i)$ in *O* as a point correspondence of p'_i. Note that we do not enforce one-to-one point correspondences. One point in *O* can have more than one point correspondence or even not a single point correspondence in *P*. The sums of squared distances $d_{ij}(P, O)$ between every point correspondence were added. In addition to that we define the quantity $\sqrt{\frac{1}{k} * d_{ij}(P,O)}$ as the mean alignment error $\bar{\varepsilon}_{ij}(O,P)$:

$$\bar{\varepsilon}_{ij}(P,O) = \sqrt{\frac{1}{k} * d_{ij}(P,O)} \tag{2}$$

with

$$d_{ij}(P,O) = \sum_{i=1}^{k} (pi'.x - NN(pi').x)^2 + (pi'.y - NN(pi').y)^2 \tag{3}$$

If the distance $d_{ji}(P, O)$ is smaller then all earlier calculated distances, $d_{min}(P, O)$ is set to $d_{ji}(P, O)$, $\bar{\varepsilon}_{\min}(O,P)$ is set to $\bar{\varepsilon}_{ij}(O,P)$ and ψ_{min} is set to ψ_{ij}.

After having cyclically aligned every possible pair of points $(p_j, o_i) \in P \times O$, we may estimate the similarity between the shape instances *P* and *O* based on the value of $d_{min}(P, O)$.

To ensure that the final measure of similarity ranges from 0 to 1, we normalize the measure $\bar{\varepsilon}_{\min}(O,P)$ to a predefined maximum distance *T*:

$$\bar{\varepsilon}'_{\min} = \frac{\bar{\varepsilon}_{\min}(P,O)}{T} \tag{4}$$

If $\bar{\varepsilon}_{\min}(O,P) = 0$ then the shape instance *P* is identical with the shape instance *O*. With an increasing value of $\bar{\varepsilon}_{\min}(O,P)$ the shape instance *P* is less similar to the shape instance *O*. If $\bar{\varepsilon}_{\min}(P,O) > T$ then the term $\frac{\bar{\varepsilon}_{\min}(P,O)}{T}$ is automatically set to value one.

It is obvious that the constant *T* has a direct influence to the value of the resulting score. The parameter *T* can be defined by the user in the settings dialog of our program *CACM*. For our calculations we set *T* to 35% of the centroid size of *O*. Our investigations showed that this value leads to good results. Figure 2 shows pair-wise aligned shapes and the calculated values of the dissimilarity measure. It can be seen that in case of identity the shapes are superposed. If the similarity score is less than one, we can see a deviation of the two shapes.

5 Clustering

The alignment of every possible pair of objects in our database leads us to *N*N* pair-wise similarity measures between the *N* shape instances. These distances can be collected in an *N x N* matrix where each row and each column corresponds to an instance of our data set.

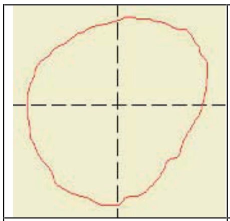
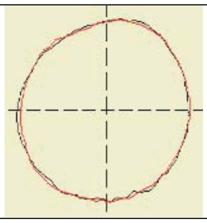
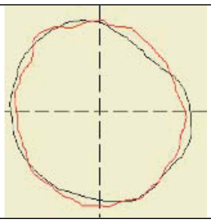
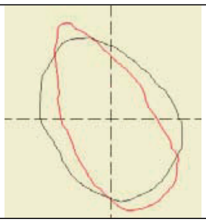
			
$\overline{\varepsilon'_{ij}} = 0$ [identical]	$\overline{\varepsilon'_{ij}} = 0.0271$ [nearly identical]	$\overline{\varepsilon'_{ij}} = 0.199$ [similar]	$\overline{\varepsilon'_{ij}} = 0.5$ [neutral]

Fig. 2. Aligned Shape Instances of Strain Ulocladium Botrytis with Distances

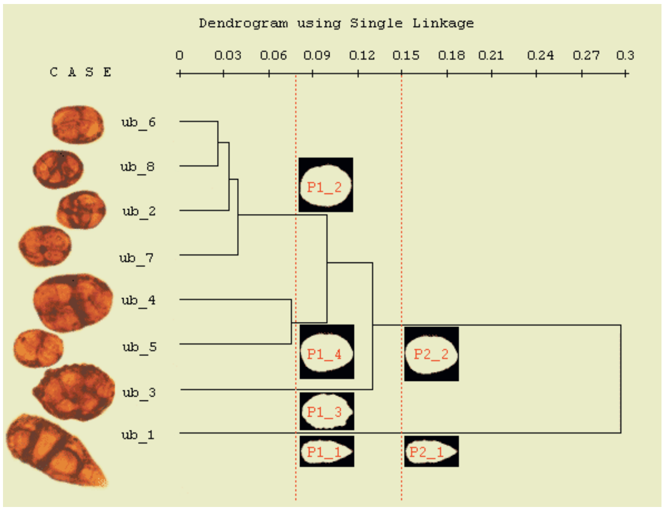


Fig. 3. Dendrogram of Shapes of the Object Ulocladium Botrytis and the calculated Prototypes

We want to point out that we do not obtain a symmetric square matrix where the distance *d(A, B)* between an individual *A* and an individual *B* is identical to the distance *d(B, A)* between individual *B* and individual *A*. This is obviously a lack of precision, but until we do not enforce a strict one-to-one mapping between corresponding points we can only assume that $d(A, B) \approx d(B, A)$.

Based on this similarity matrix we can divide our set of shape instances into groups or clusters. The clustering is done using the single linkage method [6]. The result of the hierarchical cluster analysis can be graphically represented by a dendogram. The dendogram for the shapes of the fungi strain ulocladium botrytis is shown in Figure 3. The dendogram shows the relative distances between all individuals. The merging of individuals into clusters is done with increasing distances (from left to right) until all individuals are combined in only one cluster. The exemplary cut-point (vertical red-dotted line) at a distance 0.15 results in two different clusters. The first cluster which consists only of the object with ub_1 is represented by prototype *P2_1*. The second cluster consists of the other seven objects $\{ub\_2, ub\_3, \ldots, ub\_8\}$ and is represented by prototype $P2\_2$.

6 Prototype Calculation

Each cluster consists of a subset of j shapes $S_1, S_2...S_j$. For each cluster we can now compute a prototype μ that will be the representative of the cluster. This prototype can be calculated by computing the mean over all shapes in a cluster. As result we will get an artificial prototype, a prototype that does not exist in reality. Therefore we decided to calculate the median of all shapes in a cluster.

As the median shape of that cluster we choose the shape instance which has the minimum distance to all other shape instances

$$[\hat{\mu}] = S_{\min} = \arg\inf \sum_{i=1}^{j} d(S_{\min}, S_i) \tag{5}$$

The main advantage of this solution is that the model represents a natural shape that is included in the cluster. An example of using a natural shape instance as the prototype of a cluster is shown in Figure 4a. In contrast to this the arithmetic mean as prototype is shown in Figure 4b. Visually we would favor the median shape as prototype for the cluster since it appears to be more smooth.

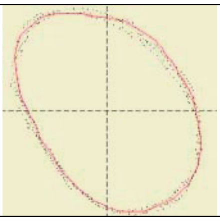
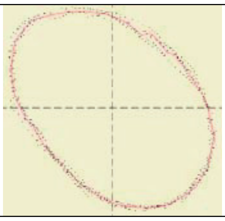
	
(a) using a natural shape instance as prototype	(b) using the arithmetic mean as prototype

Fig. 4. Median of Shapes in a Cluster and Arithmetic Mean of the Shape

7 Results

We have applied our method to six different airborne fungi spores (see Table 1). We have labelled a total of 60 objects for each of the 6 fungal strains. In the following registration process we have aligned every single object with all objects of the same strain to calculate the measure of similarity between them. As a result we have obtained six squared similarity matrices, one for each analyzed fungal strain. These matrices were the input for the following cluster analysis. The outcome of this process was a dendrogram for each of the six different fungi strains. Table 2 presents the number of models for each class at two different cut-points.

For both cut-points we calculated the corresponding set of clusters in each strain class. The prototypes of these clusters were used as models for the later recognition process which is not part of this work. Figure 5 shows as an exemple the database of models for the class Rhizopus Stolonifer at cut-point (1). We can see that a large number of models is required for good detection of the object Rhizopus Stolonifer.

Table 2. Number of Models for two different Cut Points

Classes	max. Distance	Cut-Point (1)	Number of Models	Cut-Point (2)	Number of Models
Alternaria Alternata	0.5264	0.035	23	0.031	34
Aspergillus Niger	0.2936	0.098	3	0.094	5
Rhizopus Stolonifer	0.4275	0.058	16	0.055	22
Scopulariopsis Brevicaulis	0.4911	0.095	8	0.083	10
Ulocladium Botrytis	0.5332	0.043	24	0.040	30
Wallenia Sebi	0.5202	0.050	7	0.046	10

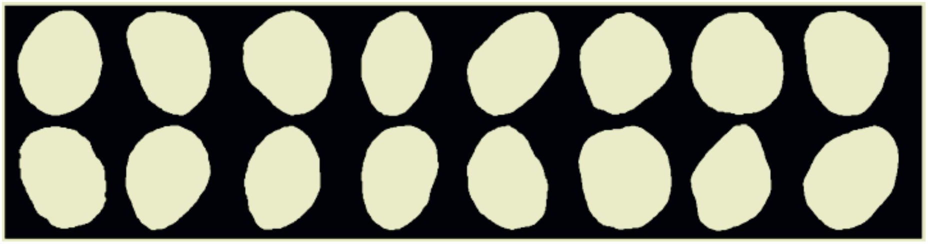


Fig. 5. Database of Models for Strain Rhizopus Stolonifer representing the 16 resulting Clusters

8 Conclusions

The recognition of objects in images can be done based on a model-based recognition procedure. That requires to have a model from the objects which should be recognized. Natural objects have a great variation in shape that does not make it easy to specify a model by hand. Therefore it is necessary to have a computerized procedure that helps to acquire the model from the real objects. We have proposed a method for the acquisition of contour instances and the learning of general shape models. We use the Procustes similarity measure for aligning and determining the similarity between different shapes. Based on the calculated similarity measure we create clusters of similar shapes by using the single linkage method. The mean shape or the median of the cluster is calculated and taken as a prototype of the cluster. The methods are implemented in the program *CACM Version 1.0* which runs on a window PC.

Acknowledgement

The project "Development of methods and techniques for the image-acquisition and computer-aided analysis of biologically dangerous substances BIOGEFA" is sponsored by the German Ministry of Economy BMWA under the grant number 16IN0147.

References

1. A. Hill, Ch. J. Taylor, and A.D. Brett, A Framework for Automatic Landmark Identification Using a New Method of Nonrigid Correspondence, IEEE PAMI, vol. 22, No. 3, pp. 241-251
2. P. Zamperoni, Methoden der digitalen Bildverarbeitung, Vieweg Verlag 1995
3. K. Wall and P.-E. Daniellson, "A fast sequential method for polygonal approximation of digitized curves", Comput. Graph. Image Process. 28, 220-227
4. I.L. Dryden and K.V. Mardia, Statistical Shape Analysis, John Wiley&Sons, 1998.
5. S.R. Lele and J.T. Richtsmeier, An Invariant Approach to Statistical Analysis of Shapes, Chapman &Hall/CRC 2001
6. P. Perner, Data Mining on Multimedia Data, Springer Verlag, lnai 2558, 2003
7. A. Rangarajan, H. Chui and F.L. Bookstein, The Softassign Procrustes Matching Algorithm, Proc. Information Processing in Medical Imaging, pp. 29-42, 1997
8. S. Belongie, J. Malik and J. Puzicha, Shape Matching and Object Recognition Using Shape Contexts, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 24, pp. 509-522, 2002

An Optimal Probabilistic Graphical Model for Point Set Matching

Tibério S. Caetano[1,2], Terry Caelli[1], and Dante A.C. Barone[2]

[1] Department of Computing Science, University of Alberta
Edmonton, AB, Canada T6G 2E8
tcaetano@cs.ualberta.ca, tcaelli@ualberta.ca
[2] Instituto de Informática, Univesidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brazil CP 15064
{caetano,barone}@inf.ufrgs.br

Abstract. We present a probabilistic graphical model for point set matching. By using a result about the redundancy of the pairwise distances in a point set, we represent the binary relations over a simple triangulated graph that retains the same informational content as the complete graph. The maximal clique size of this resultant graph is independent of the point set sizes, what enables us to perform exact inference in polynomial time with a Junction Tree algorithm. The resulting technique is optimal in the Maximum a Posteriori sense. Experiments show that the algorithm significantly outperforms standard probabilistic relaxation labeling.

1 Introduction

The Point Matching Problem is a fundamental one in structural Pattern Recognition, having many applications ranging from stereo matching techniques [4] to the analysis of electrophoresis images [3].

Many algorithms are available in the literature, both for exact and inexact matching. For exact matching, optimal polynomial time algorithms exist, most of them based on ingenious combinations of sorting and searching, inclusive for multidimensional point matching [11]. For inexact matching, there is a variety of proposed techniques, which are generally significantly different from exact techniques, since the need for a similarity measure usually requires that the problem be posed as an optimization one instead of (or in addition to) deterministic search [2].

This paper presents a principled approach for point set matching which is both applicable to exact and inexact problems. Moreover, it is assured to be optimal in the Maximum a Posteriori sense and has polynomial time dependency on the point set sizes. This is possible due to the proposed Markov Random Field (MRF) formulation, which poses point set matching as an exact inference problem that can be effectively solved by Junction Tree methods [7]. The polynomial time performance is obtained from a key observation which exploits concepts of the rigidity of straight line graph embeddings [12, 13]. We also show via a series of experiments that the proposed technique, when applied to inexact problems, presents extremely robust performance under augmentation of the point set sizes. In addition, experiments indicate that the proposed approach is significantly more robust than standard probabilistic relaxation labeling both under varying point set sizes and varying noise levels.

2 The Problem

We consider the problem in $\mathbb{R}^d$, $d \geq 2$, of finding the subset of an S-sized point set (the *codomain pattern*) that best matches another point set (the *domain pattern*) having T points, where $T \leq S$. There may or may not exist distortions due to noise, but if there are, we assume no prior knowledge of the type of noise. We restrict the matching to be invariant up to isometries. In this work, the scale of both point sets is assumed to be the same. The only constraint enforced in the mapping is that it must be a total function: every point in the domain pattern must map to one point in the codomain pattern (but the opposite may not hold).

3 Theoretical Foundation

This section presents the fundamental result that enabled us to formulate the point set matching problem as one of optimal inference in a MRF, while keeping the overall complexity of the algorithm polynomial.

3.1 Global Rigidity: Basic Definitions

We start by presenting some basic definitions of the global rigidity of graphs [12]. A *configuration* is a finite set of n labeled points, $\mathbf{p} = (p_1, \cdots, p_n)$, such that each $p_i \in \mathbb{R}^d$. A *framework* in $\mathbb{R}^d$ consists of a straight line embedding of a graph G with n vertices with configuration $\mathbf{p} = (p_1, \cdots, p_n)$, and is denoted by $G(\mathbf{p})$. In this representation the lengths of the edges correspond to the Euclidean distances between the corresponding vertices. A configuration in *general position* (or *general configuration*) in $\mathbb{R}^d$ is such that no $(d+1)$ points lie in the same $(d$-$1)$-dimensional hyperplane. In $\mathbb{R}^2$, this means that no 3 points are collinear.

Two frameworks $G(\mathbf{p})$ and $G(\mathbf{q})$ are said to be *equivalent*, denoted by $G(\mathbf{p}) \equiv G(\mathbf{q})$, if when $\{i, j\}$ is an edge of G, then $||p_i - p_j|| = ||q_i - q_j||$, where $||.||$ is the Euclidean norm. It is said that a configuration $\mathbf{p} = (p_1, \cdots, p_n)$ is *congruent* to $\mathbf{q} = (q_1, \cdots, q_n)$, and are denoted by $\mathbf{p} \equiv \mathbf{q}$, if, for all $\{i, j\} \in \{1, \cdots, n\}$, $||p_i - p_j|| = ||q_i - q_j||$. This is equivalent to saying that congruent configurations are those related by an *isometry*, or a transformation that preserves distances. A framework $G(\mathbf{p})$ is called *globally rigid* in $\mathbb{R}^d$ if $G(\mathbf{p}) \equiv G(\mathbf{q})$ implies $\mathbf{p} \equiv \mathbf{q}$. In other words, a framework is globally rigid when the specification of the edge lengths uniquely specifies the remaining pairwise distances between vertices that are not joined by an edge. In the following we present a key fact about the global rigidity of a special kind of framework, which turns out to allow for the development of an effective technique for point matching.

3.2 Global Rigidity of k-Trees

In order to present the basic result that allows us to develop a model for optimal point matching, we start by reviewing some basic definitions from graph theory [14]. In what follows a complete graph with n vertices is denoted as K_n. We recall that a *framework* is a straight line embedding of a graph.

Definition 1 (k-clique). *A* k-clique *of a graph is a complete subgraph with k vertices.*

Definition 2 (k-tree, base k-clique). *A* k-tree *is a graph that arises from K_k by zero or more iterations of adding a new vertex adjacent to each vertex of a k-clique in an older graph and nonadjacent to the remaining vertices. The k-cliques adjacent to the new vertices are called* base k-cliques*.*

Figure 1 shows the process of creating a k-tree, where $k = 3$. We start with a K_3 graph. Then we add new vertices by connecting them to 3 vertices of any existing base 3-clique. Note that all intermediate graphs generated in this way are themselves legitimate 3-trees.

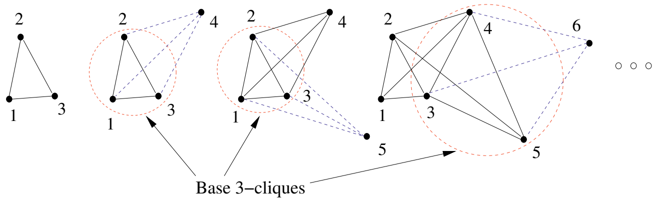


Fig. 1. The process of constructing 3-trees

From these definitions and those of the global rigidity of frameworks, it is possible to prove the following result:

Proposition 1. *Any k-tree framework having each of its base k-cliques in general position in $\mathbb{R}^{k-1}$ is globally rigid in $\mathbb{R}^{k-1}$.*

This follows from the fact that $n+1$ hyper-spheres in $\mathbb{R}^n$ which do not lie in a $(n-1)$-dimensional vector subspace intersect in at most one point. The proof is omitted for space reasons, but can be obtained by induction using the known fact that the intersection of two spheres is a sphere in a lower dimensional subspace.

The direct implication of the result is that the k-tree framework has *exactly* the same informational content than a fully connected framework (since the absent edges have uniquely determined lengths).

In the next section we show how, by taking advantage of this result, we can formulate a MRF model that has precisely the structure of a k-tree, and where exact probabilistic inference is feasible in polynomial time.

4 The Probabilistic Graphical Model

A direct consequence of the definition of a k-tree is that the size of its maximal clique is at most $k+1$, being precisely $k+1$ if the number of vertices is greater than k and being k if the number of vertices equals k.

This observation is what impelled us to propose a probabilistic formulation based on Graphical Models [6], which involves algorithms for optimal inference in probabilistic networks with exponential complexity on the size of the maximal clique of the underlying graph. Since the size of the maximal clique is fixed in $k+1$, the dependency on the number of points is only polynomial, as will be shown. As a result, we obtain a polynomial time procedure for optimal matching. The description of the model and the optimization procedure follow.

4.1 The Model

Here we present a probabilistic graphical model for point set matching. The cardinalities of the domain and codomain pattern sets are denoted, respectively, by T and S. Each point in the domain is associated with a vertex of a graph G_t, and each point in the codomain is associated with a vertex of a graph G_s. The relative distance between a pair $\{i,j\}$ of points in a pattern is seen as an *edge attribute* of the edge that connects vertices i and j in the respective graph. In this formulation, point pattern matching turns out to be an attributed graph matching problem.

The model formulation consists, initially, in defining each of the T vertices in G_t as a random variable (*r.v.*) that can assume S possible values (discrete states), corresponding to the vertices in G_s. Note that in this formulation the solution to the problem (the best match) corresponds to finding the most likely (the best) realization of the set of r.v.'s. The core of a Markov clique condition lies in the compatibilities between joint matches of two *r.v.*'s x_i and x_j in G_t to values θ_k and θ_l in G_s and is defined by

$$\psi_{ij;kl} = p(x_i = \theta_k | x_j = \theta_l)$$

or, in matrix form, for each pair $\{i,j\}$ in G_t, we define the *potential*

$$\psi_{ij} = \psi(x_i, x_j) \equiv p(x_i|x_j) = \frac{1}{Z}\begin{pmatrix} \mathcal{S}(y_t^{ij}, y_s^{11}) & \dots & \mathcal{S}(y_t^{ij}, y_s^{1S}) \\ \vdots & \ddots & \vdots \\ \mathcal{S}(y_t^{ij}, y_s^{S1}) & \dots & \mathcal{S}(y_t^{ij}, y_s^{SS}) \end{pmatrix} \tag{1}$$

where y_a^{bc} denotes the edge attribute between vertices b and c in graph G_a. Z is a normalization constant that equals the sum of all elements in the matrix, in order to keep ψ_{ij} compatible with a probability distribution. $\mathcal{S}$ is a similarity function that measures the compatibility of the two arguments. Several options are available for $\mathcal{S}$ [1], and the specific function $\mathcal{S}$ to be used is not a central issue of this work. Here we choose a similarity function based on the Hyperbolic

Tangent, for which there is evidence of better performance over a Gaussian function [1]:

$$\mathcal{S}(y_t^{ij}, y_t^{kl}) = 1 - \tanh\left[\frac{|y_t^{ij} - y_s^{kl}|}{\sigma}\right]. \tag{2}$$

As a result we can now define a Markov Random Field (MRF) graphical model over the domain graph G_t where nodes in the model correspond to the vertices in G_t whose states are defined by discrete random variables given by the set of vertices in G_s. The local cliques are defined by the connections between the *r.v.'s* of the k-tree topology and their conditional dependencies (Markov condition) are defined via Eq.(1).

Since we have shown that considering only this subset of edges is equivalent to considering all the edges, this MRF actually represents a complete model of the probabilistic interactions in the graph. Figure 2 shows an example of a particular 3-tree MRF. The result of section 3 implies that this model is equivalent to a complete model for matching tasks in $\mathbb{R}^2$, where our experiments will take place. Each connection represents the interaction between the corresponding random variables, which is given by the associated "potential" ψ_{ij}.

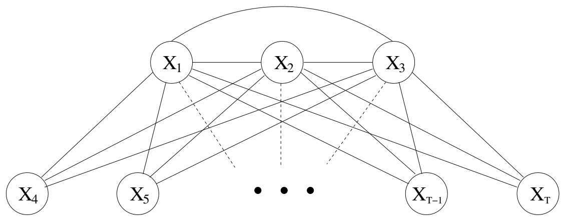


Fig. 2. A possible k-tree model for $k = 3$ (3-tree). Other 3-trees are possible, as long as their base 3-cliques correspond to non-collinear points

Now we need an optimization procedure to infer the most likely realization of the set of random variables, which is precisely the solution to the point set matching problem.

4.2 Optimization

Inference in MRFs typically capitalizes on the Gibbs' distribution to employ simulated annealing to derive the assignment [10]. However, with the above results we can use the Junction Tree (JT) framework, which provides a set of deterministic algorithms for *exact* inference in arbitrary graphical models [7, 6]. A JT of a graph is another graph where the nodes correspond to the maximal cliques of the former graph such that the *Junction Tree property* is satisfied. This property states that all the nodes in the path between any two nodes in the JT must contain the intersection of these two nodes. It is known that the condition for the existence of a JT is that the graph must be chordal [6]. A

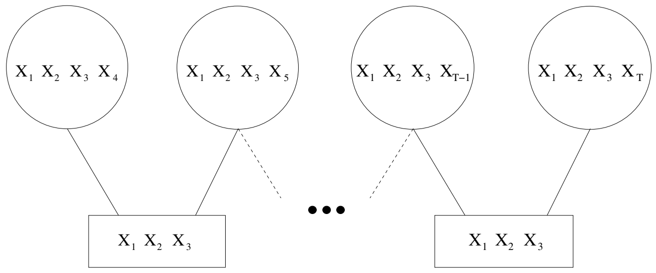


Fig. 3. The Junction Tree obtained from the model in Figure 2

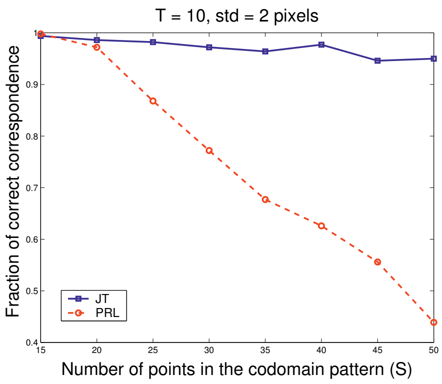
k-tree is a chordal graph, and this allows us to use the JT framework to perform optimization over the model.

Figure 3 shows a JT obtained from the model in Figure 2. The nodes of the JT are denoted by circles in which are listed the nodes of the original graph that correspond to the respective maximal cliques. The rectangles are the so-called *separators*, that contain the intersection of the nodes to which they are linked. Both the nodes and the separators are endowed with "clique potentials", and the optimization process consists in updating these potentials, as explained below. In this paper we applied the HUGIN algorithm [6], an instance of the JT framework, to accomplish exact inference in the 3-tree model shown in Figure 2. The complexity of JT using HUGIN in our k-tree model is $O(S^{d+2}T)$, where d ($d > 1$, $d = k - 1$) is the dimension of the Euclidean space. As a result, the complexity on S and T is polynomial. For the model and experiments presented here (in $\mathbb{R}^2$), we have $O(S^4T)$. The HUGIN algorithm essentially works in two steps: initialization and message-passing. During the initialization, the clique potential of each separator (Φ) is set to unity and the clique potential of each node (Ψ) is introduced. These clique potentials are assembled as an element-by-element product of the pairwise potentials (see Eq. 1) in the respective clique. For example, for the 3-tree model, $\Psi(x_i, x_j, x_k, x_l) = \psi(x_i, x_j).\psi(x_i, x_k).\psi(x_i, x_l)$.

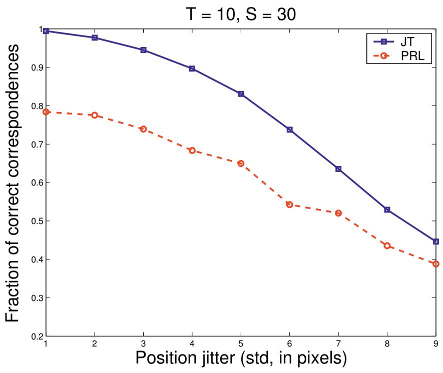
The second step is the message-passing scheme, which involves a transfer of information between two nodes V and W [7]. This operation is defined by the following equations:

$$\Phi_S^* = \max_{V \backslash S} \Psi_V \qquad\qquad \Psi_W^* = \frac{\Phi_S^*}{\Phi_S} \Psi_W$$

where we used standard notation for the current and updated (*) versions of the separator potentials (Φ) and the clique potentials (Ψ). The first equation is a maximization over all sub-configurations in Ψ_V that do not involve the singleton nodes which are common to Φ_S and Ψ_V. The second is simply a normalization step necessary to keep Ψ_W consistent with the updated version of Φ_S (division and multiplication are performed element-by-element). The above potential update rules must respect the following protocol: a node V can only send a message to a node W when it has already received messages from all its other neighbors. If this protocol is respected and the equations are applied until all clique nodes



(a)



(b)

Fig. 4. (a) Performances of JT and PRL when the size of the codomain pattern (S) is increased; $T = 10$, std (noise jitter) = 2 pixels. (b) Performances of JT and PRL when the position jitter (std) in the codomain pattern is increased; $T = 10$, $S = 30$

have been updated, the algorithm assures that the resulting potential in each node and separator of the JT is equal to the (global) maximum a posteriori probability distribution of the set of enclosed singleton nodes [7]. In our particular case, we need the maximum probability for each singleton, what can be obtained by maximizing out the remaining 3 singletons in each of the nodes. The indexes for which the final potentials are maximum are considered the vertices in G_s to which the corresponding vertices in G_t must be assigned.

5 Experiments and Results

We have carried out two experiments. In both of them, we compare our technique (denoted simply as JT) with probabilistic relaxation labeling (PRL) [8]. We have implemented the standard algorithm for PRL presented in [9], for the same reasons than those explained in [5]. PRL does not guarantee global optimal assignments like, under the right conditions and enough iterations, simulated annealing does [10]. However, it is representative of the class of methods that locally update evidence, in parallel, for assignment in terms of the compatibilities between the label of each node and those within its neighborhood. The algorithm runs in $O(S^3T^2)$.

In both experiments, we generated codomain patterns in images of size 256x256 with S points at random (but not coincident) positions. Then, we extracted randomly a subset of T of these points to build the domain pattern.

In the first experiment, we assume that there is a small amount of noise (position jitter) in the codomain pattern, and vary the size of the codomain pattern (in the absence of noise - when the domain pattern is exactly related via an isometry to a subset of the codomain pattern - our method always gives perfect results). A set of 8 increasingly complex matching tasks were carried out, where patterns of size $(T,S) = \{(10,15), (10,20), (10,25), (10,30), (10,35),$

$(10, 40)$, $(10, 45)$, $(10, 50)\}$ were matched using both JT and PRL. For each of these matching tasks, the fraction of correct assignment was calculated based on 1000 trials. The obtained performances are shown in Figure 4(a).

In the second experiment, the sizes of both graphs are kept constant, but the degree of noise in the codomain pattern is increased. We used $T = 10$ and $S = 30$, and the noise consisted in adding independent random numbers drawn from a normal distribution with zero mean and varying standard deviation (denoted as *std* in the figures) to both the x and y coordinates of each point from the codomain pattern. The standard deviation was progressively set to $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ pixels. The final performance is presented in Figure 4(b). Each point in this Figure also represents the result over 1000 trials. The only parameter in the model is σ (see Eq. 2), and it was estimated so that the minimal possible value for $\mathcal{S}(y_t^{ij}, y_s^{kl})$ was 10^{-12}. This is done to prevent underflow and also to guarantee that the similarity function $\mathcal{S}$ will effectively behave monotonically with the observed value of $|y_t^{ij} - y_s^{kl}|$.

6 Discussion

Figure 4(a) shows that, for a fixed amount of noise, the augmentation of the codomain pattern size damages severely the performance of PRL, whereas that of JT remains significantly robust. This is a very important result, since scalability is an important factor in real applications such as stereo matching and image registration. The sensitivity of PRL to matching problems involving many elements has already been reported [5].

In Figure 4(b), we observe that, for fixed sizes of the domain and codomain patterns, the increasing of additive noise still keeps JT preferable for all the experimented values of noise. It is reasonable to expect that both techniques will perform similarly for extremely severe perturbations, when the performance cannot exceed significantly that of pure choice.

7 Conclusion

This work has presented a novel technique for both exact and inexact point pattern matching in $\mathbb{R}^{k-1}$ ($k \geq 3$), which runs in polynomial time and is optimal in the Maximum a Posteriori sense. By representing a point pattern with the correspondent relative pairwise distances between them, we showed that a subset of these distances is sufficient for uniquely determining the remaining ones. From this result, a special class of graph emerges, a k-tree, which has the same representational power as the full graph, but has a maximal clique limited to size $k+1$. By exploiting the Markovian properties of this simple graph structure which has a fixed maximal clique size, we developed a probabilistic graphical model where optimal inference is realizable in polynomial time. The proposed technique presents perfect results in the absence of noise, and is much more robust than standard probabilistic relaxation labeling to varying point set sizes when under noise. The technique is also robust under augmentation of additive noise, where it clearly outperforms standard probabilistic relaxation labeling.

Acknowledgements

We are grateful to NSERC/Canada and CAPES/Brazil for financial support.

References

1. Carcassoni, M., Hancock, E. R.: Spectral correspondence for point pattern matching. Pattern Recognition **36** (2003) 193–204
2. Luo, B., Hancock, E. R.: Structural Graph Matching using the EM Algorithm and Singular Value Decomposition. IEEE Trans. PAMI **23** n.10 (2001) 1120–1136
3. Akutsu, T., Kanaya, K., Ohyama, A., Fujiyama, A.: Point matching under non-uniform distortions. Discrete Applied Mathematics **127** (2003) 5–21
4. Reimann, D., Haken, H.: Stereo Vision by Self-Organization. Biological Cybernetics, **71** n.1 (1994) 17–21
5. Gold, S., Rangarajan, A.: A Graduated Assignment Algorithm for Graph Matching. IEEE Trans. PAMI **18** n. 4 (1996) 377–388
6. Lauritzen, S. L.: Graphical Models. Oxford University Press, New York, NY, (1996)
7. Jordan, M. I.: An Introduction to Probabilistic Graphical Models. In preparation
8. Christmas, W. J., Kittler, J., Petrou, M.: Structural Matching in Computer Vision Using Probabilistic Relaxation. IEEE Trans. PAMI **17** n. 8 (1995) 749–764
9. Rosenfeld, A., Kak, A. C.: Digital Picture Processing, Vol. 1. Academic Press, New York, NY (1982)
10. Geman, S., Geman, D.: Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. IEEE Trans. PAMI **6** n. 6 (1984) 721–741
11. de Rezende, P. J., Lee, D. T.: Point Set Pattern Matching in d-dimensions. Algorithmica **13** (1995) 387-404
12. Connelly, R.: Generic Global Rigidity. Preprint privately provided
13. Jackson, B., Jordán, T.: Connected Rigidity Matroids and Unique Realizations of Graphs. citeseer.nj.nec.com/jackson03connected.html
14. West, D. B.: Introduction to Graph Theory, 2nd Edition. Upper Saddle River, NJ. Prentice Hall (2001)

A New Variational Framework for Rigid-Body Alignment

Tsuyoshi Kato[1], Koji Tsuda[2,1], Kentaro Tomii[1], and Kiyoshi Asai[3,1]

[1] AIST Computational Biology Research Center,
2-43, Aomi, Koto-ku, Tokyo, 135-0064 Japan
{kato-tsuyoshi,koji.tsuda,k-tomii,asai-cbrc}@aist.go.jp
[2] Max Planck Institute of Biological Cybernetics
Spemannstr. 36, 72076 Tübingen, Germany
[3] Graduate School of Frontier Sciences, The University of Tokyo
5-1-5, Kashiwanoha, Kashiwa, 277-8562, Japan

Abstract. We present a novel algorithm for estimating the rigid-body transformation of a sequence of coordinates, aiming at the application to protein structures. Basically the sequence is modeled as a hidden Markov model where each state outputs an ellipsoidal Gaussian. Since maximum likelihood estimation requires to solve a complicated optimization problem, we introduce a variational estimation technique, which performs singular value decomposition in each step. Our probabilistic algorithm allows to superimpose a number of sequences which are rotated and translated in arbitrary ways.

1 Introduction

In the most simple form, the protein structure is represented as a sequence of 3-dimensional vectors, each of which indicates the position of C_α atom of an amino acid [6]. A large amount of structure data are readily available e.g. in the Protein Data Bank. However, it is not easy to compare protein structures because they are translated and rotated in arbitrary ways. A set of proteins have to be superposed correctly to measure meaningful similarities among them. Here one has to estimate the *rigid-body transformation* (i.e. rotation and translation) of each protein correctly[1]. Superposition of protein structures has been a central issue in computational biology, and many methods have been proposed (e.g. [3, 11, 1]). However, most works employ ad hoc or physically-motivated approaches, and probabilistic models (e.g. HMMs) are rather out of focus. One of the reasons would be that the probabilistic models for estimating 3-dimensional rigid-body transformation get so complicated that direct maximization of likelihood e.g. by gradient descent is almost hopeless (we will show details later). However, there

[1] Notice that estimating rigid-body transformation is more difficult than estimating affine transformation [9], because we have to constrain the rotation matrix to be orthogonal. Affine transformation allows rescaling, which is obviously inappropriate for protein structures.

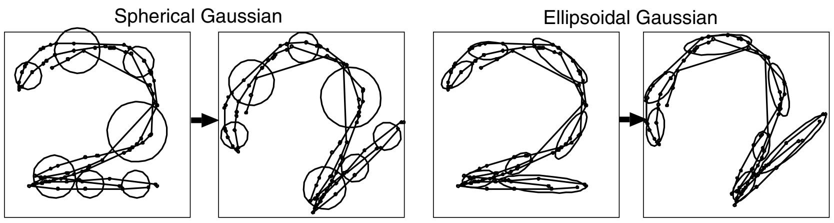


Fig. 1. Comparison of the shape models based on spherical (left) and ellipsoidal (right) Gaussians. When rotated, the covariance matrix of each Gaussian stays the same in the spherical case, but it changes nonlinearly in the ellipsoidal case. This fact makes it difficult to estimate the rotation and transformation by means of the ellipsoidal Gaussians. However, the ellipsoidal Gaussians are much better to describe string-like shapes (e.g. proteins). We will adopt a hierachical model, which combines the best of both worlds.

are crucial advantages of employing probabilistic models. For example, one can attach confidence levels on the estimated rotation and translation. Also one can embed the probabilistic model as one node of a Bayesian network for higher-level inference.

In this paper, we model protein structures by an HMM where each state outputs a 3-dimensional vector subject to an ellipsoidal Gaussian[2]. The mean vectors and covariance matrices of Gaussians have parameters corresponding to rotation and translation. The rigid-body transformation is basically estimated by maximum likelihood with respect to these parameters. The main difficulty is that the covariance matrices are highly nonlinear functions of the rotation parameter (Fig. 1, right). In order to alleviate the computational problem, we replace the ellipsoidal Gaussian with the hierarchical model, that is, a spherical Gaussian distribution whose mean is subject to an ellipsoidal Gaussian. Here we have a new set of hidden variables, that is, the means of spherical Gaussians. Fixing these hidden variables, the tranformation parameters are easily obtained [2], because the covariance matrix of a spherical Gaussian does not change by rotation and translation (Fig. 1, left). Now the estimation of transformation parameters amounts to maximize the expected log-likelihood with respect to the hidden variables, which is tractably solved by a variational technique [5].

The organization of this paper is as follows. In section 2 we describe an HMM shape model for representing a sequence of vectors. In section 3, we provide an efficient algorithm for estimating rigid-body transformation. Section 4 explains how to learn the HMM from a set of sequences. We will show several experiments in section 5 before concluding in section 6.

[•] Typically superposition is helped by side information such as amino acid sequences (i.e. `Leu-Thr-Ser-Ile-`$\cdots$). However, this paper considers more challenging setting that only a sequence of 3-dimensional vectors is available.

2 Shape Models

First of all, let us formulate the shape model without rotation/translation parameters. Let us define the sequence of d-dimensional vector sequence as $\mathbf{X} = [\mathbf{x}(1), \cdots, \mathbf{x}(L)] \in \mathcal{R}^{d \times L}$, where L denotes the length of sequence. In the case of protein structure, L is the number of residues. We use the continuous density hidden Markov model(HMM) as the shape model. The HMM has the following latent variables: $\mathbf{S} = [s(1), \cdots, s(L)]$ where $s(r) \in \{1, \cdots, M\}$ indicates the state at residue r. We use a d-dimensional Gaussian as the output distribution: $p(\mathbf{x}(r)|s(r) = j) \sim \mathcal{N}(\mathbf{m}_j^0, \mathbf{C}_j)$ where $\mathcal{N}()$ denotes a Gaussian density function and $\mathbf{m}_j^0$, $\mathbf{C}_j$ are the mean vector and the covariance matrix of state j, respectively. The density function of an observed sequence $\mathbf{X}$ is given by $f(\mathbf{X}|\Theta) \equiv \sum_{\mathbf{S}} p(\mathbf{S}|\Theta) \prod_{r=1}^{L} p(\mathbf{x}(r)|s(r), \Theta)$ where $\sum_{\mathbf{S}}$ denotes summing over all possible $\mathbf{S}$. For simplicity, let us describe all the parameters by Θ which consists of the parameters of Gaussian, $\mathbf{m}_j^0, \mathbf{C}_j$, as well as the state transition probabilities, a_{ij}, and the initial state probabilities π_i.

The density function of the rotated and translated model is described as

$$p(\mathbf{X}|\Theta, \mathbf{U}, \mathbf{c}) = f(\mathbf{U}\mathbf{X} + \mathbf{c}\mathbf{1}_{1\times L}|\Theta) \tag{1}$$

where $\mathbf{U} \in \mathcal{R}^{d\times d}$ is a rotation matrix, $\mathbf{c} \in \mathcal{R}^{d\times 1}$ is an offset vector and $\mathbf{1}_{1\times L}$ is the $1 \times L$ matrix with all elements equal to one. The rotation matrix $\mathbf{U}$ has to satisfy $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ for orthonormality and $\det(\mathbf{U}) = 1$ for preserving orientation. Assuming that Θ is known, our task is to estimate $\mathbf{U}$ and $\mathbf{c}$ by maximum likelihood:

$$\{\hat{\mathbf{U}}, \hat{\mathbf{c}}\} = \mathrm{argmax}_{\mathbf{U},\mathbf{c}} \log p(\mathbf{X}|\Theta, \mathbf{U}, \mathbf{c}). \tag{2}$$

Let us analyze the difficulty of solving this problem. Consider an easier problem when $\mathbf{S}$ is known, i.e. maximize

$$\log p(\mathbf{X}|\mathbf{S}, \Theta, \mathbf{U}, \mathbf{c}) = -\frac{1}{2}\sum_{r=1}^{L}(\mathbf{U}\mathbf{x}(r) + \mathbf{c} - \mathbf{m}_{s(r)}^0)^\top \mathbf{C}_{s(r)}^{-1}(\mathbf{U}\mathbf{x}(r) + \mathbf{c} - \mathbf{m}_{s(r)}^0) + \text{const.} \tag{3}$$

subject to $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ and $\det(\mathbf{U}) = 1$. Basically, this problem has a quadratic objective function and a set of quadratic constraints, thus it is significantly more complicated than the quadratic programming (i.e. quadratic objective function and linear constraints) [7]. Efficient algorithms such as interior point methods are not straightforwardly applicable for this problem, so typically one has to use general purpose nonlinear optimizers (e.g. gradient descent, Newton methods), which are not so efficient and prone to local minima. Here we do not insist on finding a good approximation algorithm of solving Eq. (2), but rather decompose the covariance of the shape model:

$$\mathbf{C}_j = \mathbf{V}_j^0 + \sigma^2\mathbf{I}. \tag{4}$$

Then, using the property that the convolution of two Gaussians is also a Gaussian, we have the following hierarchical model:

$$p(\mathbf{x}(r)|\mu(r),\Theta) \sim \mathcal{N}(\mu(r),\sigma^2\mathbf{I}), \qquad p(\mu(r)|s(r),\Theta) \sim \mathcal{N}(\mathbf{m}^0_{s(r)},\mathbf{V}^0_{s(r)}) \tag{5}$$

where $\mu(r)$ is a new hidden variable. Now the density function is rewritten as

$$p(\mathbf{X}|\Theta,\mathbf{U},\mathbf{c}) = \sum_{\mathbf{S}} p(\mathbf{S}|\Theta) \prod_{r=1}^{L} \int p(\mu(r)|s(r),\Theta)p(\mathbf{U}\mathbf{x}(r)+\mathbf{c}|\mu(r),\Theta)d\mu(r), \tag{6}$$

where Θ is redefined by $\Theta \equiv \{\mathbf{m}^0_j, \mathbf{V}^0_j, a_{1j}, \cdots, a_{Mj}, \pi_j\}_{j=1}^{M} \cup \sigma^2$. Fixing hidden variables $\mathbf{S}$ and $\mu(r)$, the optimization problem can be solved analytically using the singular value decomposition (SVD) [2]. As we see in the next section, this property allows us to maximize the *negative free energy functional*, which is the lower bound of the log-likelihood.

3 Variational Estimation

We will discuss how to maximize the likelihood in Eq. (6) approximately by the variational EM algorithm [5]. For any distribution $q(\mathbf{S}, \{\mu(r)\}_{r=1}^{L})$, the following inequality holds:

$$\begin{aligned}\log p(\mathbf{X}|\Theta,\mathbf{U},\mathbf{c}) \geq & \left\langle \log p(\mathbf{X},\mathbf{S},\{\mu(r)\}_{r=1}^{L}|\Theta,\mathbf{U},\mathbf{c})\right\rangle_{q(\mathbf{S},\{\mu(r)\}_{r=1}^{L})} \\ & + \mathcal{H}\left(q(\mathbf{S},\{\mu(r)\}_{r=1}^{L})\right).\end{aligned} \tag{7}$$

where $\mathcal{H}(\cdot)$ denotes the entropy function which is defined by: $\mathcal{H}(p(x)) = -\int_x dx p(x)\log p(x)$. We maximize the lowerbound by setting up a parametric model on q and optimize q and $\mathbf{U}$, $\mathbf{c}$, alternately. Typically, q is assumed to be factorized as

$$q(\mathbf{S},\{\mu(r)\}_{r=1}^{L}) = q(\mathbf{S})\prod_{r=1}^{L} q(\mu(r)). \tag{8}$$

Denote by $\mathcal{F}(\mathbf{U},\mathbf{c},q|\Theta,\mathbf{X})$ the right hand side of Eq. (7) where the parametric model is plugged in. In terms of statistical physics, $\mathcal{F}$ is often called the negative free energy functional. Then the variational EM algorithm [5] is represented as follows:

$$q(\mathbf{S}) := \text{argmax}_{q(\mathbf{S})}\,\mathcal{F}(\mathbf{U},\mathbf{c},q|\Theta,\mathbf{X}), \tag{9}$$

$$q(\mu(r)) := \text{argmax}_{q(\mu(r))}\,\mathcal{F}(\mathbf{U},\mathbf{c},q|\Theta,\mathbf{X}), \quad \forall r \tag{10}$$

$$\{\mathbf{U},\mathbf{c}\} := \text{argmax}_{\mathbf{U},\mathbf{c}}\,\mathcal{F}(\mathbf{U},\mathbf{c},q|\Theta,\mathbf{X}) \tag{11}$$

The first two belong to the E-step while the last one belongs to the M-step. Let us solve the first one in Eq. (9). Using the variational method and keeping the other parameters fixed, the current optimal posteriors $q(\mathbf{S})$ are given by: $q(\mathbf{S}) \propto \left(\prod_{r=1}^{T} b_{s(r)}(r)\right) \pi_{s(1)} \left(\prod_{r=1}^{L-1} a_{s(r)s(r+1)}\right)$, where we define $b_{s(r)}(r) \propto \mathcal{N}(\mathbf{m}_j(r)|\mathbf{m}_j^0, \mathbf{V}^0\mathrm{J}) \exp\left(-0.5\,\mathrm{tr}\left((\mathbf{V}_j^0)^{-1}\mathbf{V}_j(r)\right)\right)$. In the other two steps, the function $q(\mathbf{S})$ is not fully needed, but only the following statistics are referred: $\gamma_i(r) \equiv q(s(r) = i) = \sum_{\mathbf{S}} \delta_{i,s(r)} q(\mathbf{S})$ where $\delta_{\cdot,\cdot}$ denotes Kronecker's delta. The statistics $\gamma_i(r)$ can be computed efficiently by applying the forward-backward algorithm [8] as follows. Computing the variables, $\alpha_i(r)$ and $\beta_i(r)$, as

$$\alpha_i(r) = \begin{cases} \pi_i b_i(1) & \text{if } r = 1, \\ b_i(r) \sum_j \alpha_j(r-1) a_{ji} & \text{if } r > 1, \end{cases}$$

$$\beta_i(r) = \begin{cases} 1 & \text{if } r = L, \\ \sum_j \beta_j(r+1) a_{ij} b_j(r+1) & \text{if } r < L, \end{cases}$$

we have $\gamma_i(r) \propto \alpha_i(r)\beta_i(r)$. We can also obtain a by-product of this procedure: $\xi_{ij}(r) = \sum_{\mathbf{S}} \delta_{i,s(r)}\delta_{j,s(r+1)} q(\mathbf{S}) \propto \alpha_i(r) a_{ij} b_j(r+1)\beta_j(r+1)$. The statistics $\xi_{ij}(t)$ are utilized in the next section.

Also, the second one in Eq. (10) is solved analytically as

$$q(\mu(r)) \sim \mathcal{N}(\mathbf{m}(r), \mathbf{V}(r)) \tag{12}$$

where

$$\mathbf{V}(r) = \left(\sigma^{-2}\mathbf{I} + \sum_j \gamma_j(r)(\mathbf{V}_j^0)^{-1}\right)^{-1},$$

$$\mathbf{m}(r) = \mathbf{V}(r)\left(\sigma^{-2}\mathbf{x}(r) + \sum_j (\mathbf{V}_j^0)^{-1}\mathbf{m}_j\right). \tag{13}$$

Finally we will show how to solve the M-step in Eq. (11). Removing the terms which do not depend on $\mathbf{U}$ and $\mathbf{c}$ from $\mathcal{F}$, we have the following:

$$\mathcal{F}_0(\mathbf{U}, \mathbf{c}|\Theta, \mathbf{X}) = -\frac{1}{2\sigma^2}\sum_{r=1}^{L} \|\mathbf{m}(r) - (\mathbf{U}\mathbf{x}(r) + \mathbf{c})\|^2. \tag{14}$$

Thus maximization of $\mathcal{F}_0$ is a least squares problem, which is known to be solved by SVD [2]. Let us define a matrix $\mathbf{\Sigma} = \frac{1}{L}\sum_{r=1}^{L}(\mathbf{m}(r) - \mu_b)(\mathbf{x}(r) - \mu_a)^\top$. where $\mu_a = \frac{1}{L}\sum_{r=1}^{L}\mathbf{x}(r), \mu_b = \frac{1}{L}\sum_{r=1}^{L}\mathbf{m}(r)$. Then decompose $\mathbf{\Sigma} = \mathbf{V}\mathbf{D}\mathbf{W}^\top$ by SVD, where $\mathbf{V}$ and $\mathbf{W}$ are the matrices of left and right singular vectors and $\mathbf{D}$ is the diagonal matrix of singular values. The optimal values of $\mathbf{U}$ and $\mathbf{c}$ are obtained as

$$\mathbf{U} := \mathbf{V}\mathbf{P}\mathbf{W}^\top, \quad \mathbf{c} := \mu_b - \mathbf{U}\mu_a, \tag{15}$$

where

$$\mathbf{P} = \begin{cases} \mathbf{I} & \text{if } \det(\mathbf{V})\det(\mathbf{W}) = 1, \\ \mathrm{diag}(1, \cdots, 1, -1) & \text{if } \det(\mathbf{V})\det(\mathbf{W}) = -1. \end{cases}$$

As seen in Eq. (14), each M-step finds $\mathbf{U}$ and $\mathbf{c}$ which yields the least square error between the transformations of $\mathbf{X}$ and $\mathbf{m}(r)$. So the location of $\mathbf{m}(r)$ is extremely important in this procedure. The latent variable $\mathbf{m}(r)$ can be regarded as the intermediates between the transformation $\mathbf{x}'(r)(\equiv \mathbf{U}\mathbf{x}(r) + \mathbf{c})$ and the corresponding inner Gaussian $\mathcal{N}(\mathbf{m}^0_{s(r)}, \mathbf{V}^0_{s(r)})$. The crucial variable determining $\mathbf{m}(r)$ is σ^2. From the nature that Gaussians merely generate points outside of the circle with the variance, $\mathbf{m}(r)$ is likely to be in the circle with radius σ and centre $\mathbf{x}'(r)$ so as to explain $\mathbf{x}'(r)$ produced by $\mathcal{N}(\mu(r), \sigma^2\mathbf{I})$. Therefore, the larger σ^2 is, the closer $\mathbf{m}(r)$ is to the centre of the Gaussian and the more quickly the optimal solution is found. From these observations, we employ an annealing approach: we start with the large σ^2 and reduce the values step by step. In all simulations provided later, we used the following value of σ^2 in the t-th iteration: $\sigma^2(t) := (49\exp(-t/20) + 1)\sigma_0^2$ where σ_0^2 is the minimum of all the eigen-values in M covariance matrices. $\mathbf{V}^0_j$ are fixed at $\mathbf{V}^0_j = \mathbf{C}_j - \sigma_0^2\mathbf{I}$. This annealing is scheduled so that Eq. (4) holds in the ∞-th iteration.

4 Learning Shape Models

Here we describe a method for learning the shape model parameters Θ from a number of sequences. We again use the variational EM algorithm in order to estimate the shape model parameters, Θ, and the rotation and offset parameters, $\mathbf{U}_n$, $\mathbf{c}_n$, of each sequence simultaneously. Given a training set of sequences, $\{\mathbf{X}_n\}_{n=1}^N$, the objective function for learning is the following log-likelihood function:

$$\mathcal{L}(\Theta, \{\mathbf{U}_n, \mathbf{c}_n\}_{n=1}^N | \{\mathbf{X}_n\}_{n=1}^N) \equiv \sum_{n=1}^{N} \log p(\mathbf{X}_n | \Theta, \mathbf{U}_n, \mathbf{c}_n) \tag{16}$$

where $\mathbf{U}_n$, $\mathbf{c}_n$ are the rotation matrix and offset vector for n-th sequence $\mathbf{X}_n$, respectively. The log-likelihood function in Eq. (16) leads the following negative free energy functional:

$$\mathcal{F}_{\text{shape}}(\{\mathbf{U}_n, \mathbf{c}_n\}_{n=1}^N, q | \Theta, \{\mathbf{X}_n\}_{n=1}^N) = \sum_{n=1}^{N} \mathcal{F}(\mathbf{U}_n, \mathbf{c}_n, q | \Theta, \mathbf{X}_n) \tag{17}$$

by the similar variational approximation to Eq. (8), that is, $q(\mathbf{S}_n, \{\mu_n(r)\}_{r=1}^L) = q(\mathbf{S}_n) \prod_{r=1}^L q(\mu_n(r))$. We then obtain the variational EM algorithm as follows:

$$q(\mathbf{S}_n) := \text{argmax}_{q(\mathbf{S}_n)}\, \mathcal{F}(\mathbf{U}_n, \mathbf{c}_n, q | \Theta, \mathbf{X}_n), \quad \forall n \tag{18}$$

$$q(\mu_n(r)) := \text{argmax}_{q(\mu_n(r))}\, \mathcal{F}(\mathbf{U}_n, \mathbf{c}_n, q | \Theta, \mathbf{X}_n), \ \forall n, \forall r \tag{19}$$

$$\{\mathbf{U}_n, \mathbf{c}_n\} := \text{argmax}_{U_n, \mathbf{c}_n}\, \mathcal{F}(\mathbf{U}_n, \mathbf{c}_n, q | \Theta, \mathbf{X}_n), \quad \forall n \tag{20}$$

$$\Theta := \text{argmax}_{\Theta}\, \mathcal{F}_{\text{shape}}(\{\mathbf{U}_n, \mathbf{c}_n\}_{n=1}^N, q | \Theta, \{\mathbf{X}_n\}_{n=1}^N) \tag{21}$$

The E-step includes Eq. (18) and Eq. (19), whereas the M-step includes Eqs. (20), (21).

In the first problem in Eq. (18), we need not to solve $q(\mathbf{S}_n)$ completely. Here the statistics $q(s_n(r) = i)$ and $q(s_n(r) = i, s_n(r+1) = j)$ are required for solving Eq. (21), which are commonly described as $\gamma_{i,n}(r)$ and $\xi_{i,j,n}(r)$, respectively, in HMM literature (e.g. [8]). Again they can be computed by the forward-backward algorithm [8]. The second problem in Eq. (19) can be solved by the similar update equations as Eq. (12). In this case, we have to replace $\mu(r)$, $\mathbf{m}(r)$, $\mathbf{V}(r)$, $\mathbf{x}(r)$, $\gamma_j(r)$ with $\mu_n(r)$, $\mathbf{m}_n(r)$, $\mathbf{V}_n(r)$, $\mathbf{x}_n(r)$, $\gamma_{j,n}(r)$, respectively. The third problem in Eq. (20) can be solved in the same way as Eq. (15). In the fourth problem in Eq. (21), the optimal solution of σ^2 is described as

$$\sigma^2 := \frac{\sum_{n,r} \left\| \mathbf{U}_n \mathbf{x}_n(r) + \mathbf{c}_n - \mathbf{m}_n(r) \right\|^2 + \operatorname{tr} \mathbf{V}_n(r)}{d \sum_n L_n}. \tag{22}$$

The other variables are obtained by vanishing the derivative of Eq. (21) subject to the constraints that $\sum_j a_{ij} = 1$ and $\sum_i \pi_i = 1$. The solutions are described as follows:

$$\mathbf{m}_j^0 := \frac{\sum_{n,r} \gamma_{j,n}(r) \mathbf{m}_n(r)}{\sum_{n,r} \gamma_{j,n}(r)}, \qquad \mathbf{V}_j^0 := \frac{\sum_{n,r} \gamma_{j,n}(r) \mathbf{V}_{n,r,j}}{\sum_{n,r} \gamma_{j,n}(r)}, \tag{23}$$

$$a_{ij} := \frac{\sum_{n=1}^{N} \sum_{r=1}^{L_n - 1} \xi_{i,j,n}(r)}{\sum_{n=1}^{N} \sum_{r=1}^{L_n - 1} \gamma_{i,n}(r)}, \qquad \pi_i := \frac{\sum_{n=1}^{N} \sum_{r=1}^{L_n} \gamma_{i,n}(r)}{\sum_{n=1}^{N} L_n}, \tag{24}$$

where $\mathbf{V}_{n,r,j} \equiv \mathbf{V}_n(r) + \left(\mathbf{m}_n(r) - \mathbf{m}_j^0\right) \left(\mathbf{m}_n(r) - \mathbf{m}_j^0\right)^\top$.

5 Experiments

We first tested the algorithm on on-line handwritten digits '2' and '6', where eight 2-dimensional vector sequences are superposed for each digit (Figure 2). In all simulations in this paper, we set the number of states $M = 7$. The variational EM algorithm found the almost optimal rotations and translations and the common shape in the data set, as shown in Figure 2. Next we will show the superposition of protein sequences. We used eight 3-dimensional structures from the globin family: 4HHB:A, 4HHB:B, 5MBN:-, 1ECD:-, 2LHB:-, 2LH3:-, 2HBG:-, which have also been used in [1, 14]. Although we did not use any additional information such as amino acid sequences or the position of other atoms than C_α, almost perfect superposition was achieved (Figure 3).

One crucial advantage of probabilistic modeling is that it can be used as a building block of a larger probabilistic model. For illustrating this advantage, we actually implemented the mixture of HMMs [12] and applied it to semi-supervised learning (i.e. learning from labeled and unlabeled data) [10]. We used 46 protein structures of three classes (16 Globins, 17 Ig-likes, and 13 TIM-barrels). For each class, six structures are randomly chosen as training data, where two of them are given class labels and the other four stays unlabeled. The remaining samples are used as test data. The confusion matrices averaged over 10 trials are shown in Table 1. When unlabeled samples are involved, the classification accuracy improved significantly.

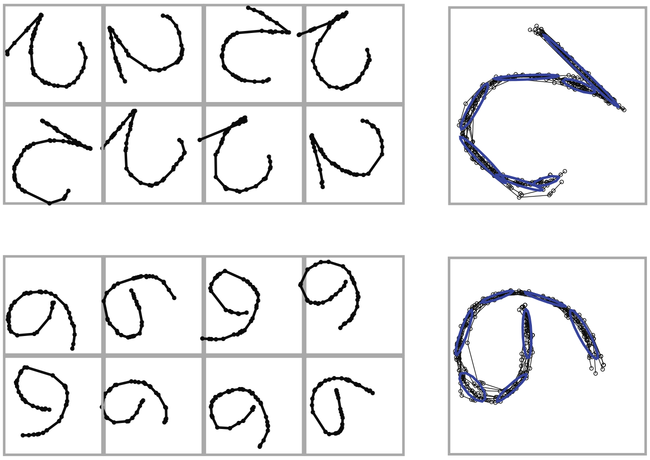


Fig. 2. Superpositions of on-line handwritten digits. Using eight on-line handwritten '2's (top left), we estimate the rotation and the offset parameters as well as common shape parameters by the variational EM algorithm described in section 3, and obtained the resultant superposition (top right). The result of superposition and common shape of eight '6's are also shown in the bottom row. In both cases, almost optimal superpositions are achieved.

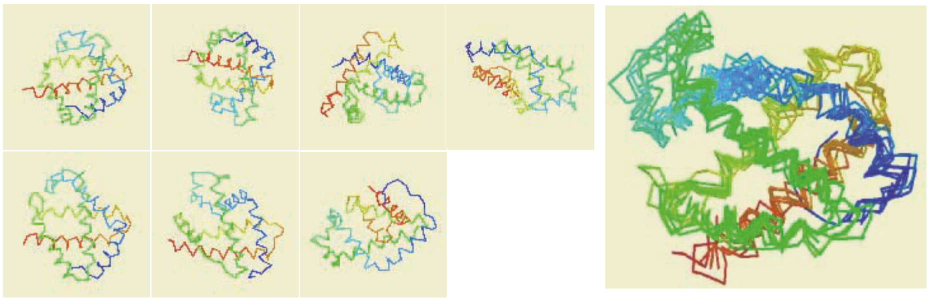


Fig. 3. Superposition of globins. We apply the variational EM algorithm described in section 3 to seven globin structures (left) and achieve almost perfect superposition (right) in spite of using only coordinates of C_α atoms.

6 Conclusion

In this paper, we presented a novel algorithm which estimates the rigid-body transformations from arbitrarily rotated and translated vector sequences. As partly suggested in the previous section, a large number of extensions can be developed from this algorithm due to its probabilistic nature, for example, clustering, detecting outliers, introducing prior knowledge, interpolating missing values, and so on.

One of the most attractive extensions is to combine discriminative methods such as support vector machines. The discriminative methods are often reported

Table 1. Confusion matrices from the semi-supervising experiment. The mixture of HMMs is trained by 2 labeled and 4 unlabeled sequences. Significant improvement is observed when unlabeled samples are incorporated.

	No unlabeled sequences			4 unlabeled sequences		
	Globin	Ig-like	TIM-barrel	Globin	Ig-like	TIM-barrel
Globin	80.0%	1.0%	19.0%	95.0%	1.0%	4.0%
Ig-like	0.0%	64.5%	35.5%	0.0%	82.7%	17.3%
TIM-barrel	0.0%	1.3%	98.8%	0.0%	0.0%	100.0%

to be superior in classification to generative models [4]. Motivated by the fact, several methods which design kernel functions for use in discriminative methods have been proposed (e.g. Fisher kernel [4], marginalized kernel [13] etc.). We might achieve the further improvement by adopting such methods.

References

1. D. Bashford and A. M. Lesk C. Chothia. Determinants of a protein fold: unique features of the globin amino acid sequences. *J. Mol. Biol.*, 196:199–216, 1987.
2. J. H. Challis. A procedure for determining rigid body transformation parameters. *J. Biomechanics*, 28:733–737, 1995.
3. Z. Weng J. D. Szustakowski. Protein structure alignment using a genetic algorithm. *Proteins: structure, function, and genetics*, 38(4):428–440, March 2000.
4. T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *NIPS*, volume 11, pages 487–493. MIT Press, 1999.
5. M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical methods. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 105–161. MIT Press, 1998.
6. D. W. Mount. *Bioinformatics: Sequence and Genome Analysis.* Cold Spring Harbor Laboratory Press, March 2001.
7. J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer Verlag, New York, 1999.
8. L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE*, 77:257–286, 1989.
9. M. D. Revow, C. K. I. Williams, and G. E. Hinton. Using generative models for handwritten digit recognition. *IEEE T. PAMI*, 18(6):592–606, July 1996.
10. M. Seeger. Learning with labeled and unlabeled data. Technical report, Institute for Adaptive and Neural Computation, University of Edinburgh, 2001.
11. I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Engineering*, 11:739–747, 1998.
12. P. Smyth. Clustering sequences with hidden markov models. In *NIPS*, volume 9, pages 648–654. The MIT Press, 1997.
13. K. Tsuda, T. Kin, and K. Asai. Marginalized kernels for biological sequences. *Bioinformatics*, 18(Suppl. 1):S268–S275, 2002.
14. T. D. Wu, T. Hastie, S. C. Schmidler, and D. L. Brutlag. Regression analysis of multiple protein structures. *J. Comput. Biol.*, 5(3):585–595, 1998.

An Error-Tolerant Approximate Matching Algorithm for Attributed Planar Graphs and Its Application to Fingerprint Classification

Michel Neuhaus and Horst Bunke

Department of Computer Science, University of Bern
Neubrückstrasse 10, CH-3012 Bern, Switzerland
{mneuhaus,bunke}@iam.unibe.ch

Abstract. Graph edit distance is a powerful error-tolerant similarity measure for graphs. For pattern recognition problems involving large graphs, however, the high computational complexity makes it sometimes impossible to apply edit distance algorithms. In the present paper we propose an efficient algorithm for edit distance computation of planar graphs. Given graphs embedded in the plane, we iteratively match small subgraphs by locally optimizing structural correspondences. Eventually we obtain a valid edit path and hence an upper bound of the edit distance. To demonstrate the efficiency of our approach, we apply the proposed algorithm to the problem of fingerprint classification.

1 Introduction

In recent years graphs have been recognized as a powerful concept to represent structural patterns. Similarity measures for graphs that are based on an exact structural correspondence such as graph isomorphism and maximum common subgraph are often elegant and quite efficient [1–3]. For real applications, however, it is often difficult to find a graph representation that deals sufficiently well with structural variations between graphs from the same class. Graph matching procedures that allow for such structural variations, so-called error-tolerant algorithms, have been introduced with the development of the graph edit distance [4, 5]. The edit distance of graphs is computed by determining the least costly way to edit one graph into another, given an underlying set of edit operations on graphs and their costs. Due to the enormous computational complexity of the matching problem for general graphs, a number of authors have studied special classes of graphs, such as trees, bounded-valence graphs, and graphs with unique node labels [6–8].

In the present paper we focus on the problem of efficiently matching large attributed planar graphs in the context of the edit distance framework. Planar graphs are interesting in many applications involving images, because common graph representations extracted from an image are planar. A well-known example is region adjacency graphs [9].

In Section 2 of this paper the graph edit distance terminology is introduced and in Section 3 the proposed approximate distance algorithm for planar graphs is described. Next, in Section 4, we demonstrate how planar graph matching can be applied to the fingerprint classification problem and present experimental results. Finally, conclusions are provided in Section 5.

2 Graph Edit Distance

Graph edit distance is an error-tolerant similarity measure for graphs [4, 5]. Structural variations between graphs are modeled with a set of edit operations such as node insertion, node deletion, node substitution, edge insertion, edge deletion, and edge substitution. The key concept is to describe structural differences with the sequence of edit operations that best explain the variations. For this purpose it is common to assign costs to edit operations such that they reflect the strength of the corresponding distortion. The edit distance $d(G, G')$ of two graphs G and G' is then defined as the cost of the least expensive edit path that transforms G into G'. Theoretically, every node of G could be matched to every node of G', as edit operations are defined such that they are able to correct any structural error, and a straight-forward pruning criterion (such as the one for graph isomorphism) does not exist. Hence, it is easy to observe that the computational complexity of the graph edit distance algorithm is exponential in the number of nodes involved. Nonetheless, for small graphs it has proven a powerful graph similarity measure [9, 10]. But for large graphs it becomes computationally infeasible due to its high running time and memory complexity.

3 Approximate Planar Graph Edit Distance

In order to overcome the difficulties arising from the high computational complexity, we propose an approximate, but efficient algorithm for the computation of the edit distance for attributed planar graphs. In the following we assume that our data graphs are provided with a planar embedding, that is, a drawing of the graph in the plane such that none of its edges intersect. An example is shown in Fig. 1. In contrast to exact graph edit distance computation, which defines the distance in terms of the least expensive of all edit paths, we restrict the number of possible edit operations and determine the least expensive member of a smaller set of candidate edit paths. This set of candidate paths is obtained in the course of a process that embeds the graphs under consideration in the plane. If the candidate generation process produces an edit path that is close the the optimal path, the planar edit distance will approximate the graph edit distance well.

For the description of the generation process of the candidate paths we need the following definition. The *neighborhood* of a node u in a graph is defined as the subgraph consisting of node u, all nodes connected to u, and all edges between these nodes. More formally, if we denote a graph by $G = (V, E, \alpha, \beta)$, where V is the set of nodes, E the set of directed edges, $\alpha : V \rightarrow L_V$ the node labeling function, and $\beta : E \rightarrow L_E$ the edge labeling function, the neighborhood $N(u)$ of u in G is defined as the induced subgraph $N(u) = (V_u, E_u, \alpha_u, \beta_u)$ of G, where

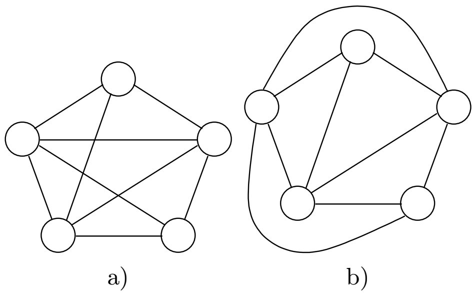


Fig. 1. Illustration of a) a planar graph and b) the same graph embedded in the plane

$$\begin{aligned}
V_u &= \{u\} \cup \{v \in V | (v,u) \in E \text{ or } (u,v) \in E\} \\
E_u &= E \cap (V_u \times V_u) \\
\alpha_u &= \alpha|_{V_U} \\
\beta_u &= \beta|_{E_U} \quad .
\end{aligned}$$

An illustration of a neighborhood is shown in Fig. 2. Note that the embedding of the planar graph is preserved in the neighborhood, that is, there is an order defined on the nodes connected to u.

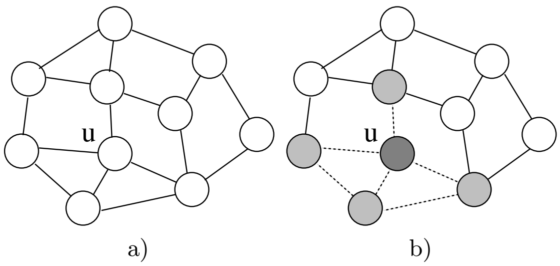


Fig. 2. a) Planar graph and b) graph with marked neighborhood of u

In order to initialize the generation of a candidate path in the process of matching graphs G and G', a *seed substitution* $u \rightarrow u'$ has to be chosen, where u is a node from G and u' a node from G'. Next an optimal matching from subgraph $N(u)$ to subgraph $N'(u')$ (where symbol N refers to graph G and symbol N' to graph G') based on the underlying set of edit operations is to be determined. All new substitutions that occur in this matching are marked for further processing. In consecutive steps the neighborhoods belonging to unprocessed substitutions are processed in the same manner, where substitutions that were previously obtained are preserved in subsequent neighborhood matchings. The matching begins with the seed neighborhood and is iteratively expanded across the two graphs. The result of this procedure is a valid edit path from the first to the second graph. The algorithm is outlined in Table 1.

Table 1. Planar edit distance algorithm

Input: Two planar graphs $G = (V, E, \alpha, \beta)$ and $G' = (V', E', \alpha', \beta')$ to be matched.
Output: A matching between G and G' and the corresponding edit distance, $d(G, G')$

```
0. Determine seed substitution $u_\bullet \rightarrow u'_\bullet$
1. Add seed substitution $u_\bullet \rightarrow u'_\bullet$ to the FIFO queue $Q$
2. Fetch next substitution $u \rightarrow u'$ from $Q$
3. Match neighborhood $N(u)$ to neighborhood $N'(u')$
4. Add new substitutions occurring in step 3 to $Q$
5. If $Q$ is not empty, go to step 2
6. Delete all unprocessed nodes and edges in both $G$ and $G'$
```

Let us consider step 3 of the algorithm, the neighborhood matching, more closely. A neighborhood consists of a center node, a set of adjacent nodes, and edges between these nodes. The set of adjacent nodes can be considered an ordered sequence of nodes due to the planar embedding of the neighborhood. In order to obtain such a node sequence, we randomly start at an adjacent node and traverse all nodes in a clockwise manner. Instead of regarding a neighborhood as a graph to be matched, we can represent a neighborhood as an ordered node sequence and match two neighborhoods simply by finding an optimal node alignment. With this restriction we assume that the optimal neighborhood matching preserves the ordering of the nodes adjacent to the center node. The node alignment can be performed with a cyclic string matching algorithm [11–15], where the sequence of nodes is regarded as a string and the string edit operation costs are derived from the corresponding graph edit operation costs. If we consider graphs with a bounded valence of v, this procedure takes $O(v^2)$. The algorithm terminates after $O(n)$ loops, where n denotes the number of nodes in the graphs. The computational complexity of string matching can further be reduced by preserving previously matched nodes. If we consider a string substitution $u \rightarrow u'$, we require that its operation costs amount to zero if $u \rightarrow u'$ has occurred previously, to infinity if a substitution $u \rightarrow v'$ or $v \rightarrow u'$ with $u \neq v$ and $u' \neq v'$ has occurred previously, and to graph edit operation costs $c(u \rightarrow u')$ otherwise. This means that the present edit path must never be violated by newly added edit operations.

The optimality of the neighborhood matching is determined with respect to the original graph edit operations. New edit operations matching previously obtained operations are added to the edit path in every neighborhood matching. When the algorithm terminates, the generation process yields a valid edit path. The approximate distance value is therefore an upper bound of the true graph edit distance. Since the resulting edit path strongly depends on the seed substitution, we suggest to use several planar distance computations with different seed substitutions and choose the one that returns the minimum matching costs. Promising seed substitution candidates can for instance be found close to the barycenter of the planar embedding in both graphs or may be determined with a local graph matching. If knowledge of the underlying application is available, it may also be utilized to find seed substitution candidates.

4 Application to Fingerprint Classification

Fingerprint recognition tasks can coarsly be divided into verification (one-to-one matching), identification (one-to-many matching), and classification. Fingerprint classification refers to the process of assigning fingerprints to classes with similar characteristics. A large number of fingerprint classification approaches have been reported in the literature, including rule-based [16, 17], syntactic [18], statistical [19], and neural-network-based [20] algorithms. Structural pattern recognition seems to be particularly well suited to the classification problem, as fingerprint analysis naturally involves the comparison of ridge and valley structures. For instance, Maio and Maltoni [9] segment the orientation field of ridge lines into homogeneous regions and convert these into a region adjacency graph. The classification is then performed with an edit distance algorithm. Due to the nature of the segmentation process, the resulting graphs are guaranteed to contain at most ten nodes. Marcialis et al. [21] describe how to improve classification results by fusing this structural algorithm with a statistical classification algorithm. In the present paper, we propose to use larger graphs for the description of the orientation field. Instead of segmenting the orientation field, we combine orientation vectors in a window of constant size and represent them as a single node. In the following, the graph extraction and classification procedure is described in detail. Experimental results are reported in Section 5.

In our fingerprint experiments we use a subset of 450 fingerprints from the NIST-4 database [22]. This database consists of 2000 pairs of grayscale fingerprint images that are classified into one of the classes *arch*, *tented arch*, *left loop*, *right loop*, and *whorl*. An example of a *whorl* image is depicted in Fig. 3a. The image background is segmented from the foreground by computing the grayscale variance in a window around each pixel. The pixels that exhibit a variance lower than a threshold are considered background. For each pixel we then estimate the discrete gradient of the grayscale surface by applying a Sobel operator in the vertical and horizontal direction. After a smoothing process we obtain a ridge orientation field as illustrated in Fig. 3b. Then we represent each pixel in a window as a graph node without attributes. From every node an edge is generated in those two, out of eight, possible directions that best match the vector orthogonal to the average window gradient. A single discrete attribute $\gamma \in \{1, 2, \ldots, 8\}$ is attached to every edge representing the orientation of the edge. The size of the resulting graph depends on the size of the pixel window. In Fig. 3c such a graph is illustrated. The 450 fingerprint graphs from the NIST-4 subset contain an average of 174 nodes and 193 edges per graph at a resolution of 32×32 pixels per window.

We use a simple edit cost function that assigns constant costs p_n to node insertions and deletions, and constant costs p_e to edge insertions and deletions. As nodes are unlabeled, there is no cost for node substitutions, and edge substitution costs are set proportional to the distance of the two involved angles, $d(\gamma, \gamma') = \min\{(\gamma - \gamma') \bmod 8, (\gamma' - \gamma) \bmod 8\}$, for $\gamma, \gamma' \in \{1, 2, \ldots, 8\}$. The ratio of the edge insertion and deletion penalty p_e and the edge substitution cost p_s, i.e. $2p_e/p_s$, determines when an edge deletion followed by an edge insertion is less expensive than an edge substitution.

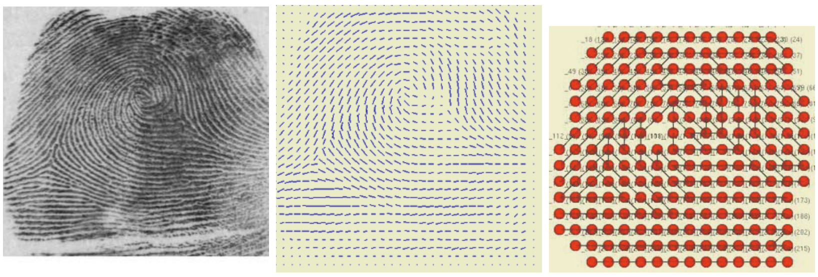


Fig. 3. a) NIST-4 whorl image `f0011`, b) averaged ridge orientation field, and c) orientation graph

Table 2. Running time of exact graph edit distance algorithm (GED, 1 run) and planar edit distance approximation (PED, 50 runs) — empty entries indicate failure due to lack of memory

Nodes	GED	PED
5	<1s	<1s
7	<1s	<1s
9	9s	1s
12	-	1s
20	-	1s
30	-	2s
42	-	5s
169	-	15s

The fingerprint classification is performed by evaluating distances of unknown input graphs to labeled prototype graphs. We adopt a nearest-neighbor paradigm and classify graphs according to a maximum similarity, or minimum edit distance, criterion with respect to the prototype graphs. Note that, with this classification procedure, we rather intend to demonstrate the applicability of the approximate planar edit distance algorithm than provide a thoroughly optimized fingerprint classification system.

5 Experimental Results

To evaluate the running time of the approximate algorithm for planar edit distance computation, we perform the standard graph edit distance computation and the planar edit distance computation for the same pair of graphs. The standard graph edit distance is a deterministic algorithm that yields the exact distance value, whereas the planar edit distance approximation requires several runs to be carried out. The results of several distance computations for pairs of fingerprint graphs are shown in Table 2. For small graphs with less than 10 nodes and edges, the exact graph edit distance computation is computationally

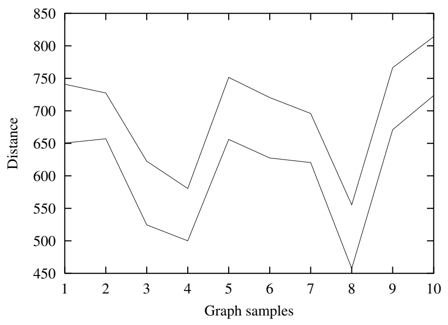


Fig. 4. Exact graph edit distance (lower curve) and approximated planar edit distance (upper curve) for 10 graphs and subgraphs with 10 nodes

feasible. For larger graphs, however, the edit distance search tree exceeds the memory capacity of our testing machine (1024MB). The planar edit distance, on the other hand, provides a result for every tested graph pair, taking only a few seconds for all 50 runs.

Due to memory contraints, the exact edit distance cannot be computed for large graphs. It is therefore difficult to directly evaluate the accuracy of the approximation algorithm. If we delete some nodes from a given graph, however, we obtain a pair of graphs for which a minimum cost edit path is known, so that we can easily compute the exact edit distance between these graphs. The planar edit distance approximation for these graphs is computed in the usual manner without utilizing any knowledge of the special form of the sample graphs. In our first experiment, we delete all but the 10 nodes located closest to the barycenter of the planar embedding from a fingerprint graph and match the resulting graph with the original one. In the second experiment, we use the same procedure to construct subgraphs with 100 nodes. The resulting (known) exact edit distance and the (computed) approximate distance of the first 10 pairs of graphs from NIST-4 are illustrated in Fig. 4. As expected the approximation yields an upper bound of the exact distance. Interestingly enough, the approximation seems to closely follow the exact distance up to an additive constant. If we compute the empiric correlation coefficient of the approximated and the exact distance of the first 100 graphs from NIST-4, we obtain a coefficient of $r = 0.99$ for the subgraphs with 10 nodes and $r = 0.85$ for the subgraphs with 100 nodes. This result indicates that the approximated and the exact distance are strongly correlated in a linear way. In Fig. 5, the correlation can clearly be observed. A regression analysis of the exact distance x and the approximation y according to the linear model $y = \alpha x + \beta$ yields a slope of $\alpha = 0.99$ and an offset of $\beta = 93$ for subgraphs with 10 nodes, and a slope of $\alpha = 1.10$ and an offset of $\beta = 803$ for subgraphs with 100 nodes. A slope of approximately $\alpha = 1$ is equivalent to the reduced linear regression model $y = x + \beta$. We conclude that the difference of the approximation and the exact distance (as illustrated in Fig. 4) is almost

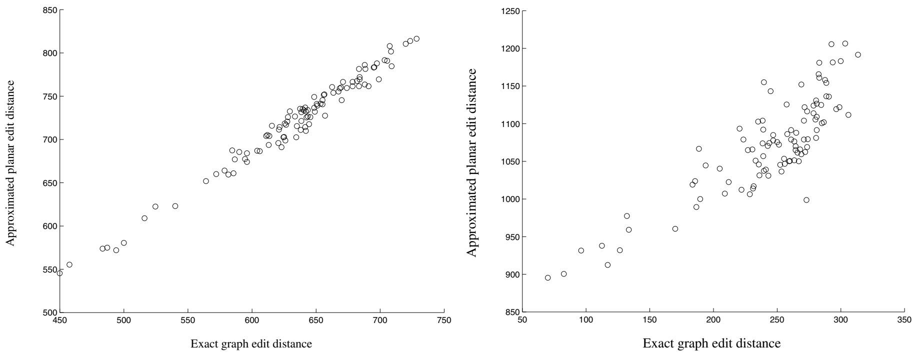


Fig. 5. Exact graph edit distance and approximated planar edit distance for subgraphs with 10 nodes (left) and subgraphs with 100 nodes (right)

Table 3. Fingerprint classification recognition rates per class

Class	Recognition rate
Arch	62.5%
Tented arch	72.5%
Left loop	77.5%
Right loop	85%
Whorl	90%

constant and that the approximation therefore reflects the structural similarity of the underlying graphs well.

In our third experiment we test the applicability of the proposed planar edit distance to the problem of fingerprint classification. The experiment proceeds as follows. For each of the five classes *arch*, *tented arch*, *left loop*, *right loop*, and *whorl* we randomly select 40 input graphs to be classified and another 50 graphs representing the respective fingerprint category. This results in a test set of size 200 and a training, or prototype, set of size 250 graphs. By computing the approximate planar edit distance, we obtain a similarity value between each input graph and each prototype and classify the input graph with a nearest-neighbor classifier. The recognition rates we achieve with this procedure are shown in Table 3. Evaluating some misclassified samples, we note that the recognition errors mainly occur on pairs of fingerprints from different classes that have a high subjective similarity.

6 Conclusions

In the present paper we propose an efficient approximate edit distance algorithm for planar graphs. The graph matching is performed by iteratively extending pairs of matching subgraphs of two given graphs. Our algorithm generates a single edit path between two graphs by locally optimizing the structure correspondence. The optimization is accomplished with an efficient cyclic string matching algorithm.

We evaluate the planar edit distance on fingerprint graphs extracted from grayscale fingerprint images from the NIST-4 database. The edit distance approximation is very fast compared to a standard edit distance computation. The approximated distance values seem to be sufficiently accurate for the measurement of the structural similarity of graphs. Particularly for larger graphs with more than 100 nodes and edges, the planar edit distance offers a good tradeoff between running time and accuracy. In the future we intend to study the influence of the set of prototypical structures on the classification performance and evaluate the fingerprint classification system on larger data sets.

Acknowledgment

This research was supported by the Swiss National Science Foundation NCCR program “Interactive Multimodal Information Management (IM)2” in the Individual Project “Multimedia Information Access and Content Protection”.

References

1. Bunke, H., Shearer, K.: A graph distance metric based on the maximal common subgraph. Pattern Recognition Letters **19** (1998) 255–259
2. Fernandez, M.L., Valiente, G.: A graph distance metric combining maximum common subgraph and minimum common supergraph. Pattern Recognition Letters **22** (2001) 753–758
3. Wallis, W., Shoubridge, P., Kraetzl, M., Ray, D.: Graph distances using graph union. Pattern Recognition Letters **22** (2001) 701–704
4. Sanfeliu, A., Fu, K.: A distance measure between attributed relational graphs for pattern recognition. IEEE Transactions on Systems, Man, and Cybernetics **13** (1983) 353–363
5. Messmer, B., Bunke, H.: A new algorithm for error-tolerant subgraph isomorphism detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 493–504
6. Hopcroft, J., Wong, J.: Linear time algorithm for isomorphism of planar graphs. In: Proc. 6th Annual ACM Symposium on Theory of Computing. (1974) 172–184
7. Luks, E.: Isomorphism of graphs of bounded valence can be tested in ploynomial time. Journal of Computer and Systems Sciences **25** (1982) 42–65
8. Dickinson, P., Bunke, H., Dadej, A., Kraetzl, M.: On graphs with unique node labels. In Hancock, E., Vento, M., eds.: Proc. 4th Int. Workshop on Graph Based Representations in Pattern Recognition. LNCS 2726 (2003) 13–23
9. Lumini, A., Maio, D., Maltoni, D.: Inexact graph matching for fingerprint classification. Machine Graphics and Vision, Special Issue on Graph Transformations in Pattern Generation and CAD **8** (1999) 231–248
10. Ambauen, R., Fischer, S., Bunke, H.: Graph edit distance with node splitting and merging and its application to diatom identification. In Hancock, E., Vento, M., eds.: Proc. 4th Int. Workshop on Graph Based Representations in Pattern Recognition. LNCS 2726 (2003) 95–106

11. Bunke, H., Bühler, U.: Applications of approximate string matching to 2D shape recognition. Pattern Recognition **26** (1993) 1797–1812
12. Lladós, J., Martí, E., Villanueva, J.: Symbol recognition by error-tolerant subgraph matching between region adjacency graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 1137–1143
13. Peris, G., Marzal, A.: Fast cyclic edit distance computation with weighted edit costs in classification. In Kasturi, R., Laurendeau, D., Suen, C., eds.: Proc. 16th Int. Conf. on Pattern Recognition. Volume 4. (2002) 184–187
14. Mollineda, R., Vidal, E., Casacuberta, F.: A windowed weighted approach for approximate cyclic string matching. In Kasturi, R., Laurendeau, D., Suen, C., eds.: Proc. 16th Int. Conf. on Pattern Recognition. (2002) 188–191
15. Robles-Kelly, A., Hancock, E.: String edit distance, random walks and graph matching. Int. Journal of Pattern Recognition and Artificial Intelligence (2004) to appear.
16. Kawagoe, M., Tojo, A.: Fingerprint pattern classification. Pattern Recognition **17** (1984) 295–303
17. Karu, K., Jain, A.: Fingerprint classification. Pattern Recognition **29** (1996) 389–404
18. Rao, K., Balck, K.: Type classification of fingerprints: A syntactic approach. IEEE Transactions on Pattern Analysis and Machine Intelligence **2** (1980) 223–231
19. Jain, A., Prabhakar, S., Hong, L.: A multichannel approach to fingerprint classification. IEEE Transactions on Pattern Analysis and Machine Intelligence **21** (1999) 348–359
20. Wilson, C., Candela, G., Watson, C.: Neural network fingerprint classification. Journal of Artificial Neural Networks **1** (1994) 203–228
21. Marcialis, G., Roli, F., Serrau, A.: Fusion of statistical and structural fingerprint classifiers. In Kittler, J., Nixon, M., eds.: 4th Int. Conf. Audio- and Video-Based Biometric Person Authentication. LNCS 2688 (2003) 310–317
22. Watson, C., Wilson, C.: NIST Special Database 4. Fingerprint Database. (1992)

Comparison of Algorithms for Web Document Clustering Using Graph Representations of Data

Adam Schenker[1], Mark Last[2], Horst Bunke[3], and Abraham Kandel[1,4]

· University of South Florida, Tampa FL 33620, USA
· Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel
· University of Bern, CH-3012 Bern, Switzerland
· Tel-Aviv University, Tel-Aviv 69978, Israel

Abstract. In this paper we compare the performance of several popular clustering algorithms, including *k*-means, fuzzy *c*-means, hierarchical agglomerative, and graph partitioning. The novelty of this work is that the objects to be clustered are represented by graphs rather than the usual case of numeric feature vectors. We apply these techniques to web documents, which are represented by graphs instead of vectors, in order to perform web document clustering. Web documents are structured information sources and thus appropriate for modeling by graphs. We will examine the performance of each clustering algorithm when the web documents are represented as both graphs and vectors. This will allow us to investigate the applicability of each algorithm to the problem of web document clustering.

1 Introduction

The topic of clustering has long been studied in the pattern recognition community. The goal of clustering is to create groups of data items in an unsupervised fashion such that items in the same cluster are similar to each other yet dissimilar to items in other clusters. Many algorithms are presented in the literature [1], such as *k*-means [2], hierarchical agglomerative clustering [3], fuzzy *c*-means [4], and graph partitioning [5]. A more recent method is the global *k*-means method, which attempts to find a "good" initialization state for the *k*-means algorithm [6].

Common to many clustering algorithms is that they are expected to work on data (usually numeric) which is represented by vectors (i.e. sets of attribute values). However, using such vector representations may lead to the loss of the inherent structural information in the original data. Consider, for example, the case of web document clustering. With the increasingly large amount of Internet-based content, it is difficult and costly to categorize and cluster every document manually. In order to deal with this problem, automated clustering of web documents, which allows them be more easily browsed, organized, and cataloged with minimal human intervention, is an important research area [7][8]. Under the vector space model of document representation [3], which is often applied to web documents, each term which may appear on a document is represented by a vector component (or dimension). The values associated with each dimension indicate either the frequency of the term or its relative importance according to some weighting scheme. A problem with representing web documents in this manner is that certain information, such as the order of term appearance, term proximity, term location within the document, and any web specific information, is lost under the vector model. Graphs are a more robust data structure, capable of capturing and maintaining this additional information.

Until recently, there have been no mathematical frameworks available for dealing with graphs in the same fashion that we can deal with vectors. Clustering algorithms require the computation of similarity (or distance) between two objects. This is easily accomplished with vectors in a Euclidean feature space, but until recently it has not been possible with graphs [9][10][11]. Further, a representative of a cluster (such as a centroid) is sometimes required for clustering; again, we have not had such tools available for graphs until lately [12].

Given these new graph-theoretic foundations, a version of the k-means algorithm which can cluster data that is represented by graphs rather than by vectors has been proposed [13]. The experimental results, which compared clustering performance with the traditional vector-based k-means, showed that the performance when representing documents by graphs usually exceeds that of the corresponding vector-based approach.

In this paper we will compare the clustering performance of several different classical clustering algorithms when using data represented by graphs. As mentioned above, it has been shown that the graph representation scheme under the k-means algorithm compares favorably with the vector approach [13] in terms of clustering performance. We have already investigated the effects of various graph distance measures [14] and graph representations [15] on clustering performance. However, the impact of changing the underlying clustering algorithms when clustering data represented by graphs has not been examined. Given graph-theoretic distance and centroid definitions, we can adapt many different clustering algorithms to work with graph-based data in addition to k-means.

Clustering data which is represented by graphs is a novel approach, and it is important for the reader to realize the difference between this method and the well known graph partitioning clustering procedure [5]. In our approach, the data items themselves (web documents for the application presented here) are represented by graphs which are then used in a classical clustering algorithm; by contrast, in graph partitioning clustering, each data item is represented by a single node in a graph representing the clustering problem, with edge weights indicating similarity between nodes (data items). Clustering with graph representations is also discussed in [16], where structural similarity is determined by the subgraph relation and graphs are restricted to having numerical-valued attributes. Our approach uses the size of the maximum common subgraph to calculate real-valued distances between pairs of graphs and does not place any restriction on the attributes or labels associated with the nodes and edges of graphs.

The remainder of the paper is organized as follows. In Sect. 2 we will briefly explain how clustering algorithms can utilize data that is represented by graphs.

We will also present one of our novel graph representations for web documents. The experimental results comparing each clustering algorithm will be given in Sect. 3. Sect. 4 contains our concluding remarks.

2 Clustering Algorithms for Graph Representations

In order to deal with data represented by graphs instead of the traditional case of vectors when using a clustering algorithm, we need two mathematical definitions: distance between graphs and a representative of a set of graphs. A distance measure for graphs, based on the maximum common subgraph (the largest graph shared in common by two graphs) has been proposed [9]:

$$dist(G_1, G_2) = 1 - \frac{|mcs(G_1, G_2)|}{\max(|G_1|, |G_2|)} \tag{1}$$

where G_1 and G_2 are graphs, $mcs(G_1, G_2)$ is their maximum common subgraph, $\max(\ldots)$ is the standard numerical maximum operation, and $|\ldots|$ denotes the size of the graph. The size of a graph is taken in the current work to be the number of nodes and edges contained in the graph. In the general case the computation of the maximum common subgraph is NP-Complete [17], but for our graph representation of web documents (described below) the computation of the maximum common subgraph can be accomplished in polynomial time due to the existence of unique node labels [18].

In order to create a representative of a set of graphs, we can use the median of set of graphs [12]:

$$g = \arg \min_{\forall s \in S} \left(\frac{1}{n} \sum_{i=1}^{n} dist(s, g_i) \right) \tag{2}$$

In basic terms, the median is the graph g in the set of graphs S (where $S = \{g_1, g_2, \ldots, g_n\}$) which has the minimum average distance to all other graphs in the set.

By using Eqs. 1 and 2 instead of vector distance calculations or centroid calculations, respectively, we can arrive at a version of a classical clustering algorithm which can utilize data represented by graphs. In order to represent web documents using graphs and maintain the information that is usually lost in a vector model representation, we use the following method. First, each term (word) appearing in the web document, except for stop words such as “the”, “of”, and “and” which convey little information, becomes a node in the graph representing that document. This is accomplished by labeling each node with the term it represents. Note that we create only a single node for each word even if a word appears more than once in the text. Thus each node in the graph represents a unique word and is labeled with a unique term not used to label any other node. Second, if word a immediately precedes word b somewhere in a “section” s of the web document, then there is a directed edge from the node corresponding to a to the node corresponding to b with an edge label s. We take into account

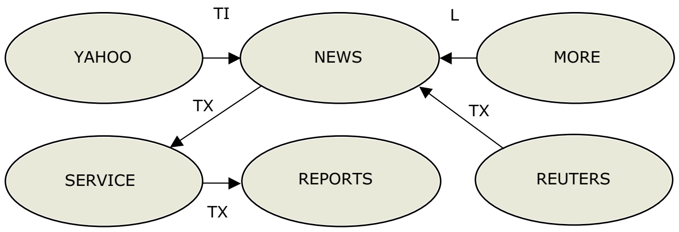


Fig. 1. Example of a graph representation of a document.

certain punctuation (such as a period) and do not create an edge when these are present between two words. Sections we have defined are: *title*, which contains the text related to the web document's title and any provided keywords; *link*, which is text appearing in clickable hyperlinks on the web document; and *text*, which comprises any of the readable text in the web document (this includes link text but not title and keyword text). Next, we remove the most infrequently occurring words for each document by deleting their corresponding nodes, leaving at most m nodes per graph (m being a user provided parameter). This is similar to the dimensionality reduction process for vector representations but with our method the term set can be different for each document. Finally, we perform a simple stemming method and conflate terms to the most frequently occurring form by re-labeling nodes and updating edges as needed. An example of this type of graph representation is given in Fig. 1. The ovals indicate nodes and their corresponding term labels. The edges are labeled according to title (TI), link (L), or text (TX). The document represented by the example has the title "YAHOO NEWS", a link whose text reads "MORE NEWS", and text containing "REUTERS NEWS SERVICE REPORTS". Note also there is no restriction on the form of the graph and that cycles are allowed. While this appears superficially similar to the bigram, trigram, or N-gram methods [19], those are statistically-oriented approaches based on word occurrence probability models. Our method does not require or use the computation of term probabilities.

3 Experimental Results

In order to evaluate the performance of our proposed method of using graphs with the various clustering algorithms, we performed several experiments on two different collections of web documents, called the F-series and the J-series[1]. Each collection contains web documents in HTML format. The F-series originally contained 98 documents assigned to one or more of 17 sub-categories of four major category areas. Since there are multiple sub-category classifications from the same category area for many of these documents, we have reduced the categories to just the four major categories in order to simplify the problem.

[1] The data sets are available under these names at: ftp://ftp.cs.umn.edu/dept/users/boley/PDDPdata/

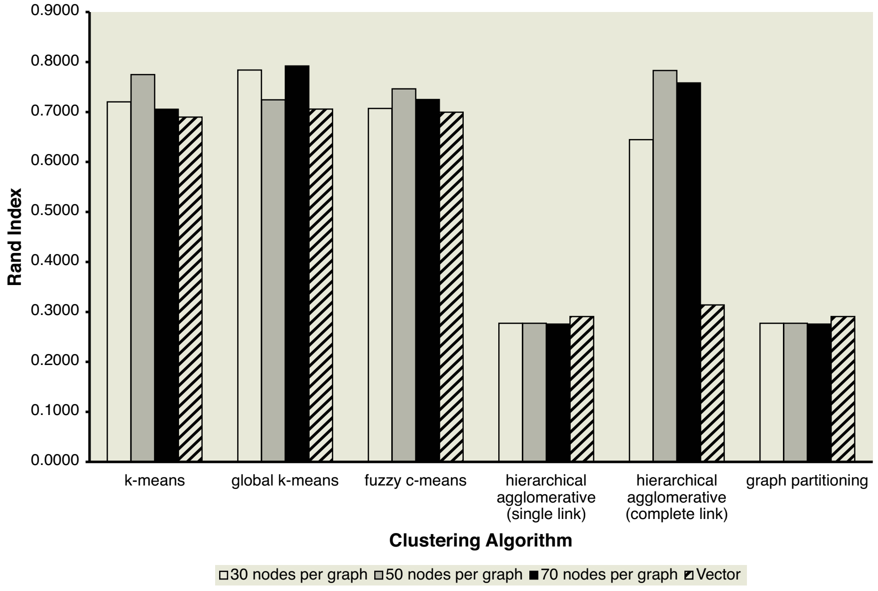


Fig. 2. Experimental results for the F-series data set.

There were five documents that had conflicting multiple classifications (i.e., they were classified to belong to two or more of the four major categories) which we removed, leaving 93 total documents. The J-series contains 185 documents and ten categories. We have not modified this data set. Clustering performance is calculated by the Rand Index [20], which is defined as the number of agreements (i.e. pairs of items which both appear together in a ground truth cluster and a cluster created by the clustering algorithm; *or* pairs of items which appear in different clusters in both ground truth and the created clusters) divided by the number agreements and disagreements (i.e. those cases that are not agreements). Thus the Rand Index is a measure of how closely the clustering created by some clustering algorithm matches ground truth (i.e. it is a measure of clustering accuracy). A value of 1.0 indicates a clustering that exactly matches ground truth.

The clustering performance results for the F-series and the J-series for the various graph-based clustering algorithms are given in Figs. 2 and 3, respectively. The charts show the performance of each algorithm as a group of four columns. From left to right the algorithms compared are: k-means, global k-means, fuzzy c-means, hierarchical agglomerative clustering (single link), hierarchical agglomerative clustering (complete link), and graph partitioning. For our experiments we varied the maximum number of nodes allowed per graph, which is the parameter m described in the previous section. Within each group of columns, the white (leftmost) bar indicates using 30 nodes per graph maximum. The grey bars correspond to using 50 nodes per graph maximum, while the black bars are the results when using 70 nodes per graph maximum. The rightmost (striped) bars represent the performance of the traditional vector-based approach using a

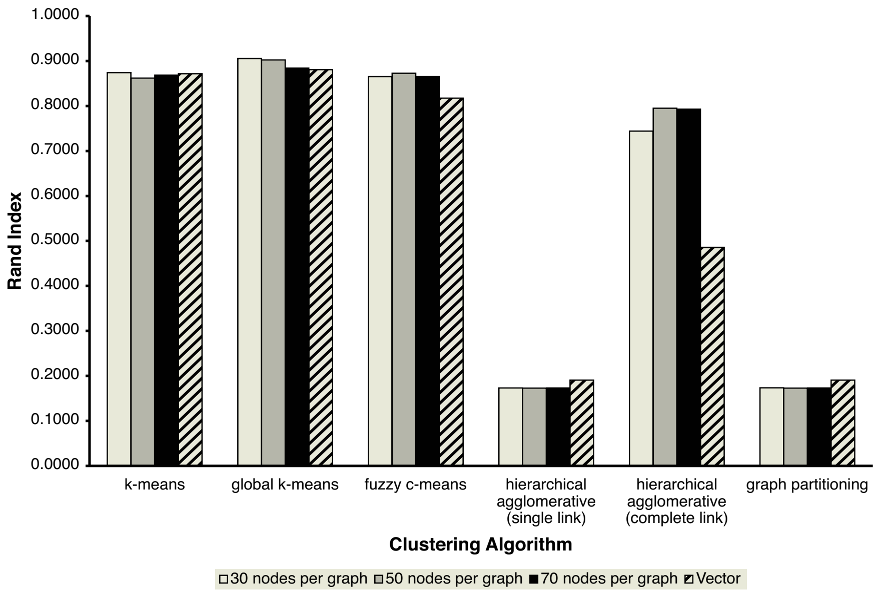


Fig. 3. Experimental results for the J-series data set.

distance measure based on Jaccard similarity [3], which was the best performing in our experiments. Nondeterministic clustering algorithms that use random initializations (k-means and fuzzy c-means) are represented by the average of ten experiments. The results indicate that the single link hierarchical clustering algorithm and the graph partitioning algorithm both performed poorly for all graph sizes and both data sets. Their similar performance is not surprising, as both methods take a similar approach of examining pairs of minimum distance objects; the hierarchical agglomerative algorithm can be seen as a bottom-up procedure while the graph partitioning method is its top-down counterpart. Both of these algorithms can lead to a “chaining effect” where most objects are placed in one large cluster with the other clusters containing only one or a few objects each. Complete link hierarchical agglomerative clustering does not suffer from this phenomenon, and thus its performance is considerably improved over the case of single link. Global k-means was the best performing algorithm overall; it only performed worse than other methods for the F-series data set when using 50 nodes per graph. The graph sizes we selected did not have a consistent influence across algorithms or data sets.

In comparing the performance of the graph-based methods to the traditional vector-based clustering, we see that in most cases clustering with data represented by graphs outperformed the clustering produced with a vector representation for the same clustering algorithm. Only for the single link hierarchical agglomerative clustering and graph partitioning algorithms did the vector approach perform better than all the graph-based experiments in the group; however, the margin of improvement was slight and clustering performance was still poor for all approaches for these two algorithms. The graph clustering approach strongly outperformed the vector model for the complete link hierarchical agglomerative clustering algorithm for both data sets.

4 Conclusions

In this paper we compared the performance of several popular clustering algorithms when using data represented by graphs rather than other conventional representation models, such as vectors. The novelty of this work is utilizing classical clustering algorithms, such as k-means or hierarchical agglomerative clustering, for clustering graph-based data. We compared six algorithms in all: k-means, global k-means, fuzzy c-means, hierarchical agglomerative clustering (single link and complete link), and graph partitioning. We used the Rand Index to measure how well the produced clusters matched ground truth when clustering two web document data sets. The results showed our graph approach outperformed the traditional vector representation methods for most clustering algorithms, strongly for the case of the complete link hierarchical agglomerative clustering algorithm.

Acknowledgments

This work was supported in part by the National Institute for Systems Test and Productivity at the University of South Florida under the U.S. Space and Naval Warfare Systems Command Contract No. N00039–02–C–3244 and by the Fulbright Foundation that has granted Prof. Kandel the Fulbright Research Award at Tel-Aviv University, College of Engineering during the academic year 2003–2004.

References

1. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Computing Surveys **31** (1999) 264–323
2. Mitchell, T.M.: Machine Learning. McGraw-Hill, Boston (1997)
3. Salton, G.: Automatic Text Processing: the Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley, Reading (1989)
4. Klir, G.J., Yuan, B.: Fuzzy Sets and Fuzzy Logic: Theory and Applications. Prentice Hall, Upper Saddle River (1995)
5. Zahn, C.T.: Graph-theoretical methods for detecting and describing gestalt structures. IEEE Transactions on Computers **C-20** (1971) 68–86
6. Likas, A., Vlassis, N., Verbeek, J.J.: The global k-means algorithm. Pattern Recognition **36** (2003) 451–461
7. Strehl, A., Ghosh, J., Mooney, R.: Impact of similarity measures on web-page clustering. In: AAAI-2000: Workshop of Artificial Intelligence for Web Search. (2000) 58–64

8. Zamir, O., Etzioni, O.: Web document clustering: A feasibility demonstration. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. (1998) 46–54
9. Bunke, H., Shearer, K.: A graph distance metric based on the maximal common subgraph. Pattern Recognition Letters **19** (1998) 225–259
10. Fernández, M.L., Valiente, G.: A graph distance metric combining maximum common subgraph and minimum common supergraph. Pattern Recognition Letters **22** (2001) 753–758
11. Wallis, W.D., Shoubridge, P., Kraetz, M., Ray, D.: Graph distances using graph union. Pattern Recognition Letters **22** (2001) 701–704
12. Jiang, X., Muenger, A., Bunke, H.: On median graphs: properties, algorithms, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 1144–1151
13. Schenker, A., Last, M., Bunke, H., Kandel, A.: Clustering of web documents using a graph model. In Antonacopoulos, A., Hu, J., eds.: Web Document Analysis: Challenges and Opportunities. Volume 55 of Machine Perception and Artificial Intelligence. World Scientific Publishing Company (2003) 3–18
14. Schenker, A., Last, M., Bunke, H., Kandel, A.: Comparison of distance measures for graph-based clustering of documents. In: Proceedings of the 4th IAPR-TC15 International Workshop on Graph-Based Representations in Pattern Recognition. Volume 2726 of Lecture Notes in Computer Science., Springer-Verlag (2003) 202–213
15. Schenker, A., Last, M., Bunke, H., Kandel, A.: Graph representations for web document clustering. In: Proceedings of the 1st Iberian Conference on Pattern Recognition and Image Analysis. Volume 2652 of Lecture Notes in Computer Science., Springer-Verlag (2003) 935–942
16. Perner, P.: Data Mining on Multimedia Data. Volume 2558 of Lecture Notes in Computer Science. Springer-Verlag (2003)
17. Messmer, B.T., Bunke, H.: A new algorithm for error-tolerant subgraph isomorphism detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 493–504
18. Dickinson, P., Bunke, H., Dadej, A., Kretzl, M.: On graphs with unique node labels. In: Proceedings of the 4th IAPR-TC15 International Workshop on Graph-Based Representations in Pattern Recognition. Volume 2726 of Lecture Notes in Computer Science. Springer-Verlag (2003) 13–23
19. Tan, C.M., Wang, Y.F., Lee, C.D.: The use of bigrams to enhance text categorization. Information Processing and Management **38** (2002) 529–546
20. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association **66** (1971) 846–850

Heat Kernels, Manifolds and Graph Embedding

Xiao Bai and Edwin R. Hancock

Department of Computer Science
University of York, UK

Abstract. In this paper, we investigate the use of heat kernels as a means of embedding graphs in a pattern space. We commence by performing the spectral decomposition on the graph Laplacian. The heat kernel of the graph is found by exponentiating the resulting eigensystem over time. By equating the spectral heat kernel and its Gaussian form we are able to approximate the geodesic distance between nodes on a manifold. We use the resulting pattern of distances to embed the trees in a Euclidean space using multidimensional scaling. The arrangement of points in this space can be used to construct pattern vectors suitable for clustering the graphs. Here we compute a weighted proximity matrix, and from the proximity matrix a Laplacian matrix is computed. We use the eigenvalues of the Laplacian matrix to characterise the distribution of points representing the embedded nodes. Experiments on sets of shock graphs reveal the utility of the method on real-world data.

1 Introduction

One of the problems that arises in the manipulation of large amounts of graph data is that of clustering. Broadly speaking, there are two approaches to the problem. The first of these is to maintain a class prototype, and to cluster by iteratively merging graphs together. The second approach, which avoids the need to maintain a class prototype, is to apply pairwise clustering methods to the edit distance between graphs. Unfortunately, both of these methods involve computing correspondences between nodes, and since this is potentially an NP-hard problem, the computational overheads can be large. An alternative, which does not involve computing explicit correspondences is to embed the nodes of individual graphs in a low dimensional space and to characterise the graph using the distribution of points corresponding to nodes. Central clustering techniques can then be applied to vectors representing the features of the point-distribution associated with the graphs.

In the mathematics literature, there is a considerable body of work aimed at understanding how graphs can be embedded in manifolds [10]. Broadly speaking there are three ways in which the problem has been addressed. First, the graph can be interpolated by a surface whose genus is determined by the number of nodes, edges and faces of the graph. Second, the graph can be interpolated by a hyperbolic surface which has the same pattern of geodesic (internode) distances as the graph [1] [6]. Third, a manifold can be constructed whose triangulation is the simplicial complex of the graph [12, 2]. A review of methods for efficiently computing distance via embedding is presented in the recent paper of Hjaltason and Samet [4].

In the pattern analysis community, there has recently been renewed interest in the use of embedding methods motivated by graph theory. One of the best known of these is ISOMAP [7]. Here a neighborhood ball is used to convert data-points into a graph, and Dijkstra's algorithm is used to compute the shortest(geodesic) distances between nodes. The matrix of geodesic distances is used as input to MDS. The resulting algorithm has been demonstrated to locate well-formed manifolds for a number of complex data-sets. Related algorithms include locally linear embedding which is a variant of PCA that restricts the complexity of the input data using a nearest neighbor graph, and the Laplacian eigenmap that constructs an adjacency weight matrix for the data-points and projects the data onto the principal eigenvectors of the associated Laplacian matrix (the degree matrix minus the weight matrix) [3]. Collectively, these methods are sometimes referred to as manifold learning theory.

One of the most interesting recent developments in this area has been to establish a link between graph-spectra and the geometry of the underlying manifold [5, 8, 9, 11]. Here considerable insight can be achieved through the analysis of the heat kernel of the graph [5, 9]. According to the heat-equation the time derivative of the kernel is determined by the graph Laplacian. The solution to the heat equation is obtained by exponentiating the Laplacian eigensystem over time. The heat kernel encapsulates the way in which information flows through the edges of the graph over time, and is closely related to the path length distribution on the graph. The graph can be viewed as residing on a manifold whose pattern of geodesic distances is characterised by the heat kernel. For short times the heat kernel is determined by the local connectivity or topology of the graph as captured by the Laplacian, while for long-times the solution gauges the global geometry of the manifold on which the graph resides.

The aim in this paper is to investigate whether the heat kernel can be used for the purposes of embedding graphs, and in particular trees, on a low dimensional manifold. When the manifold on which the graph resides is locally Euclidean, then the heat kernel may be approximated by a Gaussian function of the geodesic distance between nodes. By equating the spectral and Gaussian forms of the kernel, we can make estimates of the geodesic distances. These distances may then be used to embed the graph in a low-dimensional space. Here we follow the ISOMAP algorithm and use multdimensional scaling to locate a low-distortion embedding of the geodesic distances. Once embedded in this space, we can attempt to extract features that characterise the point-distribution of the embbeded nodes and to use them for the purposes of clustering. To do this we construct a weighted Laplacian matrix for the nodes of the embedded graph by exponentiating the negative squared-distance between nodes. The spectrum of eigenvalues of the Laplacian can be used for the purposes of tree clustering and visualisation.

2 Heat Kernels and Riemannian Manifolds

In this section, we develop a method for approximating the geodesic distance between nodes by exploiting the properties of the heat kernel. To commence, suppose that the graph under study is denoted by $G = (V, E)$ where V is the set of nodes and $E \subseteq V \times V$ is the set of edges. Since we wish to adopt a graph-spectral approach we introduce the adjacency matrix A for the graph where

$$A(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

We also construct the diagonal degree matric D, whose elements are given by $D(u, u) = \sum_{v \in V} A(u, v)$. From the degree matrix and the adjacency matrix we construct the Laplacian matrix $L = D - A$, i.e. the degree matrix minus the adjacency matrix. The normalised Laplacian is given by $\hat{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$. The spectral decomposition of the normalised Laplacian matrix is

$$\hat{L} = \Phi \Lambda \Phi^T = \sum_{i=1}^{|V|} \lambda_i \phi_i \phi_i^T \tag{2}$$

where $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_{|V|})$ is the diagonal matrix with the ordered eigenvalues as elements and $\Phi = (\phi_1 | \phi_2 | .... | \phi_{|V|})$ is the matrix with the ordered eigenvectors as columns. Since $\hat{L}$ is symmetric and positive semi-definite, the eigenvalues of the normalised Laplacian fall in the interval $[0, 2]$, i.e. they are all positive. The eigenvector assoicated with the smallest non-zero eigenvector is referred to as the Fiedler-vector. We are interested in the heat equation associated with the Laplacian, i.e.

$$\frac{\partial h_t}{\partial t} = -\hat{L} h_t \tag{3}$$

where h_t is the heat kernel and t is time. The solution is found by exponentiating the Laplacian eigenspectrum, i.e.

$$h_t = \sum_{i=1}^{|V|} \exp[-\lambda_i t] \phi_i \phi_i^T = \Phi \exp[-t\Lambda] \Phi^T \tag{4}$$

The heat kernel is a $|V| \times |V|$ matrix, and for the nodes u and v of the graph G the resulting component is

$$h_t(u, v) = \sum_{i=1}^{|V|} \exp[-\lambda_i t] \phi_i(u) \phi_i(v) \tag{5}$$

When t tends to zero, then $h_t \simeq I - \hat{L} t$, i.e. the kernel depends on the local connectivity structure or topology of the graph. If, on the other hand, t is large, then $h_t \simeq \exp[-t\lambda_m] \phi_m \phi_m^T$, where λ_m is the smallest non-zero eigenvalue and ϕ_m is the assoicated eigevector, i.e. the Fiedler vector. Hence, the large time behaviour is governed by the global structure of the graph.

It is interesting to note that the heat kernel is also related to the path length distribution on the graph. If $P_k(u,v)$ is the number of paths of length k between nodes u and v then

$$h_t(u,v) = \exp[-t] \sum_{k=1}^{|V|^2} P_k(u,v) \frac{t^k}{k!} \tag{6}$$

The path-length distribution is itself related to the eigenspectrum of the Laplacian. By equating the derivatives of the spectral and path-length forms of the heat kernel it is straightforward to show that

$$P_k(u,v) = \sum_{i=1}^{|V|} (1-\lambda_i)^k \phi_i(u)\phi_i(v) \tag{7}$$

When the graph is embedded on a manifold in Riemannian space then the pattern of geodesic distances between nodes on the manifold is the same as the path length distribution. However, when the manifold is locally Euclidean, then the heat kernel is approximated by the Gaussian

$$h_t(u,v) = [4\pi t]^{-\frac{n}{2}} \exp[-\frac{1}{4t^2} d(u,v)^2] \tag{8}$$

where $d(u,v)$ is the distance between the nodes u and v on the Euclidean manifold and n is the dimensionality of the space. The aim here is to find an approximation to the geodesic distance between nodes in the embbeding, by equating the spectral and Gaussian forms for the kernel. The result is

$$d(u,v) = 2\sqrt{-t \ln\left\{ (4\pi t)^{\frac{n}{2}} \sum_{i=1}^{|V|} \exp[-\lambda_i t]\phi_i(u)\phi_i(v) \right\}} \tag{9}$$

We can consider the behaviour of this function for large and small values of t. When t is small, making use of the fact that $h_t = I - \hat{L}t$, we have

$$d(u,v) = 2\sqrt{-t\left\{ \frac{n}{2} \ln[4\pi t] + \ln[1 - \hat{L}(u,v)t] \right\}} \tag{10}$$

Hence, the small t behaviour determined by the local topology of the graph. Moreover, since the second term under the-square-root vanishes, the behaviour near $t = 0$ is independant of the structure of the graph. On the other hand, when t is large we can write

$$d(u,v) = 2\sqrt{-t\left\{ \frac{n}{2} \ln[4\pi t] - \lambda_m t + \ln \phi_m(u)\phi_m(v) \right\}} \tag{11}$$

For very large t we have that $d(u,v) \simeq t\sqrt{\lambda_m}$, and hence the effect of local edge-structure is completely smoothed away.

Although the parameter t potentially provides a route to a graph scale-space, here we set $4\pi t = 1$ to simplify the analysis.

3 Multidimensional Scaling

Our aim is to embed the pattern of geodesic distances in a low dimensional space in a manner which minimises the distortion. For this reason we turn to multidimensional scaling(MDS), which is a procedure which allows data specified in terms of a matrix of pairwise distances to be embedded in a Euclidean space. The pairwise geodesic distances between nodes $d(u,v)$ are used as the elements of an $|V| \times |V|$ dissimilarity matrix S, whose elements are defined as follows

$$S(u,v) = \begin{cases} d(u,v) & \text{if } u \neq v \\ 0 & \text{if } u = v \end{cases} \tag{12}$$

In this paper, we use the classical multidimensional scaling method. The first step of MDS is to calculate a matrix T whose element with row r and column c is given by $T(r,c) = -\frac{1}{2}[d(r,c)^2 - \hat{d}(r,.)^2 - \hat{d}(.,c)^2 + \hat{d}(.,.)^2]$, where $\hat{d}(r,.) = \frac{1}{|V|}\sum_{c=1}^{|V|} d(r,c)$ is the average dissimilarity value over the rth row, $\hat{d}_{.c}$ is the dissimilarly defined average value over the cth column and $\hat{d}(.,.) = \frac{1}{|V|^2}\sum_{r=1}^{|V|}\sum_{c=1}^{|V|} d(r,c)$ is the average dissimilarity value over all rows and columns of the dissimilarity matrix T.

We subject the matrix T to an eigenvector analysis to obtain a matrix of embedding co-ordinates X. If the rank of T is $k, k \leq |V|$, then we will have k non-zero eigenvalues. We arrange these k non-zero eigenvalues in descending order, i.e. $l_1 \geq l_2 \geq \ldots \geq l_k > 0$. The corresponding ordered eigenvectors are denoted by $\boldsymbol{u}_i$ where l_i is the ith eigenvalue. The embedding co-ordinate system for the graphs obtained from different views is $X = [\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_s]$, where $\boldsymbol{f}_i = \sqrt{l_i}\boldsymbol{u}_i$ are the scaled eigenvectors. For the tree-nodes indexed i, the embedded vector of co-ordinates is $\boldsymbol{x}_i = (X_{i,1}, X_{i,2}, ..., X_{i,s})^T$.

4 Characterising the Embedded Point Distribution

Once the nodes of a graph have been embedded, we can attempt to characterise the structure of the graph by summarising the distribution of points associated with the nodes. Although there are many alternatives that can be used for this purpose, including statistical moments, here we opt to use a graph-spectral characterisation of the points. To this end, we commence by computing a weighted proximity matrix W with elements

$$W_{i_1,i_2} = \begin{cases} \exp[\frac{-\|\boldsymbol{x}_{i_1} - x_{i_2}\|_2^2}{2\sigma^2}] & \text{if } \|\boldsymbol{x}_{i_1} - \boldsymbol{x}_{i_2}\|_2 < r \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where σ is a scale constant and r is the radius of a neigbourhood hypersphere in the embedding space. Unfortunately, the matrix W may have negative eigenvalues. Hence, we turn our attention instead to the Laplacian matrix, since it is positive semi-definite and therefore has positive or zero eigenvalues. The Laplacian matrix is $L_E = W - \Delta$ where Δ is diagonal degree matrix with elements $\Delta(i,i) = \sum_{j \in V} W(i,j)$. The spectral decomposition of the Laplacian matrix is

$L_E = \sum_{i=1}^{n} \lambda_i \boldsymbol{e}_i \boldsymbol{e}_i^T$, where λ_i^k is the ith eigenvalue and $\boldsymbol{e}_i$ is the corresponding eigenvector of the Laplacian matrix $\hat{L}$. Our spectral characterisation of the graph is based on the vector of N leading Laplacian eigenvalues $\boldsymbol{B} = (\lambda_1, ..., \lambda_N)^T$. We can perform pattern analysis on sets of graphs by applying clustering or dimensionality reduction techniques to the the vectors of Laplacian eigenvalues.

Our aim is explore the structure of a set of graphs with pattern vectors B_k, $k = 1, M$. There are a number of ways in which the spectral pattern vectors can be analysed. Here, for the sake of simplicity, we use principal components analysis. We commence by constructing the matrix $\mathbf{V} = [\mathbf{B}_1|\mathbf{B}_2|\ldots|\mathbf{B}_k|\ldots|\mathbf{B}_M]$ with the graph feature vectors as columns. Next, we compute the covariance matrix for the elements of the feature vectors by taking the matrix product $\mathbf{C} = \mathbf{V}\mathbf{V}^T$. We extract the principal components directions by performing the eigendecomposition $\mathbf{C} = \sum_{i=1}^{M} l_i \mathbf{u}_i \mathbf{u}_i^T$ on the covariance matrix $\mathbf{C}$, where the l_i are the eigenvalues and the $\mathbf{u}_i$ are the eigenvectors. We use the first s leading eigenvectors (2 or 3 in practice for visualisation purposes) to represent the graphs extracted from the images. The co-ordinate system of the eigenspace is spanned by the s orthogonal vectors $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, .., \mathbf{u}_s)$. The individual graphs represented by the long vectors $\mathbf{B}_k, k = 1, 2, \ldots, M$ can be projected onto this eigenspace using the formula $\mathbf{x}_k = \mathbf{U}^T \mathbf{B}_k$. Hence each graph G_k is represented by an s-component vector $\mathbf{x}_k$ in the eigenspace.

5 Experiments

In this section we experiment with the application of our clustering algorithm to shock graphs. We tested our algorithm on a database of 150 silhouettes of 10 kinds of objects. A representative view of each object is shown in Figure 1.

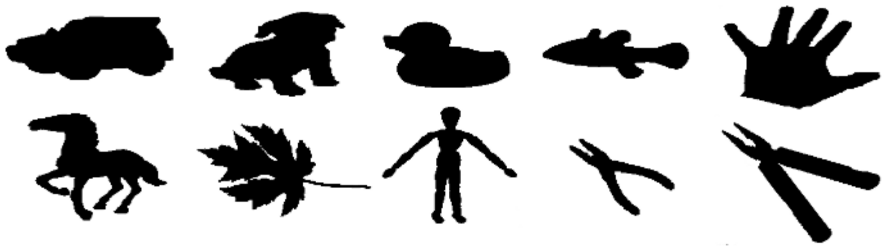


Fig. 1. Sample views of the 10 objects

In our experiments, we will compare the results obtained by using our algorithm with those obtained by using direct spectral analysis of trees. The direct spectral analysis of trees commences by first constructing the Laplacian matrix $\hat{L}$ for the tree. Then we use the leading Laplacian eigenvalues λ_i of the matrix $\hat{L}$ to construct the spectral feature vector $\boldsymbol{B} = (\lambda_1, ..., \lambda_N)^T$. After the spectral feature vectors have been extracted from the trees, we apply PCA (principal component analysis).

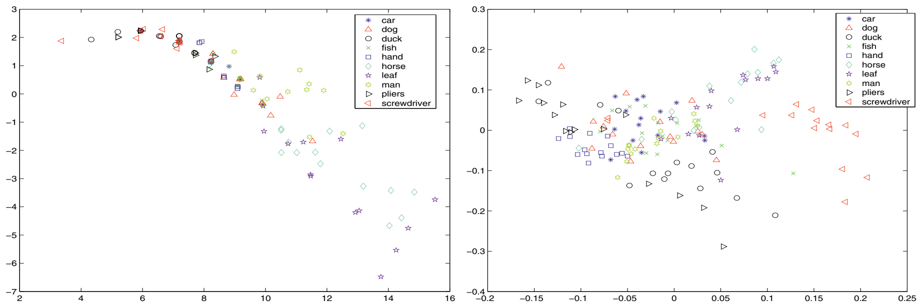


Fig. 2. Direct spectral analysis (left) and heat-kernel analysis (right)

Our first experiment compares our algorithm with the direct spectral analysis of the trees from the entire database of 150 shock trees. In Figure 2 the left-hand panel shows the result of the direct spectral analysis of the shock trees, while the right-hand panel is the result of applying our heat kernel analysis. There is a legend in the top left-hand corner of each plot that explains the shape correspondence of each of the symbols. There are a number of points that can be drawn from these plots. First, in the case of the direct spectral analysis the data distribute themselves along a trajectory in the embedding space. This may be attributable to the problem of co-spectrality of the trees. However, after the heat kernel analysis is performed, the trees distribute themselves over the 2D space. Moreover, in the case of the direct spectral analysis the different shapes are interspersed along the trajectory. It is hence not possible to allocate the shapes to reliably assign shapes to classes on the basis of their position in the plots. The possible exception is that the horses and leafs are separated at the bottom right hand corner of the plot. In the case of the heat kernel analysis, the trees could be better separated. Although there is considerable overlap near the center of the plot, it appears that there is scope for separating the screwdrivers, pliers and leafs.

In our second experiment, we have repeated the procedure above for a smaller database which contains only three representative shapes. The three shapes used for test are the hands, the leafs and the men. For each shape there are 15 different views corresponding to different viewing directions. The left-hand panel of Figure 3 shows the results of direct spectral analysis, while the right-hand panel shows the result of heat kernel analysis. In the case of the direct spectral analysis, the shapes are poorly separated. In the case of the heat kernel analysis, there is good separation.

To further investigate this three-class data, in Figure 4 the two panels show the distances $d_T(k_1, k_2) = (\boldsymbol{B}_{k_1} - \boldsymbol{B}_{k_2})^T(\boldsymbol{B}_{k_1} - \boldsymbol{B}_{k_2})$ between the vectors of eigenvalues for the trees indexed $k_1$and k_2. The left-hand panel is for the direct spectral analysis of the trees and the right-hand panel is for the spectral vectors extracted by performing the heat-kernel embedding. Here the classes emerge as clear blocks in the distance matrix for the heat-kernel embedding, while for the direct spectral analysis the block structure is more confused.

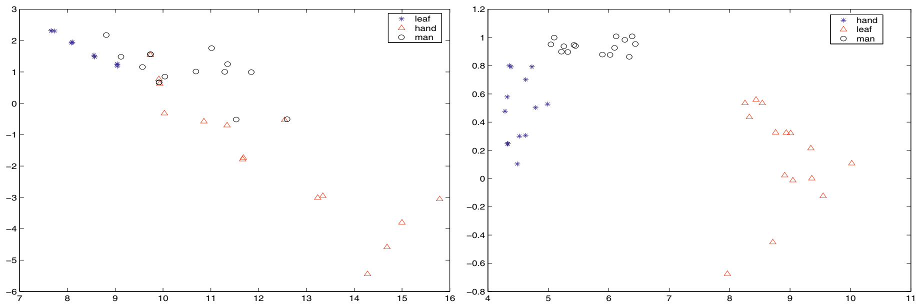


Fig. 3. Direct spectral analysis (left) and heat-kernel analysis (right)

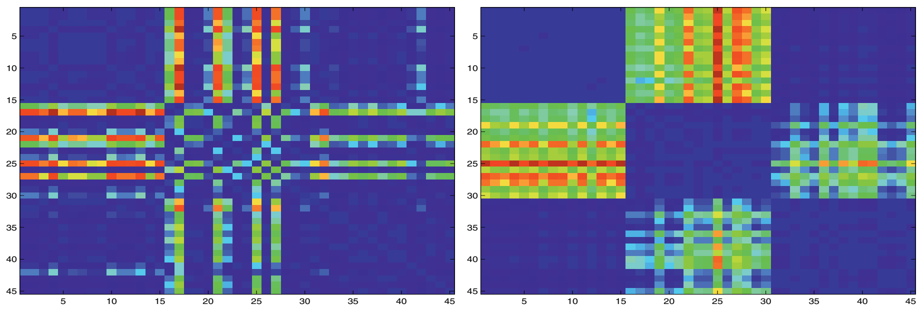


Fig. 4. Distance matrices for direct spectral analysis (left) and heat kernel analysis (right)

6 Conclusion and Future Work

In this paper we have explored how the use of heat kernels can lead to a measure of geodesic distance that can be used for the purposes of embedding graphs in low dimensional Euclidean spaces. The distance measure is found by equating the spectral and Gaussian forms of the heat kernel. We show how MDS can be used to embed the the distances, and how a spectral characterisation of the embedded graphs can be used for graph-clustering. We experiment with the method on sets of shock trees. Here the characterisation which results from the geodesic analysis is better than that obtained from the raw spectral features of the graphs. There are clearly a numbert of ways in which the work reported in this paper can be extended. For instance, it would be interesting to study the controlled effects of varying the time parameter, and to see if this leads to a natural definition of "scale-space" for the graphs.

References

1. A.D. Alexandrov and V.A. Zalgaller. Intrinsic geometry of surfaces. *Transl. Math. Monographs*, 15, 1967.
2. A. Ranicki. Algebraic l-theory and topological manifolds. *Cambridge University Press*, 1992.

3. M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Neural Information Processing Systems*, 14:634–640, 2002.
4. G.R. Hjaltason and H. Samet. Properties of embedding methods for similarity searching in metric spaces. *PAMI*, 25:530–549, 2003.
5. A. Grigor'yan. Heat kernels on manifolds, graphs and fractals. *preprint*, 2003.
6. H. Busemann. The geometry of geodesics. *Academic Press*, 1955.
7. J.B. Tenenbaum, V.D. Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:586–591, 2000.
8. J. Lafferty and G. Lebanon. Diffusion kernels on statistical manifolds. *CMU preprint*, 2004.
9. T. Coulhon M. Barlow and A. Grigor'yan. Manifolds and graphs with slow heat kernel decay. *Imperial College preprint*, 2000.
10. N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some its algorithmic application. *Combinatorica*, 15:215–245, 1995.
11. A.J. Smola and R. Kondor. Kernels and regularisation of graphs. 2004.
12. S. Weinberger. Review of algebraic l-theory and topological manifolds by a.ranicki. *BAMS*, 33:93–99, 1996.

A Syntactic Pattern Recognition Approach to Computer Assisted Translation

Jorge Civera[1], Juan M. Vilar[2], Elsa Cubel[1], Antonio L. Lagarda[1], Sergio Barrachina[2], Francisco Casacuberta[1], Enrique Vidal[1], David Picó[1], and Jorge González[1]

[1] Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de València
Instituto Tecnológico de Informática, E-46071 València, Spain
tt2iti@iti.upv.es
http://www.iti.upv.es/~prhlt

[2] Departamento de Lenguajes y Sistemas Informáticos
Universitat Jaume I, E-12071 Castellón de la Plana, Spain

Abstract. It is a fact that current methodologies for automatic translation cannot be expected to produce high quality translations. An alternative approach is to use them as an aid to manual translation. We focus on a possible way to help human translators: to interactively provide completions for the parts of the sentences already translated. We explain how finite state transducers can be used for this task and show experiments in which the keystrokes needed to translate printer manuals were reduced to nearly 25% of the original.

1 Introduction

It is becoming increasingly clear that current automatic translation methodologies cannot be expected to produce high quality translation in the near future. An alternative way to take advantage of the technologies developed is to use them in order to help human translators. One such approach, proposed by [1], can be explained as follows: the translator begins to type the translation and the system guesses the best completion for the text typed so far. The user can then accept the suggestion of the computer or part of it. This should reduce the amount of work of the translator.

This approach has two important aspects: the models need to provide adequate completions and they have to do so efficiently. To fulfill these two requirements, we have decided to use Stochastic Finite State Transducers (SFST) since they have proved in the past to be able to provide adequate translations [2–4] and, as we show in this paper, efficient parsing algorithms can be easily adapted in order to provide completions.

The rest of the paper is structured as follows. The following section presents the general setting for machine translation and finite state models. In section 3, the search procedure for an interactive translation is presented. Experimental results are presented in section 4. Finally, some conclusions and future work are explained in section 5.

2 Machine Translation with Finite-State Transducers

Given a source sentence s, the goal of MT is to find a target sentence $\hat{\mathbf{t}}$ that maximizes:

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}} \Pr(\mathbf{t} \mid \mathbf{s}) = \underset{\mathbf{t}}{\operatorname{argmax}} \Pr(\mathbf{t}, \mathbf{s}) \approx \underset{\mathbf{t}}{\operatorname{argmax}} \Pr_{\mathcal{T}}(\mathbf{t}, \mathbf{s}) \tag{1}$$

The joint distribution $\Pr(\mathbf{t}, \mathbf{s})$ can be modeled by Stochastic Finite State Transducers (SFST) $\mathcal{T}$ [5]. They have been successfully applied into many translation tasks [2–4]. Furthermore, there exist efficient parsing algorithms like Viterbi[6] for the best parse and REA [7] for the n-best parses.

One possible way of inferring SFSTs from training data is the Grammatical Inference and Alignments for Transducer Inference[1] (GIATI) technique [8]. Given a finite sample of string pairs, it works in three steps:

1. Building training strings. Each training pair is transformed into a single string from an extended alphabet to obtain a new sample of strings. The "extended alphabet" contains words or substrings from source and target sentences coming from training pairs.
2. Inferring a (stochastic) regular grammar. Typically, a smoothed n-gram is inferred from the sample of strings obtained in the previous step.
3. Transforming the inferred regular grammar into a transducer. The symbols associated to the grammar rules are transformed into source/target symbols by applying an adequate transformation, thereby transforming the grammar inferred in the previous step into a transducer.

The transformation of a parallel corpus into a corpus of single sentences is performed with the help of statistical alignments: each word is joined with the word in the target sentence it is aligned to, creating an "extended word". This joining is done taking care not to invert the order of the output words. The third step is trivial with this arrangement. In our experiments, the alignments are obtained using the GIZA software [9, 10], which implements IBM statistical models [11, 12].

3 Interactive Search

In the previous section the training process undergone to generate a SFST $\mathcal{T}$ from a parallel corpus was described. The aim of interactive search is to find a suffix of target sentence $\hat{\mathbf{t}}_s$ that maximizes the *a posteriori* probability given a SFST $\mathcal{T}$, a source sentence $\mathbf{s}$ and a prefix of the target sentence $\mathbf{t}_p$ produced by a human translator:

$$\hat{\mathbf{t}}_s = \operatorname*{argmax}_{\mathbf{t}_s} \Pr(\mathbf{t}_s \mid \mathbf{s}, \mathbf{t}_p) \approx \operatorname*{argmax}_{\mathbf{t}_s} \Pr_{\mathcal{T}}(\mathbf{t}_p \mathbf{t}_s, \mathbf{s}) \tag{2}$$

This equation is similar to the one for general translation but in this case, the optimization is performed over the set of target suffixes rather than the set of complete target sentences.

The solution to this problem has been devised in two phases. The first phase copes with the extraction of a word graph $\mathcal{W}$ from a SFST $\mathcal{T}$ given a source sentence $\mathbf{s}$. In a second phase, the search of the best translation (or translations) is performed over the word graph $\mathcal{W}$.

[1] The previous name of this technique was MGTI - Morphic-Generator Transducer Inference.

3.1 Word Graph Derivation

A word graph is a compact representation of all the possible translations that a SFST $\mathcal{T}$ can produce from a given source sentence **s** together with the probabilities of those translations. In fact, the word graph could be seen as a kind of weighted finite state automaton in which the probabilities are not normalized.

The construction of the word graph is reminiscent of the intersection of two automata: a SFST and a linear DFA (deterministic finite state automaton) representing the input sentence. We will explain it through a simple example. Assume that we have to translate the sentence “haga clic en siguiente .” (click next) using the SFST of Figure 1. The first step is to build the DFA of Figure 2. It is easy to see that the DFA has as many states as words in the source sentence plus one and the ith word of the sentence connects states $i-1$ and i.

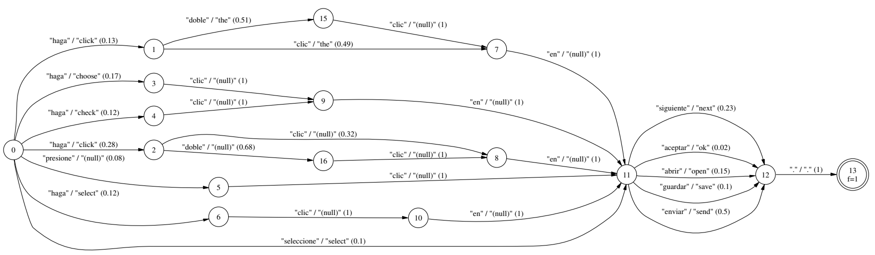


Fig. 1. A transducer inferred from a parallel corpus

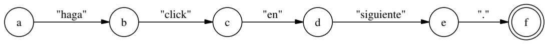


Fig. 2. A DFA representing the sentence: “haga clic en siguiente.”

From these two automata, we can easily build the word graph. For the moment, assume that the output of the arcs of the SFST have at most one word. The states of the word graph will be pairs composed of a state of the DFA and a state of the SFST. The initial state of the word graph will be the pair composed of the initial states of those automata. The probability of a pair being final will be the final probability of the corresponding state in the SFST. Now, assume that p and r are states of the DFA, p' and r' are states of the SFST and that there is an arc from p to r with input w in the DFA and another arc in the SFST from p' to r' with input w and output y. Then, the word graph will have an arc from $q \equiv (p, p')$ to $q' \equiv (r, r')$ with input y (remember that the word graph represents sentences of the output language, i.e. possible translations). This arc will have the same probability as the arc (p', w, y, r') in the SFST that we will denote as $P(q, y, q')$. The final-state probability of each state q will be denoted as $P_F(q)$. The result of this process in our example can be seen in Figure 3.

There are a couple of minor issues to deal with in this construction. On one hand, the output symbol for a given arc could be empty string (which are represented by “(null)” in the Figures) or could contain more than one word. Since the word graph generated

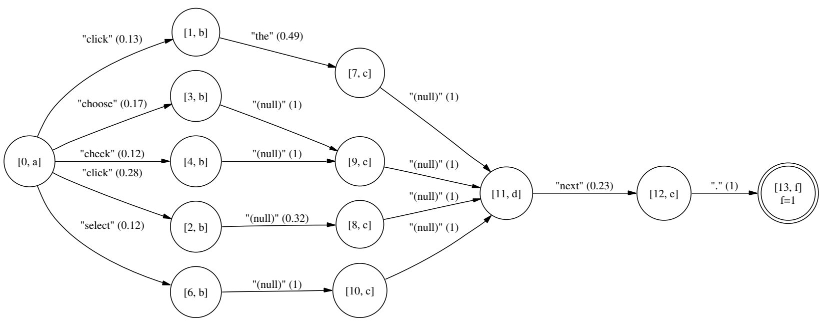


Fig. 3. Word graph resulting from the SFST in Figure 1 and the DFA in Figure 2. Isolated states are not shown

is not deterministic, the inclusion of empty outputs coming from the SFST is integrated easily. In the case of arcs with more than one word, auxiliary states were created in order to assign only one word for each arc. On the other hand, it is possible to have words in the input sentence that do not belong to the input vocabulary in the SFST. This problem is solved with the introduction of a special "unknown word" in the input vocabulary of the SFST.

3.2 Search of n-Best Translations Given a Prefix of the Target Sentence

Once the word graph is constructed, it can be used for finding the best completions for the part of the translation typed by the human translator. Not that the word graph depends only on the input sentence, so it is used repeatedly for finding the completions of all the different prefixes provided by the translator.

Ideally, the task would be to find the target suffix $\mathbf{t}_s$ that maximizes the probability *a posteriori* given a prefix $\mathbf{t}_p$ of the target sentence and the input sentence. In practice, however, it may happen that $\mathbf{t}_p$ is not present in the word graph $\mathcal{W}$. The solution is to use not $\mathbf{t}_p$ but a prefix $\mathbf{t}'_p$ that minimizes the edition distance with $\mathbf{t}_p$ and is compatible with $\mathcal{W}$. Therefore, the score of a target translation $\mathbf{t} \equiv \mathbf{t}_p \cdot \mathbf{t}_s$ is characterized by two functions, the edition cost between the target prefix $\mathbf{t}_p$ and the optimal prefix $\mathbf{t}'_p$ found in the word graph $\mathcal{W}$ and the *a posteriori* probability of $\mathbf{t}_s$ ($Pr(\mathbf{t}_s \mid \mathbf{t}'_p)$). However, the list of n-best translations has been prioritized first by minimum edition cost and then by *a posteriori* probability to value more significantly those translations that were closer to the user preferences.

Let q_p be the state(s) in $\mathcal{W}$ that is (are) reached from the initial state using $\mathbf{t}'_p$ and let $\mathcal{P}(\mathcal{W}, \mathbf{t}_p, q_p)$ be the set of possible paths $(q_0, t_1, q_1), \ldots, (q_{m-1}, t_m, q_m)$ in $\mathcal{W}$ from $q_0 = q_p$ that produce the translation suffix $\mathbf{t}_s = t_1, \ldots, t_m$ of length m . $Pr(\mathbf{t}_s \mid \mathbf{t}'_p)$ is calculated as:

$$Pr(\mathbf{t}_s \mid \mathbf{t}'_p) = \sum_{\mathcal{P}_m \in \mathcal{P}(\mathcal{W}, \mathbf{t}_s, q_p)} \prod_{1 \le i \le m} P(q_{i-1}, t_i, q_i) P_F(q_m) \tag{3}$$

The search for the $\mathbf{t}_s$ that maximize $Pr(\mathbf{t}_s \mid \mathbf{t}'_p)$ has been demonstrated to be an NP-hard problem, so the Viterbi approach [6] will be adopted to make feasible the calculation of $Pr(\mathbf{t}_s \mid \mathbf{t}'_p)$, that is:

$$\widehat{Pr}(\mathbf{t}_s \mid \mathbf{t}'_p) = \max_{\mathcal{P}_m \in \mathcal{P}(\mathcal{W}, \mathbf{t}_s, q_p)} \prod_{1 \le i \le m} P(q_{i-1}, t_i, q_i) P_F(q_m) \tag{4}$$

This simplification is imperative because of the real-time constraints under which our prototype is required to run. However, a better approximation to $Pr(\mathbf{t}_s \mid \mathbf{t}'_p)$ will be presented in next section.

The algorithm proposed to solve this search problem is an adapted version of the Recursive Enumeration Algorithm (REA) described in [7] that integrates the minimum edition cost algorithm in the search procedure. This algorithm consist on two parts:

– Forward search that calculates the 1-best path from the initial state q_0 to every state in the word graph $\mathcal{W}$. Paths in the word graph are weighted not only based on their *a posteriori* probability, but also on their edition cost respect to the target sentence prefix.

  To this purpose, ficticious edges have been inserted into the word graph to represent edition operations like insertion, substitution and deletion. These edition operations have been included in the word graph in the following way:

  - **Insertion:** An insertion edge has been inserted as a loop for each state in the word graph.
  - **Deletion:** A deletion edge is added for each arc in the word graph having the same source and target state than its sibling arc.
  - **Substitution:** Each arc in the word graph is treated as a substitution edge whose edition cost is proportional to the levenshtein distance between the symbol associated with this arc and the word prefix employed to traverse this arc during the search.

  For example, in Figure 3 if the user would type “press” as an initial prefix, we would have two different classes of translation sentences. Those ones that applying an insertion operation would start from the initial stage at state $[0, a]$, and those ones that applying a deletion or substitution operation would depart from states in the second stage.

– Backward search that enumerates candidates for the k-best path along the $(k-1)$-best path. This recursive algorithm defines the next best path that arrives at a given state q as the next best path that reaches q' plus the arc leaving from q' to q, being $(q', b, q) \in E$. If this next best path arriving at state q' has not been calculated yet, then the next best path procedure is called recursively until a 1-best path is found or no best paths are found.

To reduce the computational cost of the search, the beam-search technique has been implemented. During the word graph construction, two beam coefficients were employed to penalize those edges leading to backoff states over those ones arriving at normal states. Finally, a third beam coefficient controls how far in terms of number of edition operations a hypothesis could be from the best hypothesis in a given stage during the parsing procedure.

3.3 An Improved Approximation to the Translation Probability

A better approximation to the true translation probability of equation (3) can be obtained on the base of the Viterbi n-best path approach. This approximation is achieved by summing up the probability of those paths with the same translation $\mathbf{t}_s$ in the set of n-best paths $\mathcal{P}_N(\mathcal{W}, \mathbf{t}_s, q_p)$:

$$Pr_N(\mathbf{t}_s \mid \mathbf{t}'_p) = \sum_{\mathcal{P}_m \in \mathcal{P}_N(\mathcal{W}, \mathbf{t}_s, q_p)} \prod_{1 \leq i \leq m} P(q_{i-1}, t_i, q_i) P_F(q_m) \qquad (5)$$

Indeed, the Viterbi approximation could be considered to be a particular case of the n-best approximation, that is, when the number of paths is 1. As the reader may expect, this approximation improves as the size of $\mathcal{P}_N(\mathcal{W}, \mathbf{t}_s, q_p)$ increases. However it should be noted that $Pr_N(\mathbf{t}_s \mid \mathbf{t}'_p)$ follows a log-wise growth, since most of the probability associated with the translation $\mathbf{t}_s$ is accumulated in the first n-best paths.

4 Experimental Results

4.1 Corpus Features

We performed experiments using the so-called Xerox corpus [13]. This corpus consists in a collection of technical Xerox manuals written in English, Spanish, French and German. The English versions are the original and the rest are translation of them. The sizes (in thousands of words) of the subsets used can be seen in Table 1.

Table 1. Features of the Xerox Corpus: training, vocabulary and test sizes are measured in thousands of words

	EN / ES	EN / DE	EN / FR
TRAINING	600/700	600/500	600/700
VOCABULARY	26 / 30	25 / 27	25 / 37
TEST	8 / 9	9 / 10	11 / 10
PERPLEXITY (3-gram)	107/60	93/169	193/135

4.2 Translation Quality Evaluation

We have used three different measures in order to assess the techniques presented:

1. *Translation Word Error Rate* (TWER). It is defined as the minimum number of word substitution, deletion and insertion operations to convert the target sentence provided by the transducer into the reference translation. Also known as edit distance.

2. *Character Error Rate* (CER). Edit distance in terms of characters between the target sentence provided by the transducer and the reference translation.
3. *Key-Stroke Ratio* (KSR). Number of key-strokes that are necessary to achieve the reference translation plus the acceptance key-stroke divided by the number of running characters.

These experiments were performed with GIATI transducers based on trigrams. The results are shown in Table 2. On the leftmost column appears the language pair employed for each experiment, English (En), Spanish (Es), French (Fr) and German (De). The main two central columns compare the results obtained with 1-best translation to 5-best translations. In the latter case, the target sentence out of the five suggested translations that minimizes most the correspondent error measure was selected.

Table 2. Results for the Xerox Corpus comparing 1-best to 5-best translations

	GIATI 3-gram (1-best)			GIATI 3-gram (5-best)		
XRCE 2	KSR	CER	TWER	KSR	CER	TWER
En-Es	29.1	30.3	43.1	26.2	25.0	37.8
Es-En	33.5	35.5	51.4	29.7	28.1	45.2
En-Fr	58.5	54.3	73.8	53.7	48.5	69.6
Fr-En	58.4	55.3	71.9	54.0	49.5	67.7
En-De	66.2	62.8	81.3	60.1	56.7	77.2
De-En	59.0	61.5	78.5	53.9	55.1	73.3

The best results were obtained between English and Spanish language pairs, in which the human translator would only need to type 25% of the total reference sentences. In theory, this could result in a factor of 4 increase in the productivity of human translators.

Furthermore, in all cases there is a clear and significant improvement in error measures when we move from 1 to 5-best translations. This gain in translation quality diminishes in a log-wise fashion as we increase the number of best translations. Pair of languages as English and French present somewhat higher error rates, as is also the case between English and German, reflecting the complexity of the task faced in these experiments.

4.3 A Comparative Evaluation: Viterbi vs. n-Best Approximation

An approximation to the true translation probability based on the n-best path was introduced in section 3.3. Some experiments were performed to assess the evolution of the translation quality as the calculation of the translation *a posteriori* probability improves. To this purpose a simplified version of the Xerox corpus was employed to reduce the impact of noise due to preprocess and postprocess phases.

The most important conclusion that could be extracted from these results is the adequacy of the simpler and direct Viterbi approach as an approximation to the actual *a posteriori* probability of a target sentence. As it can be observed from Table 3, the evolution of TWER rates across an increasing number of n-best translations does not

Table 3. TWER comparative table across different number of n-best paths based on a simplified version of XRCE2

TWER	Viterbi	5-best	10-best	20-best	50-best	100-best	200-best	500-best	1000-best
En-Es	31.7	31.9	32.1	32.0	32.2	32.3	32.3	32.4	32.4
Es-En	35.9	35.3	35.4	35.5	35.6	35.7	35.7	35.7	35.7
En-Fr	60.7	60.7	61.0	61.0	60.8	60.7	60.7	60.8	60.8
Fr-En	56.1	57.0	57.0	57.1	57.0	57.2	57.1	57.2	57.3
En-De	69.7	69.8	69.7	69.8	69.8	69.7	69.7	69.8	69.9
De-En	63.1	63.2	63.3	63.5	63.5	63.4	63.6	63.6	63.6

show a consistent positive growth of the translation quality. It is even negative for large n in most cases. A possible reason for these results is that for large n translations with lower quality are more frequent among the set of n-best translations, so summing up the probability of equal translations favors those ones that even being less probable have more repetitions.

5 Conclusions and Future Work

Finite-state transducers can be used for computer assisted translation. These models can be learned from parallel corpus, but the number of states/transitions can be too high. The concept of interactive search has been introduced in this paper along with some efficient techniques (word graph derivation and n-best) that solve the parsing problem given a prefix of the target sentence undeolve the parsing problem given a prefix of the target sentence under real-time constraints.olve the parsing problem given a prefix of the target sentence under real-time constraints.

The promising results achieved in the first experiments provide a new field in machine translation still to be explored, in which the human expertise is combined with automatic translation techniques to increase productivity without sacrificing high-quality translation.

Moreover, an alternative approach to Viterbi approximation based on the n-best idea was explained and the results obtained with it confirm the appropriateness of the Viterbi approach in real applications.

Finally, the introduction of morpho-syntactic information and/or bilingual categories in finite-state transducers are topics that leave an open door to future research.

Acknowledgements

This work has been supported by the European Union under the IST Programme (IST-2001-32091) and the Spanish project TIC2003-08681-C02-02.

References

1. Langlais, P., Foster, G., Lapalme, G.: Unit completion for a computer-aided translation typing system. Machine Translation **15** (2000) 267–294
2. Amengual, J.C., Benedí, J.M., Castano, A., Castellanos, A., Jiménez, V.M., Llorens, D., Marzal, A., Pastor, M., Prat, F., Vidal, E., Vilar, J.M.: The EuTrans-I speech translation system. Machine Translation **15** (2000) 75–103

3. Casacuberta, F., Llorens, D., Martínez, C., Molau, S., Nevado, F., Ney, H., Pastor, M., Picó, D., Sanchis, A., Vidal, E., Vilar, J.M.: Speech-to-speech translation based on finite-state transducers. In: International Conference on Acoustic, Speech and Signal Processing. Volume 1., IEEE Press (2001)
4. Vidal, E.: Finite-state speech-to-speech translation. In: Int. Conf. on Acoustics Speech and Signal Processing (ICASSP-97), proc., Vol.1, Munich (1997) 111–114
5. Picó, D., Casacuberta, F.: Some statistical-estimation methods for stochastic finite-state transducers. Machine Learning **44** (2001) 121–142
6. Viterbi, A.: Error bounds for convolutional codes and a asymtotically optimal decoding algorithm. IEEE Transactions on Information Theory **13** (1967) 260–269
7. Jiménez, V.M., Marzal, A.: Computing the k shortest paths: a new algorithm and an experimental comparison. In Vitter, J.S., Zaroliagis, C.D., eds.: Algorithm Engineering. Volume 1668 of Lecture Notes in Computer Science., London, Springer-Verlag (1999) 15–29
8. Casacuberta, F., Ney, H., Och, F.J., E. Vidal, J.M.V., Barrachina, S., Garcia-Varea, I., D. Llorens, C.M., Molau, S., Nevado, F., Pastor, M., Pico, D., Sanchis., A.: Some approaches to statistical and finite-state speech-to-speech translation. Computer Speech and Language **18** (2004) 25–47
9. Al-Onaizan, Y., Curin, J., Jahr, M., Knight, K., Lafferty, J., Melamed, D., Och, F.J., Purdy, D., Smith, N., Yarowsky, D.: Statistical machine translation (1999)
10. Och, F.J., Ney, H.: Improved statistical alignment models. In: ACL00, Hongkong, China (2000) 440–447
11. Brown, P.F., Cocke, J., Pietra, S.D., Pietra, V.J.D., Jelinek, F., Lafferty, J.D., Mercer, R.L., Rossin, P.S.: A statistical approach to machine translation. Computational Linguistics **16** (1990) 79–85
12. Brown, P.F., Pietra, S.D., Pietra, V.J.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics **19** (1993) 263–312
13. SchlumbergerSema S.A., de Informática, I.T., für Informatik VI, R.W.T.H.A.L., en Linguistique Informatique Laboratory University of Montreal, R.A., Soluciones, C., Gamma, S., Europe, X.R.C.: TT2. TransType2 - computer assisted translation. Project technical annex. (2001)

GIATI: A General Methodology for Finite-State Translation Using Alignments

David Picó, Jesús Tomás, and Francisco Casacuberta

Departament de Sistemes Informàtics i Computació
Institut Tecnològic d'Informàtica
Universitat Politècnica de València
Valencia, Spain

Abstract. Statistical techniques for machine translation have experienced an increasing interest by the natural language research community in the last years. Both statistical language modeling and statistical machine translation are now well-established disciplines with solid basis and outstanding results. On the other hand, finite-state transducers have revealed as an efficient and flexible formalism for the representation of a wide range of the kind of information that arises in natural language processing.
This paper presents a powerful general framework for combining statistical techniques with grammatical inference and finite-state traducers. The GIATI methodology proposed here provides a schema for building inference algorithms that are able to generate finite-state transducers from parallel corpora of text making use of information supplied by robust statistical techniques such as n-grams and alignments. Here, the general method is presented together with two concrete inference algorithms and some experiments that show the validity of the GIATI framework for real-world translation tasks.

1 Introduction

As it is well known, the fields of statistical and syntactic pattern recognition have found one of their most outstanding applications in natural language processing. Language modeling, machine translation or document retrieval are some examples of interesting tasks that have been successfully approached with pattern recognition techniques and are the object of intensive research. In language modeling, for instance, the statistical technique of smoothed n-grams has become the most widely used solution for problems such as speech recognition, clearly beating other approaches coming from a knowledge-based framework.

Machine translation has been traditionally approached by knowledge-based methods. However, in the last years, encouraging results obtained using statistical methods have raised the claim and the subsequent discussion about the possibility of machine translation to be successfully performed by learning the models from examples, specially in restricted domains of language. Among the pioneer work in statistical machine translation, the techniques commonly known as the “IBM algorithms” [3] were the first to describe the concept of *statistical alignments*, which will center our attention in the following sections, and have produced a wide range of derived ideas that rely to some extent on that seminal work.

The high heterogeneity of the available techniques in these fields is one of the reasons for the increasing interest in formalisms that can establish a common framework for them. Finite-state automata (and the closely related finite-state transducers) are one of such formalisms. They are founded on solid mathematical basis (see, for example, [2]), have many developments in grammatical inference [9], and provide a flexible and efficient tool for representing different kinds of information generated in natural language processing. They have been used in speech recognition [1], morphology, phonotactics, dictionary compression [7], and machine translation [9], among others.

The present work presents a general methodology for representing information coming from different sources (in particular, from statistical alignments) using finite-state transducers, called GIATI (for *grammatical inference and alignments for transducer inference*). This general setting constitutes a sort of *template* that acts as a melting pot for the creation of algorithms that are able to generate machine translation models (finite-state transducers) starting from bilingual corpora of text. This methodology has been already presented in other forums [4] with an emphasis on the theoretical and algebraic aspects. This paper offers a more concise presentation of GIATI and it focuses on new practical algorithms and experimental results on a real machine translation task about instruction manuals of hardware equipment.

2 The GIATI Methodology

2.1 Preliminaries

A *weighted finite-state automata* (WFSA) is a tuple $A = (\Gamma, Q, i, f, P)$, where Γ is an alphabet of symbols, Q is a finite set of states, $i : Q \rightarrow \mathbb{R}$ and $f : Q \rightarrow \mathbb{R}$ give a weight to the possibility of each state to be an initial or final state, respectively, and $P : Q \times \{\Gamma \cup \lambda\} \times Q \rightarrow \mathbb{R}$ defines a set of transitions between pairs of states in such a way that each transition is assigned a weight and it is labeled with a symbol from Γ or with the empty string, λ. A semi-ring can be defined on the set of weights so that under some particular conditions (such as the weight values ranging from 0 to 1) the automaton defines a probability distribution over the free monoid $\Gamma^\star$ and is called a *stochastic* finite-state automata. This has been studied in detail in [7].

A *weighted finite-state transducer* (WFST) is defined similarly to weighted finite-state automata, with the difference that transitions between states are labeled with *pairs* of symbols that belong to a Cartesian product of two different (*input* and *output*) alphabets, $\Sigma \times \Delta$.

A WFSA is essentially a device that can assign a weight to all the strings of symbols that label a path that goes from some input state to some output state. The total weight is usually computed by an accumulated addition or multiplication of the corresponding weights of the transitions that are found in the path. Analogously, a WFST is able to assign weights to *pairs of strings* following a similar procedure. This opens the door for the possibility of using WFST for translation. If we have an input string and want to find a translation of it, we can perform a search for the set of string pairs within the transducer such that the input string coincides with the one we want. The standard procedure is to choose the pair with the highest (or lowest) weight and yield the output part as the translation result. This search is not obvious and has been studied in different works (see, for example, [5]).

When an automaton or a transducer are unweighted they behave as accepting machines that accept the strings or string pairs, respectively, that label a path going from some input state to some output state. The set of strings accepted by an automaton $\mathcal{A}$ is called the *language* accepted by that automaton, $L(\mathcal{A})$. The set of string pairs accepted by a transducer $\mathcal{T}$ is called the *translation* accepted by that transducer, $T(\mathcal{A})$.

Given two finite alphabets, Γ and Γ', a *morphism* $h : \Gamma^\star \to \Gamma'^\star$ is a function that satisfies the following conditions: a) $h(\bar{x}, \bar{x}') = h(\bar{x}) \cdot h(\bar{x}')\ \forall \bar{x}, \bar{x}' \in \Gamma^\star$, and b) $h(\lambda) = \lambda$, where λ is the empty string. An *alphabetic morphism* h is a morphism that verifies: $h(a) \in \Gamma'$, $\forall a \in \Gamma$. The definition of GIATI in the following subsection makes use of morphisms as a way to denote rewriting transformations between the different set of symbols involved.

2.2 GIATI

The goal of GIATI is to define a general inference method for obtaining a finite-state transducer from a corpus of parallel text[1]. The aim is to produce a transducer that is able to generalize the training data and can find the correct translation of new input sentences that have not been seen during the training process.

The process defined by GIATI is illustrated in Figure 1. Given a parallel corpus consisting in a finite sample A of string pairs $(\bar{s}, \bar{t}) \in \Sigma^\star \times \Delta^\star$:

1. Each training pair $(\bar{s}, \bar{t})$ from A is transformed into a string z from an *extended alphabet* Γ yielding a sample S of strings, $S \subset \Gamma^\star$.
2. A (stochastic) finite-state transducer $\mathcal{A}$ is inferred from S.
3. The symbols (from Γ) of edges in $\mathcal{A}$ are transformed back into pairs of strings of source/target symbols (from $\Sigma^\star \times \Delta^\star$).

The first transformation is modeled by some labeling function $\mathcal{L} : \Sigma^\star \times \Delta^\star \to \Gamma^\star$, while the last transformation is defined by an "inverse labeling function" $\Lambda(\cdot)$, such that $\Lambda(\mathcal{L}(A)) = A$. Typically, $\Lambda(\cdot)$ consists of a couple of morphisms, h_Σ, h_Δ, such that for any string $z \in \Gamma^\star$, $\Lambda(z) = (h_\Sigma(z), h_\Delta(z))$. This guarantees some conditions on the transformations which are helpful to demonstrate some interesting algebraic properties (see [4]).

[1] A corpus of parallel text, also called a parallel corpus or a *bicorpus*, is a collection of pairs of word strings, usually sentences in the linguistic sense, that belong to two different languages and are the translation from one another.

$$\begin{tikzcd}
\begin{array}{c} A \subset \Sigma^\star \times \Delta^\star \\ \text{Sample of training pairs} \end{array} \arrow[r, "\mathcal{L}"] & \begin{array}{c} S \subset \Gamma^\star \\ \text{Sample of training strings} \end{array} \arrow[d, "\text{GI}"', "\text{algorithm}"] \\
\begin{array}{c} \mathcal{T}\colon A \subset T(\mathcal{T}) \\ \text{A finite-state transducer} \end{array} & \begin{array}{c} \mathcal{A}\colon S \subset L(\mathcal{A}) \\ \text{A finite-state automaton} \end{array} \arrow[l, "\Lambda"']
\end{tikzcd}$$

Fig. 1. Basic scheme for the inference of finite-state transducers. A is a finite sample of training pairs. S is a finite sample of strings. $\mathcal{A}$ is an automaton inferred from S such that S is a subset of the language $L(G)$. $\mathcal{T}$ is a finite-state transducer whose translation $T(\mathcal{T})$ includes the training sample A.

The interest of GIATI comes from the fact that it establishes a general template the instances of which are different inference algorithms. So, for example, the first transformation, $\mathcal{L}$, can aim at grouping pairs of segments of words into bigger, meaningful units that can be considered as the new tokens for the sample of training strings S. In the practical applications that we are presenting in Section 4 for machine translation this is done in different manners but taking profit of the information given by statistical alignments of words (see the next section). The grammatical inference algorithm that is needed in the second step can be any algorithm able to infer a WFSA from a corpus of text. For example, different variants of smoothed n-grams can be used for this [6].

A Toy Example: Inferring a Canonical Transducer. In this example, the algorithm produced by GIATI infers an unweighted finite-state transducer that will only accept the source sentences of the training set and will produce the target sentences as the corresponding translation.

- Transformation of string pairs into strings: Given a training pair $(\bar{s}, \bar{t})$, each symbol of $\bar{s}$ is labeled as itself, except for the last one, x, which is labeled as $x_{\bar{s}}$, i.e., a new symbol composed by the symbol itself and the target sentence as subindex:
  $\Gamma = \{a, b, a\_\{00\}, a\_\{101\}, a\_\{011\}, a\_\{0\}\}$
  $S = \{(abba\_\{00\}), (aaabbaa\_\{101\}), (bbaaa\_\{011\}), (bba\_\{0\})\}$

- Inference of a finite-state automaton: We will use a prefix tree-acceptor. The automaton produced by this method is shown in Fig. 2.
- Inverse transformation:

	a	b	a_{00}	a_{101}	a_{011}	a_0
h_Σ	a	b	a	a	a	a
h_Δ	λ	λ	00	101	011	0

The resulting canonical finite-state transducer is shown in Fig. 3.

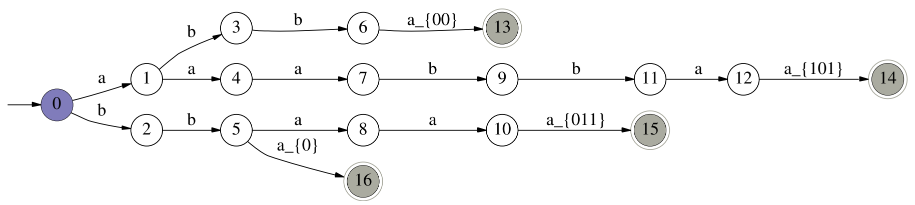


Fig. 2. A prefix–tree acceptor for the training sample of Example 1.

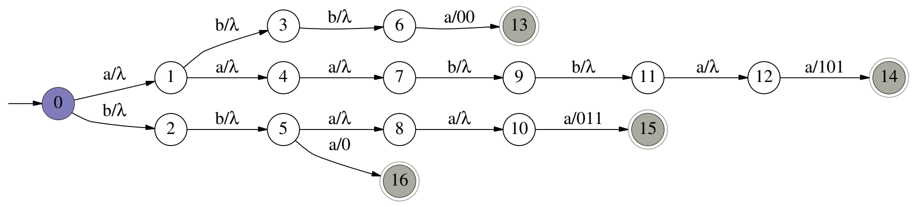


Fig. 3. The resulting finite-state transducer for Example 1.

3 Statistical Alignments

Our aim when building a combined corpus is to condense meaningful information about the relations that lay between the input and output words. This is a problem that has been thoroughly studied in statistical machine translation and has well-established techniques for dealing with it. The concept of *statistical alignment* [3] formalizes this problem. An alignment is a correspondence between words from an input text to words from an output text. Whether this is a one-to-one, a one-to-many or a many-to-many correspondence depends on the particular definition that we are using. The interesting thing is the availibily of algorithms for learning such correspondences from bilingual corpora. Constraining the definition of alignment simplifies the learning but subtracts expressive power to the model. The available algorithms try to find a compromise between complexity and expressiveness.

Formally, an alignment between a pair of sentences $(\bar{s}, \bar{t})$ is a mapping $i \rightarrow j = a_j$ that assigns a word s_j in position j to a word t_i in position $i = a_j$. Alignments are used as a hidden variable in statistical machine translation models such as IBM models [3] or hidden Markov models [10].

4 Two Translation Algorithms

In order to explain the following algorithms clearly we will denote a pair from the parallel corpus by $(\bar{s}, \bar{t})$ and we will be using a very small example of alignment taken from a real English-to-Spanish corpus: We will consider that the English phrase *the configuration program* is aligned with the Spanish phrase *el programa de configuración* with the alignment $\{1 \rightarrow 1, 2 \rightarrow 4, 3 \rightarrow 2\}$.

Algorithm #1: Using a Language of Segment Pairs.

- Transformation of string pairs into strings: the composed string is a sequence of $|\bar{s}|$ pairs, $(u_i, \bar{v}_i)$, where $u_i = s_i$ and $\bar{v}_1\bar{v}_2 \ldots \bar{v}_{|\bar{s}|} = \bar{t}$. Each of these pairs is considered to be *a single symbol*. We refer the reader to [4] for a complete description of this algorithm and other minor details. Applying this algorithm to the alignment of our example would produce the following corpus containing one string:

  S = {(the, el) (configuration, λ), (program, programa de configuración)}
- Inference of a finite-state automaton: a smoothed n-gram model can be inferred from the corpus of strings obtained in the previous step. Such a model can be expressed in terms of a WFSA [6].
- Inverse transformation: the transitions in the inferred automaton are labeled with compound symbols which are pairs of strings. A transducer can be obtained directly by considering these symbols as the pair of strings that label a transducer transition.

Algorithm #2: Using a Corpus of Bilingual Phrases.

- Transformation of string pairs into strings: this transformation obtains a set of bilingual phrases from each alignment, where many reasonable (and overlapping) possibilities are included. The compound corpus of strings only contains strings of length one and the symbols are pairs of strings as in the previous algorithm. Let us illustrate this with our small example. The alignment above will produce a corpus of phrases such as the following one, containing 7 strings of length 1:

  S = {(the, el), (configuration, configuración), (configuration, configuración de), (program, programa), (program, de programa), (configuration program, programa de configuración), (the configuration program, el programa de configuración)}

  This transformation function is inspired in recent work done in phrase-based statistical machine translation. We refer the reader to [10, 8] for details on different methods for extracting bilingual phrases from alignments.
- Inference of a finite-state automaton: we use a smoothed unigram on S with a normalization on the probability of appearance of the input part in each bilingual phrase in S.
- Inverse transformation: the same as in algorithm #1.

5 Experimental Results

We have performed some experiments in a real-word machine translation task so as to test the feasibility of the algorithms explained in the previous section. This corpus consists of a collection of technical manuals of hardware equipment by the Xerox company and have been processed by the team of the TransType2 project [11]. We are using the English-to-Spanish version of the corpus. The characteristics of this corpus are shown in Figure 4.

Training set	English	Spanish
Number of sentences	56,773	56,773
Running words	679,678	768,564
Size of the vocabulary	7,976	11,094
Test set	English	Spanish
Number of sentences	1,125	1,125
Running words	10,106	8,370
Size of the vocabulary	1,132	1,215

Fig. 4. Size values of the TT2 training and test corpora.

We have used the measure known as *word error rate* (WER), calculated as the percentage of insertions, deletions and substitutions of words that are necessary to obtain the reference output sentence from the translation calculated by the algorithm. Our results are a WER of 34.0% for the algorithm #1 using 4-grams smoothed by back-off, and 30.8% for the algorithm #2 limiting the length of phrases to 6 words. These experiments are still in a rudimentary stage but the results are not in very different range of error that those obtained by other more sofisticated methods. For example, phrase-based statistical translation with monotone search [8] obtained for this task a WER of 24.87% using much longer phrases and lexical weighting.

6 Conclusions and Further Work

The GIATI methodology for inferring finite-state transducers from parallel corpora has been presented here. GIATI is a general way of defining transducer inference algorithms making use of automata induction and other kinds of useful information such as statistical alignments. Two inference algorithms described within the GIATI framework have been presented here and some encouraging experimental results have been reported.

Further work on GIATI points towards searching other algorithms that make a clever use of statistical alignments in combination with n-grams or other finite-state automata inference methods. GIATI and finite-state models seem to be specially well suited for highly sequential translation tasks. Reordering words is usually a problem. We want to explore the possibilities for incorporating non-monotonous information (recursive alignments, statistical non-monotonous search) through some extension of GIATI.

Acknowledgements

The authors are grateful to the anonymous reviewers for their criticism and suggestions. This work has been partially supported by the Spanish project TIC2003-08681-C02-02 and the AVCiT project *Ayuda para grupos I+D+I* (GRUPOS03/031).

References

1. J. C. Amengual, J. M. Benedí, F. Casacuberta, A. Castaño, A. Castellanos, V. M. Jiménez, D. Llorens, A. Marzal, M. Pastor, F. Prat, E. Vidal and J. M. Vilar. *The EUTRANS-I Speech Translation System*, Machine Translation Journal, vol. 15. 2000
2. J. Berstel, *Transductions and context-free languages*, Teubner Stuttgart. 1979
3. P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer and P. S. Roosin. *A Statistical Approach to Sense Disambiguation in Machine Translation.* Computational Linguistics, 79–86. 1990
4. F. Casacuberta, E. Vidal and D. Picó. *Inference of finite-state transducers from regular languages.* Acepted for publications in Pattern Recognition.
5. F. Casacuberta and C. de la Higuera, *Linguistic Decoding is a Difficult Computational Problem*, Pattern Recognition Letters, vol. 20. 1999
6. D. Llorens, *Suavizado de autómatas y traductores finitos estocásticos*, Phd Thesis, Universitat Politècnica de València. 2000
7. M. Mohri, *Finite-State Transducers in Language and Speech Processing*, Computational Linguistics, vol. 23. 1997
8. J. Tomás, Casacuberta, F.: *Monotone Statistical Translation using Word Groups.* Proceedings of the Machine Translation Summit VIII, Santiago de Compostela, Spain (2001)
9. J. M. Vilar, *Improve the Learning of subsequential Transducers by Using Alignments and Dictionaries*, Grammatical Inference: Algorithms and Applications, Springer-Verlag, vol. 1891. 2000
10. R. Zens, F. J. Och and H. Ney. *Phrase-based statistical machine translation.* `http://citeseer.nj.nec.com/zens02phrasebased.html`
11. TransType 2 Project. `http://tt2.sema.es`

A Coupled Relaxation Method for Finding Perceptual Structures

Richard C. Wilson

Dept. of Computer Science
University of York, UK

Abstract. In this paper we describe a method for determining the existence and parameters of a geometric structure in an image by using a relational representation of its structure. The relational model is matched to image structure in order to find possible instances of the model in the image. Matches between the relational model and image primitives are then used to determine probability distributions for the parameters of a geometric transformation. This transform maps a geometric model of the structure onto an instance in the image. This distribution may then be used to infer a probability map for pixel-based information such as edge responses. When combined with the original edge responses, an enhanced image is produce with more salient edge structure. Iteration of the procedure results in a consistent set of models and edge structure. The method is demonstrated on rectangles undergoing affine transforms.

1 Introduction

The Generalized Hough Transform is a method of recovering the parameters of a model from information such as edge positions in the image array. Although it is very effective in recovering model parameters, it has a number of drawbacks. The computational cost and storage requirements of the method rise rapidly as the number of model parameters increases, making it unsuitable for complex models. Since it's inception[5], many variants have been suggested to alleviate these problems. The randomised Hough transform(RHT)[2, 3] uses pairs of points to compute single entries in the parameter map. The probabilistic Hough transform[4] makes use of the fact that not all edge pixels must be processed to obtain an accurate model. Edge points are therefore sampled from the image array at random. Kalviainen and Hirvonen[1] exploit the connective structure of line points to improve the speed and accuracy of the method.

These methods may be viewed as encompassing a spectrum between the use of raw edge points and segmental entities such as connected line points. Although the use of more salient entities such as line segments increases the efficiency of the method, it renders it vulnerable to errors in the segmentation phase. It is our observation that image segmentation can be improved by utilising global models of possible structure, and these are exactly the kind of models produced by the parametric methods described above. Grimson[6, 7] and Grimson and Lozano-Pérez[8, 9] have looked at the problem of interpreting segmental output, which may include noise and occulsion errors, by using global models of structure. They employ the interpretation tree[8] and a search algorithm to find matches between segmental structure and the model. This match is used to infer global tranformations of the 2D models.

Relational structure is a common and powerful way of representing the structure of objects in scenes. Methods of matching such structures to image segments have recently been developed which are tolerant to uncertainty and error[14, 10, 11, 15, 13, 12]. The key element of this approach is a relational graph which describes the structure of an object by the relationship between sub-parts. This graph represents a model of the object. The image may then be segmented and their inter-relationships found in order to form a image graph. The task of object location is then one of finding a sub-graph isomorphism between the model and image graphs. Despite these advances, the power of such representations is often limited by the quality of the segmental output. It is the aim of this paper to couple the processes of model location and segmentation, with the aim of finding reliable pairs of mutually consistent arrangments of segmental structure and model parameters. Initially, the method is similar to that of Grimson in that it interprets segmentation in terms of a global structural model, which in our case is a relational model based on perceptual groupings of segments. However, once a putative relational model has been found, the correspondences are used to find possible transforms which map a geometric model onto the image. In fact, the accuracy of the segmentation is used to determine a probability distribution for the tranformation. This in turn allows us to establish a probability map for the location of edge structure in the image which can be used to enhance the original edge responses. Iteration of the method leads to the enhancement of salient edge structure while the remainder is discarded.

2 Relational Model

We begin with a description of the image, consisting of a set of segmental entities $V = \{v_0, v_1, ..., v_n\}$ representing, for example segments such as straight lines. These segments may be obtained by an edge detector followed by a standard edge polygonisation method. Each segment has a measurement vector associated with it which is derived from its properties in the image; thus for each segment v_i there is a measurement vector $\boldsymbol{x}_i$. In the case of line segments this vector consists of the start and end points of the segment. Our task then is to determine any arrangements of these segments which are consistent with a relational model of the object of interest. A relational graph $G = \{V, E\}$ consists of a set of nodes $V = \{v_0, v_1, ..., v_n\}$, representing segments, and set of edges $E = \{e_0, e_1, \ldots\}$. Each edge $e_i = \{(v_a, v_b), \omega_i\}$ represents a perceptual relation between two segments, where ω_i is a label denoting the type of the relation. Such a graph can describe the mutual consistency constraints between segments in a compact way. These edges may either represent geometric relations such as Voronoi neighbours or perceptual structure such as colinearity or parallelism. Two such graphs G_0, G_1 are completely consistent when there exists a mapping between the node sets $\mathcal{M} : V_0 \rightarrow V_1$ which is one-to-one and is such that the mapped edges have the same labels. In a scene there are many objects and resulting segments. The task of locating structure consistent with a particular object model becomes one of mapping the relational graph for the object onto a sub-graph of the scene segments. Depending upon the complexity of the object model, missing and spurious segments may also become an issue, and inexact sub-graph mapping may be required. There are many algorithms available in the literature to achieve this end[14, 10, 11, 15, 13, 12].In this paper we have used the method described in Wilson and Hancock [15].

2.1 Affine Rectangle Model

In the experiments conducted in section 5, we use an affine rectangle model. The relational part of this model is shown in figure 1.

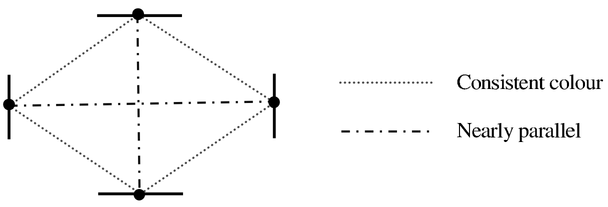


Fig. 1. Model of an affine rectangle

The nodes of this graph are straight line segments which represent the edges of the rectangle. We do not employ a fully perspective model of the rectangle because such a model is too unconstrained, since any four lines will make such a rectangle. Instead, we confine our attention to the set of rectangles whose projections are nearly affine, i.e. whose opposite sides are nearly parallel. Our first graph relation is therefore the 'parallel' relation between line segments. The other property of a consistent rectangle is that the interior of the rectangle is one colour. The second relation in the graph connects edges which have similar colours on the interal side of the edge they represent. In other words, they represent the constraint that the interior of the rectangle must be the same colour. We also exclude very narrow parallelograms with small corner angles.

3 Geometric Model

Once a mapping between an object model and a sub-graph in the scene has been located, the co-incident segments provide information about the geometry of the object. In particular, they may be used to infer a transform between the object model and it's realisation in the scene. In the case of our affine rectangle model, the affine mapping between the canonical model of a unit square and image is found by associating each one of the sides with a line segment identified by a match to the relational model. We can then use this transform to infer where the edges should be in the image, and enhance the edge responses at these locations.

However, there are some problems with using a precise geometric model for projection in this way. There are likely to be errors in the positions of the image segments and, due to discrete nature of the image, edge structure may not lie precisely on the line specified by the transformed geometric model. In order to successfully locate the edge structure, we need to take account of the uncertainty in the position of line segments and therefore in the affine mapping. The analysis of the appropriate probability distributions is very complex, even under simple models such as the affine transformation. Instead, we adopt an approach based on finding the first order variations of segment parameters and then adopt a normal distribution of transform parameters. We commence by looking at the errors in line segments extracted from the image.

3.1 Errors in Segments

For each segment in the image, we can calculate not only the segment measurement parameters, but also the parameter variances using a least-squares approach. For example in the case of a segment containing the points $\{(x_i, y_i), i = 0 \dots n\}$ and a straight line segment model, the variances are given by, for a mainly horizontal line,

$$\begin{aligned} var(m) &= \frac{\sum (y_i - mx_i - c)^2}{n \sum x_i^2 - (\sum x_i)^2 / n} \\ var(c) &= \frac{\sum (y_i - mx_i - c)^2}{n - (\sum x_i)^2 / \sum x_i^2} \\ cov(m, c) &= -\frac{\sum (y_i - mx_i - c)^2 \cdot \sum x_i}{n^2 \sum x_i^2 - (\sum x_i)^2} \end{aligned} \tag{1}$$

The variance in the centre-points of the segments is $var(y_c) = \sum (y_i - mx_i - c)^2 / n$. If the line is mainly vertical, the roles of the x and y points must be reversed.

3.2 Affine Rectangle Model

The geometric model of the rectangle consists of the four corner points and four lines joining the corner points (figure 2).

The segments are straight lines, and therefore the errors in the segment parameters are those given in equation 1. However, because of the affine transformation, we require the four corner points to define the transform. We are therefore interested in the errors in the crossing points of the line segments which make up the rectangle. These are given by equation 2 where corner n is formed by the intersection of lines a and b.

$$\begin{aligned} \mathrm{var}(x_n) = v_n &= x_n^2 \left[\frac{\mathrm{var}(c_a) + \mathrm{var}(c_b)}{(c_a - c_b)^2} + \frac{\mathrm{var}(m_a) + \mathrm{var}(m_b)}{(m_a - m_b)^2} \right] \\ \mathrm{var}(y_n) = w_n &= y_n^2 \Big[\frac{c_b^2 \mathrm{var}(m_a) + c_a^2 \mathrm{var}(m_b) + m_a^2 \mathrm{var}(c_b) + m_b^2 \mathrm{var}(c_a)}{(m_a c_b - c_a m_b)^2} \\ &\quad + \frac{\mathrm{var}(m_a) + \mathrm{var}(m_b)}{(m_a - m_b)^2} \Big] \end{aligned} \tag{2}$$

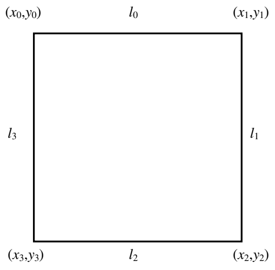


Fig. 2. The geometric model of the rectangle

By using a normal distribution and the variance provided by (2), the probability of a transform G which maps the square onto the endpoints $(x_0, y_0) \ldots (x_3, y_3)$ is given by

$$P(G|\mathcal{M}^*, \mathcal{L}) = \frac{1}{2\pi\sqrt{\prod_i v_i w_i}} \exp\big[-\sum_i (x_i - \overline{x}_i)^2/2v_i + (y_i - \overline{y}_i)^2/2w_i\big] \quad (3)$$

Here $\mathcal{M}^*$ represents the match found between image and relational model, and $\mathcal{L}$ represents the edge pixels involved in the lines which make up the model.

4 Enhancing Contours

Our final aim is to iteratively re-compute the edge probability maps, image segments and relational mappings to find a consistent description of the model and scene. To achieve this aim, we exploit a relaxation method[16] which makes successive approximations to the MAP estimate of the features and labels.

We begin by expanding over the set of possible transformations G and factorising the probability model:

$$P(F, \mathcal{M}^*, \mathcal{L}) = \sum_G P(F, \mathcal{M}^*, \mathcal{L}) = P(F) \sum_G P(G|\mathcal{M}^*, \mathcal{L}, F) P(\mathcal{M}^*, \mathcal{L}|F)$$

Here F is the field of edge responses. The tranform probability may be considered to be conditionally independent of the feature space given a particular match and set of line segments, i.e. $P(G|\mathcal{M}^*, \mathcal{L}, F) = P(G|\mathcal{M}^*, \mathcal{L})$. The quantity $P(\mathcal{M}^*, \mathcal{L}|F)$ is precisely that which we optimise to find the best match between relational model and image. Our iterative relaxation algorithm is then specified by

$$P^{(n+1)}(F) = P^{(n)}(F) P(\mathcal{M}^*, \mathcal{L}|F) \sum_G P(G|\mathcal{M}^*, \mathcal{L}) \quad (4)$$

5 Results

In the first set of experiments, we apply the technique to a synthetic image of a quadrilateral. The image is shown in figure 3. As it stands, this is a trivial

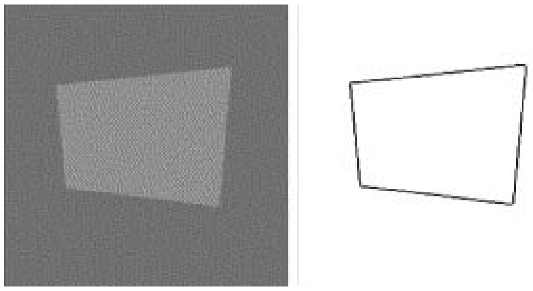


Fig. 3. Left: Quadrilateral image, Right: Result of edge detection

problem since the initial edge configuration is very good. The results of edge detection are also shown in figure 3. In order to provide a challenging task, we have added various levels of Gaussian noise to the image in figure 3. The new images have signal to noise ratios of 2, 1, 0.5 and 0.25 respectively. The final image therefore has noise with standard deviation four times the size of the edge step. Figure 4 shows the results of applying our method; the image in the left hand column is the original image; the central column is the edge detection result, and the image in the right hand column represents the final edge configuration after application of the algorithm. The method is able to accurately reconstruct the correct edge configuration even under very high levels of noise.

Figure 5 shows these results quantitatively. Here we have used a pixel based measure of accuracy based on average minimum distance. If $P = \{p_0, p_1, \ldots, p_m\}$ is the set of pixels in a perfect edge configuration, and $Q = \{q_0, q_1, \ldots, q_n\}$ is the set of pixels in the test image, the forward error e_f is given by $e_f = \frac{1}{m}\sum_{i=0}^{m} \min_j d(p_i, q_j)$, where $d(.)$ is the Euclidean distance. This error measures both misplaced and missing structure in the test image. The backwards error is given by $e_b = \frac{1}{n}\sum_{i=0}^{n} \min_j d(q_i, p_j)$ and measures misplaced and extra spurious structure. By combining these measures in $e = (e_f + e_b)/2$, we obtain a measure which reflects misplaced, spurious and missing edge structure. When $e = 0$, the configurations are identical. The signal level is 16. In this simple case, our model can accurately reconstruct the edge configuration even in the presence of extreme noise because of the global model of structure.

In the second set of experiments, we add a set of background non-quadrilateral distractors. These distractors both disrupt the initial edge configuration and test the algorithms ability to discard irrelevant structure. The initial image is shown in figure 6. Again we add Gaussian noise to the image and attempt to reconstruct the original rectangle. The results are shown in figure 6(right).In this case, although the original configurations are worse, the final result is superior because the background contrast is increased by the distractors.

In the third set of experiments, we apply the quadrilateral model to an scene from an office desk. There are a number of prominent quadrilaterals present in the image, and a considerable amount of clutter, including some non-rectangular objects and background texture. Figure 7 shows the original image and the resulting edge strengths and edge segmentation.

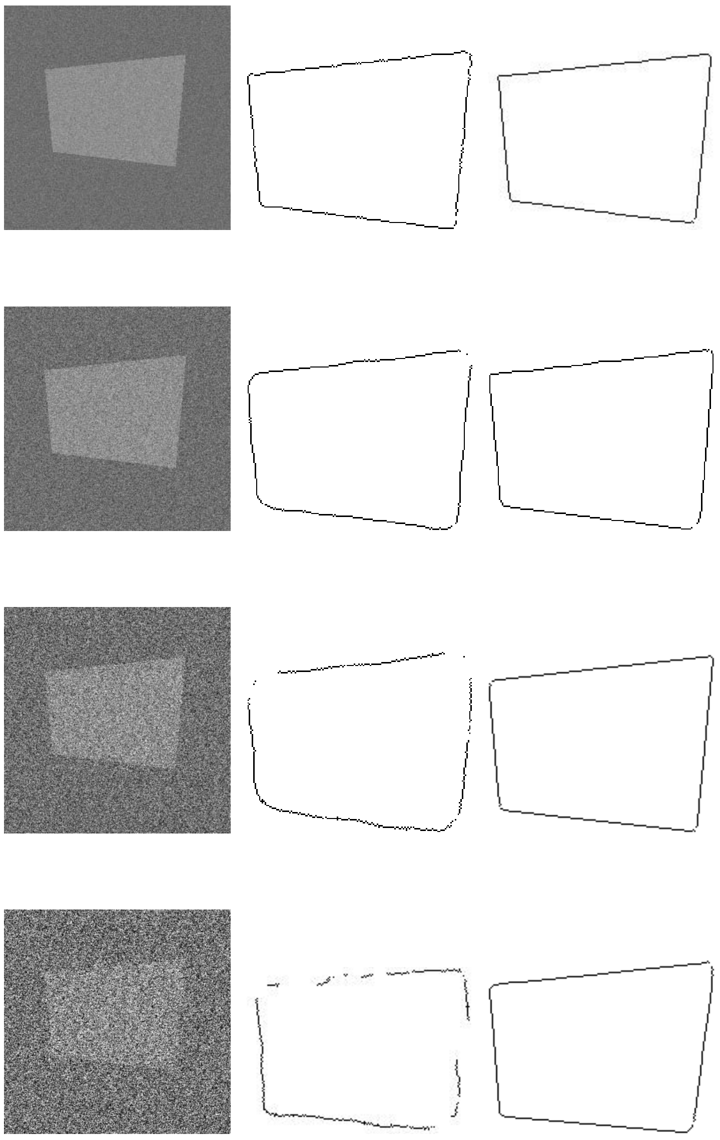


Fig. 4. Left column: original image; Central column: original edge detection results; Right column: final edge configuration

The final image is the result from the tenth and final iteration. Both the background and non-rectangular objects have been eliminated. However, some spurious linear structures remain, resulting from the participation of some edges in multiple quadrilaterals.

This experiment reveals some important points about the method and the quadrilateral model. Firstly, gaps in the edges can only be closed where some

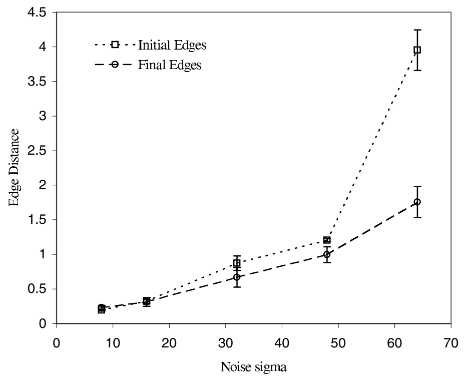


Fig. 5. Performance of method on simple rectangle image

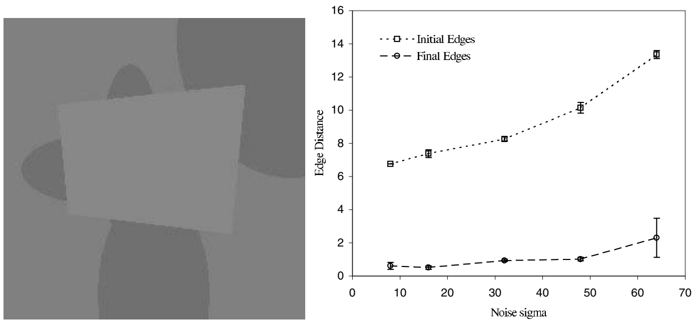


Fig. 6. Left: original distractors image; Right: Performance of method at different noise levels

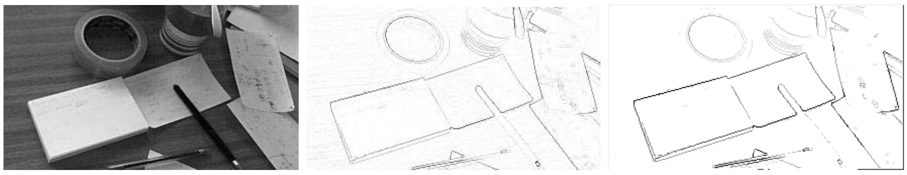


Fig. 7. The original image (left), the initial edge field (centre) and the final edge field (right)

evidence exists in the responses of the edge detector. For example, where the pen crosses the notepad, no evidence for the notepad edge exists. Secondly, the quadrilateral model is limited in the sense that the four edges may appear anywhere in the image, and so if one edge is missing, it becomes impossible to reconstruct the model. We intend to address these issues in future work by constructing a more sophisticated model of feature probability and employing more sophisticated perceptual groups.

References

1. H. Kalviainen and P. Hirvonen,"An Extention to the Randomized Hough Transform", *Pattern Recognition Letters*,18:1 pp 77–85 1997.
2. L. Xu and E. Oja and P. Kultanen,"A new curve detection method - Randomized Hough Transform(RHT)", *Pattern Recognition Letters*,11:5 pp 331–338 1990.
3. L. Xu and E. Oja,"Randomized Hough Transform(RHT) - Basic mechanisms, algorithms and computational complexities", *CVGIP: Image understanding*, 57:2 pp 131–154 1993.
4. J. R. Bergen and H. Shvaytser,"A probabilistic algorithm for computing Hough Transforms", *Journal of Algorithms*, 12:4 pp 639–656 1991.
5. R. O. Duda and P. E. Hart, "Pattern Recognition and Scene Analysis", Wiley, New York 1973
6. W. E. L. Grimson, "On the Recognition of Curved Objects", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:6 pp632–643 1989
7. W. E. L. Grimson, "On the Recognition of Parameterized 2D Objects", *International Journal of Computer Vision*, 2:4 pp353–372 1989
8. W. E. L. Grimson and T. Lozano–Pérez, "Localizing overlapping parts by searching the interpretation tree", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:4 pp469–482 1987
9. W. E. L. Grimson and T. Lozano–Pérez, "Model–Based recognition and localization from sparse range or tactile data", *International Journal of Robotics Research*, 3:3 pp3–35 1984
10. K. Boyer and A. Kak, "Structural Stereopsis for 3D vision", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp 144-166, 1998
11. W. Christmas and J. Kittler and M. Petrou, "Structural Matching in Computer Vision using Probabilistic Relaxation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp 749-764, 1995
12. B. T. Messmer and H. Bunke, "Efficient Error-Tolerant Subgraph Isomorphism Detection", *Shape, Structure and Pattern Recognition*, pp 231–240, 1994
13. A. Sanfeliu and K. S. Fu, "A Distance Measure Between Attributed Relational Graphs", *IEEE Systems, Man and Cybernetics*, vol. 13, pp 353–362, 1983
14. L. Shapiro and R. M. Haralick, "Structural descriptions and inexact matching", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp 504–519, 1981
15. R. C. Wilson and E. R. Hancock, "Structural Matching by Discrete Relaxation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:6, pp 634–648, 1997
16. J. Kittler and E. R. Hancock, "Combining evidence in Probabilistic Relaxation", *International Journal of Pattern Recognition and Artificial Intelligence* 3, pp29–52, 1989

A Comparison of Unsupervised Shot Classification Algorithms for News Video Segmentation

Massimo De Santo[1], Gennaro Percannella[1], Carlo Sansone[2], and Mario Vento[1]

[1] Dipartimento di Ingegneria dell'Informazione e di Ingegneria Elettrica
Università di Salerno - Via P.te Don Melillo, 1 I-84084, Fisciano (SA), Italy
{desanto,pergen,mvento}@unisa.it
[2] Dipartimento di Informatica e Sistemistica, Università di Napoli "Federico II"
Via Claudio, 21 I-80125 Napoli (Italy)
carlosan@unina.it

Abstract. Automatic classification of shots extracted by news video plays an important role in the context of news video segmentation. In spite of the efforts of the researchers involved in this field, a definite solution for the shot classification problem does not yet exist. Moreover, the authors of each novel algorithm usually provide results supporting the claim that their method performs well on a set of news videos, without facing the problem of making a wide comparison with other algorithms in terms of key performance indexes.
In this paper, we present an experimental comparison of three shot classification algorithms. We considered only techniques that do not require the explicit definition of a model of the specific news video. In such a way the obtained performance should be quite independent of the news program's style. For testing the selected algorithms, we built up a database significantly wider than those typically used in the field.

1 Introduction

In order to allow a faster and more appealing use of news video databases, indexing and retrieval are essential issues to be addressed. A first step towards an effective indexing is the segmentation of a news video. It implies, at a first stage, the partition of the video into sequences of frames, called *shots*, obtained by detecting transitions that are typically associated to camera changes. Once the shots have been individuated, they can be classified on the basis of their content. Two different classes are typically considered, *anchor shot* and *news report shot* classes. Then, a news video can be segmented into *stories*; each story is obtained by linking a given anchor shot together with all successive news report shots, until another anchor shot occurs.

In this paper, we address the shot classification problem. In the literature, most of the approaches that exploit the video source information use a model matching strategy [1]. For each shot, a distinctive frame, called *key-frame*, is extracted. Then, it is matched against a set of predefined models of an anchor shot frame in order to classify it. This approach is strongly dependent on the model of the specific video program. This is a severe limitation, since it is difficult to construct all the possible models for the different news videos and the style of a particular news program can change over the time.

Other authors use a face detection approach to identify anchor shots [2]. However, face detection in video is generally too time-consuming for practical application. A different approach based on the frame statistics is presented in [3], where the authors use a Hidden Markov model (HMM) to classify frames. The features used are the difference image between frames, the average frame color and also the audio signal. In this case the HMM parameters are evaluated during a training phase.

Finally, some authors [4,5,6] propose methods that are substantially unsupervised and do not require the explicit definition of an anchor shot model. In particular, in [4] a graph-theoretical cluster analysis method is employed. As pointed out by the authors, this approach fails when identical or very similar news-report shots appear in different stories of the same news program. In [5] shot classification is firstly performed on the basis of a statistical approach and then refined by considering motion features. Here the authors assume that in an anchor shot both the camera and the anchor are almost motionless. In our opinion, however, this hypothesis is not completely acceptable. In [6] a template-based method is proposed. The template is found in an unsupervised way and it does not depend on a particular threshold. However, the authors assume that different anchor shot models share the same background. This is not true for most news stations: because of different camera angles, different models can have different backgrounds.

From the previous analysis it appears evident that a definite solution for the shot classification problem does not yet exist, since it is always possible to find a case in which the assumptions of a given technique fail. Moreover, the authors of each algorithm usually provide results supporting the claim that their method performs well on a set of news videos, without facing the problem of making an extensive comparison with other algorithms in terms of key performance indices.

Starting from these considerations, in this paper we present an experimental comparison of three shot classification algorithms among those presented so far in the literature. We have chosen to consider only techniques that do not require the explicit definition of a specific model of the anchorperson shot. In such a way the obtained performance should be quite independent of the specific style of the news program.

In order to test the selected algorithms in a significant way, we built-up a database that is twice the biggest database reported until now in the scientific literature [4]. Namely, we used a news video database consisting of about 10 hours with 464 anchor shots and 5705 news report shots.

The organization of the paper is as follows: in section 2 the three selected shot classification algorithms are presented. In section 3 the database used is reported together with the tests carried out in order to assess the performance of the selected algorithms. Finally, in section 4, some conclusions are drawn.

2 The Selected Algorithms

Among all the techniques described in the introduction, we choose to consider for comparison only those that neither use a model-based approach nor require a specific training phase. This choice is justified by the consideration that the variety of existing news programs in the world is so high to make it practically infeasible building a good general model. Even when limiting the attention to a single broadcaster, as it could be reasonable in a real application, the news model can repeatedly change over the time. In this case, a model-based approach could provide good results only for a small period of time, while requiring a continuous re-modeling to guarantee acceptable performance for a longer period. For the same reason, methods requiring a specific training phase are not very suitable for the application at hand, given that it would be necessary re-training the system for each change in the news style.

In particular, on the basis of the best results available in the literature, we considered the shot classification algorithms proposed by Bertini *et al.* in [5], Gao and Tang in [4] and Hanjalic *et al.* in [6]. Hereinafter, for the sake of simplicity, we will refer to these algorithms with the terms BER, GAO and HAN, according to the first three letters of the first author.

In the following we will briefly recall the rationale inspiring these algorithms.

2.1 BER Algorithm

The shot classification is here performed on the basis of a statistical approach and of the motion features of the anchor shots, without requiring any model [5]. The statistical approach is based on the consideration that anchor shots are repeated at variable length throughout the video. The obtained classification is successively refined by considering also motion features: according to the authors, it is reasonable to assume that in an anchor shot both the camera and the anchorperson are almost motionless. More in details, the first step of the process is based on the computation, for each video shot S_k, of the so-called *shot lifetime* $L(S_k)$. It measures the shortest temporal interval that includes all the occurrences of shots with similar visual content within the video and is used to perform a first shot classification. In this case, for each video shot the first frame is chosen as key-frame. Having defined a suitable similarity measure between two frames, the similarity between two shots is evaluated as the similarity between their key-frames. By indicating with t_i the value of a time variable corresponding to the occurrence of the key-frame of the shot S_i, we can build, for each shot S_k, the set T_k. It contains all the values t_i relative to the shots S_i whose similarity with the shot S_k is lower than a suitable threshold τ_S; in other words, it contains the time occurrences of all the shots similar to S_k. The shot lifetime $L(S_k)$ can be then defined as the difference between the last and the first value of t_i

Since anchor shots occur repeatedly through the video, the shot classification is performed by attributing to the anchor shot class all the shots S_k having the value of $L(S_k)$ greater than a suitably chosen threshold τ_l. The value of τ_l can be determined according to the statistics of the specific video database. In [5] it is fixed to 4.5 s.

This classification is then refined by computing an index Q_S that measures the quantity of motion for each candidate anchorperson shot. This index Q_S is calculated as the sum of all the frame-to-frame difference between the key-frame and all the subsequent frames in the shot. So, only those shots whose Q_S value does not exceed a threshold τ_Q are definitely classified as anchor shots.

Note that for this method three thresholds need to be evaluated: τ_S, τ_l, and τ_Q.

2.2 GAO Algorithm

In this case [4], video shots are classified by using an algorithm based on graph-theoretical cluster (GTC) analysis. More in details, the authors propose an anchor shot detection scheme composed of four steps: short shot filtering, key-frame extraction, GTC analysis and post-processing.

In general, an anchorperson shot should last for more than 2 s, since this shot should involve at least one sentence pronounced by the reporter. Therefore, if a shot lasts less than 2 s it is considered as a news-report shot. Otherwise, it is further analyzed through later steps. The second step is the key-frame extraction: the authors propose that the middle frame is taken as the key-frame. These key-frames are the input to the GTC analysis module. It considers them as vertices in a feature space and then constructs the minimum spanning tree (MST) on these vertices. To do that, a distance between key-frames, based on the color histograms, is defined. It is used for weighting each edge connecting two vertices. Successively, by removing from the MST all the edges with weights greater than a threshold γ, a *forest* containing a certain number of subtrees (*clusters*) is obtained. In this way, the GTC method automatically groups similar vertices (i.e., key-frames) into clusters.

The key-frames composing a cluster are classified as potential anchorperson frames if the size of the cluster is greater or equal to 2. Starting from this set of potential anchorperson frames, the last step of the proposed detection scheme operates a further filtering. In fact, in some situations, the key-frames in a cluster may have similar color histograms but different content. To detect this situation, a spatial difference metric (SDM) between two key-frames is proposed. If a cluster has an average SDM higher than a threshold λ, the whole cluster is removed from the anchorperson frame list.

It is worth noticing that, in this case, two thresholds, γ for the GTC algorithm and λ for the post-processing step, need to be specified in advance.

2.3 HAN Algorithm

In [6] a template-based method is proposed. It is based on the assumption that an anchor person shot is the only shot that has multiple match of most of its visual content along the whole video, and consists of two steps: a unsupervised procedure for finding the template shots and its use to detect all the anchor person shots in the video sequence by applying an adaptive thresholding.

The authors assume that the first anchor shot in a news video appears within the first N shots (in the paper N is fixed to 5). A dissimilarity measure is defined between two shot, as it will be specified later. Each shot S_k with $k \in [1,N]$ is then matched with all the other shots, obtaining a set of dissimilarity values for each S_k. For each S_k the P best matches (i.e., the lowest values) out of its set of dissimilarities are averaged to compute the overall matching value of the shot S_k. The shot with the lowest overall matching value is assumed to be an anchor shot and is used as template.

The dissimilarity measure between two shots is defined as follows: each shot is represented by means of two frames, one close to the beginning of the shot and the other close to the end of the shot. These frames are merged into the so-called *shot image*. Each shot image is divided into blocks of $M1 \times M2$ pixels and a distance in the

L^*u^*v color space between two blocks belonging to different shot images is defined. The dissimilarity measure between two shot images S_i and S_j, is then defined as the minimum value among all the average distances obtainable by considering each possible matching of blocks b_k belonging to S_i and blocks b_h belonging to S_j. In the paper, for minimizing the computational effort of the exact evaluation of this measure, only the C blocks more similar each other are considered.

Once the anchor shot template has been found, all the remaining shots are checked for individuating the other anchor shots. In particular, all the shots whose similarity with the template shot is lower than an adaptive threshold M are detected as anchor shots. Such threshold is proportional to a suitably chosen parameter w.

It is worth noticing that the values of the parameters $M1$, $M2$, P, C and w need to be fixed for this algorithm. In [6], for two video sequences with key-frames of size 165x144 and 180x144 respectively, $M1$ and $M2$ were both fixed to 8, while P was set to 3, w to 3.0 and C was considered as the 70% of the total number of blocks.

3 Experimental Results

Some efforts have been spent in the recent past by other researchers in building video databases for benchmarking purposes; in particular in [7] a database was built in order to characterize the performance of shot change detection algorithm. This database, however, is not adequate for our aims, since it is made up not only of news videos but also of sport events and sitcom videos, and the duration of news videos is only 20 minutes.

So, we decided to build-up a new database. The acquisition was performed by means of the digital satellite decoder *emme esse 6000pvr*. It has an internal hard disk that allowed us to record in the DVB MPEG-2 format. Then, the videos were transferred on a PC, so preserving the broadcasting quality. We encoded the videos in the MPEG-1 format using the TMPGEnc encoder (ver. 2.01). The parameters used to encode the videos were selected taking into account the storage requisites, without decreasing the performance of the algorithms with respect to those obtainable with the original full-quality videos. In particular, we selected four videos from our database for analyzing the dependence of the algorithms' performance on both the frame size and the bit-rate. Two frame sizes have been considered: 704x576 and 352x288. We verified that the information loss due to the 352x288 frame size does not allow the algorithms to perform the same as on the full-quality videos. On the other hand, if the video is coded with at least 1500 kbit/s and the frame size is 704x576, all the algorithms perform as well as on the original videos. In order to keep a tolerance margin, we decided to encode the videos with approximately 2000 kbit/s.

To reproduce as much as possible the variability of the phenomenon under study, different news video editions of a single broadcaster should be considered, as well as news videos of different broadcasters. However, while in the first case the different editions are usually less than ten, in the latter the number of different models is significantly large. As a consequence, we preferred to test our system on a database composed by all the different news videos captured from a single broadcaster, rather than perform tests with only few samples belonging to a large number of different broadcasters. Even if some archiving companies work with large quantities of videos from different sources, this approach fits the most realistic use case for the proposed system. A typical broadcaster, in fact, should be interested to employ such a system for analyzing all the editions of its news videos.

The database used in this paper is composed by more than thirty news videos from the main Italian public TV-network (namely, RAI 1). As it can be easily noted from Table 1 its size is large; this is more evident if it is compared with the databases used in the papers of BER, GAO and HAN.

Table 1. Composition of the databases used in this paper and in [4], [5] and [6].

Paper	Total length (hh:mm:ss)	Number of videos	Number of Broadcasters	Number of Anchor/News-report shots
This	09:24:19	34	1	464 / 5705
[4]	05:05:17	14	2	253 / 3654
[5]	02:41:00	12	6	66 / 665
[6]	00:37:00	2	-	22 / --

As a first step for the assessment of the performance of the three anchor shot classification algorithms, we calculated their *Precision-Recall* curves [7]. Each point of these curves represents the performance in terms of *Precision* and *Recall* obtained by the algorithm using a specified set of thresholds. Each considered technique is characterized by several thresholds; hence, for each technique a family of *Precision-Recall* curves can be drawn. Each curve is obtained by varying the value of a threshold, holding fixed the remaining ones. In particular, in Figures 1-3 the operating curves for GAO, BER and HAN, respectively, are shown.

The operating curves for GAO were obtained by varying both the thresholds λ (in the range 1500-3500 with step 500) and γ(in the range 20000-45000 with step 5000). Differently, the curves for BER were obtained by varying the value of τ_S in the range 25000-50000 with step 5000 and τ_Q in the range 25-150 with step 25, while holding fixed $\tau_l = 4.5$. This threshold, in fact, does not significantly influence the overall algorithm performance. Finally, there is a single operating curve for HAN. It is obtained by varying the value of w (in the range 1.5-4.0 with step 0.1), while holding fixed the other parameters. We used the same values suggested by the authors for N, P and C, while $M1$ and $M2$ were both fixed to 32, so as to have the same number of blocks per shot image of the original paper. In this case it is founded that variations in the values of P and C do not implies variations on the operating curve for HAO.

So, a first result is that for BER and HAN only a subset of the parameters are really significant. In fact, it is possible to allow some parameters to vary within wide ranges of values without significant changes in the obtainable performance. However, the families of curves shown in Figs. 1-3 are not suitable for a comparison of the three algorithms. Depending on the chosen operating point, it is possible to detect a different curve which allows maximizing the performance. This also implies that a unique set of values that maximizes the performance of each algorithm does not exist, except that for the HAN algorithm. Hence, the comparison of the three algorithms has been carried out on the basis of the envelope of the *Precision-Recall* curves. For each algorithm the points of the envelope were obtained by considering for each value of the *Precision* the values of the parameters that maximized the *Recall*.

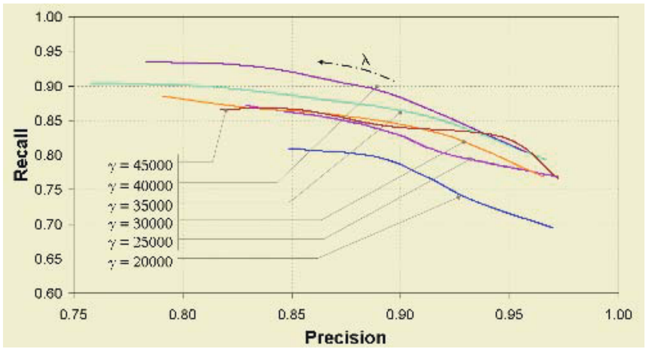


Fig. 1. Operating curves for GAO. Each curve is obtained by varying the value of λ while holding fixed the value of γ. The dashed arrow indicates the direction of increasing values of λ. The six curves accounts for six different values of the threshold γ.

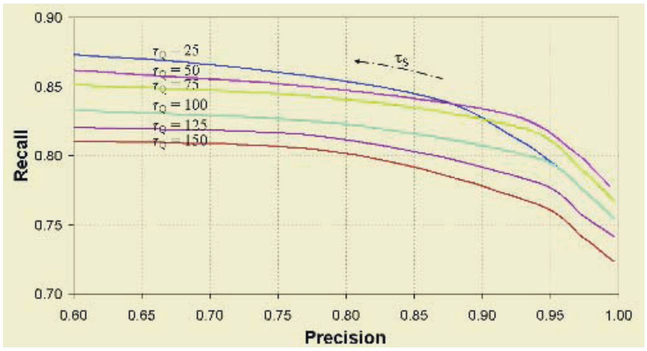


Fig. 2. Operating curves for BER. Each curve is obtained by varying the value of τ_S while fixing the other parameters' values. The dashed arrow indicates the direction of increasing values of τ_S. The eight curves are relative to eight different values of the threshold τ_Q.

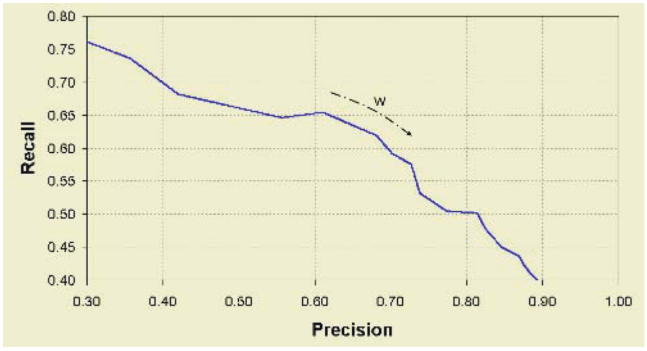


Fig. 3. Operating curve for HAN. It is obtained by varying the value of *w* while holding fixed the other parameters' values. The dashed arrow indicates the direction of increasing values of *w*.

In Figure 4 the envelopes of the operating curves of the three algorithms are reported. Curves in Figure 4 clearly show that HAN performs much worse than BER and GAO for all the operating conditions. Differently, the behaviour exhibited by both the other two algorithms is characterized by a high and stable value of the *Recall* (above 0.80) for almost all the values of the *Precision*. Furthermore, it is interesting to note that the GAO algorithm is preferable if we are looking for high *Recall* values (i.e., higher than 0.90). On the contrary, if it is mandatory to have very high *Precision* values (i.e., almost no false alarms), the BER algorithm must be chosen.

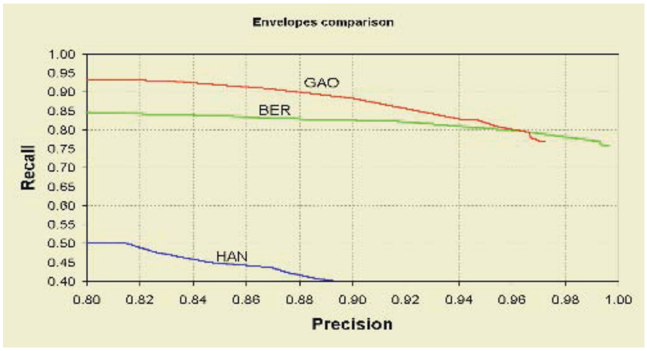


Fig. 4. Envelopes of the operating curves for the three considered algorithms.

In order to provide a more global comparison among the three different algorithms and for comparing the results obtained on our database with those achieved by the algorithms in their original papers, a unique figure of merit, as the parameter F defined in [8], can be used. It combines *Precision* and *Recall* as in the following:

$F = (2 * Precision * Recall) / (Precision + Recall)$.

For choosing the operating conditions of BER and GAO algorithms to be used on the whole database, a preliminary tuning phase was required. In particular, we chose the optimal values of the thresholds by means of an empirical optimization, i.e., by maximizing F over a predefined set of videos. In this set we included the same videos already used for setting up the MPEG-1 coding parameters for the whole database. These videos were not included in the successive tests.

Table 2 reports the global performance of each algorithm; a first result is the discrepancy between the performance reported in Table 2 and the results presented in the original papers (reported for the sake of comparison in parenthesis). All the algorithms perform worse on our database: this is particularly true for the HAN algorithm. Moreover, GAO algorithm performs better in terms of *Precision*, while in [4] its behavior was the opposite one. Analogously, BER algorithm exhibits on our database a *Recall* value higher than the *Precision* one, differently from the behavior described in [5]. Since the anchor shots/news-report shots ratio of our database is quite similar to those of the databases used in the original papers (excluding HAN, where the ratio is not specified), such discordances are mainly due to the different size of the database used for the testing the algorithms in this paper. This confirms the importance of testing algorithms on a large and significant database. Finally, the results reported in Table 2 point out that, even if BER algorithm outperforms the remaining two in terms of *Precision* and F, GAO exhibits the best value of the *Recall*.

Table 2. The performance of the three considered algorithm in terms of *Precision*, *Recall* and *F*. For the sake of comparison, the results obtained in the original papers are reported in parenthesis.

	Recall	***Precision***	***F***
GAO	0.929 (*0.973*)	0.842 (*0.976*)	0.881 (*0.974*)
BER	0.816 (*0.970*)	0.987 (*0.955*)	0.892 (*0.962*)
HAN	0.623 (*1.000*)	0.692 (*0.917*)	0.655 (*0.957*)

4 Conclusions

In this paper an experimental comparison of three unsupervised algorithms for anchor shot classification was presented. The comparison has been carried out on a news video database consisting of about 10 hours with 464 anchor shots and 5705 news report shots. As it could be expected, it does not exist an algorithm that is definitively better than the others. While HAN algorithm performs always the worst, BER algorithm typically outperforms the remaining two in terms of *Precision*, while GAO exhibits in general higher values of *Recall*. However, the choice of the most suitable algorithm at hand strongly depends on the selected operating conditions.

Future steps of this activity will involve the comparison of other anchor shot detection algorithms. We are also planning to investigate in more details how the similarity measure between key-frames influences the algorithms' performance. To this aim, other similarity measures will be also considered.

References

1. B. Furht, S.W. Smoliar, H. Zhang, Video and Image Processing in Multimedia Systems, Kluwer Publishers, Boston (MA), 1996.
2. Y. Avrithis, N. Tsapatsoulis, S. Kollias, "Broadcast news parsing using visual cues: A robust face detection approach", Proc. IEEE Intern. Conf. on Multimedia and Expo, vol. 3, pp. 1469–1472, 2000.
3. S. Eickeler, S. Muller, "Content-based video indexing of TV broadcast news using Hidden Markov Models", Proc. IEEE International Conference on ASSP, pp. 2997-3000, 1999.
4. X. Gao, X. Tang, "Unsupervised Video-Shot Segmentation and Model-Free Anchorperson Detection for News Video Story Parsing", IEEE Trans. on Circuits and Systems for Video Technology, vol. 12, no. 9, pp. 765-776, 2002.
5. M. Bertini, A. Del Bimbo, P. Pala, "Content-based indexing and retrieval of TV News", Pattern Recognition Letters, vol. 22, pp. 503-516, 2001.
6. A. Hanjalic, R.L. Lagendijk, J. Biemond, "Semi-Automatic News Analysis, Indexing, and Classification System Based on Topics Preselection", Proc. of SPIE: Electronic Imaging: Storage and Retrieval of Image and Video Databases, San Jose (CA), 1999.
7. U. Gargi, R. Kasturi, S.H. Strayer, "Performance Characterization of Video-Shot-Change Detection Methods", IEEE Trans. on Circuits and Systems for Video Technology, vol. 10, no. 1, pp. 1-13, 2000.
8. L. Chaisorn, T.-S. Chua, C.-H. Lee, "A Multi-Modal Approach to Story Segmentation for News Video", World Wide Web, vol. 6, pp. 187–208, 2003.

Diagnosis of Lung Nodule Using the Semivariogram Function

Aristófanes C. Silva[1], Perfilino Eugênio F. Junior[2], Paulo Cezar P. Carvalho[2], and Marcelo Gattass[1]

[1] Pontifical Catholic University of Rio de Janeiro - PUC-Rio
R. Marquês de São Vicente, 225, Gávea
22453-900, Rio de Janeiro, RJ, Brazil
ari@visgraf.impa.br, mgattass@tecgraf.puc-rio.br

[2] Institute of Pure and Applied Mathematics - IMPA
Estrada D. Castorina, 110, Horto, 22460-320
Rio de Janeiro-RJ, Brazil
perfeuge@visgraf.impa.br, pcezar@impa.br

Abstract. This paper proposes using the semivariogram function, to help characterize lung nodules as malignant or benign in computerized tomography images.
The tests described in this paper were carried out using a sample of 36 nodules, 29 benign and 7 malignant. Fisher's Linear Discriminant Analysis (FLDA), Multilayer Perceptron (MLP) and Support Vector Machine (SVM) were performed to evaluate the ability of these features to predict the classification for each nodule. A leave-one-out procedure was performed to provide a less biased estimate of the classifiers performance. All analyzed classifers have value area under ROC curve above 0.9, which means that the results have excellent accuracy. The preliminary results of this approach are very promising in characterizing nodules using semivariogram function.

1 Introduction

Lung cancer is known as one of the cancers with shortest survival after diagnosis [1]. Therefore, the sooner it is detected the larger the patient's chance of cure. On the other hand, the more information physicians have available, the more precise the diagnosis will be.

Lung nodules have a structure of very complex tissue. There can be nodules with tissue alterations almost imperceptible to the human eye and other presenting very noticeable alterations. Tissue variation and, sometimes, the not apparent development of the nodule's shape, make diagnosis very difficult. Pattern variations in a nodule's texture (distribution of attenuation coefficients) provide indications about its malignancy or benignity. Nodule calcifications in the shape of a popcorn, laminate concentric, diffuse or central will probably be benign. However, if the nodule does not have calcifications and presents necrosis areas, it is likely to be malignant [1]. The top row in Figure 1 shows the texture for two benign (a and b) and two malignant (c and d) nodules.

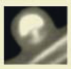
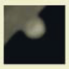
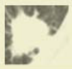
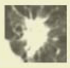
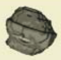
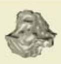
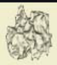
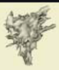
	Benign		Malignant	
Label	(a)	(b)	(c)	(d)
1 Slice				
2 3D Reconstruction				

Fig. 1. Examples of benign lung nodules and malignant lung nodules.

This work intends to investigate the semivariogram function (a geostatistical function), applied to CT images of three-dimensional nodules and to determine whether they are effective in the diagnosis of lung nodules. The nodule's malignancy or benignity is determined by applying Fisher's Linear Discriminant Analysis, Multilayer Perceptron and Support Vector Machine. The validation of the classifiers is done by means of the leave-one-out technique. The analysis and evaluation of tests are done using the area under the ROC curve.

2 Methods

2.1 Image Acquisition

The images were acquired with a Helical GE Pro Speed tomography under the following conditions: tube voltage 120 kVp, tube current 100 mA, image size 512×512 pixels, voxel size $0.67 \times 0.67 \times 1.0$ mm. The images were quantized in 12 bits and stored in the DICOM format [2].

2.2 3D Extraction of Lung Nodules

In most cases, lung nodules are easy to be visually detected by physicians, since their shape and location are different from other lung structures. However, the nodule's voxel density is similar to that of other structures, such as blood vessels, which makes automatic computer detection difficult. This happens especially when a nodule is adjacent to the pleura. For these reasons, we have used the 3D region-growing algorithm with voxel aggregation [3], which provides physicians greater interactivity and control over the segmentation and determination of required parameters (thresholds, initial and final slice, and seed).

Two other resources help and provide greater control in the segmentation procedure: the barrier and the eraser. The barrier is a cylinder placed around the nodule by the user with the purpose of restricting the region of interest and stopping the segmentation by voxel aggregation from invading other lung structures. The eraser is a resource of the system that allows physicians to erase undesired structures, either before or after segmentation, in order to avoid and correct segmentation errors [4]. The bottom row in Figure 1 shows the 3D reconstruction of the nodules in the top row and exemplifies the nodule segmentation.

2.3 Semivariogram Function

Semivariance is a measure of the degree of spatial dependence between samples. The magnitude of the semivariance between points depends on the distance between the points. A smaller distance yields a smaller semivariance and a larger distance results in a larger semivariance. The plot of the semivariances as a function of distance from a point is referred to as a semivariogram. The semivariogram function summarizes the strength of associations between responses as a function of distance, and possibly direction [5].

A semivariogram has three main features: its sill, range, and nugget (Figure 2). The sill is the ordinate value at which the semivariogram levels off, that is, its asymptotic value; the range is the distance at which this leveling off occurs, that is, the spatial extent of the structure in the data; and the nugget is the semivariance at a distance 0.0, that is, the intercept. A nonzero nugget can imply either intrinsic variability in the data (the component typically ascribed to "sampling error"), or it might indicate that the sampling was conducted at an inappropriate spatial scale, that is, there is considerable variability at scales smaller than the smallest between-point distance.

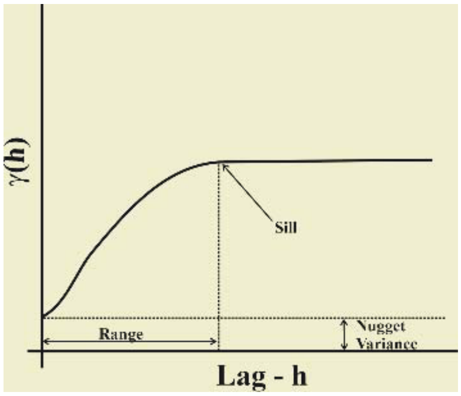


Fig. 2. Semivariogram and its main features: range, sill, and nugget.

The semivariogram is defined by

$$\gamma(h) = \frac{1}{2N(h)} \sum_{i=1}^{N(h)} (x_i - y_i)^2 \tag{1}$$

where h is the lag (vector) distance between the head value (target voxel), y_i, and the tail value (source voxel), x_i, and N(h) is the number of pairs at lag h.

When computing directional experimental semivariograms in 3D, two angles are used to define the direction vector: azimuth and dip. To define the rotation of a vector, we assume the unrotated vector starts in the +y direction. The azimuth angle is the first angle of rotation and it represents a clockwise rotation in the horizontal plane starting from the +y axis. The dip angle is the second angle of rotation and it represents a downward rotation of the vector from the horizontal plane. Other parameters used for semivariogram calculations as lag space, lag tolerance, direction, angular tolerance, maximum bandwidth are exemplified in the Figure 3.

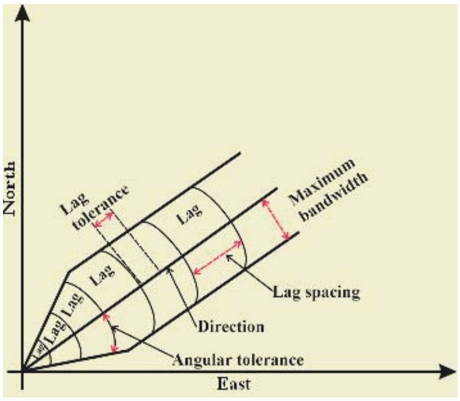


Fig. 3. Parameters used for semivariogram calculations.

2.4 Classification Algorithms

A wide variety of approaches has been taken towards the classification task. Three main historical strands of research can be identified [6]: statistical, neural network and machine learning. This section give an overview of Fisher's Linear Discriminant Analysis, Multilayer Perceptron and Support Vector Machine based on paradigms cited above.

Fisher's Linear Discriminant Analysis - FLDA: Linear discrimination, as the name suggests, looks for linear combinations of the input variables that can provide an adequate separation for the given classes. Rather than look for a particular parametric form of distribution, LDA uses an empirical approach to define linear decision planes in the attribute space i.e. it models a surface. The discriminant functions used by LDA are built up as a linear combination of the variables that seek to somehow maximize the differences between the classes [7]:

$$y = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n = \beta' x \qquad (2)$$

The problem then reduces to finding a suitable vector β. There are several popular variations of this idea, one of the most successful being the Fisher Linear Discriminant Rule. Fisher's Rule is considered a "sensible" classification, in the sense that it is intuitively appealing. It makes use of the fact that distributions that have a greater variance between their classes than within each class should be easier to separate. Therefore, it searches for a linear function in the attribute space that maximizes the ratio of the between-group sum-of-squares (B) to the within-group sum-of-squares (W). This can be achieved by maximizing the ratio

$$\frac{\beta' B \beta}{\beta' W \beta} \tag{3}$$

and it turns out that the vector that maximizes this ratio, β, is the eigenvector corresponding to the largest eigenvalue of $W^{-1}B$ i.e. the linear discriminant function y is equivalent to the first canonical variate. Hence the discriminant rule can be written as:

$$x \in i \text{ if } \left|\beta^T x - \beta^T u_i\right| < \left|\beta^T x - \beta^T u_j\right|, \text{for all } j \neq i \tag{4}$$

where $W = \sum n_i S_i$ and $B = \sum n_i (x_i - x)(x_i - x)'$, and n_i is class i sample size, S_i is class i covariance matrix, x_i is the class i mean sample value and x is the population mean.

Multilayer Perceptron: The Multilayer Perceptron - MLP, a feed-forward back-propagation network, is the most frequently use neural network technique in pattern recognition [8], [9]. Briefly, MLPs are supervised learning classifiers that consist of an input layer, an output layer, and one or more hidden layers that extract useful information during learning and assign modifiable weighting coefficients to components of the input layers. In the first (forward) pass, weights assigned to the input units and the nodes in the hidden layers and between the nodes in the hidden layer and the output, determine the output. The output is compared with the target output. An error signal is back propagated and the connection weights are adjusted correspondingly. During training, MLPs construct a multidimensional space, defined by the activation of the hidden nodes, so that the two classes (benign and malignant nodules) are as separable as possible. The separating surface adapts to the data.

Support Vector Machine: The Support Vector Machine (SVM) introduced by V. Vapnik in 1995 is a method to estimate the function classifying the data into two classes [10], [11]. The basic idea of SVM is to construct a hyperplane as the decision surface in such a way that the margin of separation between positive and negative examples is maximized. The SVM term come from the fact that the points in the training set which are closest to the decision surface are called support vectors. SVM achieves this by the structural risk minimization principle that is based on the fact that the error rate of a learning machine on the test data is bounded by the sum of the training-error rate and a term that depends on the Vapnik-Chervonenkis (VC) dimension.

The process starts with a training set of points $x_i \in \Re^n, i = 1, 2, \cdots, l$ where each point x_i belongs to one of two classes identified by the label $y_i \in \{-1, 1\}$. The goal of maximum margin classification is to separate the two classes by a hyperplane such that the distance to the support vectors is maximized. The construction can be thinked as follow: each point x in the input space is mapped to a point $z = \Phi(x)$ of a higher dimensional space, called the feature space, where the data are linearly separated by a hyperplane. The nature of data determines how the method proceeds. There is data that are linearly separable, nonlinearly separable and with impossible separation. This last case be still tracted by the SVM. The key property in this construction is that we can write our decision function using a kernel function $K(x, y)$ which is given by the function $\Phi(x)$ that map the input space into the feature space. Such decision surface has the equation:

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i K(x, x_i) + b \qquad (5)$$

where $K(x, x_i) = \Phi(x).\Phi(x_i)$, and the coefficients α_i and the b are the solutions of a convex quadratic programming problem [10], namely

$$\begin{array}{ll} \min\limits_{w,b,\xi} & \frac{1}{2} w^T \cdot w + C \sum\limits_{i=1}^{l} \xi_i \\ \text{subject to} & y_i \left[w^T \cdot \phi(x_i) + b \right] \geq 1 - \xi_i \\ & \xi_i \geq 0. \end{array} \qquad (6)$$

where $C > 0$ is a parameter to be chosen by the user, which corresponds to the strength of the penality errors and the ξ_i's are slack variables that penalize training errors.

Classification of a new data point x is performed by computing the sign of the right side of Equation 5. An important family of kernel functions is the Radial Basis Function, more commonly used for pattern recognition problems, which has been used in this paper, and is defined by:

$$K(x, y) = e^{-\gamma \|x-y\|^2} \qquad (7)$$

where $\gamma > 0$ is a parameter that also is defined by the user.

2.5 Validation and Evaluation of the Classification Methods

In order to validate the classificatory power of the discriminant function, the leave-one-out technique [12] was employed. Through this technique, the candidate nodules from 35 cases in our database were used to train the classifier; the trained classifier was then applied to the candidate nodules in the remaining case. This technique was repeated until all 36 cases in our database had been the "remaining" case.

In order to evaluate the ability of the classifier to differentiate benign from malignant nodules, the area (AUC) under the ROC (Receiver Operation Characteristic) [13] curve was used. In other words, the ROC curve describes the ability of the classifiers to correctly differentiate the set of lung nodule candidates into two classes, based on the true-positive fraction (sensitivity) and false-positive fraction (1-specificity).

Sensitivity is defined by $TP/(TP+FN)$, specificity is defined by $TN/(TN+FP)$, and accuracy is defined by $(TP+TN)/(TP+TN+FP+FN)$, where TN is true-negative, FN is false-negative, FP is false-positive, and TP is true-positive.

3 Results

The tests described in this paper were carried out using a sample of 36 nodules, 29 benign and 7 malignant. It is important to note that the nodules were diagnosed by physicians and that the diagnosis was confirmed by means of surgery or based on their evolution. Such process takes about two years, which explains the reduced size of our sample.

There were no specific criteria to select the nodules. The sample included nodules with varied sizes and shapes, with homogeneous and heterogeneous characteristics, and in initial and advanced stages of development.

SPSS (*Statistical Package for the Social Sciences*) [14], LIBSVM [15] and NeuralPower [16] were used to training and classification of lung nodules to FLDA, MLP and SVM, respectively. ROCKIT [17] software was used to compute and compare the area under the ROC curve.

Stepwise discriminant analysis [7] was used to select the best variables to differentiate between groups. These measures were used in the FLDA, MLP and SVM classifiers.

In this study, analytical models for the semivariogram were not used; instead, empirical semivariograms were employed. The measures (variables) extracted, considered as texture signatures, were obtained by computing the semivariogram function for a set of directions: dip (Z) $0°$,$-45°$, and $-90°$; for each dip the azimuth (X and Y) is $0°$, $45°$, $90°$, and $135°$. The adopted lag separation distance (h) was 1, tolerance angle of $\pm 22.5°$, and tolerance lag of ± 0.45. The maximum number of lags depends on the dimensions of each image (volume). We have selected the first three and the last lags (h) in a specific direction for each function. These lags were selected because we were interested in verifying slight variations in small distance, but without rejecting the information of larger distances. This way, we had 48 measures (3 dips $\times$ 4 azimuths $\times$ 4 lags) for semivariogram function. The GSLIB [18] software was used to perform these calculations.

We use the following parameters in the MLP classifier: one hidden layer with four units, hiperbolic tangent as the activation function, the value of 0.15 for the learning ratio, the value of 0.75 for the momentum. These parameters were determined through empirical tests.

In the classification via SVM a proposed procedure by the authors of LIBSVM [15] was used to obtain the best constants C and γ with a process of 36-fold cross-validation. In our case, $C = 2048.0$ and $\gamma = 0.03125$.

Figure 4 shows the application of experimental semivariograms to the volumes represented by Figures 1(a), (b), (c) and (d). We verify that benign nodules have an higher sill than malignant nodules, and that the initial slope is much more accentuated. The graph analysis shows the presence of greater dispersion in benign nodules than in malignant nodules. Table 1 shows the results of semivariogram function and studied classifiers. Based on the area of the ROC curve, we have observed that all classifiers have value *AUC* above 0.9, which means results with excellent accuracy [19]. There is not statistically significant difference among ROC curves of the classifiers.

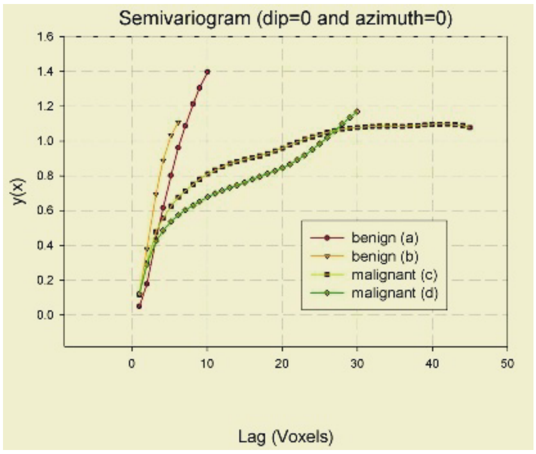


Fig. 4. Semivariogram applied to the example in Figure 1.

Table 1. Analysis of FLDA, MLP and SVM classifiers.

Classifiers	Specificity %	Sensitivity %	Accuracy %	*AUC*
FLDA	86.7	100.0	88.9	0.926 ± 0.071
MLP	93.1	100.0	94.4	0.970 ± 0.046
SVM	100.0	85.7	97.2	0.995 ± 0.019

4 Conclusion

This paper has presented the semivariogram function with the purpose of characterizing lung nodules as malignant or benign. The measures extracted from semivariogram function were analyzed and had excellent discriminatory power, using FLDA, MLP and SVM to classify and the ROC curve to evaluate the obtained results. Based on these results, we have observed that the number of nodules studied in our dataset is too small to allow us to reach definitive conclusions, but preliminary results from this work are very encouraging, demonstrating the potential for multiple variables used in a pattern classification approach to discriminate benign from malignant lung nodules. Nevertheless, there is the need to perform tests with a larger database and more complex cases in order to obtain a more precise behavior pattern.

Despite the good results obtained only by analyzing the texture, further information can be obtained by analyzing the geometry. As a future work, we propose a combination of texture and geometry measures for a more precise and reliable diagnosis.

Acknowledgments

We would like to thank CAPES and FAPERJ for the financial support, Dr. Rodolfo A. Nunes and his team for the clinical support, and the staff from Instituto Fernandes Figueira, particularly Dr. Marcia Cristina Bastos Boechat, for the images provided.

References

1. Tarantino, A.B.: 38. In: Nódulo Solitário Do Pulmão. 4 edn. Guanabara Koogan, Rio de Janeiro (1997) 733–753
2. Clunie, D.A.: DICOM Structered Reporting. PixelMed Publishing, Pennsylvania (2000)
3. Nikolaidis, N., Pitas, I.: 3-D Image Processing Algorithms. John Wiley, New York (2001)
4. Silva, A.C., Carvalho, P.C.P.: Sistema de análise de nódulo pulmonar. In: II Workshop de Informática aplicada a Saúde, Itajai, Universidade de Itajai (2002) Available at http://www.cbcomp.univali.br/pdf/2002/wsp035.pdf.
5. Clark, I.: Practical Geostatistics. Applied Sience Publishers, London (1979)
6. Michie, D., Spiegelhalter, D.J., Taylor, C.C.: Machine Learning, Neural and Statistical Classification. Ellis Horwood Series in Artificial Intelligence, NJ, USA (1994)
7. Lachenbruch, P.A.: Discriminant Analysis. Hafner Press, New York (1975)
8. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. Wiley-Interscience Publication, New York (1973)
9. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press, New York (1999)
10. Haykin, S.: Redes Neurais: Princípios e Prática. 2 edn. Bookman, Porto Alegre (2001)
11. Burges, C.J.C.: A Tutorial on Support Vector Machines for Pattern Recognition. Kluwer Academic Publishers. (1998)
12. Fukunaga, K.: Introduction to Statistical Pattern Recognition. 2 edn. Academic Press, London (1990)
13. Erkel, A.R.V., Pattynama, P.M.T.: Receiver operating characteristic (ROC) analysis: Basic principles and applicattions in radiology. European Journal of Radiology **27** (1998) 88–94
14. Technologies, L.: SPSS 11.0 for windows. Available at http://www.spss.com (2003)
15. Chang, C.C., Lin, C.J.: LIBSVM – a library for support vector machines (2003) Available at http://www.csie.ntu.edu.tw/ cjlin/libsvm/.
16. Software, C.X.: Neuralpower professional v. 1.0. Available at http://www.geocities.com/neuralpower/ (2003)
17. Metz, C.E.: ROCKIT software. Available at http://www-radiology.uchicago.edu/krl/toppage11.htm (2003)
18. Deutsch, C.V., Journel, A.G.: GSLIB. Geostatistical Software Library and User's Guide. Oxford University Press, New York (1992)
19. Greinera, M., Pfeifferb, D., Smithc, R.: Principles and practical application of the receiver-operating characteristic analysis for diagnostic tests. Preventive Veterinary Medicine **45** (2000) 23–41

Dictionary-Based Syntactic Pattern Recognition Using Tries

B. John Oommen[1] and Ghada Badr[2]

• Carleton University, *Fellow of the IEEE*
School of Computer Science, Carleton University, Ottawa, Canada : K1S 5B6
oommen@scs.carleton.ca
• Carleton University, Ph.D student
School of Computer Science, Carleton University, Ottawa, Canada : K1S 5B6
badrghada@hotmail.com

Abstract. This paper deals with the problem of estimating a transmitted string X^* by processing the corresponding string Y, which is a noisy version of X^*. We assume that Y contains substitution, insertion and deletion errors, and that X^* is an element of a finite (but possibly, large) dictionary, H. The best estimate $X^{\bullet}$ of X^*, is defined as that element of H which minimizes the Generalized Levenshtein Distance $D(X, Y)$ between X and Y, for all $X \in H$. All existing techniques for computing $X^{\bullet}$ requires a separate evaluation of the edit distances between Y and every $X \in H$. In this paper, we show how we can evaluate $D(X, Y)$ for every $X \in H$ simultaneously, without resorting to any parallel computations. This is achieved by resorting to the use of an additional data structure called the Linked List of Prefixes (LLP), which is built "on top of" the trie representation of the dictionary. The computational advantage (for a dictionary made from the set of 1023 most common words augmented by computer-related words) gained is at least 50% and 80% measured in terms of the time and the number of operations required respectively. The accuracy forfeited is negligible.

1 Introduction

We consider the traditional problem involved in the syntactic Pattern Recognition (PR) of strings, namely that of recognizing garbled words (sequences). Let Y be a misspelled (noisy) string obtained from an unknown word X^*, which is an element of a finite (but possibly, large) dictionary H, where Y is assumed to contain Substitution, Insertion and Deletion (SID) errors. Various algorithms have been proposed to obtain an appropriate estimate X^+ of X^*, by processing the information contained in Y.

Damarreau [3], [13], [19] was probably the first researcher to observe that most of the errors found in strings were either a single substitution, insertion, deletion or a reversal (transposition) error. Thus the question of computing the dissimilarities between strings was reduced to that of comparing them using these edit transformations. In much of the existing literature, the transposition operation has been modelled as a sequence of a single insertion and deletion.

The first breakthrough in comparing strings using the three (the SID) edit transformations was the concept of the Levenshtein metric introduced in coding theory [11], and its computation. The Levenshtein distance, $D(X, Y)$, between two strings, X and Y, is defined as the minimum number of edit operations required to transform one string to another. Many researchers, among whom are Wagner and Fisher [23], generalized it by using edit distances which are symbol dependent, and can be perceived as a metric [3], [19], [20]. The latter distance goes by many names, but we shall call it the Generalized Levenshtein Distance (GLD). The GLD has also been studied for parameterized [4],[17] inter-symbol distances. Wagner and Fisher [23] and others [19] also proposed an efficient algorithm for computing this distance by utilizing the concepts of dynamic programming. This algorithm is optimal for the infinite alphabet case. Various amazingly similar versions of the algorithm are available in the literature, a review of which can be found in [3], [19], [20]. Masek and Paterson [12] improved the algorithm for the finite alphabet case, and Ukkonen [21] designed solutions for cases involving other inter-*substring* edit operations. Related to these algorithms are the ones used to compute the Longest Common Subsequences (LCS) of two strings [3], [8], [19], [20]. String correction using GLD-related criteria has been done for noisy strings [3], [18], [19], [20], substrings [19], [20], and subsequences [13], and also for strings in which the dictionaries are treated as grammars [19], [20], [22]. Besides these, various probabilistic methods have also been studied in the literature [2], [18]. Indeed, more recently, probabilistic models which *attain the information theoretic bound* have also been proposed [15], [16].

All the algorithms proposed earlier for estimating X^+, requires the separate evaluation of the edit distance between Y and every element of $X \in H$. However, they do not generally utilize the information it has obtained in the process of evaluating any one $D(X_i, Y)$, to compute any other $D(X_j, Y)$. Suppose X_i and X_j have the same prefix $X^{(P)} = a_1a_2...a_p$. Then, previous algorithms would compute the distance $D(a_1a_2...a_p, Y)$ for both of X_i and X_j, and would thus unnecessarily repeat the same comparisons and minimizations for the substring $a_1a_2...a_p$ and *all its prefixes*. Thus, the previous algorithms usually, have many redundant computations.

The first pioneering attempt to avoid the repetitive computations for a finite dictionary, was the one which took advantage of this prefix information, as proposed by Kashyap *et al.* [10]. The authors of [10] proposed a *set-based* algorithm which we refer to as *Algorithm Prefix-Set-Based*, to compute $X^+ \in H$, which minimizes $D(X, Y)$ for a given Y. In contrast to the previous algorithms, in Algorithm Prefix-Set-Based, $D(X, Y)$ was *not* individually evaluated for every $X \in H$. Rather, it calculated $D(X, Y)$ for all $X \in H$ simultaneously, and this was done by treating the dictionary as one integral unit and by using "dictionary-based" dynamic programming principles, as explained presently. It thus took maximum advantage of the information contained in the prefixes of the words of the dictionary. However, the algorithm in [10] was computationally expensive, as we shall see presently, because it required *set-based* operations in its entire execution.

In this paper, we shall show how we can get a feasible implementation for the concepts introduced in [10]. This is achieved by the introduction of a new data structure called the Linked Lists of Prefixes (LLP), which can be constructed when the dictionary is represented using a trie. The LLP is an enhanced, but modified, representation of the trie, which can be used to facilitate the "dictionary-based" dynamic programming calculations. The LLP-based algorithm for the syntactic PR of strings has been rigorously tested. The dictionaries are subsets of a file consisting of the 1023 most common words augmented by words used in the computer literature. The algorithm was tested by recognizing noisy strings generated using the model discussed in [15]. Numerous experiments were done using these noisy strings to compare the accuracy, the time and the number of computations required by the LLP-based enhanced method, and the sequential current-day algorithms. The results demonstrate that by forfeiting a negligible PR accuracy, we can often reduce the time and the number of operations by about 50% and 80% respectively.

In terms of notation, A is a finite alphabet, H is a finite (but possibly large) dictionary, and μ is the null string, distinct from λ, the null symbol. The left derivative of order one of any string $Z = z_1 z_2 \dots z_k$ is the string $Z_p = z_1 z_2 \dots z_{k-1}$. Z_g, the left derivative of order two of Z, is the left derivative of order one of Z_p, and so on. Also, in the interest of brevity, the pertinent results are merely cited here. Their details can be found in [14].

2 Individual and Dictionary-Based Computations

In string-processing applications, the distance metrics employed traditionally quantify $D(X, Y)$, the minimum cost of transforming one string , X, into the other. Y. This distance is intricately related to the costs associated, typically with the individual edit operations, the SID operations. As mentioned earlier, these inter-symbol distances can be of a 0/1 sort, parametric [4], [17] or entirely symbol dependent [10], [19], in which case, they are usually assigned in terms of the confusion probabilities. In all of these cases, the primary dynamic programming rule used in computing the inter-string distance $D(X, Y)$ is:

$$\begin{aligned} D(x_1 \dots x_N,\ \ y_1 \dots y_M) = \min\ [\ &\{D(x_1 \dots x_{N-1},\ \ y_1 \dots y_{M-1}) + d(x_N, y_M)\}, \\ &\{D(x_1 \dots x_N,\ \ y_1 \dots y_{M-1}) + d(\lambda, y_M)\}, \\ &\{D(x_1 \dots x_{N-1},\ \ y_1 \dots y_M) + d(x_N, \lambda)\}]. \quad (1) \end{aligned}$$

Recognition using distance criteria is obtained by essentially evaluating the string in the dictionary which is "closest" to the noisy one as per the metric under consideration.

Rather than compute the individual string edit distance separately, Kashyap *et.al.* in [10], developed a recursive procedure to compute $D(X, Y)$ for all the relevant prefixes in the entire dictionary. This involved only a fixed finite number of the prefixes of X and the left derivative of Y. They introduced a new distance measure, $D_1(X, Y)$, (an intermediate computational tool called a *pseudodistance* because it assumes that the last symbol of X was not inserted during editing) between X and Y. The measure $D_1(X,Y)$ has the desirable properties that it can be computed "recursively" and that the final distance $D(X,Y)$ can be obtained from *it* using only a single additional symbol comparison. The relationship between $D_1(X,Y)$ and $D(X,Y)$ is formalized below (see [10] and [14] for the details):

$D(X,Y) = \min[D_1(X,Y), \{D_1(X_p,Y) + d(x_N,\lambda)\}]$.

Similarly, the recursive properties of the pseudodistances between an arbitrary string $X \in A^*$ and $Y^{(K)}$ can be summarized as follows (see [10] and [14]).

If $X = X_1bc$ (where $|X| \geq 2$) with $|X_1| \geq 0$, since c is not inserted:

$$\begin{aligned} D_1(X_1bc, Y^{(K+1)}) = \min\ [\ & \{D_1(X_1bc, Y^{(K)}) + d(\lambda, y_{K+1})\}, \\ & \{D_1(X_1b, Y^{(K)}) + d(c, y_{K+1})\}, \\ & \{D_1(X_1, Y^{(K)}) + d(b,\lambda) + d(c, Y^{(K+1)})\}]. \end{aligned} \quad (2)$$

Observe that in the above expressions the number of terms included to obtain the distance between X_1bc and $Y^{(K)}$ is merely three, and is superior to the expression valid for finite-state representations for Regular Languages [22]. The reasons for this are explained in detail in [10] and [14].

3 Procedure and Data Structure for Obtaining X^+

In [10], Kashyap *et al.* showed that the pseudodistances $D_1(X, Y^{(K)})$ can be recursively computed using the dynamic programming principle given by Equation (2). For this purpose, they defined two sets $R^{(K)}$ and $S^{(K)}$, where $R^{(K)}$ is the set of prefixes of H into which $Y^{(K)}$ can be transformed with finite pseudodistances, and $S^{(K)}$ associates every element in $R^{(K)}$ with its corresponding pseudodistance with $Y^{(K)}$, for all $K = 1, ..., M$ where $M = |Y|$. The problem with this method of calculation (i.e., using these two sets) , is that the computation is so complicated and is not feasibly implemented. It also requires extensive *set-based computations*. The details of the conceptual representation of the solution of [10], its FSM model, and how it differs from other FSM models given in the literature, are included in [14]. We shall, however, use the dynamic programming recursive relation explained above, and enhance it using a new structure called the Linked List of Prefixes (LLP), which makes the computations feasible.

A *trie* is an alternative to a BST for storing strings in a sorted order [7]. Tries are both an abstract structure and a data structure that can be superimposed on a set of strings over some fixed alphabet [5]. As an abstract structure, they are based on a splitting scheme, which, in turn, is based on how the letters are encountered in the strings. In any specific implementation, the nodes of the trie can be modelled and represented in different ways. They can be implemented using an array [5], [9], a linked list, or even a binary search tree [1]. Fig. 1 (left side) shows an example of a simple dictionary represented as a trie. Observe the relationships between the prefixes of a string and its ancestors in the trie.

We now explain a feasible way of obtaining X^+ when the dictionary is represented using tries. To calculate the best estimate X^+, what we need is to divide

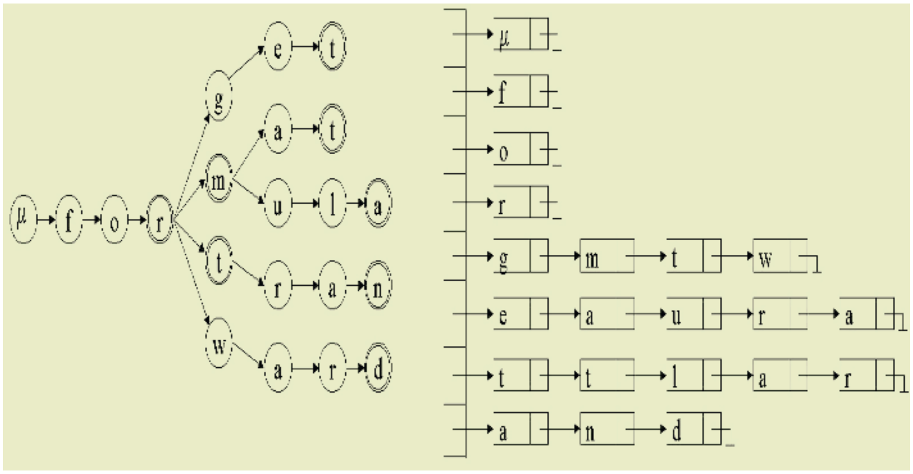


Fig. 1. An example of a dictionary stored as a trie and the corresponding LLP with the words {for, form, fort, fortran, formula, format, forward, forget}.

the dictionary into its sets of prefixes. Each set $H^{(p)}$ is the set of all the prefixes of H of length less than or equal to p, for $1 \leq p \leq N_m$, where N_m is the length of longest word in H. The trie itself divides the prefixes and the dictionary in the way we want, as each sub-trie starting from the root to level p corresponds to all the prefixes in the set $H^{(p)}$. What we need is a data structure that facilitates the trie traversal, and gives us a unique data structure that can always be used to effectively compute the pseudodistances for the prefixes. We called this data structure the Linked Lists of Prefixes (LLP).

The LLP consists of a linked list of levels, *where each level is a level in the corresponding trie.* Each level, in turn, consists of a linked list of all prefixes that have the same length p. The levels are ordered in an increasing order of the length of the prefixes, exactly as in the case of the trie levels. The figure on the right side of Fig. 1 shows the corresponding LLP for the trie shown on the left side of Fig. 1. The character written in each node is actually a pointer to the node of the trie itself, and so we can access the parent nodes in the trie in a straightforward manner, as will be seen in the algorithm presently. The values of D_1 used during calculations is stored in the trie nodes. Thus, actually the LLP is a data structure used to facilitate the traversing of the trie in the proposed string correction algorithm.

Finally, we need to store a linked list of pointers to all the nodes in the trie which corresponds to the words in the dictionary. This list is called"dictionary-words". The pseudo-code for constructing the LLP from a trie and the correctness of the algorithm are given in [14].

3.1 The Procedure for Obtaining X^+

Let U be a prefix corresponding to some node in the trie T. It can be seen from (2) that if the pseudodistance between a certain node U and $Y^{(K+1)}$ has to be computed, it can be done with merely the knowledge of the pseudodistances between each of U, U_p (the parent of U), Ug (the grandparent of U), and the string $Y^{(K)}$ respectively. In each node of the trie we store two values for the pseudodistances, namely the previous value of D_1 and its new value, new_D_1, (both of which are initialized to ∞ for every node in the trie, or rather, in the conceptual LLP). Observe that we need both of these quantities because the computation of D_1 will require the "old" values of D_1 calculated in the previous iteration, which we refer to as $D_1(U, Y^{K-1})$. We then proceed along the trie from the root towards the leaves, and at each iteration, fill in the distances for nodes at the same level and for nodes which are two levels deeper than at the previous iteration. Indeed, it is at this stage that the LLP becomes useful. Rather than work with *set based operations* as in [10], the LLP permits the access to nodes *level by level*, while details of the parents and grandparents are gleaned from the trie itself. The pseudo-code for the computation that formalizes this along the trie and LLP and are omitted here. They can be found in [14].

4 Experimental Results

To investigate the power of our new method with respect to computation various experiments were conducted. The results obtained were remarkable with respect to the gain in time and the number of computations. The new method was compared with Algorithm_GLD, a PR scheme which used any traditional editing [10], [11], [12], [18], [19], [23] algorithm using symbol-dependent costs as described in Section 2, where the costs were assigned for elementary distances using the GLD, and the inter-string distances were computed sequentialy.

The Dictionary consisted of 342 words obtained as a subset of the most common English words [6] augmented with words used in computer literature[1]. The length of the words was greater than or equal 7 and the average length of a word was approximately 8.3 characters. Other experiments (see [14]) were also done for larger subsets of the most common English words [6].

From these 342 words, five sets, SA, SB, SC, SD, and SE, of 1368 words each were generated using the noise generator model described in [15]. We assumed that the number of insertions was geometrically distributed with parameter $\beta = 0.7$. The conditional probability of inserting any character $a \in A$ given that an insertion occurred was assigned the value 1/26; and the probability of deletion was set to be 1/20. The table of probabilities for substitution (typically called the confusion matrix) was based on the proximity of character keys on the standard QWERTY keyboard and is given in [15][2]. The statistics associated with each of the five sets are given in Table 1. Some of the words in the dictionary are very similar even before garbling such as "official" and "officials"; "attention", "station" and "situation". These are words whose noisy versions can themselves easily be mis-recognized. The errors per word associated with these five sets was bounded by 30%, 40%, 50%, 60% and 100% respectively.

[1] This file is available at `www.scs.carleton.ca/~oommen/papers/WordWldn.txt`

[2] It can be downloaded from `www.scs.carleton.ca/~oommen/papers/QWERTY.doc`

Table 1. Noise statistics of the set SA, SB, SC, SD, and SE.

Errors	SA	SB	SC	SD	SE
Number of insertions	873	1105	1545	1766	2763
Number of deletions	461	503	549	567	633
Number of substitutions	808	931	1028	1051	1120
Total number of errors	2142	2539	3122	3384	4516
Average % error	18.91	22.41	27.26	29.87	39.87
Maximum % error per word	30.00	40.00	50.00	60.00	100.00

Table 2. The experimental results obtained from each of the three sets for the 1368 noisy words. The results are given in terms of the number of operations needed, the time and accuracy. The results also shows the percentage of savings in the total number of operations and the time used when utilizing the LLP-based method as opposed to the sequential method using Algorithm-GLD.

Operation	SA		SC		SE	
	GLD	LLP	GLD	LLP	GLD	LLP
Add	430529184	69466624	451174752	73954200	491263920	82668160
Min	132990720	18770224	139606272	19892118	152452224	22070608
Oper.	563519904	88236848	590781024	93846318	643716144	104738768
Savings	84.34		84.11		83.72	
Time (sec.)	19	8	19	8	21	10
Savings	57.89		57.89		52.38	
Accuracy	98.83	98.54	97.59	97.37	96.05	95.76

The two algorithms, Algorithm_GLD (the algorithm which sequentially computed the GLD for the entire dictionary) and our algorithm, Algorithm_LLP, were tested with the five sets of noisy words. We report the results obtained in terms of the number of computations (additions and minimizations), the time, and the accuracy for only the three sets SA, SC, SE, in Table 2. The results shows the significant benefits of the LLP-based method with respect to the time and number of computations. For example, for the set SA, the number of operations is 563,519,904 for Algorithm_GLD, and 88,236,848 for the LLP-based method with a saving of 84.34%. The time taken (on a Pentium II processor, 1000 GHZ) is 19 seconds for Algorithm_GLD and just 8 seconds for the LLP-method, which is a saving of 57.89%.

The savings in the computations are more than 80% for all sets which is interesting, and the saving in time is more than 50%. The accuracy is very slightly less than what can be obtained from the sequential computation, because some of the words which contained two successive deletions, were not correctly recognized by the LLP method. Indeed if these words were removed from the test data, the accuracy will be the same for both schemes. For example, for the set SE the test results shows that both schemes give the same recognition except for one string in the LLP method, namely for the word "property". The noisy word in this case was "opertzg" in which the first two symbols were successively deleted. X^*, which generated Y, was estimated as the word "experts" by the LLP method.

5 Conclusion and Future Work

In this paper we have presented a feasible solution for the problem of estimating a transmitted string X^* by processing the corresponding string Y, which is a noisy version of X^*, an element of a finite (but possibly, large) dictionary H, when the whole dictionary is considered simultaneously. The method builds on the concepts introduced by Kashyap *et al.* [10], where the set model used in the computations was not feasible. We enhanced this by the introduction of a new data structure called the Linked Lists of Prefixes (LLP), which can be constructed when the dictionary is represented using a trie. The LLP is an enhanced, but modified, representation of the trie, which can be used to facilitate the "dictionary-based" dynamic programming calculations. The LLP-based algorithm for the syntactic PR of strings has been rigorously tested. The results showed significant benefits (with respect to the time and number of computations) when compared with Algorithm_GLD, the algorithm which sequentially computes the GLD for the entire dictionary.

As a future work we would like to extend this method for probabilistic computations and "two-sided tries".

References

1. J. Bentley and R. Sedgewick. Fast algorithms for sorting and searching strings. *Eighth Annual ACM-SIAM Symposium on Discrete Algorithms New Orleans*, January 1997.
2. P. Bucher and K. Hoffmann. A sequence similarity search algorithm based on a probabilistic interpretation of an alignment scoring system. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology, ISMB-*, volume 96, pages 44–51, 1996.
3. H. Bunke. Structural and syntactic pattern recognition. *in Handbook of Pattern Recognition and Computer Vision. Edited by C.H.Chen, L.F.Pau and P.S.P. Wang, World Scientific, Singapore*, 1993.
4. H. Bunke and J. Csirik. Parametric string edit distance and its application to pattern recognition. *IEEE Trans. Systems, Man and Cybern, SMC-*, 25:202–206, 1993.
5. J. Clement, P. Flajolet, and B. Vallee. The analysis of hybrid trie structures. *Proc. Annual A CM-SIAM Symp. on Discrete Algorithms, San Francisco, California*, pages 531–539, 1998.
6. G. Dewey. *Relative Frequency of English Speech Sounds.* Harvard Univ. Press, 1923.
7. S. Heinz, J. Zobel, and H. Williams. Burst tries: A fast, efficient data structure for string keys. *ACM Transactions on Information Systems*, 20(2):192–223, 2002.
8. J. W. Hunt and T. G. Szymanski. A fast algorithm for computing longest common subsequences. *Comm. Assoc. Comput. Mach.*, 20:350–353, 1977.

9. P. Jacquet and W. Szpankowski. Analysis of digital tries with markovian dependency. *IEEE Trans. Information Theory, IT-*, 37(5):1470–1475, 1991.
10. R. L. Kashyap and B. J. Oommen. An effective algorithm for string correction using generalized edit distances -i. description of the algorithm and its optimality. *Inf. Sci.*, 23(2):123–142, 1981.
11. A. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Phys. Dokl.*, 10:707–710, 1966.
12. W. J. Masek and M. S. Paterson. A faster algorithm computing string edit distances. *J. Comput. System Sci.*, 20:18–31, 1980.
13. B. J. Oommen. Recognition of noisy subsequences using constrained edit distances. *IEEE Trans. on Pattern Anal. and Mach. Intel.,PAMI-*, 9:676–685, 1987.
14. B. J. Oommen and G. Badr. Dictionary-based syntactic pattern recognition using tries. Unabridged version of the present paper. Can be made available by contacting the authors.
15. B. J. Oommen and R. L. Kashyap. A formal theory for optimal and information theoretic syntactic pattern recognition. 31:1159–1177, 1998.
16. B. J. Oommen and R. K. S. Loke. Syntactic pattern recognition involving traditional and generalized transposition errors: Attaining the information theoretic bound. Submitted for Pubication.
17. B. J. Oommen and R. K. S. Loke. Designing syntactic pattern classifiers using vector quantization and parametric string editing. *IEEE Transactions on Systems, Man and Cybernetics SMC-*, 29:881–888, 1999.
18. J. L. Peterson. Computer programs for detecting and correcting spelling errors. *Comm. Assoc. Comput. Mach.*, 23:676–687, 1980.
19. D. Sankoff and J. B. Kruskal. *Time Warps, String Edits and Macromolecules: The Theory and practice of Sequence Comparison*. Addison-Wesley, 1983.
20. G. A. Stephen. *String Searching Algorithms*, volume 6. Lecture Notes Series on Computing, World Scientific, Sihgapore, NJ, 2000.
21. E. Ukkonen. Algorithm for approximate string matching. *Information and control*, 64:100–118, 1985.
22. R. A. Wagner. Order-n correction for regular languages. *Comm. ACM*, 17:265–268, 1974.
23. R. A. Wagner and M. J. Fisher. The string to string correction problem. *J. Assoc. Comput. Mach.*, 21:168–173, 1974.

Improving Probabilistic Automata Learning with Additional Knowledge

Christopher Kermorvant[1], Colin de la Higuera[2], and Pierre Dupont[3]

[1] Dept. IRO, Université de Montréal, Canada
kermorvc@iro.umontreal.ca
[2] EURISE, Université Jean Monnet, Saint-Etienne, France
cdlh@univ-st-etienne.fr
[3] INGI, Université de Louvain, Louvain-la-Neuve, Belgique
pdupont@info.ucl.ac.be

Abstract. In this paper, we propose a way of incorporating additional knowledge in probabilistic automata inference, by using typed automata. We compare two kinds of knowledge that are introduced into the learning algorithms. A statistical clustering algorithm and a part-of-speech tagger are used to label the data according to statistical or syntactic information automatically obtained from the data. The labeled data is then used to infer correctly typed automata. The inference of typed automata with statistically labeled data provides language models competitive with state-of-the-art n-grams on the Air Travel Information System (ATIS) task.

1 Introduction

Grammatical inference consists in learning formal grammars for unknown languages when provided with examples of strings belonging (or not) to the language. Regular grammatical inference, in which the target grammar is supposed to be regular, has received most of the attention. If one is provided with positive and negative examples, the Rpni algorithm [OG92] can be used to infer deterministic finite automata.

In the case where only positive examples are available, theoretical results [Gol67] show that the task becomes considerably harder. An alternative is to learn a probabilistic finite automaton from the data, learning the regularities of the distribution rather than those of the language: several algorithms have been proposed [CO94,SO94,TDdlH00] for this task.

These algorithms generally perform well on small tasks but are not currently able to obtain significant results on real world tasks where the size of the alphabet and the noise are serious obstacles. Furthermore, very often the complexity of the intended model is such that the quantity of learning data is insufficient. The success of a model and a learning algorithm depends on their ability to include prior knowledge, in order to compensate for the lack of data. Alternatively, with only a fixed set of data, prior knowledge allows to learn more complex functions.

In the specific context of probabilistic model learning, the success of Hmms in several application domains, like speech recognition or computational biology, is partly due to the use of additional knowledge to design the structure of the models: in speech recognition, the knowledge on the phonemic structure of utterances and in computational biology, additional knowledge regarding the mean length of proteins and additional distribution of amino acids are used to design the models.

We believe that the use of additional knowledge in grammatical inference can bring a number of advantages. Firstly to reduce the search space by excluding automata which do not conform with this knowledge; secondly to complete the learning data with additional knowledge, for example by providing implicit counter-examples: strings known not to belong to the target language; thirdly to introduce in a simple way real world constraints that the induced formal language must satisfy.

Kermorvant & de la Higuera [KdlH02] have proposed a framework based on state typing to include additional knowledge into the automaton inference process. State typing both reduces the search space of the probabilistic finite automata (Pfa) induction (hence decreasing the practical complexity of learning), and guarantees the compatibility of the learned models with this knowledge. The additional knowledge considered in the present paper either comes from statistical clusters or part-of-speech tags.

We compare the use of these two kinds of additional knowledge in the framework of language modeling: on the Air Travel Information System (Atis) task, the results we report are comparable with those achieved by state-of-the art n-grams models.

2 Regular Language Learning from Tagged Data

The search space for the problem of regular languages inference is finite but huge [DMV94]: it is a partition lattice defined by the set of states of the prefix tree acceptor (Pta). The Pta is the smallest tree-shaped deterministic automaton accepting exactly I_+. Under the hypothesis of the presence of a structurally complete sample (*i.e.* a set of strings that makes use of all edges, nodes and final states of the target), the target automaton is guaranteed to belong to this lattice. A negative sample is generally used to control the generalization while searching for the target. However, since the size of the lattice is exponential in the size of the sample set, a good strategy is required to explore this lattice. We choose to learn probabilistic finite automata so that we can, in principle, handle two additional difficulties raised by real data: the lack of negative information and the presence of noise. Besides, background knowledge of the application domain is often available. In [KdlH02] a framework that includes additional knowledge in the automaton inference algorithms is proposed. This framework is the application of typing, as known for terms and trees, to finite state automata.

2.1 Typed Probabilistic Finite State Automata

We consider probabilistic finite state automata (Pfa), which provide a probabilistic extension of finite state automata. A Pfa $\mathcal{A}$ is a tuple $< Q, \Sigma, \delta, \tau, q_0, F >$

Algorithm 1 Generic PFA induction algorithm.

```
Require:
  $I_+$ , training set (sequences)
  $\alpha$, a precision parameter
Ensure: a probabilistic finite state automaton
  A ← build_PPTA($I_+$ )
  while $(q_i, q_j) \leftarrow$ choose_states(A) do
    if is_compatible($q_i, q_j, \alpha$) then
      merge(A,$q_i, q_j$)
  return A
```

where: Q is a finite set of states, Σ is an alphabet, $\delta : Q \times \Sigma \rightarrow Q$ is a transition function, $\tau : Q \times \Sigma \rightarrow]0..1]$ is a function which returns the probability associated with a transition; q_0 is the initial state, $F : Q \rightarrow [0..1]$ is a function which returns the probability for a state to be final. A typed PFA [KdlH02] is defined with the addition of S, a set of types and σ, a typing function which associates a single type to each state of the automaton.

Furthermore, we only consider PFA which are structurally deterministic and which define define a probability distribution on Σ^* (trimmed and satisfy the consistency constraint: $\forall q \in Q, \left[\sum_{a\in\Sigma} \tau(q,a)\right] + F(q) = 1$). The probability assigned by the automaton to a string is classically the product of the transition probabilities along any accepting path and the final probability.

There are several ways to define the typing function σ. In [KdlH02], it was proposed to define the typing function σ by a type automaton constructed by an expert, on the basis of some knowledge he may have of the domain. In this paper, we propose to automatically infer the typing function from strings where the symbols are tagged for example according to a part-of-speech tagger for natural language sentences.

Typing functions could, in theory, be as complex as one may want. Practically we do not want typing to be a burden to learning. Our current choice is to make type-checking easy, even if this limits the expressiveness of the typing function. The typing function (σ) must be able to type all states, and therefore possible strings in a regular manner. Therefore two conditions must be met:

- if L is the set of all possible strings: $\forall u \in L, \sigma(u)$ is defined;
- if we denote by $\sigma_{uv}(u)$ the label of prefix u in the context of string uv: $\forall uv, uw \in L \Rightarrow \sigma_{uv}(u) = \sigma_{uw}(u)$.

A typing function σ is *admissible* if the above conditions hold.

Hence one can associate various types to a symbol, but only one type to a string. It should be noticed that this is a strong condition: in usual cases tags are computed by taking into account both left and right-hand contexts. In section 4 we discuss various ways of relaxing this condition.

2.2 Learning Typed Automata from Automatically Labeled Data

Several algorithms have been proposed to infer PFA from examples using frequencies [CO94,RST95,TDdlH00]. All these algorithms are based on a similar scheme, which is presented in algorithm 1. Our inference algorithm for typed automata from labeled data is also derived from this scheme that we explicit below.

Given a set of labeled positive examples I_+, the algorithm first builds the typed *probabilistic prefix tree acceptor* (PPTA). The typed PPTA is an automaton accepting all examples of I_+, in which the states corresponding to common prefixes are merged and such that a training count is attached to each state and each transition. This count denotes the number of times this state, or transition, is used while parsing the sample.Let $C(q)$ (respectively $C(q,a)$ and $C_f(q)$) denotes the number of times the state q (respectively the transition (q,a) and the final state q) is used while parsing I_+. An estimate $\hat{\tau}$ (resp. $\hat{F}$) of the function τ (resp. F) can be computed from these counts:

$$\forall a \in \Sigma, \hat{\tau}(q,a) = \frac{C(q,a)}{C(q)} \qquad \hat{F}(q) = \frac{C_f(q)}{C(q)}$$

The typing function of the typed PPTA is defined by the labels of the strings in I_+. When a string with label l is used to reach a state q of the typed PPTA, the label l is the type of the state q: $\sigma(q) = l$. Note that the fact that the typing function is admissible implies that a typed PPTA can always be built from a given labeling.

The second step of the algorithm consists in visiting the states of the PPTA (function *choose_states(A)*), and testing whether the states are compatible and can be merged. The compatibility criterion (defined in algorithm 1) by function *is_compatible*(q_i, q_j, α) depends on a precision parameter α. If the states are compatible, they are merged (function *merge(A,*q_i, q_j*)*). Usually, several consecutive merging operations are made in order to maintain the deterministic structure of the automaton. The algorithm halts when no more merging is possible. In the case of algorithm ALERGIA [CO94], the compatibility of two states is based on testing the compatibility of their associated probabilities.

By using only admissible typing functions, the introduction of type constraints in the learning algorithm is straightforward. Every time a merging operation is considered, we check whether the states have the same type. This constraint can easily be implemented in constant time: it is sufficient to test the equality of the types of q_i and q_j in function *is_compatible(*q_i, q_j, α*)*. With this constraint, the type of any string in the inferred language is guaranteed to be consistent with the types of the strings in the learning set.

3 Experiments

We have tested our approaches on a language modeling task with the Air Travel Information System (ATIS) corpus. This corpus has been widely used in the speech recognition community and specifically for probabilistic automaton induction tasks (see *e.g.* [DC98,TDdlH00,LVC02]). This corpus consists in information requests performed in American English.

We use the Atis-2 sub corpus which is composed of a training set containing 13,044 utterances (130,773 tokens) and two test sets containing respectively 974 utterances (10,636 tokens) and 1001 utterances (11,703 tokens). The task vocabulary is composed of 1,296 different words. All the automata are inferred on the train set, the parameters are tuned on the first test set (named validation set), and the second test set (named test set) is used for independent final evaluation.

The usual quality measure in language modeling tasks is *test set perplexity*:

$$PP = 2^{LL} = 2^{-\frac{1}{||S||}\sum_{w\in S}\log_2 P(w)}$$

where $P(w)$ denotes the predicted probability of word w and $||S||$ denotes the number of words in the test set. The smaller the perplexity the better the automaton can predict the strings observed in the test set. It is generally agreed that perplexity is a good quality criterion for language models.

In order to guarantee that every word can be predicted with a non null probability, the inferred automaton must be smoothed. We interpolate the automaton with a unigram model, which defines the probability $P_1(w)$ of each word w in the training set, independently of its context. The probability of a word w assigned by the smoothed automaton is then:

$$P(w) = \beta P_{PFA}(w) + (1-\beta)P_1(w)$$

This smoothing technique is very rudimentary but, as such, it best reflects the quality of the induced Pfa alone. Finally, as some words of the application vocabulary may not occur in the training set[1], the unigram probability itself is smoothed by absolute discounting [KN95].

3.1 Comparison of Two Typing Functions for Automata Inference

In this section, we compare the use of two kind of additional knowledge for typed automata inference: Pos tags and statistical clustering. Our baseline is a standard PFA inference algorithm (Alergia) not using state typing.

The Pos information was obtained by tagging the training set using the Brill tagger [Bri92]. As a first approach, each word was tagged with its most likely tag, disregarding the context rules. The resulting tagged training set contains 32 different Pos types.

The statistical information leading to the class tagging was obtained by the clustering algorithm presented in [DC98]. For a given number of clusters, the clustering algorithm iteratively constructs the classes so that the average mutual information between the classes is maximized. Values for the number of clusters ranging from 10 to 1000 have been tested.

The best standard automata inferred with Alergia yields a perplexity of 66 on the Atis test set. On the same test set, typed automaton inferred using POS tags typing yields a perplexity of 57 and the best perplexity score, 42, is obtained with the typed automaton using statistical clusters. The influence of 3 learning parameters was studied on the Atis development set:

[1] This is the case for 131 out of 1,296 words.

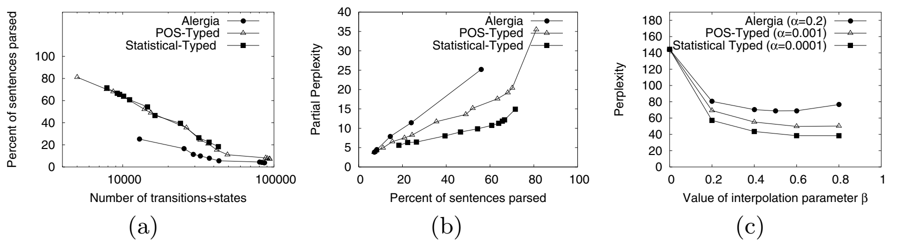


Fig. 1. Results on the validation set: Percentage of sentences correctly parsed versus the number of parameters of the automaton 1(a), perplexity of the sentences correctly parsed 1(b), and perplexity with inferred typed automaton and unigram interpolation 1(c).

- the precision parameter α which controls the compatibility criterion and therefore the number of compatible merging operations,
- the number k of distinct types,
- the interpolation parameter β.

Note that k can only be tuned when the types correspond to statistically induced classes. In this case, the optimal number of classes is 90. In the case of Pos tagging, the number of distinct types is defined *a priori* by the tagger and cannot be tuned. The parameters α and k control the degree of generalization allowed during the typed automaton induction. Hence these parameters control the number of parameters of the inferred model (number of states and transitions of the Pfa). The parameter β controls the weight of the induced Pfa in the combined smoothed automaton.

Figure 1(a) shows the percentage of sentences from the validation set fully parsed by the inferred typed automaton for the two kind of additional knowledge with respect to the number of parameters of the typed automaton (number of states and number of transitions). In this case, no smoothing is used. The typed Ppta parses only 7% of the validation set whereas the universal automaton which accepts all the sentences built with words of the training set, parses 94% of the sentences in the validation set (6% of the sentences are not fully parsed since they contain out-of-vocabulary words). For a fixed number of parameters, the use of typed automata increase the number of sentences that can be parsed compared to standard automata inferred with Alergia.

Figure 1(b) presents the perplexity obtained by the inferred typed automata with respect to the number of sentences parsed. The best results are situated in the bottom right corner of figure 1(b) as they correspond to high coverage and small perplexity. The smoothed unigram parses 100% of the sentences but yields a perplexity of 145. For a given number of parsed sentences, both the Pos-based and cluster-based typed automata yield a smaller perplexity and the typed automaton inferred with statistical classes yields the smallest perplexity. It should be stressed that, as no smoothing is performed in this case, the perplexity is only partial as it is computed over those strings that can be parsed.

Table 1. Two approaches to use two different kinds of information and their best perplexity results on the test set with interpolation to unigram.

	typed automata inference	inference on classes + expansion
POS tags	POS-typed automata perplexity: 57	POS-class automata perplexity: 112
Statistical clusters	cluster-typed automata perplexity: 42	cluster-class automata perplexity: 52

Figure 1(c) shows the perplexity obtained by the inferred typed automaton interpolated with the smoothed unigram. The best perplexity reduction (39% as compared with standard Alergia) is obtained when using typed inference with 90 statistically defined classes with inference parameter $\alpha = 1.10^{-4}$ and interpolation parameter $\beta = 0.8$.

3.2 Comparison with Class-Based Inference

In this section we compare our approach with a method previously introduced to improve automata inference.

Dupont & Chase [DC98] proposed to use statistical clustering of symbols to improve grammatical inference on large vocabularies. The first step of their approach consists in building classes of symbols from the learning samples. Once the classes are defined, each symbol is associated to a class and the probability of each symbol w in its class $g(w)$, denoted by $\hat{P}(w|g(w))$, can easily be computed. The learning samples are then relabeled in terms of classes and an automaton is inferred on the class labels using a classical inference algorithm such as ALERGIA. Finally the automaton is expanded by replacing each class by all the symbols it contains. More formally, once an automaton is inferred on the classes, each transition (q, G) from a state q with label G is replaced by as many transitions as there are symbols w such that $g(w) = G$. The probability estimates $\hat{\tau}(q, w)$ of these transitions are given by $\hat{\tau}(q, w) = \hat{\tau}(q, G) \cdot \hat{P}(w|G)$.

We propose to use the same scheme but with Part-of-Speech classes instead of statistical clusters. The class automaton is inferred with ALERGIA on POS tags and expanded to words afterward. This yields to compare four approaches that are summarized in Table 1. The perplexity results of these four approaches is also shown on Table 1. Our approach, based on typed automata yields better results than the approach based on class inference, both when using POS tags or statistical clusters.

3.3 Improved Smoothing Methods

The smoothing technique used in the evaluations described in section 3.1 and 3.2 is rudimentary. We argued that interpolation with a smoothed unigram guarantees to bound the perplexity while best reflecting the predictive power of the inferred PFA alone. However, if the objective is to minimize test set perplexity, more sophisticated smoothing techniques are required.

A very competitive language model on the ATIS task is a trigram model with Kneser-Ney back-off smoothing [KN95]. This smoothed trigram model combines a trigram model and two back-off distributions, respectively based on a bigram and a unigram model. The ATIS test set perplexity of this combined model is 14.

Current results for the best typed automata inferred with 90 statistically defined classes and smoothed with a simple back-off to unigram (a simplified version of the smoothing scheme described in [LVC02]) gives a perplexity of 20. The trigram model smoothed with the same method (back-off to unigram) gives a perplexity of 17. Further improvements of the smoothing techniques for automata should therefore decrease the perplexity.

It should be noted that the number of parameters needed by the best typed automata combined with a smoothed unigram is $1.1 * 10^5$. The trigram model with Kneser-Ney smoothing to both bigram and unigram needs $6 * 10^5$ parameters. The smoothed typed automata needs less parameters to obtain a similar perplexity on this task.

4 Discussion

It has been shown in [DC98] that the use of statistical class information improves the quality of probabilistic automata used as language models. The present work illustrates that this is even more true when statistically induced classes are combined with typed PFA inference.

The results obtained when using POS tag information are less convincing, even though it has been shown that grammatical information can help language models. Let us stress however that we did not use here the full information provided by the POS tagger as each word was tagged according to its most likely tag, disregarding the contextual rules. This approximation was required to construct a typed PPTA, which is a deterministic PFA as explained in section 2.2. In order to fully take into account POS information, several extensions of the present approach are possible. Firstly inference algorithms could be developed to infer (possibly) non-deterministic structures. Secondly extended typing functions allowing several types per states and inducing multi-typed automata could be developed.

Finally, the framework of typed automata is general and could easily be adapted to other grammatical inference algorithms which have been shown to have better performances than ALERGIA.

5 Conclusion

We have proposed a way to use additional knowledge in grammatical inference with typed automata. When manually or automatically labeled data is available, the labels can be used as types and the inference algorithm we have proposed guarantees that the inferred automaton is compatible with the labeled data. We have compared the use of two kinds of labeling for probabilistic typed automata inference. Part-of-speech labeling provided by a Pos tagger and statistical clustering of words have been compared as labeling for natural language data. The use of statistical word classes information allows us to infer better automata. Our approach provides models which are competitive with state-of-the-art n-grams with similar smoothing techniques while being more compact and needing less parameters.

References

[Bri92] E. Brill. A simple rule-based part-of-speech tagger. In *Proc. of the Conf. on Applied Natural Language Processing*, pages 152–155, 1992.

[CO94] R. C. Carrasco and J. Oncina. Learning stochastic regular grammars by means of a state merging method. In *Proc. Int. Coll. on Grammatical Inference*, volume 862 of *LNAI*, pages 139–152. Springer-Verlag, 1994.

[DC98] P. Dupont and L. Chase. Using symbol clustering to improve probabilistic automaton inference. In *Proc. Int. Coll. on Grammatical Inference*, volume 1433 of *LNAI*, pages 232 – 243. Springer-Verlag, 1998.

[DMV94] P. Dupont, L. Miclet, and E. Vidal. What is the search space of the regular inference? In *Proc. Int. Coll. on Grammatical Inference*, volume 862 of *LNAI*, pages 25–37. Springer-Verlag, 1994.

[Gol67] E. M. Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967.

[KdlH02] C. Kermorvant and C. de la Higuera. Learning languages with help. In *Proc. Int. Coll. on Grammatical Inference*, volume 2484 of *LNAI*, pages 161–173. Springer-Verlag, 2002.

[KN95] R. Kneser and H. Ney. Improved backing-off for N-gram language modeling. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, pages 181–184, 1995.

[LVC02] D. Llorens, J. M. Vilar, and F. Casacuberta. Finite state language models smoothed using N-grams. *Int. J. of Pattern Recognition and Artificial Intelligence*, 16(3):275–289, 2002.

[OG92] J. Oncina and P. García. Identifying regular languages in polynomial time. In H. Bunke, editor, *Adv. in Structural and Syntactic Pattern Recognition*, pages 99–108. World Scientific, 1992.

[RST95] D. Ron, Y. Singer, and N Tishby. On the learnability and usage of acyclic probabilistic finite automata. In *Proc. of the Annual Conference on Computational Learning Theory*, pages 31–40. ACM Press, 1995.

[SO94] A. Stolcke and S. Omohundro. Inducing probabilistic grammars by bayesian model merging. In *Proc. Int. Coll. on Grammatical Inference*, LNAI, pages 106–118. Springer-Verlag, 1994.

[TDdlH00] F. Thollard, P Dupont, and C. de la Higuera. Probabilistic DFA inference using Kullback-Leibler divergence and minimality. In *Proc. Int. Conf. on Machine Learning*, pages 975–982. Morgan Kaufmann, 2000.

Distances between Distributions: Comparing Language Models

Thierry Murgue[1,2] and Colin de la Higuera[2]

• RIM, Ecole des Mines de Saint-Etienne 158, Cours Fauriel
42023 Saint-Etienne cedex 2 – France
• EURISE, University of Saint-Etienne 23, rue du Dr Paul Michelon
42023 Saint-Etienne cedex 2 – France
`murgue@emse.fr, cdlh@univ-st-etienne.fr`

Abstract. Language models are used in a variety of fields in order to support other tasks: classification, next-symbol prediction, pattern analysis. In order to compare language models, or to measure the quality of an acquired model with respect to an empirical distribution, or to evaluate the progress of a learning process, we propose to use distances based on the $L_{\bullet}$ norm, or quadratic distances. We prove that these distances can not only be estimated through sampling, but can be effectively computed when both distributions are represented by stochastic deterministic finite automata. We provide a set of experiments showing a fast convergence of the distance through sampling and a good scalability, enabling us to use this distance to decide if two distributions are equal when only samples are provided, or to classify texts.

1 Introduction

A common task to machine translation [1], speech recognition [2], optical character recognition [3] or computational biology [4] is that of constructing a language model. Typical language models are n-grams [5], but HMMs also can be used [6]. Finite automata have been considered as alternative language models for nearly 10 years [7], with an extension to stochastic automata more adapted to the task [8]. In a more general setting, stochastic or probabilistic automata have been used in structural and syntactic pattern recognition for a number of years [9]. How to derive a grammar or an automata from data is usually called grammatical inference or grammar induction: This has been studied in the framework of pattern recognition [10, 11], with applications to textures in images, fingerprints classification, dynamic systems or recognition of pictures of industrial objects. The problem of learning a stochastic finite automaton is generally considered to be a hard but important one [12], with smoothing a decisive component of language modeling. Clearly, a better understanding of stochastic automata, and moerover of their topological properties is necessary in order to better learn them or compare them. Not much work has been done in this direction: Work linked with the results we report here has been done by Fred [13], who computes the weight of a language following a distribution, or Lyngsø *et al.* [4] who prove that computing the L_2-distance is tractable between distributions over finite sets, or Carrasco who computes the Kulback-Leibler divergence between regular distributions [14] or the L_2-distance between distributions over trees [15].

We can thus identify the following tasks related with language modeling: measure how well a sample corresponds to a distribution or how close a hypothesis language is from a target, or even how close two samples are one from the other. Traditionally researchers in the field have used *perplexity* as its measure for the above questions, with the *Kullback-Leibler divergence* closely related to this measure between a true or target distribution $\mathfrak{D}$ and a hypothesis or candidate distribution $\mathfrak{D}'$:

$$d_{KL}(\mathfrak{D}, \mathfrak{D}') = \sum_{w \in \Sigma^\star} \mathrm{Pr}_{\mathfrak{D}}(w) \log \frac{\mathrm{Pr}_{\mathfrak{D}}(w)}{\mathrm{Pr}_{\mathfrak{D}'}(w)}$$

The Kullback-Leibler divergence suffers from a number of drawbacks:

1. It is not a distance. Therefore topological operations are not easy, and using samples to represent a distribution is not reasonable.
2. In the case where some string has a null probability in $\mathfrak{D}'$, but not in $\mathfrak{D}$, then the Kullback-Leibler divergence is infinite. This implies that over-generalization is necessarily going to appear. The language model can only be considered when it is properly *smoothed.* But in this case it is hard to know if (when concerned with testing) what we are measuring is the quality of the model or that of the smoothing or alternatively of both, combined, which may be what we are looking for.

These seems to be good reasons to propose an alternative measure to see how close one distribution is to another. Furthermore this measure should be a distance, computable and easy to calculate, and not require the models to be smoothed for computation to be possible. We give in section 2 the definitions of stochastic deterministic finite automata (Dpfa), regular distributions, and the probability functions that are associated. Distances between Dpfa are defined and studied in section 3. In section 4, we experimentally study the distances: A first set of experiments (section 4.1) on the well-known Reber grammar [16] show that our measure can give a good estimation of the quality of the learned automata. In section 4.2, we show that the speed of convergence of the distance of a sample to the model is sufficiently fast to be able to decide in practice from which automaton a sample is generated. These experiments are made on artificial data corresponding to parity functions, which are usually considered as a hard case for learning. Then, section 4.3 deals with experiments on real data: firstly, we show the L_2 distance compares favorably to perplexity on a typical speech language modeling task; a second set of experiments (section 4.4) on French poems shows that the distance can be used in classification tasks. They also show the scalability of these methods. In section 5 we discuss further work and conclude.

2 Probability Distributions over Sequences

Definition 1 (Distribution over strings). *Let Σ be an alphabet i.e. a finite set of letter, $\{a, b, \ldots\}$. A string w is an element from Σ^*. The length of a string w is denoted by $|w|$. The unique string with length 0 is λ, $|\lambda| = 0$. A probability function is a function such that $\forall w \in \Sigma^*$, $0 \le \Pr_{\mathfrak{D}}(w)$ and $\sum_{w \in \Sigma^*} \Pr_{\mathfrak{D}}(w) = 1$.*

Definition 2 (Dpfa). *A* DPFA *is a tuple $\mathcal{A} = \langle \Sigma_{\mathcal{A}}, Q_{\mathcal{A}}, q_{I_{\mathcal{A}}}, \delta^{\bullet}_{\mathcal{A}}, p^{\bullet}_{\mathcal{A}}, f_{\mathcal{A}} \rangle$ where $\Sigma_{\mathcal{A}}$ is a finite alphabet, e.g. $\{a, b\}$, $Q_{\mathcal{A}}$ is a set of states, $\{q_0, q_1, q_2, \ldots, q_{|\mathcal{A}|-1}\}$, $q_{I_{\mathcal{A}}}$ is the unique initial state, $\delta^{\bullet}_{\mathcal{A}} : Q_{\mathcal{A}} \times \Sigma_{\mathcal{A}} \mapsto Q_{\mathcal{A}}$ defines a transition function, $p^{\bullet}_{\mathcal{A}} : Q_{\mathcal{A}} \times \Sigma_{\mathcal{A}} \mapsto [0, 1]$ is a probability function, $f_{\mathcal{A}} : Q_{\mathcal{A}} \mapsto [0, 1]$ defines the probability that a state is final.*

Non-deterministic automata exist, but shall not be used in this paper. In the following, when no ambiguity is possible, we will forget subscript $_{\mathcal{A}}$.

To be able to deal with strings, we generalize the automaton probability and transition functions:

Definition 3 (Generalized probability and transition functions). *Let $w \in \Sigma^*$ be a string, let $\mathcal{A}$ be an* DPFA, *and q a state,*

$$p_{\mathcal{A}}(q, w) = \begin{cases} f_{\mathcal{A}}(q) & \text{if } w = \lambda \\ p^{\bullet}_{\mathcal{A}}(q, a) \cdot p_{\mathcal{A}}(\delta^{\bullet}_{\mathcal{A}}(q, a), x) & \text{else } (w = ax, a \in \Sigma, x \in \Sigma^*) \end{cases}$$

$$\delta_{\mathcal{A}}(q, w) = \begin{cases} q & \text{if } w = \lambda \\ \delta_{\mathcal{A}}(\delta^{\bullet}_{\mathcal{A}}(q, a), x) & \text{else } (w = ax, a \in \Sigma, x \in \Sigma^*) \end{cases}$$

Definition 4. *$\pi_{\mathcal{A}}$ is the prefix probability function:*

$$\forall w \in \Sigma^*,\ \pi_{\mathcal{A}}(w) = \sum_{x \in \Sigma^*} p_{\mathcal{A}}(wx)$$

The definitions are linked as follows:

$$\forall w \in \Sigma^*,\ p(w) = \pi(w) \cdot f(\delta(q_I, w))$$
$$\forall w \in \Sigma^*, \forall a \in \Sigma,\ \pi(wa) = \pi(w) \cdot p^{\bullet}(\delta(q_I, w), a)$$
$$\forall w \in \Sigma^*,\ \pi(w) = p(w) + \sum_{a \in \Sigma} \pi(wa)$$

3 Distances between Two Dpfa

In this section we define two measures over string probability distributions. Technically a first step is to compute, given $\mathcal{A}$ and $\mathcal{A}'$ two DPFA, the product probability of reaching two states q in $\mathcal{A}$ and q' in $\mathcal{A}'$:

Definition 5. *Let $(q, q') \in Q_{\mathcal{A}} \times Q_{\mathcal{A}'}$,*

$$\eta_{qq'} = \sum_{\substack{w \in \Sigma^*: \\ \delta_{\mathcal{A}}(q_{I_{\mathcal{A}}}, w) = q \\ \delta_{\mathcal{A}'}(q_{I_{\mathcal{A}'}}, w) = q'}} \pi_{\mathcal{A}}(w) \cdot \pi_{\mathcal{A}'}(w)$$

We describe here a method to compute $\eta_{qq'}$. When Σ^* is not a finite set of strings, the above sum may not be easily computable. We adapt the efficient method that computes the d_2 distance over trees [15] to the string case. The method computes iteratively $\eta_{qq'}$.

Note that λ is the only null-length prefix and $\pi_{\mathcal{A}}(\lambda) = \pi_{\mathcal{A}'}(\lambda) = 1$. That allows us to write:

$$\eta_{q_{I_{\mathcal{A}}} q_{I_{\mathcal{A}'}}} = 1 + \sum_{q \in Q_{\mathcal{A}}} \sum_{q' \in Q_{\mathcal{A}'}} \sum_{\substack{a \in \Sigma: \\ \delta_{\mathcal{A}}(q,a) = q_{I_{\mathcal{A}}} \\ \delta_{\mathcal{A}'}(q',a) = q_{I_{\mathcal{A}'}}}} \eta_{q,q'} \cdot p^{\bullet}_{\mathcal{A}}(q,a) \cdot p^{\bullet}_{\mathcal{A}'}(q',a)$$

and $\forall q \in Q_{\mathcal{A}}, q' \in Q_{\mathcal{A}'} \neq (q_{I_{\mathcal{A}}}, q_{I_{\mathcal{A}'}})$

$$\eta_{qq'} = \sum_{s \in Q_{\mathcal{A}}} \sum_{s' \in Q_{\mathcal{A}'}} \sum_{\substack{a \in \Sigma: \\ \delta_{\mathcal{A}}(s,a) = q \\ \delta_{\mathcal{A}'}(s',a) = q'}} \eta_{s,s'} \cdot p^{\bullet}_{\mathcal{A}}(s,a) \cdot p^{\bullet}_{\mathcal{A}'}(s',a)$$

The above relation corresponds to a system of linear equations. To solve it efficiently, when convergence is clear, terms are computed iteratively during a predefined number k of steps. Complexity then is $\mathcal{O}(|Q_{\mathcal{A}}| \cdot |Q_{\mathcal{A}'}| \cdot |\Sigma| \cdot k)$.

In order to compare two distributions, we first evaluate the co-emission probability of each string, *i.e.* the probability that $\mathcal{A}$ and $\mathcal{A}'$ generate independently a same string.

Definition 6. *The* Co-emission probability *of $\mathcal{A}$ and $\mathcal{A}'$ is*

$$\begin{aligned}\mathrm{CoEm}(\mathcal{A}, \mathcal{A}') &= \sum_{w \in \Sigma^*} (p_{\mathcal{A}}(w) \cdot p_{\mathcal{A}'}(w)) \\ &= \sum_{q \in Q_{\mathcal{A}}} \sum_{q' \in Q_{\mathcal{A}'}} \eta_{qq'} \cdot f_{\mathcal{A}}(q) \cdot f_{\mathcal{A}'}(q')\end{aligned}$$

If comparing samples, co-emission may be null when large vocabulary and long strings are used. It is then reasonable to compare not only the whole strings, but their prefixes:

Definition 7. *As above, the* Prefixial Co-emission probability *of $\mathcal{A}$ and $\mathcal{A}'$ is*

$$\mathrm{CoEmPr}(\mathcal{A}, \mathcal{A}') = \sum_{w \in \Sigma^*} (\pi_{\mathcal{A}}(w) \cdot \pi_{\mathcal{A}'}(w))$$

Note that CoEmPr *is directly linked to η coefficients.*

The above co-emission measures allow us to define two distances for the L_2 norm:

Definition 8 (d_2 distance between two models). *The* distance for the L_2 norm*, denoted by* d_2 *is defined as:*

$$\mathrm{d}_2(\mathcal{A}, \mathcal{A}') = \sqrt{\sum_{w \in \Sigma^*} (p_{\mathcal{A}}(w) - p_{\mathcal{A}'}(w))^2}$$

which can be computed easily by using:

$$d_2(\mathcal{A},\mathcal{A}') = \sqrt{\mathrm{CoEm}(\mathcal{A},\mathcal{A}) + \mathrm{CoEm}(\mathcal{A}',\mathcal{A}') - 2\,\mathrm{CoEm}(\mathcal{A},\mathcal{A}')}$$

Definition 9 (d_{2p} distance between two models). *The* prefixial distance for the L_2 norm*, denoted by* d_{2p} *is defined as:*

$$d_{2p}(\mathcal{A},\mathcal{A}') = \sqrt{\sum_{w\in\Sigma^*} (\pi_{\mathcal{A}}(w) - \pi_{\mathcal{A}'}(w))^2}$$

Theorem 1. d_2 *and* d_{2p} *are distances over* Σ^* *(see [17] for the proof).*

4 Experiments – Results

4.1 Identification Task

The Reber artificial grammar [16] has been used to prove that human language learning is implicit (no conscious rules). Figure 1 shows a DPFA representing this grammar. We compare here distance d_2 and perplexity pp between a learned automaton and the target one.

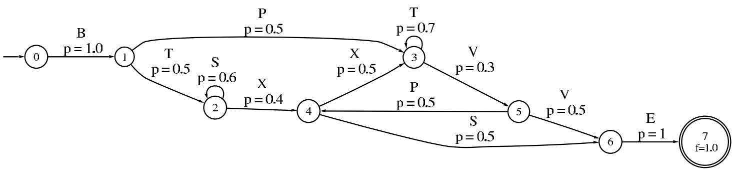


Fig. 1. Reber grammar

We built learning samples with different sizes (100, 500, 1000 strings). Algorithm MDI [8] is used to learn an automaton from these samples. MDI is a state merging algorithm with currently the best results for language modeling by DPFA tasks; It makes use of a parameter (α) to adjust learning.

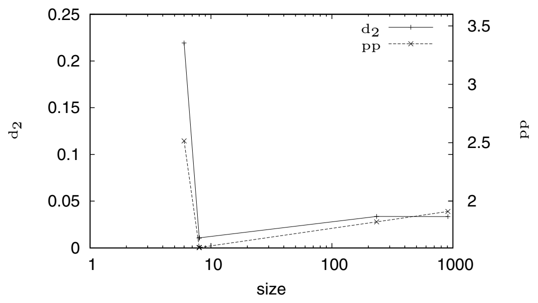


Fig. 2. Comparison between perplexity and distance d. in the exact identification task

Figure 2 shows the results for a sample of size 500, but other sizes give similar results: The structure of the automaton is well identified, but the probabilities are better estimated when sample size becomes larger. Size is drawn in log-scale format. The point corresponding to a 8 states automaton is actually reached by 5 automata learned with 5 different settings of the α parameter (in the range $\{0.005, 0.08\}$). Both pp and d_2 are minimal with the same values of α (*i.e.* on the same learned automaton) corresponding to the case where the target structure is obtained. Over- and under-generalization (which does not reach the correct structure) yields worse results. Again, it should be noted that the d_2 results have been obtained without smoothing the automaton; This is an advantage of the technique.

4.2 Discrimination and Sample Robustness

In this section, the aim is to show that the d_2 distance is robust for sampling, or, in other words, is able to help us deciding if a sample is generated from one automaton or another. A set of experiments and evaluation of the distance behavior was carried out on artificial data. Parity functions [18] have been used in various occasions as examples of functions that are hard to learn. A parity function accepts strings if the total number of bs in a certain number of predefined positions is even. To define this kind of functions as automata, we use *inversion positions expressions* which are strings of n digits from $\{0, 1\}$. They classify strings over a 2-letter alphabet $\Sigma = \{a, b\}$. A parity function f will either accept a string of length n with an even number of bs in inversion positions (marked by a 1), or a string of length $n + 1$ with an odd number of bs in those positions.

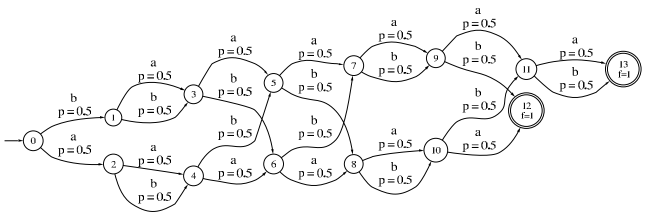


Fig. 3. f. DPFA representation

For the experiment we created three functions/languages: f_1 which represents expression 101101 (represented Figure 3). f_2 corresponds to expression 011010 and f_3 has the same structure as f_1 with small differences on transition probabilities. Computation of the distance d_2 is made on 15 samples (s_i) of different sizes (from 10 to 10000 strings).

Results are shown on Figure 4. X axis represents the size of the data sets for each function. Both X and Y axis are in logarithmic scale.

Figure 4 shows that $\lim_{|s_i|\to\infty} d_2(s_i, f_j) = d_2(f_i, f_j)$. Moreover the convergence is fast. A second point is that with $n \geq 200$ we have a real difference between $d_2(s_1, f_1)$ and the other computations. That allows us to hope to be able to detect if a set has been sampled from a given automaton. We can also,

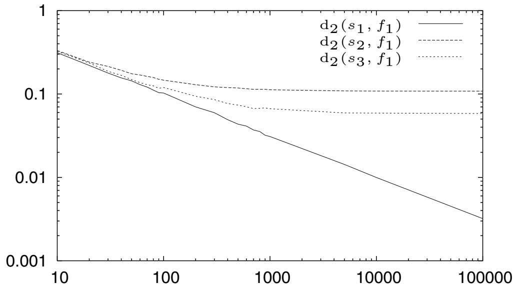


Fig. 4. Convergence of d.

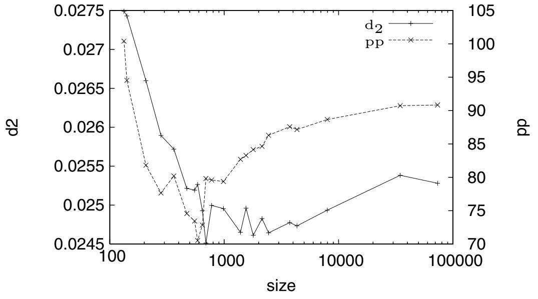


Fig. 5. Behavior of the perplexity and the d. distance on ATIS database

as a relative result, note that the $d_2(s_3, f_1)$ curve is always under the one for $d_2(s_2, f_1)$. We note that f_1 is closer to f_3 than to f_2. Actually f_1 and f_3 represent the same language with small differences on string probabilities.

4.3 Evaluation of the Quality of a Language Model

In this section, we show that the d_2 distance can be a used as a good measure to evaluate if a learned automaton is a good language model or not. As described in the introduction, perplexity is often used to estimate this by using a test sample for validation. Here, we compare d_2 and perplexity between a learned automaton and a test sample. We use the ATIS database [19]. The learning algorithm is this time `Alergia` [20]: The merging process depends on Hœffding bounds (a statistical test is performed) and a generalization parameter α. The original algorithm is slightly modified in order to learn only acyclic automata. After learning automata with different α values, we compute perplexity and d_2 between the learned automaton and a test sample. To compute perplexity we need to smooth the automaton; we choose a simple smoothing method: interpolation with a 1-gram. The results of the experiment are synthesized on Figure 5. X axis is displayed in log scale format.

Table 1. Success rates

Collection	d. rate	d.. rate	# poems	# strings	# symbols
c. .	2.30 %	70.11 %	87	2381	46686
c. .	8.82 %	75.00 %	68	3067	61322
c. .	100.00 %	73.33 %	15	1079	27243
c. .	100.00 %	53.33 %	30	1398	26483

Globally, d_2 behavior is close to that of pp. Moreover, we note that the optimal automata for d_2 and pp have similar sizes. Thus, distance d_2 can be considered as a good measure to estimate the quality of language learned model. Again, we do not need to smooth the automaton to use this measure, which allows it to be more objective when we only want to compare learning methods.

4.4 Classifying Authors

In this section, we use distance d_2 to classify texts from different authors. We used four collections of poems, two from Victor Hugo (c_{H1}, c_{H2}) and the two others from Alphonse de La Martine (c_{M1}, c_{M2}). Each collection is used to learn a language model (intended to represent the poet). From each collection units of text (poems) are extracted. We then pair the collections (c_{Hi} with c_{Mi}) and for each poem from collection 1 (resp. 2) compare it with poets c_{H2} and c_{M2} (resp. c_{H1} and c_{M1}). The experiment is successful if the distance between the poem and the poet is less than the distance from the poem to the other language model.

We present in table 1 the success rates of good classification for all the four collections, with percentages of good classification and number of poems, sentences (strings), word (symbols) of each collection. Globally, distance d_2 provides poor results: this is due to the fact that individual verses (the *string* unit) are seldom repeated in different poems. In this case, the basic d_2 classifier returns the poem collection with the smallest co-emission probability. Distance d_{2p} obtains more convincing results: Common prefixes are reused by a poet. It should be noted that results are only preliminar; We expect a distance measuring common sub-strings to obtain better results.

5 Conclusion

The results reported in this paper indicate that distance d_2 is an alternative to perplexity for language modeling tasks. A certain number of questions arise from our experiments: A theoretical study of the convergence rate would be helpfull; Identification of the type of automata for which each distance is best suited is an important task; We would like to continue the work on authors in order to enter the debate over affiliations; And finally it would be nice to extend the prefix distance, corresponding to a sub-string distance, and (this is harder) compute it in a similar way?

References

1. Amengual, J.C., Sanchis, A., Vidal, E., Benedí, J.M.: Language Simplification Through Error-Correcting and Grammatical Inference Techniques. Machine Learning Journal **44** (2001) 143–159
2. Ney, H.: Stochastic Grammars and Pattern Recognition. In: Proc. of the NATO Advanced Study Institute. Springer-Verlag (1992) 313–344
3. Lucas, S., Vidal, E., Amari, A., Hanlon, S., Amengual, J.C.: A Comparison of Syntactic and Statistical Techniques for Off-Line OCR. In: Proc. of ICGI 94. (1994) 168–179
4. Lyngsø, R.B., Pedersen, C.N.S., Nielsen, H.: Metrics and Similarity Measures for Hidden Markov Models. In: Proc. of ISMB '99. (1999) 178–186
5. Jelinek, F.: Statistical Methods for Speech Recognition. The MIT Press (1998)
6. Morgan, N., Bourlard, H.: Continuous Speech Recognition: an Introduction to the Hybrid HMM/connectionnist Approach. IEEE Signal Processing Magazine **12** (1995) 24–42
7. García, P., Segarra, E., Vidal, E., Galiano, I.: On the Use of the Morphic Generator Grammatical Inference (MGGI) Methodology in Automatic Speech Recognition. International Journal of Pattern Recognition and Artificial Intelligence **4** (1994) 667–685
8. Thollard, F., Dupont, P., de la Higuera, C.: Probabilistic DFA Inference using Kullback-Leibler Divergence and Minimality. In: Proc. of ICML '00. (2000) 975–982
9. Bunke, H., Sanfeliu, A., eds.: Syntactic and Structural Pattern Recognition, Theory and Applications. Volume 7. World Scientific (1990)
10. Fu, K.S., Booth, T.L.: Grammatical Inference: Introduction and Survey. Part I and II. IEEE Transactions on Syst. Man. and Cybern. **5** (1975) 59–72 and 409–423
11. Miclet, L.: Grammatical Inference. In: Syntactic and Structural Pattern Recognition, Theory and Applications. World Scientific (1990) 237–290
12. McAllester, D., Schapire, R.: Learning theory and language modeling. In: Exploring Artificial Intelligence in the New Millenium. Morgan Kaufmann (2002)
13. Fred, A.: Computation of Substring Probabilities in Stochastic Grammars. In: Grammatical Inference: Algorithms and Applications. Volume 1891. Springer-Verlag (2000) 103–114
14. Carrasco, R.C.: Accurate Computation of the Relative Entropy Between Stochastic Regular Grammars. RAIRO **31** (1997) 437–444
15. Carrasco, R.C., Rico-Juan, J.R.: A Similarity Between Probabilistic Tree Languages: Application to XML Document Families. Pattern Recognition, in press **36** (2002) 2197–2199
16. Reber, A.S.: Implicit Learning of Artificial Grammars. Journal of verbal learning and verbal behaviour **6** (1967) 855–863
17. Murgue, T., de la Higuera, C.: Distances Between Distributions: Comparing Language Models. Technical Report RR-0104, EURISE, Saint-Etienne, France (2004)
18. Kearns, M., Valiant, L.: Cryptographic Limitations on Learning Boolean Formulae and Finite Automata. In: 21st ACM Symposium on Theory of Computing. (1989) 433–444
19. MADCOW: Multi-Site Data Collection for a Spoken Language Corpus. In: Proc. DARPA Speech and Natural Language Workshop 92. (1992) 7–14
20. Carrasco, R., Oncina, J.: Learning Stochastic Regular Grammars by Means of a State Merging Method. In: Proc. of ICGI'94. (1994) 139–150

A New Uniform Translocation Distance

Carlos Martín-Vide[2] and Victor Mitrana[1,2]

[1] Faculty of Mathematics and Computer Science, University of Bucharest
Str. Academiei 14, 70109, Bucharest, Romania
[2] Research Group in Mathematical Linguistics, Rovira i Virgili University
Pça. Imperial Tarraco 1, 43005, Tarragona, Spain
{cmv,vmi}@fll.urv.es

Abstract. A basic problem in the area of combinatorial algorithms for genome evolution is to determine the minimum number of large scale evolutionary events (genome rearrangements) that transform a genome into another. The present paper is a contribution to the algorithmic study of genom evolution by translocations which is an area related to pattern recognition. Furthermore, it may be viewed as a contribution to other areas related to pattern recognition like: error estimation, genetic programming, disease diagnosis. In this paper we consider chromosomes as being linear strings that exchange each other prefixes in the translocation process. A new type of translocation distance between a pair of multi-chromosomal genomes is introduced; we examine the complexity of computing this distance in the case of uniform translocation, that is at each step the strings exchange prefixes of the same length. We present an exact polynomial algorithm based on the "greedy" strategy when the target set is a singleton while a 2-approximation algorithm is provided when considering arbitrary target sets. Some open problems are finally formulated.

1 Introduction

Genomes of arbitrarily complex organisms are organized into chromosomes that contain genes which may be considered as being arranged in linear order. Sequence alignment was actually the first step in molecular evolution studies but in many cases sequence alignment is quite unreliable which makes further evolutionary tree reconstruction almost impossible. It has been often found that the order of genes is much more conserved than the DNA base sequence. In the course of evolution, the genome of an organism mutates not only by processes at the level of individual genes (point mutations: insertion, deletion or substitution of individual bases) but also by some large-scale rearrangements in one evolutionary event. Recently much attention has been given by this phenomenon. One may argue that evolutionary and functional relationships between genes can be captured by taking into considerations only local mutations. However, the analysis of the genomes of some viruses (Epstein-Barr and Herpes simplex viruses, see for instance [7, 13]) have revealed that the evolution of these viruses involved a number of large-scale rearrangements in one evolutionary event. Furthermore, comparing plant and animal mitochondrial DNA, the point mutation is estimated to be 100 times slower in plant than in animal, many genes are nearly identical (more than 99% of them are identical) in related species [17]. These molecules which are almost identical in gene sequence differ fundamentally in gene order. At this level, point mutations are less meaningful compared to arrangements of gene fragments. See also [2, 7], for further discussions on this topic.

Chromosomal rearrangements include pericentric and paracentric inversions, intrachromosomal and interchromosomal transpositions, translocations, etc. For a description of these rearrangements, the reader is referred to [21]. Translocation is the biological process of exchanging material of the end of two chromosomes and could result in a different genotype[12]. Such non-local operations might permit an investigation of the evolutionary history for rather diverged organisms that cannot be identified from the study of point mutations alone [19, 20]. Recent developments in large-scale comparative genetic mapping seem to offer exciting prospects for understanding mammalian genome evolution [4]. A grammatical model based one these non-local operations can be found in [5] and [6]

The aim of this paper is to introduce a new type of translocation distance and to investigate the complexity of computing this distance for uniform (equal-length) translocation. This is a particular type of translocation which takes part just between chromosomes that exchange prefixes of equal length.

Prior work dealing with the combinatorial analysis of genome operations has focused on evolution distance in terms of inversions, transpositions or translocations for chromosomes formed from different markers which correspond to unique segments of DNA. From the formal point of view this means that in traditional genome rearrangement sorting problems the input data consists of permutations of n labels, but this approach cannot capture duplication events. In this paper, we considered chromosome as a nucleotide sequence unlike a unique marker sequence. Perhaps, this approach will not be practical for a while for lack of such data, but we looked to this problem from a mathematical point of view only. Kececioglu and Sankoff ([14, 15]) developed exact and approximation algorithms for two types of inversion distance which was shown to be *NP-complete* [3]. More recently [9] proposes a polynomial algorithm for signed inversion distance. Bafna and Pevzner reported approximation algorithms for transposition distance [1]. Two highly relevant papers which present the first polynomial algorithms for computing translocation distances are [10, 11]. Kececioglu and Ravi [16] discussed exact and approximation algorithms for distance involving translocations alone as well as together with inversions. Some applications of these results to biological data are now underway [2, 8]. The reader is also referred to [18] for a review of open combinatorial problems motivated by genome rearrangements.

Our work differs from the aforementioned approaches in many respects: the strings representing chromosomes may have multiple occurrences of the same symbol, they may have common symbols, the number of copies of all strings in the initial set is assumed arbitrarily large.

2 Preliminaries

Let V be a given alphabet (practically this alphabet is the DNA alphabet $\{A, T, C, G\}$); chromosomes may be viewed as strings over this alphabet. The set of all nonempty strings over V is denoted by V^+. For each string $x \in V^+$, whose length is denoted by $|x|$, $x[i, j]$ delivers the substring of x that starts at position i and ends at position j in x, $1 \leq i \leq j \leq |x|$. Conventionally, $x[i, j]$ is the empty string in all cases $j < i$. For two strings x, y over an alphabet V and two integers $1 \leq i < |x|, 1 \leq j < |y|$, we define the translocation operation

$$(x, y) \vdash_{(i,j)} (z_1, z_2) \text{ iff } x = tu, y = vw, z_1 = tw, z_2 = vu, \text{ and } |t| = i, |v| = j.$$

The pair of natural numbers (i, j) indicates the length of the prefixes they interchange with each other. When we are not interested about the length of these segments, we write simply $\vdash$. Let us note that, from a chromosome and its replica, say xyz, one may get two other chromosomes $xyyz$ and xz. It is worth mentioning here that this type of recombination is known as crossover between "sister" chromatids and it is the main way of producing tandem repeats or block deletions in chromosomes. We extend the translocation operation to a finite set of strings $A \subseteq V^+$ by $TO(A) = \bigcup_{x,y \in A}\{z, w | (x, y) \vdash (z, w)\}$.

Let A be a finite set of strings such that each string of A has arbitrarily many available copies. In other words, A may be viewed as the support of a multiset of strings each of them having arbitrarily many copies. Define iteratively

$$TO_0(A) = A, \quad TO_{k+1}(A) = TO_k(A) \cup TO(TO_k(A)), \quad TO_*(A) = \bigcup_{k \geq 0} TO_k(A).$$

A *translocation sequence* in $TO_*(A)$ is a sequence $S = s_1, s_2, \ldots, s_n$, where for each $1 \leq i \leq n$ $s_i = (x_i, y_i) \vdash_{(k_i, p_i)} (u_i, v_i)$, for some $x_i, y_i, u_i, v_i \in TO_*(A)$ and $1 \leq k_i < |x_i|$, $1 \leq p_i < |y_i|$. Given a translocation sequence S as above and $x \in TO_*(A)$ we define

$$P_i(S, x) = card\{j \leq i | x = x_j \text{ or } x = y_j\} + card\{j \leq i | x_j = y_j = x\},$$

$$F_i(S, x) = \begin{cases} card\{j \leq i | u_j = x_j \text{ or } v_j = y_j\} + card\{j \leq i | u_j = v_j = x\}, \text{if } x \notin A, \\ \infty, \text{ otherwise.} \end{cases}$$

The *length* of a translocation sequence $S = s_1, s_2, \ldots, s_n$ is denoted by $lg(S)$ and equals n. A translocation sequence S as above is *contiguous* iff the following two conditions are satisfied:

(i) $x_1, y_1 \in A$,

(ii) $F_{i-1}(S, x_i) > P_{i-1}(S, x_i)$, and $F_{i-1}(S, y_i) > P_{i-1}(S, y_i)$, for all $1 \leq i \leq n$.
The second condition is very natural if one considers that the copies of the two strings that exchange prefixes are not available anymore for further translocation steps; it claims that at each translocation step at least one copy for any of the two strings involved in this step is available. By CTS we mean a contiguous translocation sequence. Let B be a finite subset of $TO_*(A)$; a CTS S as above is B-producing if $F_n(S, z) > P_n(S, z)$ for all $z \in B$. In other words, S is B-producing if at the end of all translocation steps form S we have at least one copy

at each string in B. Roughly speaking, the *translocation distance* from A to B ($TD(A, B)$ shortly) is defined as the minimal number of steps strictly necessary to get B starting from A, providing that at each step just one translocation takes place. Formally,

$$TD(A, B) = \min\{lg(S) | S \text{ is a } B- \text{ producing } CTS \text{ in } TO_*(A)\}.$$

Sometimes we refer to B as the target set. In the sequel we are dealing with the complexity of computing the translocation distance defined above for the case of uniform translocation i.e. all strings exchange prefixes of equal length. We distinguish two cases depending on the cardinality of target sets: singleton target sets and arbitrary target sets.

3 Singleton Target Sets

As we said above, by uniform translocation we mean a special type of translocation so that prefixes which are to be exchanged are of the same length. Formally, the translocation operation $\vdash_{(i,j)}$ is said to be uniform iff $i = j$, so that we shall simply write $\vdash_i$.

In the case of uniform translocation with a singleton target set, without loss of generality we may assume that the initial set of strings contains only strings of the same length, that is the length of the target string. The simple proof of this statement is left to the reader. In conclusion, throughout this section the strings in the initial set and the target string will be all of the same length.

Suppose that $A = \{x_1, x_2, \ldots, x_n\}$ and let z be an arbitrary string of length k; the following measure will be very useful in the sequel:

$$MaxSubLen(A, z, p) = \max\{q | \exists\, 1 \le i \le n \text{ such that } x_i[p, p+q-1] = z[p, p+q-1]\}.$$

Note that with uniform translocation, a letter at position i in a string remains at position i after moving to another string. Assume that $z \in TO_*(A)$; define iteratively the set $H(A, z)$ of intervals of natural numbers as follows:

1. $H(A, z) = \{[1, MaxSubLen(A, z, 1)]\}$;
2. Take the interval $[i, j]$ having the largest j; if $j = k$, then stop, otherwise put into $H(A, z)$ the new interval $[j + 1, j + MaxSubLen(A, z, j + 1)]$.

Note that we allow intervals of the form $[i, i]$ for some i to be in $H(A, z)$; moreover, for each $1 \le i \le k$ there are $1 \le p \le q \le k$ (possibly the same) such that $i \in [p, q] \in H(A, z)$.

Lemma 1 *Let S be a z-producing CTS in $TO_*(A)$. Then,*

$$lg(S) \ge card(H(A, z)) - 1.$$

Proof. We prove this assertion by induction on the length k of z. For $k = 1$ the assertion is trivially true because z must be in A, hence $H(A, z)$ contains just one element. Assume that the assertion is true for any string shorter than k. Let us consider a CTS $S = s_1, s_2, \ldots, s_q$ in $TO_*(A)$ producing z. Moreover, we may assume that $s_i = (x_i, y_i) \vdash_{p_i} (u_i, v_i)$, $1 \le i \le q$, and z has been obtained in S at the last step, that is either $u_q = z$ or $v_q = z$. Let

$A' = \{x[MaxSubLen(A,z,1)+1,k] | x \in A\}$, $z' = z[MaxSubLen(A,z,1)+1,k]$. For simplicity denote $r = MaxSubLen(A,z,1)$. Clearly, $H(A',z') = \{[i-r, j-r] | [i,j] \in H(A,z) \setminus \{[1,r]\}\}$, hence $card(H(A',z')) = card(H(A,z)) - 1$. Starting from S we construct a CTS in $TO_*(A')$, producing z' $S' = s'_1, s'_2, \dots s'_m$ in the way indicated by the following procedure:

```
Procedure Construct_CTS(S,r);
begin
$m := 0$;
for $i := 1$ to $q$ begin
     if  $(p_i > r)$  then
        $m := m+1$;  $s'_m = (x_i[r+1,k], y_i[r+1,k]) \vdash_{p_i - r} (u_i[r+1,k], v_i[r+1,k])$;
     endif;
endfor;
end.
```

Claim 1: *S' is a CTS.*
Proof of the claim. Firstly, we note that for each $1 \le i \le q$ so that $p_i \le r$, the relations $u_i[r+1,k] = y_i[r+1,k]$ and $v_i[r+1,k] = x_i[r+1,k]$ hold. Assume that $p_{i_1}, p_{i_2}, \dots, p_{i_m}$ are all integers from $\{p_1, p_2, \dots, p_q\}$ bigger than r. Because all $p_1, p_2, \dots, p_{i_1 - 1}$ equal at most r, it follows that both $x_{i_1}[r+1,k], y_{i_1}[r+1,k]$ are in A'. Now, it suffices to prove that for a given $2 \le j \le m$, the relations
$F_{j-1}(S', x_{i_j}[r+1,k]) > P_{j-1}(S', x_{i_j}[r+1,k])$,
$F_{j-1}(S', y_{i_j}[r+1,k]) > P_{j-1}(S', y_{i_j}[r+1,k])$,
hold. We shall prove the first relation only. It is not hard to see that

$$F_{j-1}(S', x_{i_j}[r+1,k]) = \sum_{x[r+1,k] = x_{i_j}[r+1,k]} F_{i_j - 1}(S,x) - card(X) - card(Y),$$

$$P_{j-1}(S', x_{i_j}[r+1,k]) = \sum_{x[r+1,k] = x_{i_j}[r+1,k]} P_{i_j - 1}(S,x) - card(X) - card(Y),$$

where

$$X = \{t \le i_j - 1 | p_t \le r,\ u_t[r+1,k] = v_t[r+1,k] = x_{i_j}[r+1,k]\},$$
$$Y = \{t \le i_j - 1 | p_t \le r,\ u_t[r+1,k] = x_{i_j}[r+1,k] \text{ or } v_t[r+1,k] = x_{i_j}[r+1,k]\}.$$

In conclusion, as S is a CTS, it follows that $F_{j-1}(S', x_{i_j}[r+1,k]) > P_{j-1}(S', x_{i_j}[r+1,k])$, and the proof of the claim is complete.

Claim 2: *S' is z'-producing.*
Proof of the claim. More generally, we shall prove by induction on i that S' is producing $u_i[r+1,k]$ and $v_i[r+1,k]$ for all $1 \le i \le q$. The assertion is trivially true for $i = 1$. Assume that the assertion is true for all $t \le i$; we shall prove it for $i+1$. If $u_{i+1}[r+1,k]$ is in A' or $p_{i+1} > r$, we are done. If $p_{i+1} \le r$, then $u_{i+1}[r+1,k] = y_{i+1}[r+1,k]$; for $y_{i+1} \notin A$ we have $F_i(S, y_{i+1}) > 0$, hence there exists $t \le i$ such that $u_t = y_{i+1}$ or $v_t = y_{i+1}$. By the induction hypothesis, S' is producing $u_t[r+1,k]$ which concludes the proof of the second claim.

But there exists at least one i such that $p_i \le r$, it follows that $m \le q-1$. By the induction hypothesis, $m \ge card(H(A',z')) - 1$, and the proof is complete. □

The next result is a direct consequence of this lemma.

Theorem 1 *Let z be a string of length k and A be a set of cardinality n. There is an exact algorithm that computes TD(A,z) in O(kn) time and O(kn) space.*

Proof. The following algorithm indicates how to construct a CTS $S = s_1, s_2 \dots,$ s_m in $TO_*(A)$ producing z, when $z \notin A$, whose length is exactly $card(H(A,z)) - 1$.

```
Procedure Uniform_translocation_CTS_construction(A,z);
begin
$p := MaxSubLen(A, z, 1)$; let $x$ be a string in $A$ with $x[1,p] = z[1,p]$;
$m := 0$;
while  $p < k$  begin
   $r := MaxSubLen(A, z, p+1)$;
     if  $r = 0$  then THE STRING $z$ CANNOT BE OBTAINED FROM $A$;
                 stop
        else
          let $y$ be a string in $A$ with $y[p+1, p+r] = z[p+1, p+r]$;
          $m := m + 1$
          $s_m = (x,y) \vdash_p (u,v)\}$;
          $p := p + r$;
          $x := u$;
     endif
endwhile;
end.
```

It is easy to see that if the algorithm successfully terminates, then either u or v is exactly z, and the length of the CTS determined by the algorithm is exactly $card(H(A,z)) - 1$. By the previous lemma, this in an optimal value. As one can easily see the time complexity of this algorithm is given by the complexity of computing the values $MaxSubLen(A, z, p)$, which is $O(kn)$. Obviously, it requires $O(kn)$ memory. □

4 Arbitrary Target Sets

We shall try to adapt the techniques used in the previous section for arbitrary target sets, too. Let A be a finite set of strings and $z \in TO_*(A)$; denote by

$$\begin{aligned}
MaxPrefLen(A,z) &= \begin{cases} |z|, \text{ iff } z \in A, \\ \max(\{q \mid q < |z|, \text{ there exists } x \in A, |x| > q, \\ \qquad \text{so that } x[1,q] = z[1,q]\} \cup \{0\}), \end{cases} \\
MaxSufLen(A,z) &= \max(\{q \mid \text{ there exists } x \in A, |x| \geq |z|, \\
&\qquad \text{so that } x[|x|-q+1, |x|] = z[|z|-q+1, |z|]\} \cup \{0\}), \\
ArbMaxSubLen(A,z,p) &= \max(\{q \mid \text{ there exists } x \in A \text{ and } |x| \geq p+q \\
&\qquad \text{such that } x[p, p+q-1] = z[p, p+q-1]\} \cup \{0\}).
\end{aligned}$$

We define iteratively the set $ArbH(A,z)$ of intervals of natural numbers as follows, provided that all parameters defined above are nonzero:

1. $ArbH(A,z) = \{[1, MaxPrefLen(A,z)]\}$;
2. Take the interval $[i,j]$ having the largest j; if $j = |z|$, then stop. If $j < |z| - MaxSufLen(A,z)$, then put the new interval $[j+1, j+ArbMaxSubLen(A,z,j+1)]$ into $ArbH(A,z)$; otherwise put $[j+1, |z|]$ into $ArbH(A,z)$.

Theorem 2 *1. Let A be a finite set of strings and B be a finite subset of $TO_*(A)$. Then* $\frac{\sum_{z\in B}(card(ArbH(A,z))-1)}{2} \leq TD(A,B) \leq \sum_{z\in B}(card(ArbH(A,z))-1)$.
2. There exist A and $B \subseteq TO_*(A)$ such that $TD(A,B) = \frac{\sum_{z\in B}(card(ArbH(A,z))-1)}{2}$.
3. There exist A and $B \subseteq TO_*(A)$ such that $TD(A,B) = \sum_{z\in B}(card(ArbH(A,z))-1)$.

Proof. 1. We shall prove the first assertion by induction on the length of the longest string in B, say k. The non-trivial relation is

$$\frac{\sum_{z\in B}(card(ArbH(A,z))-1)}{2} \leq TD(A,B). \qquad (*)$$

If $k=1$, then $B \subseteq A$, hence $card(H(A,z))=1$ for all $z \in B$, therefore the relation $(*)$ is satisfied. Assume that the relation $(*)$ holds for any two finite sets X and Y, $Y \subseteq TO_*(X)$, all strings in Y being shorter than k. Assume that $B \setminus A = \{z_1, z_2, \dots, z_m\}$ and let $S = s_1, s_2, \dots, s_q$, $s_i = (x_i, y_i) \vdash_{p_i} (u_i, v_i)$, $1 \leq i \leq q$, be a $B \setminus A$-producing CTS in $TO_*(A)$. Note that at least one string in $B \setminus A$ should exist, otherwise the relation $(*)$ being trivially fulfilled.

Consider m new symbols $a_1, a_2, \dots, a_m$ and construct the sets:
$A' = \{x[1,r]a_i x[r+2,|x|] | x \in A, 1 \leq i \leq m\}$, $B' = \{z_i[1,r]a_i z_i[r+2,|z_i|] | 1 \leq i \leq m\}$,, where $r = \min\{p_i | 1 \leq i \leq q\}$. One can construct a B'-producing CTS in $TO_*(A')$ of the same length of S, say S' by applying the next procedure.

```
Procedure Convert(S);
begin
for $j := 1$ to $m$ begin
   $z := z_j$; $t := q$;
    while $z \notin A$ begin
      find the maximal $l \leq t$ such that $u_l = z$ or $v_l = z$;
      $t := l - 1$;
       if $u_l = z$ then replace $u_l$ by $u_l[1,r]a_j u_l[r+2,|u_l|]$;
          if $p_l > r$ then $z := x_l$;
            replace $x_l$ by $x_l[1,r]a_j x_l[r+2,|x_l|]$;
          else $z := y_l$;
            replace $y_l$ by $y_l[1,r]a_j y_l[r+2,|y_l|]$;
          endif;
       else replace $v_l$ by $v_l[1,r]a_j v_l[r+2,|v_l|]$;
          if $p_l \leq r$ then $z := x_l$;
            replace $x_l$ by $x_l[1,r]a_j x_l[r+2,|x_l|]$;
          else $z := y_l$;
            replace $y_l$ by $y_l[1,r]a_j y_l[r+2,|y_l|]$;
          endif;
       endif;
    endwhile;
   replace $z$ by $z[1,r]a_j z[r+2,|z|]$;
endfor;
replace the symbol on the position $r+1$ in all strings in $S$ that
have not been replaced so far by $a_\bullet$;
end.
```

Now we apply the procedure in the proof of Lemma 1 to the sequence S' for r previously defined. The obtained sequence S'' is a B''-producing CTS in $TO_*(A'')$, where

$A'' = \{a_i x[r+2, |x|] | x \in A, 1 \leq i \leq m\}$, $B'' = \{a_i z_i[r+2, |z_i|] | 1 \leq i \leq m\}$, due to the two claims from the proof of Lemma 2.
For each $1 \leq i \leq m$ $card(ArbH(A'', a_i z_i[r+2, |z_i|]))$ is either $card(ArbH(A, z_i))$ or $card(ArbH(A, z_i)) - 1$. Moreover, for each i such that $card(ArbH(A'', a_i z_i[r+2, |z_i|])) = card(ArbH(A, z_i)) - 1$ there exist at least one step in S' where the strings exchange prefixes of length at most r. It follows that $lg(S'') \leq lg(S') - \lceil t/2 \rceil$, where $t = card(\{i | card(ArbH(A'', a_i z_i[r+2, |z_i|])) = card(ArbH(A, z_i)) - 1\})$. Consequently,

$$lg(S) = lg(S') \geq lg(S'') + \lceil t/2 \rceil \geq \frac{\sum_1^m (card(ArbH(A'', a_i z_i[r+2, |z_i|])) - 1)}{2} + \lceil t/2 \rceil \geq \frac{\sum_1^m (Arbcard(H(A, z_i)) - 1)}{2}.$$

The reader may easily find sets A and B fulfilling the last two assertions. □

An α-approximation algorithm for a minimization problem is a polynomial algorithm that delivers a solution whose value is at most α times the minimum. From the previous theorem we have:

Theorem 3 *There is a 2-approximation algorithm for computing the translocation distance from two sets of strings.*

Proof. Obviously, an algorithm that computes $\sum_{z \in B}(card(ArbH(A, z)) - 1)$ requires $O(n|B|)$, where $n = card(A)$ and $|B|$ is the sum of the lengths of all strings in B. □

5 Conclusion

We have introduced a new translocation distance between two finite sets of strings and proposed an algorithm, based on the "greedy" strategy, for computing this distance when the target set is a singleton. This is a constraint that does not exclude many interesting biological applications. All results presented here are valid for a particular type of translocation, namely the uniform translocation where the strings exchange with each other prefixes of the same length. This restriction might be considered rather far from reality, but, even so, the problem of finding a polynomial algorithm to compute the translocation between two finite sets remains open. The next step is naturally to consider the case of arbitrary translocation which is a more appropriate abstraction of the practical problem. We hope to return to this topic in a forthcoming paper.

References

1. V. Bafna, P.A. Pevzner, Sorting by transpositions. In *Proceedings of the 6th ACM-SIAM Symposium on Discrete Algorithms*, 1995.
2. V. Bafna, P.A. Pevzner, Sorting by reversals: genome rearrangements in plant organelles and evolutionary history of X chromosome, *Mol. Biol. Evol.* 12 (1995), 239–246.

3. A. Caprara, Sorting by reversal is difficult. In *Proc. of the First Annual International Conference on Computational Molecular Biology (RECOMB 97)*, ACM New York, 1997, 75–83.
4. N. G. Copeland et al. A genetic linkage map of the mouse: Current applications and future prospects. *Science*, 262(1993), 57–65.
5. J. Dassow, V. Mitrana, A. Salomaa, Context-free evolutionary grammars and the language of nucleic acids. *BioSystems*, 4(1997) 169–177.
6. J. Dassow, V. Mitrana, Operations and grammars suggested by the genome evolution, *Theoretical Computer Science*, 270, 1-2 (2002), 701–738.
7. D.J. McGeoch, Molecular evolution of large DNA viruses of eukaryotes. *Seminars in Virology* 3 (1992) 399–408.
8. S. Hannenhalli et al. Algorithms for genome rearrangements: herpesvirus evolution as a test case. In *Proc. of the 3rd International Conference on Bioinformatics and Complex Genome Analysis*, 1994.
9. S. Hannehalli, P. A. Pevzner, Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversal, *J. of the ACM* 46, 1 (1999) 1–27.
10. S. Hannehalli, P. A. Pevzner, Transforming men into mice (polynomial algorithm for genomic distance problem). In *Proc. of the IEEE 36th Annual Symposium on Foundation of Computer Science*, 1995, 581–592.
11. S. Hannehalli, Polynomial algorithm for computing translocation distance between genomes. In *Combinatorial Pattern Matching, Proc. of the 6th Annual Symposium (CPM'95)*, LNCS, Springer-Verlag, 162–176.
12. D. L. Hartl, D. Freifelder, L. A. Snyder, *Basic Genetics*, Jones and Bartlett Publ., Boston, Portola Valley, 1988.
13. S. Karlin, E. S. Mocarski, G. A. Schachtel. Molecular evolution of herpesviruses: genomic and protein comparisons. *J. of Virology*, 68(1994),1886–1902.
14. J. Kececioglu, D. Sankoff, Exact and approximation algorithms for sorting by reversals, with application to genome rearrangements. In *Proc. of the 4th Symposium on Combinatorial Pattern Matching*, Springer-Verlag, LNCS 684, 1993, 87–105.
15. J. Kececioglu, D. Sankoff, Efficient bounds for oriented chromosome-inversion distance. In *Proc. of the 5th Symposium on Combinatorial Pattern Matching*, Springer-Verlag, LNCS 807, 1994, 307–325.
16. J. Kececioglu, R. Ravi, Of mice and men: Algorithms for evolutionary distances between genomes with translocation. In *Proceedings of the 6th ACM-SIAM Symposium on Discrete Algorithms*, 1995, 604–613.
17. J.D. Palmer, L.A. Herbon, Plant mitochondrial DNA evolves rapidly in structure, but slowly in sequence. *Journal of Molecular Evolution*, 27(1988), 87–97.
18. P. A. Pevzner, M. S. Waterman, Open combinatorial problems in computational molecular biology. In *Proc. of the 3rd Israel Symposium on Theory of Computing and Systems*, IEEE Computer Computer Society Press, Los Alamitos, California, 1995, 158–163.
19. D. Sankoff, Edit distance for genome comparison based on non-local operations. In *Proc. of the 3rd Symposium on Combinatorial Pattern Matching*, Springer-Verlag, LNCS 644, 121–135, 1992.
20. D. Sankoff et al. Gene order comparisons for phylogenetic inference: Evolution of the mitochondrial genome, *Proc. Natl. Acad. Sci. USA*, 89(1992),6575–6579.
21. E. Therman, M. Susman, *Human Chromosomes, Structure, Behavior, and Effects.* Springer-Verlag, 1993.

Understanding Human-Computer Interactions in Map Revision*

Jun Zhou, Walter F. Bischof, and Terry Caelli

Department of Computing Science, University of Alberta
Edmonton, Alberta, Canada T6G 2E8
{jzhou,wfb,tcaelli}@cs.ualberta.ca

Abstract. It is difficult to track, parse and model human-computer interactions during editing and revising of documents, but it is necessary if we are to develop automated technologies that will aid or replace humans. This paper introduces a system for accessing and recording a stream of events related to human actions in a real-time cartographic map revision system. The recorded events are parsed into a sequence of meaningful user actions and an action representation in XML format is generated. We also report results of experiments on predicting user actions such as view changes, edits, road tracking/production using hidden Markov models.

1 Introduction

Analyzing and understanding human behaviour in software systems has gained increasing interest in Artificial Intelligence, Pattern Recognition, Human - Computer Interaction, and Cognitive Psychology. Surprisingly, very few research results have been utilized in real-world applications [1, 2]. A typical example is that of semi-automatic road tracking in images where the human is only used to initialize automated processes or as an editor at the end, with only a few or no human-computer interactions along the way [3]. Because of the deficiencies of computer vision algorithms, the performance of road tracking systems is very limited, usually not reliable, and less efficient than humans performing the task alone.

We have adopted a different paradigm: we study and model human performance on such tasks, identify key actions and difficulties, and then develop algorithms that improve the efficiency and accuracy of human operators. To achieve this, we need to track, parse and model the user actions in real world tasks, something that is rarely done in the document processing area [4, 5]. In this paper, we introduce a real world application for computer aided map revision. The software environment and the development tools in this project make it easy to keep track of both the temporal and spatial information of the user actions

* This project is funded by a grant from the Natural Sciences and Engineering Research Council of Canada.

in system-level events. These events are parsed into time-segmented, meaningful and complete user action data that are stored in XML format. These data can be used to model user behavior patterns, and to support and automate the map revision process. In the next section, we briefly introduce the Raster Graph Revision (RGR) system in the United States Geological Survey (USGS), the class of maps of interest to this work. In Sections 3 and 4, we present a system for tracking and parsing user actions. In Section 5 we present and discuss the experimental results in modeling human viewing behaviour and evaluate user performance using hidden Markov models (HMMs).

2 RGR System

One of the main paper products of the USGS topographic maps for the USA is 7.5-minute quadrangle topographic map, which is the only uniform map series that covers the entire area of the continental United States in considerable detail [6]. This map series consists of about 53,000 map sheets.

Current USGS maps are printed on white paper with six colors: black, red, brown, green, blue and purple, one for each feature. The RGR system uses existing film separates as the primary input and creates new film separates as the primary output. Cronapaque positives are produced photographically from the map separates and are scanned at 1000 dpi as raster images. The images, in addition to the digital orthophoto quads (DOQs) of the area to be mapped, are registered to the control file and displayed simultaneously on a computer screen as the source for revision. DOQs are orthogonally rectified images from aerial photos taken at height of $20,000$ feet with approximate scale of $1 : 40,000$. DOQ can distinguish ground objects of 1 meter, which is enough for ground object detection. Cartographers then make a visual comparison of the raster image and DOQs. When a discrepancy is found between a feature on the raster image and the DOQ, cartographers can add to, delete from, or modify the raster image to match the DOQ. Figure 1 shows the environment of the RGR system.

The standard CAD tool for RGR systems is a Bentley Microstation. Bentley I/RAS B is used to display and manipulate the scanned map layers, Z/I Imaging

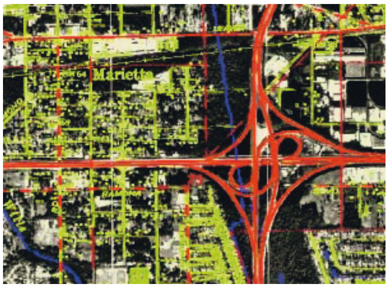


Fig. 1. Map revision environment. Here previous map layers are aligned with current digital image data.

I/RAS C is used to display the DOQs, USGS RGR software provides CAD tools to draw, delete and modify specialized graphic symbols on maps, and MVES converts vectors and points to a symbolized raster format.

3 Tracking User Actions

In Microstation, interaction with the system is by keyboard or mouse. A simple drawing operation may be achieved by clicking a tool icon on the tool bar or by inputting a "key-in" command. To facilitate map revision, RGR uses a set of tools for cartographic symbols, each of which, along with mouse actions, encompasses a sequence of system events (key-ins). Each key-in is considered as an event. Events from both inside or outside Microstation are processed by an input handler and are sent to an input queue where a task ID is assigned to each event.

With the imbedded Microstation Development Language environment, we can keep track of the states of the event queue and extract detailed information on each event, as described in Table 1. These time-stamped sequences of system-level events contain inter-action and intra-action information. The task, then, is to parse these sequences into meaningful higher-level user action sequences.

Table 1. Data structure for system-level event.

event ID	ID of the event
event name	the key-in command
event type	Is it a keyin, coordinate, or reset?
event time	the time when the event is captured
event source	where does the event come from
x coordinate	x coordinate of the mouse clicking
y coordinate	y coordinate of the mouse clicking

4 Analyzing and Parsing User Actions

Altogether there are 278 tools in the RGR software, each corresponding to a human action. This number could be increased when new standards are adopted in the USGS. Analysis of all these tools is not necessary. First, some tools are used for features that rarely appear, or need not be revised in most cases. Second, some tools relate to registration of the scans and DOQs, environment setting, file input/output, and map plotting, and are not involved in the feature collection process. Third, we only process actions related to feature collection, at least at this initial stage of the project. As a result the number of tools is reduced to 144, each being composed of a sequence of events. A complete action may contain a tool selection, a sequence of coordinate clicks, and a reset operation. View changes may occur before and after each coordinate click.

The actions can be grouped into 17 groups, each action group being defined by the number of permissible coordinates and occurrence of reset operations. Groups G0 to G11 contain system setting and drawing actions (e.g. draw a class-1 road or undo a previous action), groups V1 to V2 contain viewing actions (e.g. zoom-in or pan-view), group R contains the "reset" action, and group CO correspond to a coordinate click not involved in any action.

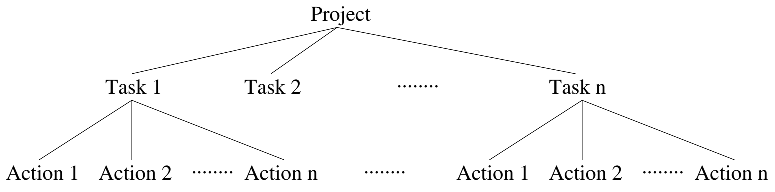


Fig. 2. Hierarchy of the action database.

A sequence of system-level events is parsed into a tree structure as shown in Figure 2. The root of the tree is a project, which is defined as the revision of a map. A task is defined as the revision of specific ground object (e.g. a road, a block of buildings, or a lake). Semantic information is contained at both project and task level and can be tagged by human input. Finally, the user actions are stored in a XML format database.

5 Issues in Computer Aided Map Revision

The purpose of computer aided map revision is to improve both the speed and accuracy of the revision process. The solution to the first task is to reduce human involvement in the drawing task by using feature tracking based on computer vision. Until now, all research efforts fall in this area. However, the efficiency and accuracy of the computer vision algorithms are not necessarily consistent with human performance, suggesting that humans should be part of the feature tracking process. On the other hand, humans too are not always accurate. Consequently we envisage a tightly coupled, real-time, error-correcting interaction between human and machine in order to make map revision more efficient. To implement this we need to better understand these interactions.

5.1 Predicting Human Gaze with Hidden Markov Models (HMM)

One of the ways of reducing the human workload in map revision is to reduce drawing actions and viewing changes in the map revision process. This can be achieved by predicting when humans are likely to change viewing. To do so one can use Markov or Hidden Markov Models [7, 1].

Along with the parsed human action sequences, we can also record the sequence of viewing locations ("gaze"). We have performed several experiments with HMMs in order to study such viewing patterns. The states in the HMM were defined as groups of actions where the groups were defined syntactically and semantically. The syntactic groups were the 17 groups (17 states) defined in Section 4, and the semantic groups were obtained by clustering actions based on their functions as used in the RGR system, such as a group of actions for drawing transportation symbols or water body symbols. In this case, the actions were divided into 6 groups (6 states). The observations were calculated from the movements of the mouse. These movements were classified into either 9 (45^o step) or 17 (22.5^o step) directions, with one direction in each group being used for the no-movement case.

Two participants were required to perform three drawing tasks twice, one for training and another for testing. The tasks involved the modification of roads, buildings, water bodies, etc. The average time taken to finish each task was 46 minutes. Altogether 34644 system-level events with time-stamps were captured and 9025 of these events were coordinate moves (changes in gaze). These events were parsed into 2157 actions. Each task sequence contained 180 actions on average. These task sequences were further cut into shorter sequences with 2 actions each. Finally, we obtained 560 training sequences and 540 test sequences sampled at 1-second intervals, with an average length of 27 observations.

With the given number of states and observations, the degrees to which each observation sequence could be predicted from the trained and untrained models was determined by the degree to which the HMM could reproduce the movements over a number of Monte Carlo trials. The results of these experiments are shown in Table 2.

Table 2. The results of predicting next viewing change as a function of different number of states (S) and observations (O) for two training sets and two test sets. Values correspond to probabilities of correctly predicting the observation sequences given the model and the Viterbi MAP solutions. The right column shows chance performance levels. Average length of the observation sequences was 27.

	Training1	Training2	Testing1	Testing2	Chance
17S 17O	0.72	0.63	0.61	0.63	0.06
6S 9O	0.72	0.63	0.61	0.63	0.11
2S 9O	0.79	0.69	0.67	0.69	0.11

The numbers in Table 2 refer to the average probabilities of correctly predicting the observation sequences given the model and the Viterbi MAP solution over 100 Monte Carlo trials. This reduces to sampling from the state-dependent observation vectors, given the Viterbi-predicted state for each observation value. Three models were tested. Each model was estimated by first obtaining initial estimates from the training sequences as all observations were automatically labeled (states known). The model was then updated using the Baum-Welch algorithm. We also tested a two-state model in which the states were either system setting actions or drawing actions. The results show that in each state-observation combination, probabilities of correctly predicting the observation sequences, given the models and Viterbi solutions, were significantly above chance (last column of Table 2). These results are quite informative given the lengths of the sequences (27 in average) and prediction rates significantly above chance.

In real applications, however, predictions of viewing changes need to be much more accurate. Further, complete viewing prediction requires not only a direction, but also an exact location on the map. This suggests that a simple analysis of the user actions is not sufficient. The better prediction model should involve the recognition of features from image and maps, as described below.

Current research results in semi-automatic or fully automatic road tracking systems can be combined into the above model in order to provide support for complete viewing prediction and automatic tracking of roads. In automatic road tracking systems, the road seeds are found by the system without need to preselect points along the road [8]. It is normally difficult to extend these automatic methods robustly and efficiently to very large images, such as the DOQs in this project [9]. Our viewing prediction result provide the possibility to reduce the searching area in a large image to relatively small areas in the predicted directions so as to generate a semi-automatic road tracker. How large the predicted area is can be decided by calculating the average length of the human viewing change steps. In the next subsection, we introduce a simple semi-automatic road tracking system on the assumption that target small area image has been extracted. Then we compare the performance of human and computer in road tracking.

5.2 Comparing the Performance of Human and Computer Vision-Based Road Tracking

In semi-automated road tracking systems, a human operator typically provides initial parameters, including a starting point, a direction and a road width [8]. Baumgartner implemented a system with more extensive human computer interactions [10]. When the system detects a possible failure of the tracking module, it stops to allow the operator to select from a list of continuation options. In such semi-automatic systems, it is assumed that the human can perform the task correctly and precisely. Further, what the computer determines as "incorrect" is also unclear and subject to error. This is not necessarily the case. To analyze this, we developed a simple road tracker and compared its performance with that of humans.

A road segment is determined by two consecutive coordinates (mouse clicks), the axis joining the coordinates defines the human detected road. To compare this axis with that detected by computer, we cropped the neighborhood image of this road segment from the DOQ to reduce the search area (the size of a DOQ is more than 2MB) and then performed Canny edge detection [11]. As a result, points at maxima of gradient magnitude in the gradient direction are marked as edges, which may include both road and non-road edges, such as contour of cars. This edge operator is used because both straight and curved roadsides can be detected. Abrupt greylevel changes caused by surface material changes can also be detected and do not affect the extraction of candidate road edge points. Figure 3(a) and 3(b) show an example of the cropped image and the image after Canny edge detection.

For each point on the axis defined by the human operator, we constructed a line perpendicular to the axis and determined the intersection points to the

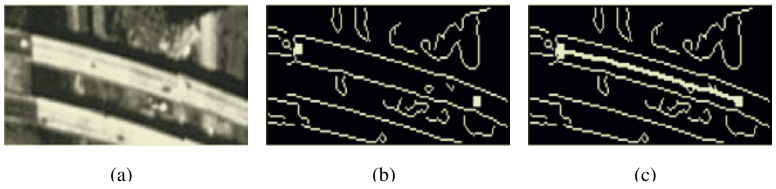


Fig. 3. (a) Cropped image from DOQ. (b) Human input (white blocks) and edges detected by Canny edge operator. (c) Axis detected by computer.

edges. These points were the candidates of roadside. To reduce the disturbances on the road, like cars and shadows, points on the edges with short length were removed from the candidate list. If two or more candidates were found, the road width limits defined by USGS was used as the upper and lower bounds of the distances between roadsides [12]. The two closest intersections to the axis detected by human, which also met the width limit, were selected as the roadsides corresponding to the axis point. Connection of the mid-points of these intersection pairs formed the axis detected by the computer. Figure 3(c) shows the result of Canny-edge axis detection of the image in Figure 3(a).

Two kinds of errors occurred during Canny-edge axis detection. One kind was caused by deficiencies in the Canny edge detection. When the road and the background have similar greylevels, the Canny operator failed to mark the road edges. To reduce this error, the roadsides were predicted by fitting a parabola to the most recent road points, as described in [13]. This error can also be avoided by detecting weak edges from gradient images using distance limits [14]. Another type of error comes from disturbances on the road that have not been removed. Some of them are connected with the roadsides, which made the road to be thinner than the lower bound of the road width limit. Consequently, correct road side candidates could not be selected by the system. This kind of error could be avoided by jumping to the next axis point along the road.

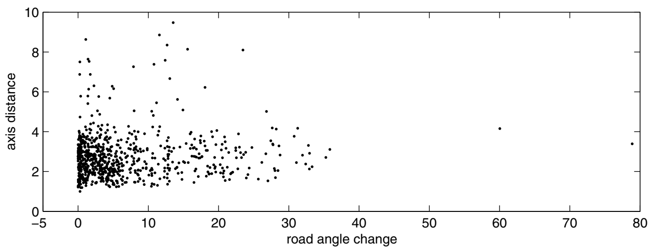


Fig. 4. Distances of road axis versus road angle changes in the training sets.

To compare human and computer performance, the mean distance of the axis detected by computer and by human was calculated. Figure 4 shows the distribution of the distances versus road angle changes on two training sets described in Section 5.1. The road angle change was obtained from the angle between two consecutive axes detected by human. We expected that the distances between the axis would increase along with the road angle changes, but the results show that there is no relationship between the two. In most cases, the distances between the axis were less than 4 pixels despite the change of road angles. It is within the tolerance of positional accuracy defined by USGS (maximum 6 pixels, average 3 pixels) [15]. In the cases where the distances were too large to be acceptable, analysis shows that although most errors were caused by the deficiency of the Canny-edge axis detection, some are caused by the inaccuracy of human in georeferencing an ground object in DOQ with a map feature. The human input may lie much closer to one roadside, or even falls outside of the road. In these cases, the road tracking system may not select the correct roadside candidates. An example of this deviation is shown in Figure 5.

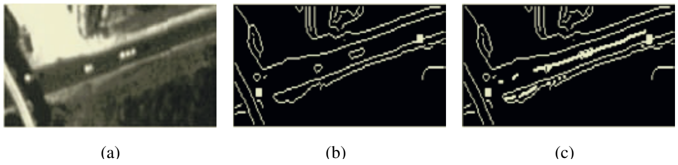


Fig. 5. (a) Cropped image from DOQ. (b) Human input and edges detected by Canny edge operator. (c) The white blocks at the end of the road are the input from human. The small white dots show the axis detected by computer. Because the human input road end points are shifted from the true centers, the computer can not detect all the axis points correctly.

6 Conclusion

This paper has introduced a real world environment for map revision. The user actions in the map revision are tracked and recorded as a time-stamped sequence of events at the system level. These events are parsed into an event sequence that is represented in XML format. This is - as far as we know - the first open database on user behavior in real world applications involving document processing, feature tracking, or pattern recognition.

Two experiments based on the human data are reported on predicting human viewing changes and comparison of the performances of human and computer in road tracking. It is clear from these studies that in order to build a human-machine system capable of improving human performance, we need a more tightly coupled interaction between human and machine.

References

1. Horvitz, E., Kadie, C., Paek, T., Hovel, D.: Models of attention in computing and communication: From principles to applications. Communications of the ACM **46** (2003) 52–59

2. Encarnacao, M., Stoev, S.: An application independent intelligent user support system exploiting action-sequence based user modeling. In: Proceedings of the Seventh International Conference on User Modeling, Banff, Canada (1999) 245–254
3. Baltsavias, E., Hahn, M.: Integrating spatial information and image analysis one plus one makes ten. In: International Archives of Photogrametry and Remote Sensing. Volume 33. (2000) 63–74
4. Sandanayake, P.T., Cook, D.J.: ONASI: Online agent modeling using a scalable Markov model. International Journal of Pattern Recognition and Artificial Intelligence **17** (2003) 757–779
5. Mayfield, J.: Controlling inference in plan recognition. User Modeling and User-Adapted Interaction **2** (1992) 83–115
6. Moore, L.: The USGS Geological Survey's revision program for 7.5-minute topographic maps. http://mcmcweb.er.usgs.gov/topomaps/revision.html (2000)
7. Hacisalihzade, S.S., Stark, L.W., Allen, J.S.: Visual perception and sequences of eye movement fixation: a stochastic modeling approach. IEEE Transactions on Systems, Man, and Cybernetics **22** (1992) 474–481
8. Fortier, M., Ziou, D., Armenakis, C., Wang, S.: Survey of work on road extraction in aerial and satellite images. Technical report, Université de Sherbrooke (2000)
9. Geman, D., Jedynak, B.: An active testing model for tracking roads in satellite images. IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1996) 1–14
10. Baumgartner, A., Hinz, S., Wiedemann, C.: Efficient methods and interfaces for road tracking. In: PCV02. (2002) B: 28
11. Canny, J.: A computational approach to edge detection. IEEE Transaction on Pattern Analysis and Machine Intelligence **8** (1986) 679–698
12. U.S. Geological Survey, U.S. Department of the Interior: Standards for 1:24000-Scale Digital Line Graphs and Quadrangle Maps. (1996)
13. McKeown, D.M., Denlinger, J.: Cooperative methods for road tracking in aerial imagery. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Ann Arbor, Michigan (1988) 662–672
14. Wiedemann, C., Mayer, H.: Automatic verification of roads in digital images using profiles. In: DAGM-Symposium. (1996) 609–618
15. U.S. Geological Survey, U.S. Department of the Interior: Standards for Raster Feature Separates Version 1.0. (2002)

Multiscale Curvature Assessment of Postural Deviations

Cornélia J.P. Passarinho[1], Fátima N.S. Medeiros[1], Jilseph Lopes Silva[1], Luís Henrique Cintra[2], and Rafael B. Moreira[1]

[1] Federal University of Ceara
Campus do PICI, Bloco 705, C.P. 6007
60455-760 - Fortaleza, CE, Brazil
{cjanayna,fsombra,jilseph,rafaelbarbosa}@deti.ufc.br
http://www.gpi.deti.ufc.br/index.html

[2] Somma Physiotherapy Center
Rua Júlio Siqueira, 989, Dionísio Torres, CEP 60130-090
Fortaleza-Ce, Brazil
Fax +55 85 272 8678
lhcintra@secrel.com.br

Abstract. This paper presents and discusses an effective approach to assessing human postural deviations based on curvature measurements. Multiscale curvature values are calculated from both parametric contours extracted from the silhouette image in sagittal plane and from the medial axis. The algorithms were applied to digital images of patients who have been submitted to Global Postural Reeducation (GPR) physiotherapy treatment. Features such as area, perimeter, center of mass, spreading, thinness degree and angulations are also obtained for similarity shape analysis between images taken before and after the GPR treatment. The medial axis is evaluated to investigate how it can be used to infer the spine alignment of patients with postural deviations prior to taking exams such as x-ray or tomography.

1 Introduction

Unlike traditional physiotherapy approach, Global Postural Reeducation (GPR) physiotherapy treatment assesses human postural balance assuming human body as a complex set of interlinked segments. Thus, independent of the deviation's location, it is important to observe the entire human silhouette image. The images used in this methodology are taken at different planes by a digital camera. Images taken on sagittal plane provide knowledge of postural deviations such as thoracic kyphosis, lumbar hyperlordosis, cervical hyperlordosis, tibia angulation related to the feet, knee angulations (flexion and extension) and pelvis (anterior and posterior tilts) [1].

In this work the performance of the GPR treatment is evaluated according to a feasible curvature measure calculated to the sagittal plane contour of patient pictures that are taken before and after the treatment. This work adopts the multiscale curvature approach developed in [2].

Other approaches related to this paper include the method proposed by Koara et al. [3] that generates the contour model of each link element of an articulated object by using hierarchical part decomposition method (HPD). The HPD method can divide the body regions into characteristic elements without any model of link parameters or shape based on the distribution of high-curvature points of the object's contour in video sequences [3].

A new approach for object recognition based on polygonal approximation of the contour object was proposed in [4]. The authors adopted the polygon vertices as being the high curvature points of the contour applying the wavelet transform.

Chang and Liu [5] presented a modified version of the curvature scale space (CSS) image of the contour [6,7] to recognize hand posture.

The traditional clinical assessment of postural deviations in patients under physioteraphy treatment is achieved by the physiotherapist observing them in different image planes, including the profile one (sagittal plane) before, during and after the treatment. The patterns of normal postures [4] are used to guide the exam that includes head projection analysis, alignment of thorax and back, alignment of arms relative to trunk, flexion of the knees and lumbar region curvature of the patient. This clinical assessment depends on the specialist ability, specially for subtle cases. Thus, this work provides a novel assessment method that measures the contour curvature for both sagittal plane image contour (SPIC) and the external contour of the medial axis (CMA).

The procedure presented in this paper consists of two stages. In the first one, the curvature for both SPIC and CMA are calculated. In the second stage, the postural deviations are evaluated according to some features extracted from the patterns.

The remainder of this paper is organized as follows. Section 2 presents a brief description of the curvature method used to assess posture patterns. Section 3 describes the geometric features used to evaluate the human shapes. Section 4 presents the experimental results and Section 5 provides the concluding remarks.

2 Background

The curvature defines the orientation changes in the tangential direction in each point of a given curve. A plot of a boundary curvature function can reveal sharp peaks corresponding to convex or concave regions in the boundary**.** Thus, the curvature analysis can be used to assist the physiotherapist in detecting human postural deviations in sagittal plane images taken in the clinic according to some rules.

The most popular method for multiscale curvature estimation uses a series of Gaussian to convolve with the curve contour. The Gaussian standard deviation controls the smoothing effect and consequently the amount of details in the curve contour.

The representation of the original human contour is given in terms of the x and y coordinates along it. The signals x and y are obtained starting from the highest left point in the contour and following it in the counterclockwise manner.

The parametrized form of a given regular curve *C* is $C(t) = (x(t),y(t))$, with $t \in \Gamma \subset \Re$, i.e. the parameter *t* has values over an interval Γ of the real line $\Re$. The curvature of any point $C(t_0) = (x(t_0),y(t_0))$, $t_0 \in \Gamma$, is given by Equation (1). It measures direction changes on the contour and is defined as:

$$k(t) = \frac{\dot{x}(t)\,\ddot{y}(t) - \ddot{x}(t)\,\dot{y}(t)}{[(\dot{x}(t))^2 + (\dot{y}(t))^2]^{3/2}} \,. \tag{1}$$

where $\dot{x}(t)$, $\dot{y}(t)$, $\ddot{x}(t)$ and $\ddot{y}(t)$ are the first and the second order derivatives of *x* and *y* with respect to *t*.

The curvature of a sagittal plane image contour can be estimated in terms of the Fourier transform of its *x*- and *y*- signals. Multiplying a set of Gaussian functions, with distinct standard deviations, by the Fourier spectra of the parametric *x* and *y* signals of the contour yields a shrinkage effect on the original shape. To overcome this effect, a scalar coefficient Ω for energy correction is multiplied by the *x* and *y* signals, according to César Junior et al [3]. This scheme is used before applying Equation (1).

The proposed method consists in evaluating the curvature in each point of the patient's contour, searching for local maxima values, i.e. high curvature points (HCP). These values reflect the most significant changes in the posture balance due to the GPR treatment.

2.1 Symmetry Analysis

Skeletonization [8], also known as the medial axis transform, is a process for reducing foreground regions in a binary image to a skeleton that largely preserves the extent and connectivity of the original region while throws away most of the original foreground pixels. The medial axis transform (MAT) is particularly interesting in this paper to addressing posture issue since it is closely related to the feasible alignment of the human spine. This section presents the MAT, also called symmetry axis transform (SAT).

We have assumed the medial axis as being the symmetry axis for the sagittal plane image contour. We have observed that the medial axis reflects relevant changes in the posture balance and it can be used by the physiotherapist to infer the patient's posture.

The overlapping of the medial axes of the patients and their respective contours are displayed in Fig. 2. The patients present sagittal deviations that are reflected in their medial axes taken before the GPR treatment as show Figs. 2(a), 2(c) and 2(e)). These postures in Figs. 2(b), 2(d) and 2(f) were flattened by the GPR treatment. The patients exhibit a stretched silhouette and significant differences can be noted bycomparing the cervical and lumbar regions of the patients.

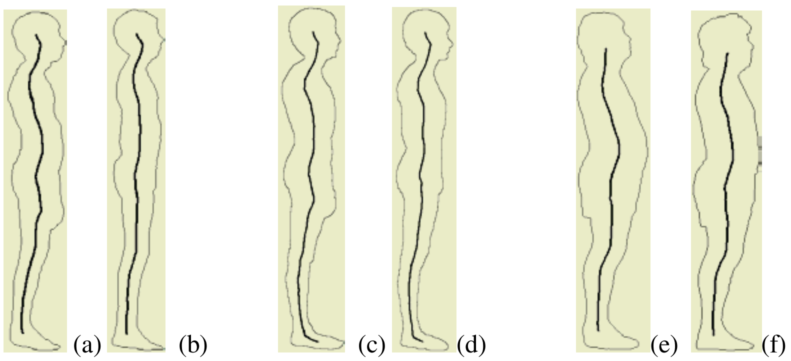


Fig. 1. The medial axis for sagittal plane image (*bold line*) of the (a) patient I before GPR, (b) patient I after GPR, (c) patient II before GPR, (d) patient II after GPR, (e) patient III before GPR and (f) patient III after GPR.

3 Shape Analysis

The aim of this section is to present features used for shape analysis. The idea consists in establishing the differences between the silhouette taken before and after the physiotherapy treatment. In the following, there are the relevant characteristics addressed to point out shape modifications due to the physiotherapy treatment.

In this approach, the uncalibrated area (A) and perimeter (P) in pixels represent the geometrical features that describe the human silhouette. Derived from them, the complexity (C) achieves a small numerical value for simple geometrically silhouettes and larger values for complex ones. It is defined as:

$$C = \frac{A}{2\sqrt{\pi P}} \ . \tag{2}$$

In [8] the area to perimeter ratio (APR) and thinness ratio (TR) are described by equations (3) and (4), respectively.

$$APR = \frac{A}{P} \ . \tag{3}$$

$$TR = 4\pi\left(\frac{A}{P^2}\right) . \tag{4}$$

The center of mass (C.M.) can be used to show the stretching effect upon the patient posture. Derived from the center of mass, the maximum (D_{max}) and minimum distances (D_{min}) from the boundary points to the center of mass are useful to determine qualitatively the stretching degree of the posture. The spreading measure (S) is derived from the principal component analysis [9], and it is presented in Equation (5) where λ_1 and λ_2 are the two eigenvalues associated with the covariance matrix and $\lambda_1 > \lambda_2$. This measure will be low for long and thin silhouettes and high for flattened profiles.

$$S = \frac{100\lambda_2}{\lambda_1 + \lambda_2}. \qquad (5)$$

4 Experimental Results

The images were taken by a digital camera in different planes in the beginning, in the middle and at the end of the GPR treatment. The selected patients wore shorts to reduced tactile hints from garments and stood barefoot on a fixed platform. Furthermore, they had short hair to avoid hiding the cervical spine. These procedures were taken to improve the assessment of the sagittal deviations.

Upon posture examination, patient I in Fig. 2(a) presents: lumbar hyperlordosis, thoracic hyperkiphosis and cervical hyperlordosis, head, hands and abdomen forward projected. After the GPR treatment, the new profile did not exhibit thoracic hyperkiphosis and the hands were aligned with the trunk. The thoracic and cervical regions presented standard flexion of the spine.

Relevant points in the sagittal plane image contour were marked as show Figs. 3(a) and 3(c) to call attention of the remarkable changes achieved in their surroundings by GPR. These points were selected because they were located in regions of the human contours that presented significant changes in the global postural balance, such as neck, abdomen, hands, thoracic kyphosis, lumbar lordosis and cervical lordosis. The modified posture of the patient I in Fig.3(c) due to the physiotherapy treatment was straightforwardly reflected in the curvature graphs of Figs. 3(d) and 3(g). The amplitude of the curvature values for these points displayed in Fig. 3(d) was decreased after GPR and the same effect was observed for the curvature (filled line) of the medial axis closed contour in Fig. 3(g).

By observing the curvature values and shape measures in Table 1 taken before and after GPR it can be noticed that all patients presented satisfactory results after the physiotherapy treatment. The curvature values for the points located in the cervical spine region confirmed the smoothing effect over the cervical hyperlordosis provided by the treatment. Similar results were observed in the lumbar spine, neck, abdomen, hands and thorax regions.

The shape measures in Table 1 pointed to an increasing tendency of the spreading measure, except for patient II, but an otherwise decreasing tendency of the thinness ratio for all patients. It implies that there was a stretching effect in the silhouette of the patients I, II and III derived from the physiotherapy treatment. The decreasing effect on the *x*- coordinates of the center of mass indicates that the patient III did not present a compressed silhouette. It can be observed that the spinal curves were smoothed after GPR treatment. Even though the *x*- coordinates for the other two patients did not decrease, the postural balance improvement became evident.

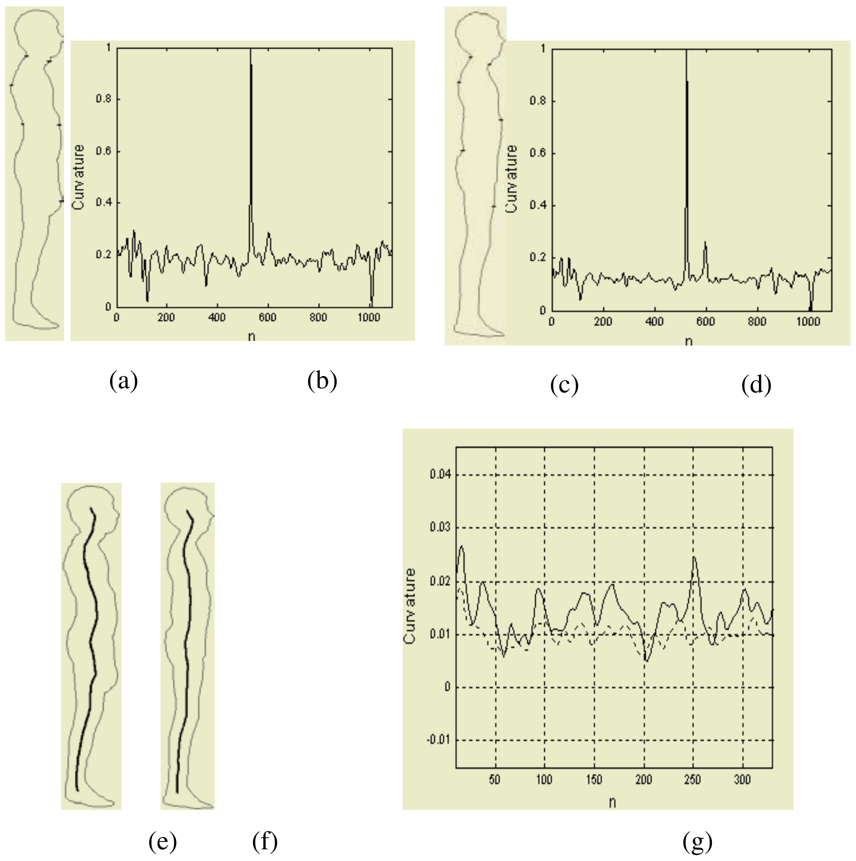
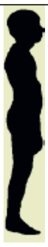
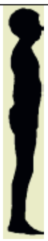
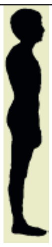
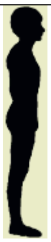
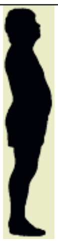
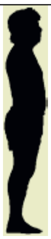


Fig. 2. Patient I and **(a)** its contour before GPR, **(b)** its respective curvature graph, (**c**) its contour after GPR **(d)** its respective curvature graph , (**e**) the medial axis (*bold line*) for the sagittal plane image before GPR, (**f**) the medial axis (*bold line*) for sagittal plane image after GPR and (**g**) a close detail of the overlapped curvatures calculated for the medial axis contour taken before *(filled line)* and after *(dotted line)* GPR.

5 Concluding Remarks

The smoothing effect over the curvatures caused by the GPR treatment was more straightforwardly visible for the contour curvature values related to lumbar and thoracic regions of the patients using this kind of physiotherapy. The external contour curvature of the medial axis was measured to substantiate the curvature results of the sagittal plane image contour and to infer the spine alignment. The remarkable changes on the postural balance of the patients pointed out to the benefits of the treatment.

Table 1. Shape measures and curvature values of the patient's profiles taken before and after the GPR treatment.

	Patient I		Patient II		Patient III	
	 Before GPR	 After GPR	 Before GPR	 After GPR	 Before GPR	 After GPR
Shape Measures						
A	25132	22663	45046	45393	34284	29850
P	1099	1098	1503	1575	1164	1143
C.M.	(228,54)	(234,59)	(308,58)	(308,57)	(245,49)	(234,47)
D_{min}	27.203	26.401	37.74	35.51	34.99	27.78
D_{max}	264.19	264.89	364.79	377.93	279.33	267.17
D_{med}	138.63	139.26	202.49	195.83	151.21	145.36
S	28.818	33.052	25.94	22.70	27.48	27.24
C	19.556	20.575	19.98	20.85	17.73	18.66
TR	0.075633	0.067268	0.071	0.065	0.095	0.084
APR	22.868	20.64	29.97	28.82	29.45	26.12
k(t) - Curvature Values						
Neck	0,1643	0,0519	0,3147	0,2217	0,2013	0,1420
Abdomen	0,1706	0,1399	0,4027	0,3256	0,3760	0,2935
Hands	0,2327	0,1305	0,4931	0,3915	0,4263	0,3087
Lumbar	0,1423	0,0738	0,3429	0,2194	0,2837	0,2650
Thorax	0,2142	0,1354	0,4500	0,4258	0,3911	0,3178
Cervical	0,0113	0,0221	0,1616	0,1161	0,1221	0,0379
K(t) - Curvature Values for the Medial Axis						
Neck	0.0086	0.0069	0.7000	0.6000	0.0084	0.0091
Abdomen	0.0168	0.0106	0.7500	0.6667	0.0099	0.0109
Hands	0.0145	0.0097	0.5750	0.5333	0.0073	0.0128
Lumbar	0.0176	0.0083	0.7500	0.6667	0.0106	0.0081
Torax	0.0088	0.0071	0.3250	0.4000	0.0042	0.0049
Cervical	0.0187	0.0095	0.8250	0.7778	0.0095	0.0091

The proposed method seems to be a promising approach for automatic evaluation of posture deviations because it can support visual diagnosis performed by the physiotherapist. It reduces the subjective evaluation of postural deviations establishing a more precise clinical assessment tool. The advantage of this system over other medical imaging ones is the fact that it does not use ionizing radiation, it is portable, versatile, noninvasive and relatively low-cost.

Acknowledgment

The authors are grateful to CAPES and CPNq for financial help.

References

1. Souchar, P.E.: Ginástica Postural Global. 2nd edn. Martins Fontes, São Paulo (1988)
2. César Junior, R. M., Costa, L. F.: Towards effective planar shape representation with multiscale digital curvature analysis based on signal processing techniques. Pattern Recognition. Vol. 29 (9) (1996) 1559-1569
3. Koara, K., Nishikawa, A., Miyazaki, F.: Hierarchical part decomposition method of articulated body contour, and its application to human body motion measurement. Proc. on IEEE/RSJ International Conference on Intelligent Robots and Systems. Vol. 3 Takamatsu Japan 31 October 5 November (2000) 2055 -2060
4. Cheikh, F., Quddus, A., Gabbouj, M.: Shape recognition based on wavelet-transform modulus maxima. The 7th IEEE International Conference on Electronics, Circuits and Systems. Vol. 1 Jounieh Lebanon December (2000) 461-464
5. Chang, C.C., Liu, C.: Modified curvature scale space feature alignment approach for hand posture recognition. Proc. on International Conference on Image Processing. Vol. 3 Barcelona September (2003) 309 -312
6. Mokhtarian, F., Mackworth, A.K.: Scale-based description and recognition of planar curves and two-dimensional shapes. IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol. 8 (1) (1986) 34-43
7. Mokhtarian, F., Mackworth, A.K.: A theory of multiscale, curvature-based shape representation for planar curves. IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol. 14 (8) (1992) 789-805
8. Costa, L.F., César Junior, R.M.: Shape Analysis and Classification: Theory and Practice. CRC Press USA 2001
9. Jollife, I. T.: Principal Component Analysis. Springer-Verlag New York 1986

Modelling Human Shape with Articulated Shape Mixtures

Abdullah A. Al-Shaher and Edwin R. Hancock

University of York, York YO1 5DD, UK

Abstract. This paper describes a statistical framework for recognising 2D shapes with articulated components. The shapes are represented using both geometrical and a symbolic primitives, that are encapsulated in a two layer hierarchical architecture. Each primitive is modelled so as to allow a degree of articulated freedom using a polar point distribution model that captures how the primitive movement varies over a training set. Each segment is assigned a symbolic label to distinguish its identity, and the overall shape is represented by a configuration of labels. We demonstrate how both the point-distribution model and the symbolic labels can be combined to perform recognition using a probabilistic hierarchical algorithm. This involves recovering the parameters of the point distribution model that minimise an alignment error, and recovering symbol configurations that minimise a structural error. We apply the recognition method to human moving skeleton.

1 Introduction

The task of recognising articulated shapes has attracted considerable interest in computer vision. The main problem is how to robustly recover correspondence when the object being tracked undergoes deformations and the detected feature points defining the object are subject to noise. One of the most effective ways of developing matching techniques is to draw on probabilistic and statistical methods. This approach has lead to the development of point distribution models [1], deformable templates [2] and condensation [3].

There are a number of ways in which object articulation can be modelled. Perhaps the simplest of these is to decompose the shape into a skeletal form, consisting of limbs or branches, and to model the articulation of the branches. The mechanical movement of the resulting shape can be captured by the rotation of the components. However, in order to constrain the change in shape to be physically realistic bounds, or distributions, must be imposed on the rotation angles [4, 5]. Hence, the mechanical constraints on articulation must be combined with a statistical model of limb movement. In addition to movement of the limbs, the articulated shape also has a structural composition, since the limbs can be assigned labels to distinguish them, and the arrangement of the labels can be used to provide further constraints for shape-recognition.

The aim in this paper is to develop a statistical framework that can be used to recognise articulated shapes using information concerning limb movement and symbolic constraints concerning the overall shape structure. To do this, we develop a hierarchical algorithm. Each shape is represented as an arrangement of articulated limbs. The movement of the limbs is represented by a polar point distribution model. The structural component of the model is represented by a configuration of limb-labels. The recognition architecture has two intercommunicating layers. The first of these is concerned with limb alignment, and this aims to recover the lengths and polar angles of the limbs. The second aims to assign limb-labels so that the overall structure is recovered consistently.

2 Point Distribution Models

The point distribution model of Cootes and Taylor commences from a set training patterns. Each training pattern is a configuration of labelled point co-ordinates or landmarks. The landmark patterns are collected as the the object in question undergoes representative changes in shape. To be more formal, each landmark pattern consists of L labelled points whose co-ordinates are represented by the set of position co-ordinates $\{X_1, X_2, ..., X_l\} = \{(x_1, y_1), ...(x_L, y_L)\}$. Suppose that there are T landmark patterns. The t^{th} training pattern is represented using the long-vector of landmark co-ordinates $X_t = (x_1, y_1, x_2, y_2, \cdots, x_L, y_L)^T$, where the subscripts of the co-ordinates are the landmark labels. For each training pattern the labelled landmarks are identically ordered. The mean landmark pattern is represented by the average long-vector of co-ordinates $Y = \frac{1}{T}\sum_{t=1}^{T} X_t$. The covariance matrix for the landmark positions is

$$\Sigma = \frac{1}{T}\sum_{t=1}^{T}(X_t - Y)(X_t - Y)^T \tag{1}$$

The eigenmodes of the landmark covariance matrix are used to construct the point-distribution model. First, the unit eigenvalues E of the landmark covariance matrix are found by solving the eigenvalue equation $|\Sigma - EI| = 0$ where I is the $2L \times 2L$ identity matrix. The eigen-vector ϕ_i corresponding to the eigenvalue E_i is found by solving the eigenvector equation $\Sigma\phi_i = E_i\phi_i$. According to Cootes and Taylor [1], the landmark points are allowed to undergo displacements relative to the mean-shape in directions defined by the eigenvectors of the covariance matrix Σ. To compute the set of possible displacement directions, the M most significant eigenvectors are ordered according to the magnitudes of their corresponding eigenvalues to form the matrix of column-vectors $\Phi = (\phi_1|\phi_2|...|\phi_M)$, where $E_1, E_2, ..., E_M$ is the order of the magnitudes of the eigenvectors. The landmark points are allowed to move in a direction which is a linear combination of the eigenvectors. The updated landmark positions are given by $\hat{X} = Y + \Phi\gamma$, where γ is a vector of modal co-efficients. This vector represents the free-parameters of the global shape-model. When fitted to an observed set of landmark measurements X_o, the least-squares estimate of the parameter vector is

$$\gamma = \frac{1}{2}(\Phi + \Phi^T)(X_o - Y)$$

3 Shape Representation

Our aim is to use point distribution models to account for shape deformations due to limb articulation. The model is a two component one. First, we have a limb-model. This accounts for the variations in shape of each of the individual limbs using a point distribution model to describe the modes of variation of the landmark points about a mean shape. Second, we have a limb arrangement model. This is an augmented point distribution model that describes the arrangement of the centre points of the limbs, and their polar angles.

We are concerned with recognising 2D shapes by modelling segment movement around the centre of the shape. The shape under study is assumed to be segmented into a set of K jointed and non-overlapping limbs. The k^{th} limb is represented by a long-vector of consecutive landmark points

$$X_k = (x_1^k, y_1^k, x_2^k, y_2^k, ...x_{n_k}^k, y_{n_k}^k)^T$$

The centre-of-gravity of the limb indexed k is

$$\boldsymbol{c}_k = \frac{1}{|n_k|} \sum_{i=1}^{n_k} (x_i^k, y_i^k)^T$$

The overall shape is represented by a long-vector of consecutive limb centres $C = (\boldsymbol{c}_1^T, \boldsymbol{c}_2^T, .., \boldsymbol{c}_K^T)^T$. The centre of articulated shape is computed by averaging the centre of the limbs

$$\boldsymbol{U} = \frac{1}{|K|} \sum_{k=1}^{K} \boldsymbol{c}_k$$

To model the articulated shape, we use a polar variant of the standard point distribution model [6]. This model allows the primitives to move about the centre of articulation According to this model the shape is viewed as an arrangement of non-overlapping primitives. Each primitive is represented by mean point $\boldsymbol{c}_k$. Limb articulation is represented by a set of limb-angles. For the k^{th} limb the angle is defined to be

$$\theta_k = \tan^{-1} \frac{U(y) - c_k(y)}{U(x) - c_k(x)}$$

and the angular arrangement of the limbs is represented by the vector $\Theta = (\theta_1, \theta_2, ..., \theta_K)^T$. The global movement of the limbs within a shape is specified by the concatenated long-vector of angles and the centres-of-articulation, i.e. by the vector $S = (\Theta^T, C^T)^T$.

To augment the geometric information, we assign symbols to the articulated components. Each training pattern is assigned to a shape class and each component primitive is assigned to a primitive class. The set of shape-labels is Ω_c and the set of articulated component or limb labels is Ω_s. The symbolic structure of each shape is represented a permissible arrangement of limb-labels. For shapes of class $\omega \in \Omega_c$ the permissible arrangement of limbs is denoted by $\Lambda_\omega =< \lambda_1^\omega, \lambda_2^\omega, ... >$.

4 Learning Mixtures of PDM's

In Cootes and Taylor's method [7], learning involves extracting a single covariance matrix from the sets of landmark points. Hence, the method can only reproduce variations in shape which can be represented as linear deformations of the point positions. To reproduce more complex variations in shape either a non-linear deformation or a series of local piecewise linear deformations must be employed.

In this paper we adopt an approach based on mixtures of point-distributions. Our reasons for adopting this approach are twofold. First, we would like to be able to model more complex deformations by using multiple modes of shape deformation. The need to do this may arise in a number of situations. The first of these is when the set of training patterns contains examples from different classes of shape. In other words, we are confronted with an unsupervised learning problem and need to estimate both the mean shape and the modes of variation for each class of object. The second situation is where the shape variations in the training data can not be captured by a single covariance matrix, and a mixture is required.

Our approach is is based on fitting a Gaussian model to the set of training examples. We commence by assuming that the individual examples in the training set are conditionally independent of one-another. We further assume that the training data can be represented by a set of shape-classes Ω. Each shape-class $\omega \in \Omega_s$ has its own mean point-pattern Y_ω and covariance matrix Σ_ω. With these ingredients, the likelihood function for the set of training patterns is

$$p(X_t, t = 1, ..., T) = \prod_{t=1}^{T} \sum_{\omega \in \Omega_s} p(X_t | Y_\omega, \Sigma_\omega) \tag{2}$$

where $p(X_t | Y_\omega, \Sigma_\omega)$ is the probability distribution for drawing the training pattern X_t from the shape-class ω. According to the EM algorithm, we can maximise the likelihood function above, by adopting a two-step iterative process. The process revolves around the expected log-likelihood function

$$Q_{n+1} = \sum_{t=1}^{T} \sum_{\omega \in \Omega_s} P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)}) \ln p(X_t | Y_\omega^{(n+1)}, \Sigma_\omega^{(n+1)}) \tag{3}$$

where $Y_\omega^{(n)}$ and $\Sigma_\omega^{(n)}$ are the estimates of the mean pattern-vector and the covariance matrix for class ω at iteration n of the algorithm. The quantity $P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)})$ is the *a posteriori* probability that the training pattern X_t belongs to the class ω at iteration n of the algorithm. The probability density for the pattern-vectors associated with the shape-class ω, specified by the estimates of the mean and covariance at iteration $n+1$ is $p(X_t | Y_\omega^{(n+1)}, \Sigma_\omega^{(n+1)})$. In the M, or maximisation, step of the algorithm the aim is to find revised estimates of the mean pattern-vector and covariance matrix which maximise the expected log-likelihood function. The update equations depend on the adopted model for the class-conditional probability distributions for the pattern-vectors.

In the E, or expectation, step the *a posteriori* class membership probabilities are updated. This is done by applying the Bayes formula to the class-conditional density. At iteration $n+1$, the revised estimate is

$$P(t \in \omega | X_t, Y_\omega^{(n)}, \varSigma_\omega^{(n)}) = \frac{p(X_t | Y_\omega^{(n)}, \varSigma_\omega^{(n)}) \pi_{t,\omega}^{(n)}}{\sum_{\omega \in \varOmega} p(X_t | Y_\omega^{(n)}, \varSigma_\omega^{(n)}) \pi_{t,\omega}^{(n)}} \tag{4}$$

where

$$\pi_{t,\omega}^{(n+1)} = \frac{1}{T} \sum_{t=1}^{T} P(t \in \omega | X_t, Y_\omega^{(n)}, \varSigma_\omega^{(n)}) \tag{5}$$

4.1 Mixtures of Gaussians

We now consider the case when the class conditional density for the training patterns is Gaussian. Here we assume that the pattern vectors are distributed according to the distribution

$$p(X_t | Y_\omega^{(n)}, \varSigma_\omega^{(n)}) = \frac{1}{(2\pi)^L \sqrt{|\varSigma_\omega^{(n)}|}} \exp\left[-\frac{1}{2} (X_t - Y_\omega^{(n)})^T (\varSigma_\omega^{(n)})^{-1} (X_t - Y_\omega^{(n)}) \right] \tag{6}$$

At iteration $n+1$ of the EM algorithm the revised estimate of the mean pattern vector for class ω is

$$Y_\omega^{(n+1)} = \sum_{t=1}^{T} P(t \in \omega | X_t, Y_\omega^{(n)}, \varSigma_\omega^{(n)}) X_t \tag{7}$$

while the revised estimate of the covariance matrix is

$$\varSigma_\omega^{(n+1)} = \sum_{t=1}^{T} P(t \in \omega | X_t, Y_\omega^{(n)}, \varSigma_\omega^{(n)}) (X_t - Y_\omega^{(n)})(X_t - Y_\omega^{(n)})^T \tag{8}$$

When the algorithm has converged, then the point-distribution models for the different classes may be constructed off-line using the procedure outlined in Section 2.

We apply this learning procedure separately to the landmark data for the individual limbs, and to the combined limb angle and limb-centre data. For the limb with label λ, the estimated modal matrix is $\varPhi_\lambda$ and the estimated parameter vector is γ_λ. For the shape-class with label ω, on the other hand, the combined modal matrix for the articulation angles and limb-centres is $\tilde{\varPhi}_\omega$ and the result of fitting to data is a parameter vector $\varGamma_\omega$. The first K rows of $\tilde{\varGamma}_\omega$ correspond to the limb angles, and the remaining $2K$ to the long-vectors of limbs centres. However, we need to constrain the parameters corresponding to the limb angles. Suppose that the mean-vector for the limb-angles is $\hat{\varTheta}_w$ and the corresponding covariance matrix is $\varSigma_w$. The angular deformations are constrained to avoid flipping by limiting the deformation vector. We use the variance associated with the eigen-modes to constrain the deformation. The k^{th} component of the parameter vector is constrained to fall in the interval $-3\sqrt{E_k} \leq \mathit{\Gamma}_{(k)} \leq 3\sqrt{E_k}$ The articulation angles lie in the range $-180°$ to $180°$ to avoid discontinuities associated with the flip from $0°$ to $360°$. A similar procedure for learning is followed to learn the variation in the polar representation of the limb and limb classes.

5 Hierarchical Architecture

With the limb-articulation and limb-centre point distribution models to hand, our recognition method proceeds in a hierarchical manner. Our aim is to classify the set of limb landmark long-vectors $X = \{\boldsymbol{z}_1, .., \boldsymbol{z}_k, ..., \boldsymbol{z}_K)$ representing a test-shape. To commence, we make maximum likelihood estimates of the best-fit parameters of each limb-model to each set of limb-points. The best-fit parameters γ_λ^k of the limb-model with class-label λ to the set of points constituting the limb indexed k is

$$\gamma_\lambda^k = \arg\max_{\gamma} p(\boldsymbol{z}_k|\mathit{\Phi}_\lambda, \gamma) \tag{9}$$

We use the best-fit parameters to assign a label to each limb. The label is that which has maximum a posteriori probability given the limb parameters. The label assigned to the limb indexed k is

$$l_k = \arg\max_{l \in \mathit{\Omega}_s} P(l|\boldsymbol{z}_k, \gamma_\lambda, \mathit{\Phi}_\lambda) \tag{10}$$

In practice, we assume that the fit error residuals follow a Gaussian distribution. As a result, the class label is that associated with the minimum squared error. This process is repeated for each limb in turn. The class identity of the set of limbs is summarised by the string of assigned limb-labels $L =< l_1, l_2, ... >$. Hence, the input layer is initialised using maximum likelihood limb parameters and maximum a posteriori probability limb labels. The shape-layer takes this information as input. The goal of computation in this second layer is to refine the configuration of limb labels using global constraints on the arrangement of limbs to form consistent shapes. The constraints come from both geometric and symbolic sources. The geometric constraints are provided by the fit of a polar limbs point distribution model. The symbolic constraints are provide by a dictionary of permissible limb-label strings for different shapes.

The parameters of the limb-centre point distribution model are found using the EM algorithm [8]. Here we borrow ideas from the hierarchical mixture of experts algorithm [9], and pose the recovery of parameters as that of maximising a gated expected log-likelihood function for the distribution of limb-centre alignment errors $p(X|\mathit{\Phi}_\omega, \mathit{\Gamma}_\omega)$. The likelihood function is gated by two sets of probabilities. The first of these are the a posteriori probabilities $P(\lambda_k^\omega|\boldsymbol{z}_k, \gamma_{\lambda_k^\omega}, \mathit{\Phi}_{\lambda_k^\omega})$ of the individual limbs. The second are the conditional probabilities $P(L|\mathit{\Lambda}_\omega)$ of the assigned limb-label string given the dictionary of permissible configurations for shapes of class ω. The expected log-likelihood function is given by

$$\mathcal{L} = \sum_{\omega \in \mathit{\Omega}_c} P(L|\mathit{\Lambda}_\omega) \left\{ \prod_k P(\lambda_k^\omega|\boldsymbol{z}_k, \gamma_{\lambda_k^\omega}, \tilde{\mathit{\Phi}}_{\lambda_k^\omega}) \right\} \ln p(X|\tilde{\mathit{\Phi}}_\omega, \mathit{\Gamma}_\omega) \tag{11}$$

The optimal set of polar limb arrangement parameters satisfies the condition

$$\Gamma_\omega^* = \arg\max_{\Gamma} P(L|\Lambda_\omega)\left\{\prod_k P(\lambda_k^\omega|\boldsymbol{z}_k, \gamma_{\lambda_k^\omega}, \tilde{\Phi}_{\lambda_k^\omega})\right\} \ln p(X|\tilde{\Phi}_\omega, \Gamma_\omega) \tag{12}$$

From the maximum likelihood alignment parameters we identify the shape-class of maximum *a posteriori* probability. The class is the one for which

$$\omega^* = \arg\max_{\omega\in\Omega_c} P(\omega|X, \tilde{\Phi}_\omega, \Gamma_\omega^*) \tag{13}$$

The class identity of the maximum *a posteriori* probability shape is passed back to the limb-layer of the architecture. The limb labels can then be refined in the light of the consistent assignments for the limb-label configuration associated with the shape-class ω

$$l_k = \arg\max_{\lambda\in\Omega_s} P(\lambda|\boldsymbol{z}_k, \gamma_l^k, \tilde{\Phi}_\lambda)P(L(\lambda,k)|\Lambda_\omega) \tag{14}$$

Finally, the maximum likelihood parameters for the limbs are refined

$$\gamma_k = \arg\max_{\gamma} p(\boldsymbol{z}_k|\tilde{\Phi}_{l_k}, \gamma, \Gamma_\omega^*) \tag{15}$$

These labels are passed to the shape-layer and the process is iterated to convergence.

6 Models

To apply the model to shape-recognition, we require models of the alignment error process and the label error process.

6.1 Alignment Errors

To develop a useful alignment algorithm we require a model for the measurement process. Here we assume that the observed position vectors, i.e. $\boldsymbol{z}_k$ are derived from the model points through a Gaussian error process. According to our Gaussian model of the alignment errors,

$$p(\boldsymbol{z}_k|\tilde{\Phi}_\lambda, \gamma_\lambda) = \frac{1}{2\pi\sigma}\exp\left[-\frac{1}{2\sigma^2}(\boldsymbol{z}_k - \hat{X}_\lambda - \tilde{\Phi}_\lambda\gamma_\lambda)^T(\boldsymbol{z}_k - \hat{X}_\lambda - \tilde{\Phi}_\lambda\gamma_\lambda)\right] \tag{16}$$

where σ^2 is the variance of the point-position errors which for simplicity are assumed to be isotropic. A similar procedure may be applied to estimate the parameters of the polar limb-angle distribution model.

6.2 Label Assignment

The distribution of label errors is modelled using the method developed by Hancock and Kittler [10]. To measure the degree of error we measure the Hamming distance between the assigned string of labels L and the dictionary item Λ. The Hamming distance is given by

$$H(L, \Lambda_\omega) = \sum_{i=1}^{K} \delta_{l_i, \lambda_i^\omega} \tag{17}$$

where δ is the Dirac delta function. With the Hamming distance to hand, the probability of the assigned string of labels L given the dictionary item Λ is

$$P(L|\Lambda_\omega) = K_p \exp[-k_p H(L, \Lambda_\omega)] \tag{18}$$

where $K_p = (1-p)^K$ and $k_p = \ln \frac{1-p}{p}$ are constants determined by the label-error probability p.

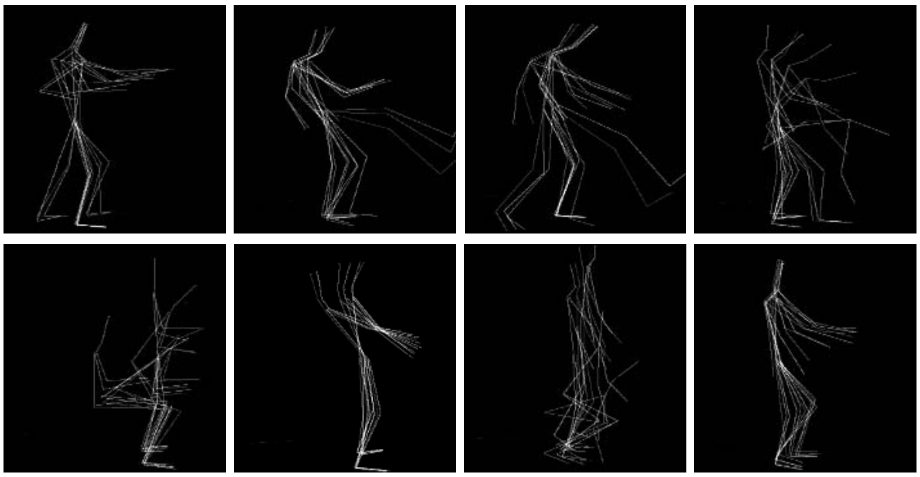


Fig. 1. Human Training Sets

7 Experiment

We have evaluated our approach on human motion sequences. Figure 1 shows some of the data used for the purpose of learning. In total, we used 14 distinct classes of human motion for learning purposes. We made use of 18 frames per class where each frame is segmented into 13 feature points representing the centre of a limb. In figure 2, we show four example shapes recovered as the output of our learning stage.

To evaluate recognition performance, we have used 1200 frames corresponding to different motion classes for testing. Table 1 shows the label number of

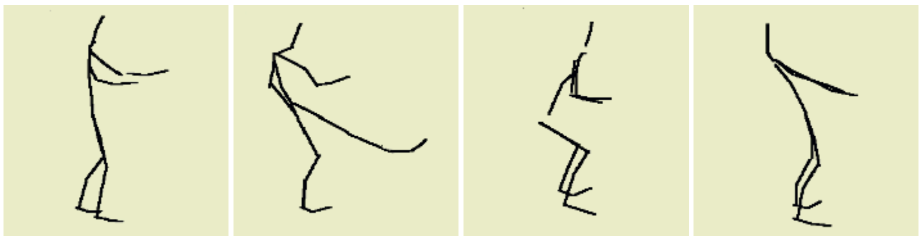


Fig. 2. Learnt Shapes:(a) Boxing, (b) Kicking, (c) Relaxing, (d) Stretching

Table 1. Label replacement for shape classes

	Boxing		**Kicking**		**Relaxing**		**Picking**		**Stretching**		**Shooting**	
Iter. No.	**Cor**	**Wro**	**Cor**	**Wro**	**Cor**	**Wro**	**Cor**	**Wro**	**Cor**	**Wro**	**Cor**	**Wro**
1	4	9	2	11	3	10	5	8	4	9	8	5
2	2	11	1	12	2	11	3	10	3	10	10	3
3	1	12	0	13	0	13	2	11	2	12	12	1
4	0	13	0	13	0	13	0	13	0	13	13	0
5	0	13	0	13	0	13	0	13	0	13	13	0

Table 2. Recognition rate for shape classes

Shape	**Samples**	**Correct**	**Wrong**
Boxing	200	198	2
Kicking	200	163	37
Relaxing	200	197	3
Picking	200	181	19
Stretching	200	200	0
Shooting	200	179	21
Recognition Rate		93.16%	6.83%

correct and incorrect limb label assignment as a function of iteration number. It is apparent that the error rate decreases for classes that are irrelevant to the shape under recognition, while it increases for the correct class. Table 2 shows the recognition rate for frames of the six classes of motion. The recognition rate is of 93.16%. The poorest recognition occurs for the kicking, the picking and the shooting classes. Since these classes share similar limb movement, we can conclude that recognition is reasonably high. In Figure 3, we show the alignment process as a function of iteration number. The different curves are for different motion classes. It is clear from the graph that the a *posteriori* probability converges on a clear ambiguous assignment from a state in which there is a considerable ambiguity. Figure 4 shows the recognition rate for different classes when random jitter is present. The recognition rate is reasonably high even under noise.

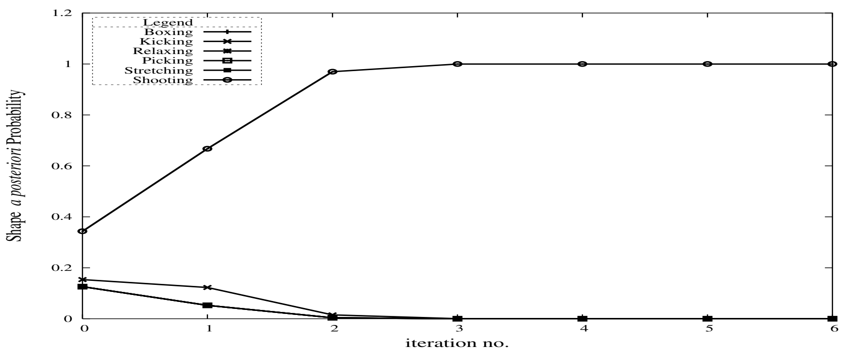


Fig. 3. Alignment as a function of iteration number

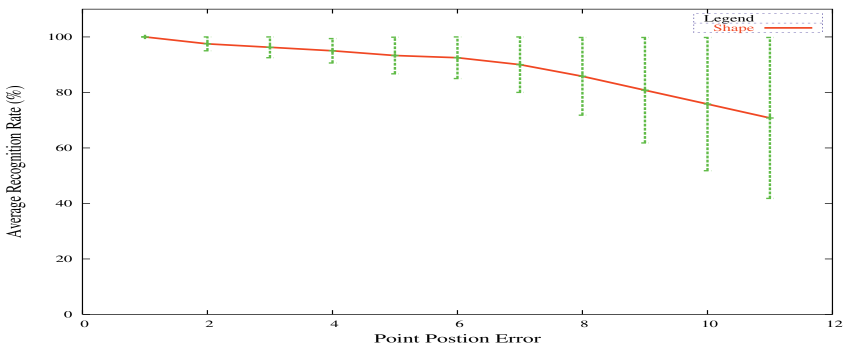


Fig. 4. Recognition Rates with respect to random point movement

8 Conclusion

In this paper, we have described a method for fitting articulated shape-models to landmark point data. The shape deformation process adopted is based on point distribution models. The model representation is a hierarchical one. There is a Cartesian deformation model for the limbs and the limb-centres, together with a polar model which represents limb articulation. We develop a probabilistic framework for fitting a mixture of articulated models to data. The method delivers good results of human shape modelling.

References

1. Cootes T.; Taylor C. Combining point distribution models with shape models based on finite element analysis. *IVC*, 13(5):403–409, 1995.
2. Duta N.; Jain A.; Dubuisson P. Learning 2d shape models. *International Conference on Computer Vision and pattern Recognition*, 2:8–14, 1999.
3. Michael Isard; Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *Proc. ECCV*, pages 343–356, 1996.

4. J Gonzales; J Varona; F Roca; and J Villanueva. aspace: Action space for recognition and synthesis of human actions. *2nd IWAMDO, Spain*, pages 189–200, 2002.
5. James Rehg ; Takeo Kanade. Visual tracking of high dof articulated structures: an application to human hand tracking. *3rd ECCV, Sweden*, pages 35–46, 1994.
6. Heap T.; Hogg D. Extending the point distribution model using polar coordinates. *Image and Vision Computing*, 14:589–599, 1996.
7. Cootes T.; Taylor C. A mixture models for representing shape variation. *Image and Vision Computing*, 17:403–409, 1999.
8. Dempster A.; Laird N.; Rubin D. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Soc. Ser.*, 39:1–38, 1977.
9. Jordan M.; Jacobs R. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6:181–214, 1994.
10. Edwin R. Hancock; Josef Kittler. Edge-labelling using dictionary-based relaxation. *IEEE Transaction on PAMI*, 12(2):165–181, 1990.

Learning People Movement Model from Multiple Cameras for Behaviour Recognition

Nam T. Nguyen[1], Svetha Venkatesh[1], Geoff A.W. West[1], and Hung H. Bui[2]

[1] Department of Computing, Curtin University of Technology
GPO Box U1987 Perth, 6845 Western Australia
{nguyentn,svetha,geoff}@cs.curtin.edu.au
[2] Artificial Intelligence Center, SRI International
333 Ravenswood Ave, Menlo Park, CA 94025, USA
bui@ai.sri.com

Abstract. In surveillance systems for monitoring people behaviours, it is important to build systems that can adapt to the signatures of people's tasks and movements in the environment. At the same time, it is important to cope with noisy observations produced by a set of cameras with possibly different characteristics. In previous work, we have implemented a distributed surveillance system designed for complex indoor environments [1]. The system uses the Abstract Hidden Markov mEmory Model (AHMEM) for modelling and specifying complex human behaviours that can take place in the environment. Given a sequence of observations from a set of cameras, the system employs approximate probabilistic inference to compute the likelihood of different possible behaviours in real-time. This paper describes the techniques that can be used to learn the different camera noise models and the human movement models to be used in this system. The system is able to monitor and classify people behaviours as data is being gathered, and we provide classification results showing the system is able to identify behaviours of people from their movement signatures.

1 Introduction

Monitoring people behaviours in large and complex environments using multiple cameras for automated surveillance is an important research area. Approaches to this problem usually divide the solution into two layers of processing: a low-level tracking component processes low-level visual data from the cameras and produces a stream of events which are then interpreted by a recognition module to produce high-level description of the people activities in the environment [2–5]. Oliver *et al* [2] propose a Layered Hidden Markov Model (LHMM), where the classification results of the lower layer are used as inputs to the higher layer. Ivanov and Bobick [3] proposed a two-stage strategy to recognise the interactions of humans and vehicles. At the lowest level, the system recognises simple events, which are used as inputs for a stochastic context-free grammar parsing mechanism to recognise multi-object interactions at the higher level.

It is well-known that current low-level tracking techniques are not robust, and their outputs are inherently noisy due to a variety of environmental and processing conditions. Thus, it is up to the "high-level" module to deal with the imperfect output produced by low-level processing to produce robust and accurate descriptions of the observed activities. We thus argue that the high-level behaviour recognition module needs to be based on a framework that facilitates the modelling of and reasoning with uncertainty. Previously, we proposed the use of the Abstract Hidden Markov mEmory Model (AHMEM) for this purpose [1, 6]. In the AHMEM, behaviours are organized into a stochastic hierarchy. Each behaviour can be refined into a sequence of more simple behaviours at lower levels. In addition, the rules for refinement can be made non-deterministic or stochastic. The model is as expressive as other grammar-based models such as the Probabilistic Context Free Grammar (PCFG) [7], and can model state-dependent goal-directed behavours. At the same time, it supports online, and efficient probabilistic inference of high-level behaviours from low-level data. Furthermore, the hierarchical nature of the model makes it suitable for the natural hierarchy existing in spatial regions, making it scalable to larger and more complex environments.

The AHMEM framework also provides the flexibility for integrating with a noisy low-level tracking module via a HMM-like model at the bottom level of the behaviour hierarchy. This acts like an interface between the AHMEM and the low-level tracking module. This paper addresses the problem of learning the necessary parameters for building this interface between the high-level and low-level tracking module. We provide techniques for estimating the obsevation models of the cameras, and to estimate the movement models of people on one floor of a building. We then describe a complete distributed surveillance system combining a low-level tracking module with the AHMEM for behaviour recognition. We provide experimental results showing that the system is able to monitor and robustly classify complex human behaviours in an indoor environment.

The paper is organised as follows. An overview of the surveillance system is provided in Section 2. The techniques to learn observation models and movement models for the surveillance system are presented in Sections 3 and 4, respectively. The system implementation is described in Section 5. Finally, Section 6 presents the experimental results of the implemented system in a real office-like environment.

2 Overview of the Surveillance System

The surveillance system has two major components: the distributed low-level tracking module and the high-level behaviour recognition module. The distributed tracking module extracts people trajectories using multiple static cameras. The trajectories are inputs for the high-level behaviour recognition module. The implementation of the surveillance system is described in [1].

In the behaviour recognition module, the AHMEM and its parameters define a conditional distribution over the observation sequences given a behaviour: $\Pr(\tilde{o}|\pi^k)$. In recognising the behaviour of a person in the scene, we are given a sequence of observations from the low-level tracking module: $\tilde{o}_{t-1} = (o_1, \ldots, o_{t-1})$ up to the current time t, and need to compute the probability $\Pr(\pi_t^k|\tilde{o}_{t-1})$, where π_t^k represents the policy being executed at level k and time t. This provides the distribution of the possible behaviours that might be currently executed at level k in the hierarchy. The computation needs to be done at every time instance t when a new observation o_t arrives. The problem is termed *policy recognition* [8], and is equivalent to the on-line inference (filtering) problem in the AHMEM. An efficient approximate inference algorithm based on the Rao-Blackwellised Particle Filter (RBPF) [9] for computing the probabilities is given in [10, 8]

The necessary parameters for building the interface between the high-level and low-level tracking modules are the observation models of cameras and the movement models of people on one floor of a building [1]. In the following sections, we will describe the techniques to learn these parameters in detail.

3 Learning Observation Models

The observation model for a camera C is defined as $B = \Pr(o|s, C)$, where s is the state of a person and o is the observation. Usually, there are a large number of states in the field of view (FOV) of the camera C. Thus, we have difficulty in creating enough sample video sequences to learn B. Assume that the observation o is in one of the cells within the neighbourhood of the state s (including s), and $\Pr(o|s, C)$ is unchanged for all states s in the FOV of camera C, i.e. the statistics are spatially invariant. We can compress the observation model B to a compressed observation model B^c, which is a 3×3 matrix given as:

$$B^c = \begin{bmatrix} \Pr(o_{northwest}|C) & \Pr(o_{north}|C) & \Pr(o_{northeast}|C) \\ \Pr(o_{west}|C) & \Pr(o_{center}|C) & \Pr(o_{east}|C) \\ \Pr(o_{southwest}|C) & \Pr(o_{south}|C) & \Pr(o_{southeast}|C) \end{bmatrix}$$

where o_{north}, $o_{northeast}$, o_{east}, $o_{southeast}$, o_{south}, $o_{southwest}$, o_{west}, $o_{northwest}$ and o_{center} are nine possible observations of a true state s (see Fig. 1). Instead of learning the observation model B, we can learn the compressed observation model B^c from a set of sample video sequences.

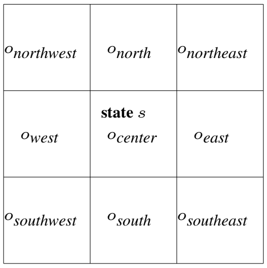


Fig. 1. The possible observations of a state s.

We learn the compressed observation model B^c for a camera C from sample video sequences, which are created by recording people in the environment from the views of all cameras. We run the tracking system to obtain the real world coordinates of people in the sample video sequences. Among these coordinates, we randomly choose N

coordinates $(x_1,y_1),\ldots,(x_N,y_N)$ that are originally generated from camera C. We consider these coordinates as the observations of the people. We then manually extract the corresponding person's true positions: $(x^t_1, y^t_1),\ldots,(x^t_N, y^t_N)$. These coordinates are mapped to the cells (states) in the environment, i. e. $(x_1, y_1),\ldots,(x_N, y_N)$ are mapped to $o_1,\ldots,o_N$ and $(x^t_1, y^t_1),\ldots,(x^t_N, y^t_N)$ are mapped to $s_1,\ldots,s_N$.

We estimate the compressed observation model B^c from the N observations $o_1, \ldots, o_N$ and the N corresponding states $s_1,\ldots,s_N$. Note that $B^c = \Pr(o|C)$, where $o \in \{o_{north}, o_{northeast}, o_{east}, o_{southeast}, o_{south}, o_{southwest}, o_{west}, o_{northwest}, o_{center}\}$ (see Fig. 1). To estimate $\Pr(o_{north}|C)$, we count the number of times that o_i is the northern neighbouring state of s_i from the N observations and the N corresponding states. $\Pr(o_{north}|C)$ then equals the frequency that o_i is the northern neighbouring state of s_i. The remaining probabilities of B^c are estimated in a similar manner.

4 Learning Movement Models

For a bottom level behaviour π in a region R, we need to learn the movement model $A = \Pr(s' \mid s, \pi)$, which is defined for all states s in R and for all neighbouring states s' of s [1].

4.1 Dealing with Large Transition Models

For the case in which region R is small, we can learn A from a number of training video sequences. However, when region R is large and has many states, the number of training video sequences required to learn A is large due to the size of the state space. Therefore, instead of learning the complete movement model A, which is a difficult task, we compress A to a compressed movement model A^c and learn A^c.

In large regions, we are only interested in behaviours representing the action of a person going to a destination such as going to a printer, going to a computer, and so on. Thus, we can assume that each behaviour defined in a large region has a destination. The compressed movement model A^c is defined as a 3×3 matrix specifying the probabilities that a person moves to the next state, assuming that the direction to a destination is the *up-front* vector. The compressed movement model A^c is given as:

$$A^c = \begin{bmatrix} p_{northwest} & p_{north} & p_{northeast} \\ p_{west} & p_{center} & p_{east} \\ p_{southwest} & p_{south} & p_{southeast} \end{bmatrix} \tag{1}$$

Given a direction to reach the destination of π which is say East, A^c can be rotated to apply the probabilities.

4.2 Compressing the Movement Model

The compressed movement model A^c can be computed from the movement model A. We compute the probability p_{south} of A^c as:

$$p_{south} = \frac{\sum_{s,s',\text{ where } s'=south(s,\pi)} \Pr(s'|s,\pi)}{\sum_{s,s'} \Pr(s'|s,\pi)}$$

where the state *south*(s, π) is computed as follows: The set of directions $\{$*North*, *Northeast*, *East*, *Southeast*, *South*, *Southwest*, *West*, *Northwest* $\}$ is rotated such that the *up-front* vector (North) becomes the direction to reach to the destination of π from s. Then, *south*(s, π) is the neighbouring state of the state s in the new *South*. In a similar way, we can define *north*(s, π), *northeast*(s, π) and so on. The other probabilities of A^c are computed in a similar manner.

4.3 Expanding Compressed Movement Models

The movement model A can be computed from the compressed movement model A^c as follows: Note that $A = \Pr(s'|s, \pi)$, where s' is a neighbouring state of s (including s), and A^c is shown in Eq. 1. We first determine the relation among s, s' and π. If $s' = \mathit{north}(s, \pi)$, then $\Pr(s'|s, \pi) = p_{\mathit{north}}$, if $s' = \mathit{northeast}(s, \pi)$, $\Pr(s'|s, \pi) = p_{\mathit{northeast}}$, and so on.

4.4 Learning the Compressed Transition Models

The movement model of the behaviour π, i.e. $A = \Pr(s' \mid s, \pi)$, and the observation of each camera C, i.e. $B = \Pr(o|s, C)$, form a Hidden Markov Model. We propose an algorithm based on the expectation maximisation (EM) algorithm for the Hidden Markov Model (HMM) to learn the compressed movement model A^c. We term this algorithm the EM algorithm for the HMM with compressed parameters (Algorithm 1.1).

The inputs for Algorithm 1.1 are the compressed observation models for the cameras and a set of training sequences. The compressed observation models of the cameras are learned as in Section 3. They remain unchanged throughout the algorithm. To generate the required training data for the algorithm, we determine all cameras that can view the execution of the behaviour π, and record a set of video sequences of people executing π using these cameras. We take each video sequence as input to the tracking system to extract the person's trajectory. The trajectory is converted to a sequence of cells or observations. As a result, we have a set of observation sequences for the behaviour π.

In the beginning, the algorithm initialises the initial state probability π and the compressed movement model A^c. It also expands the compressed observation model $B^c(C_k)$ to the observation model $B(C_k)$ ($k = 0, \ldots, \mathit{no\_camera} - 1$). In the main loop, for each observation sequence o^j_1, o^j_2,...,$o^j_{m_j}$ ($j = 1, 2, \ldots, \mathit{no\_seq}$), the algorithm finds the camera that generates this observation sequence. Assume that camera C_k is found. The algorithm expands the compressed movement model A^c to the full movement model A. Then, it computes the sufficient statistics, which are the expected frequency (number of times) in a state s at time $t = 1$, i.e. $\bar{\pi}_j$, and the expected number of transitions from a state s to a state s', i.e. $\bar{A}_j$. $\bar{A}_j$ is compressed to $\bar{A}^c_j$. After obtaining the expected sufficient statistics for all observation sequences, we estimate the parameters of the HMM for the next iteration as $\pi = \mathit{normalise}(\sum_{j=1}^{\mathit{no\_seq}} \bar{\pi}_j)$ and $A^c = \mathit{normalise}(\sum_{j=1}^{\mathit{no\_seq}} \bar{A}^c_j)$. The algorithm terminates when the likelihood score has reached a local minimum, and we obtain the compressed movement model A^c.

Algorithm 1.1 The EM algorithm for the HMM with compressed parameters.

```
input
  Obs. sequences $o_{\bullet}^{j}$, $o_{\bullet}^{j}$,...,$o_{m_j}^{j}$ $(j = 1, \dots, no\_seq)$
  $B^c(C_k)$, $k = 0, \dots, no\_camera - 1)$
output
  Compressed movement model $A^c$
begin
  Initialise $\pi^{\bullet\bullet\bullet}$, $A^{c\bullet\bullet\bullet}$
  Expand $B^c(C_k) \rightarrow B(C_k), \forall k = 0, \dots, no\_camera - 1$
  for $i$=1 to $N$
    for $j$=1 to $no\_seq$
      Get camera $C_k$ which generates $o_{\bullet}^{j}$, $o_{\bullet}^{j}$,...,$o_{m_j}^{j}$
      Expand $\rightarrow A^{\bullet i \bullet}$
      From $\pi^{\bullet i \bullet}$, $A^{\bullet i \bullet}$, $B(C_k)$ and $o_{\bullet}^{j}$, $o_{\bullet}^{j}$,...,$o_{m_j}^{j}$, compute:
        $\bar{\pi}_j^{\bullet i \bullet}$: expected frequency in state $s$ at time $t = 1$
        $\bar{A}_j^{\bullet i \bullet}$: expected number of transitions from $s$ to $s'$
        Compress $\bar{A}_j^{\bullet i \bullet} \rightarrow \bar{A}_j^{c \bullet i \bullet}$
    end
    Compute $\pi^{\bullet i \bullet\ \bullet\bullet}$, $A^{c \bullet i \bullet\ \bullet\bullet}$ as:
      $\pi^{\bullet i \bullet\ \bullet\bullet} = normalise(\sum_{j \bullet\ \bullet}^{no\_seq} \bar{\pi}_j^{\bullet i \bullet})$
      $A^{c \bullet i \bullet\ \bullet\bullet} = normalise(\sum_{j \bullet\ \bullet}^{no\_seq} \bar{A}_j^{c \bullet i \bullet})$
  end
  return $A^c$ = $A^{c \bullet N \bullet\ \bullet\bullet}$
end
```

5 System Implementation in Real Environments

The implementation of the surveillance system in an office-like environment is described in [1]. The environment has a Corridor, a Staff room and a Vision lab. The surveillance system has six static cameras, in which two are in the Corridor, one in the Staff room and the last three in Vision lab.

A three-level behaviour hierarchy is defined in the system (Fig. 2). The behaviours at the bottom level represent the movement of a person within a single region (Corridor, Staff room or Vision lab). The behaviours at the middle level represent the movement of a person in the whole environment. The top level behaviours represent the different tasks that a person might perform during the entire interval that the person stays in the environment, i.e. printing the documents, using the library or an unclassified task.

The set of parameters of the behaviour hierarchy are described in [1]. The parameters of the middle level and high level behaviours are defined manually, but they can be learned easily by observing many real scenarios. The movement models of the bottom level behaviours and the camera observation models are specified as follows:

5.1 Specifying the Compressed Observation Models for the Cameras

We learn the compressed observation models for the six cameras $C_0,\dots,C_5$ as in Section 3. We let people walk in the environment and record a set of video sequences seen

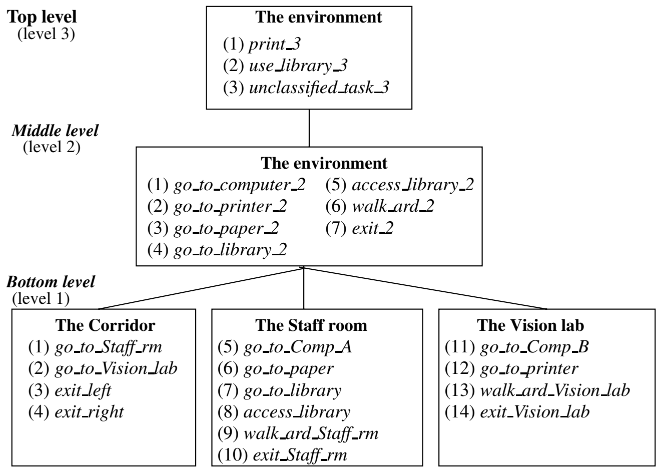


Fig. 2. The behaviour hierarchy.

from the six cameras. With each camera, we obtain 100 coordinates of the people returned from the tracking system and manually get the corresponding true coordinates. From these coordinates, we estimate the compressed observation model for the camera.

Fig. 3(a)-(f) show the compressed observation models learned for the six cameras. Note that a coordinate of a person returned from the tracking system (the person's observation) is the centre of the bottom edge of the person's bounding box. Therefore, the observation of a person is usually nearer the camera than the person's true position. This explains why in the compressed observation model for camera C_0, probabilities $\Pr(o_{north}|C_0)$ and $\Pr(o_{east}|C_0)$ are quite high (see Fig. 3(a)). The probabilities of the compressed observation models for the other cameras show the same property.

5.2 Specifying the Movement Models for Bottom Level Behaviours

The bottom level behaviours, which translate to the full or compressed movement models are learned as described in Section 4.

We learn the compressed movement model of behaviour *go_to_printer* as follows: Behaviour *go_to_printer* can be viewed from cameras C_0, C_1 and C_5. We record 30 video sequences of people executing behaviour *go_to_printer* from cameras C_0, C_1 and C_5. Then, we use Algorithm 1.1 to obtain the compressed movement model of behaviour *go_to_printer* (see Fig. 4(a)).

The movement models of other bottom level behaviours are learned in a similar way. For example, the results of the learning steps for behaviours *go_to_Linux*, *exit_Vision*, *go_to_paper*, *go_to_library* and *exit_Staff* are shown in Fig. 4(b)-(f).

6 Experiments and Results

To demonstrate that the parameters specified in Sections 5.1 and 5.2 allow the surveillance system to recognise and monitor people behaviour reliably, we tested the system

$\Pr(o_{north}|C_0)$ $\Pr(o_{east}|C_0)$

0.01	0.16 o_{north}	0.03
0.01	0.54 state s	0.18 o_{east}
0.02	0.03	0.02

(a) camera C_0

0.01	0.06	0.01
0.05	0.64	0.08
0.01	0.12	0.02

(b) camera C_1

0.01	0.06	0.01
0.07	0.66	0.08
0.02	0.08	0.01

(c) camera C_2

0.05	0.07	0.01
0.22	0.57	0.03
0.01	0.03	0.01

(d) camera C_3

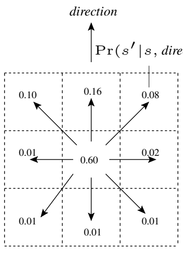
0.02	0.11	0.06
0.04	0.56	0.09
0.01	0.10	0.01

(e) camera C_4

0.01	0.07	0.01
0.16	0.62	0.01
0.01	0.10	0.01

(f) camera C_5

Fig. 3. The observation models for the six cameras C•,...,C•.



(a) *go_to_printer*

0.05	0.10	0.04
0.07	0.70	0.01
0.01	0.01	0.01

(b) *go_to_Linux*

0.01	0.15	0.07
0.01	0.68	0.05
0.01	0.01	0.01

(c) *exit_Vision*

0.08	0.04	0.01
0.14	0.64	0.01
0.05	0.02	0.01

(d) *go_to_paper*

0.03	0.13	0.11
0.01	0.62	0.07
0.01	0.01	0.01

(e) *go_to_library*

0.07	0.18	0.04
0.04	0.59	0.02
0.03	0.02	0.01

(f) *exit_Staff*

Fig. 4. The compressed movement models of behaviours *go_to_printer*, *go_to_Linux*, *exit_Vision*, *go_to_paper*, *go_to_library* and *exit_Staff*.

with 16 video sequences. In each video sequence, a person performs a task of printing, using the library or an unclassified task. The results of recognising these behaviours over time are shown in Fig. 5. As in the figure, with each video sequence and at each time slice, the system can recognise the most likely behaviour being executed by the person. The *winning* top level behaviour is available only at the end of the corresponding video sequence.

We compare the *winning* top level behaviours recognised by the system in the 16 video sequences with the groundtruth. The system correctly recognises the top level behaviours in 15 video sequences and misclassifies the top level behaviour in video sequence 11 (see Table 1). In video sequence 11, a person is executing behaviour *use_library_3*, but the system recognises *unclassified_task_3* as the *winning* behaviour (see Fig. 5, seq 11). This is because the person changes direction suddenly just before leaving the environment. A re-examination of the diagram in Fig. 5 (seq 11) does show that $p_{use\_library\_3}$ is approximately 0.4, and is significantly better than p_{print_3}. These results show that the system is able to robustly recognise people activities.

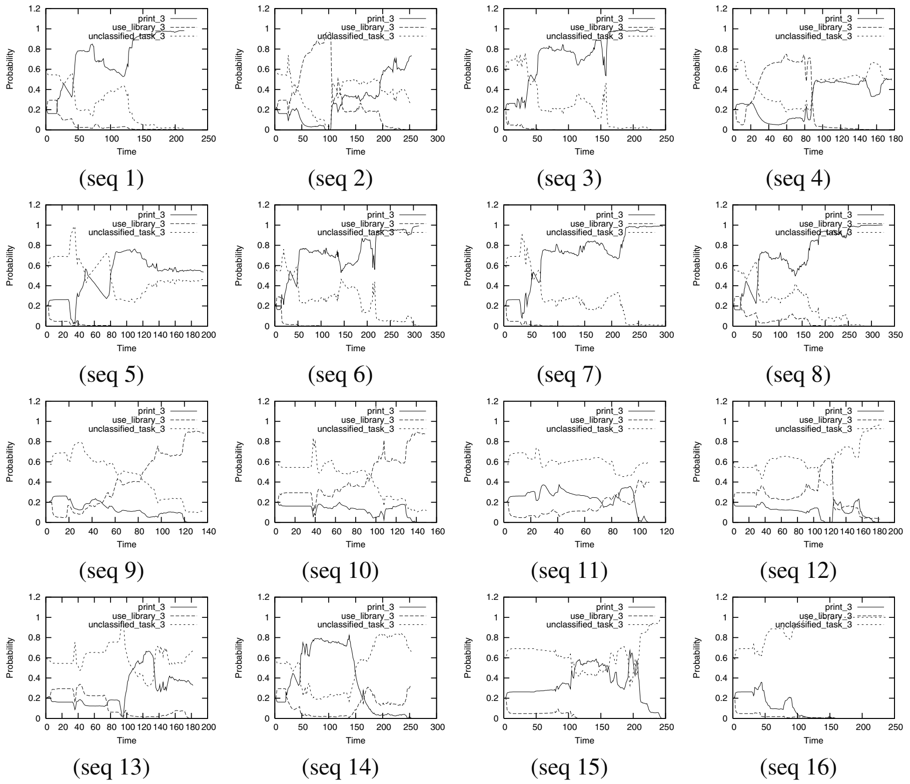


Fig. 5. Querying the top level behaviour in the 16 video sequences.

7 Conclusion

We have presented the techniques that can be used to learn camera observation models and human movement models. These techniques are used in a surveillance system for recognising and monitoring high-level human behaviours from multi-camera surveillance data. The system can query the high-level behaviours executed by a person over time. Behaviour classification results in a real environment demonstrate the ability of the system to provide real-time monitoring of high level behaviours in complex spatial environments with large state spaces.

Table 1. The results of recognising the top level behaviour in the 16 video sequences.

Seq.	*Winning* behaviour	Time periods that the *winning* behaviour has the highest probability	Compared with groundtruth
1	*print_3*	26-27, 42-213=END	correct
2	*print_3*	194-253=END	correct
3	*print_3*	38-45, 47-231=END	correct
4	*print_3*	89-93, 145-146, 170-170, 172-172, 174-176=END	correct
5	*print_3*	48-51, 79-194=END	correct
6	*print_3*	30-36, 48-311=END	correct
7	*print_3*	51-51, 53-57, 68-294=END	correct
8	*print_3*	54-323=END	correct
9	*use_library_3*	82-136=END	correct
10	*use_library_3*	95-149=END	correct
11	*unclassified_task_3*	1-107=END	wrong
12	*unclassified_task_3*	1-106, 123-181=END	correct
13	*unclassified_task_3*	1-106, 133-181=END	correct
14	*unclassified_task_3*	1-46, 149-252=END	correct
15	*unclassified_task_3*	1-108, 158-159, 162-191, 201-203, 207-246=END	correct
16	*unclassified_task_3*	1-239=END	correct

References

1. Nguyen, N.T., Bui, H.H., Venkatesh, S., West, G.: Recognising and monitoring high-level behaviours in complex spatial environments. In: IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin (2003) 620–625
2. Oliver, N., Horvitz, E., Garg, A.: Layered representations for human activity recognition. In: Proceedings of the Fourth IEEE International Conference on Multimodal Interfaces. (2002) 3–8
3. Ivanov, Y., Bobick, A.: Recognition of visual activities and interactions by stochastic parsing. IEEE Transactions on Pattern Recognition and Machine Intelligence **22** (2000) 852–872
4. Galata, A., Johnson, N., Hogg, D.: Learning variable length Markov models of behaviour. Intenational Journal of Computer Vision and Image Understanding **81** (2001) 398–413
5. Hoey, J.: Hierarchical unsupervised learning of event categories. In: IEEE Workshop on Detection and Recognition of Events in Video, Vancouver, Canada (2001) 99–106
6. Bui, H.H.: A general model for online probabilistic plan recognition. In: The 18th International Joint Conference on Artificial Intelligence (IJCAI 2003), Acapulco, Mexico (2003)
7. Pynadath, D.V., Wellman, M.P.: Generalized queries on probabilistic context-free grammars. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 65–77
8. Bui, H.H., Venkatesh, S., West, G.: Policy recognition in the Abstract Hidden Markov Model. Journal of Artificial Intelligence Research **17** (2002) 451–499
9. Doucet, A., de Freitas, N., Murphy, K., Russell, S.: Rao-Blackwellised particle filtering for dynamic Bayesian networks. In: Proceedings of the Sixteenth Annual Conference on Uncertainty in Artificial Intelligence, Stanford, California, AAAI Press (2000) 176–183
10. Bui, H.H.: Efficient approximate inference for online probabilistic plan recognition. In: AAAI Fall Symposium on Intent Inference for Users, Teams and Adversaries, Falmouth, Massachusetts (2002)

A Comparison of Least Squares and Spectral Methods for Attributed Graph Matching

JianfengLu[1,2], Terry Caelli[1], and Jingyu Yang[2]

[1] Department of Computing Science, University of Alberta, Edmonton
Alberta, Canada, T6G 2E8
lujf@mail.njust.edu.cn, tcaelli@ualberta.ca
[2] Department of Computer Science, Nanjing University of Science & Technology
Nanjing, P.R. China, 210094

Abstract. In this paper, least squares and spectral methods for attributed graph matching are compared. For the least squares method, complete graphs and decomposed graph models are considered in conjunction with the least squares approximations to optimal permutation matrices. We have used a version of Umeyama's spectral method for comparison purposes. Results clearly demonstrate how both these methods are affected by additive noise but that, in general, least squares methods are superior.

1 Introduction

Graphs are a very powerful tool for many practical problems including pattern recognition and computer vision [1-8]. Graph matching algorithms can be divided into search-based methods and methods based on optimization [3]. Least squares (LSM) and spectral methods (SM) are two recent optimization methods for attributed graph matching. However, to this date, there has not been a direct comparison of their relative benefits and deficits.

The method proposed by van Wyk et al.[1] is a more recent formulation of the least squares method. The essence of their method is to construct an interpolation function to approximate the best fitting permutation matrix (Interpolator-Based Kronecker Product Graph Matching (IBKPGM)) that maps vertices of one graph into the other. However, the method is less than ideal under additive noise conditions. Consequently it makes sense to integrate this method with graph models that are known to scale well with noise, in particular, those based on decomposition models. El-Sonbaty and Ismail [2] proposed a decomposition-based method where isomorphisms are established between sets of subgraphs – to be explored within the context of the LSM in this paper. Grewe and Kak[11] also adopted the similar model to extract local feature set for 3D object recognition.

An alternative to least squares methods is the spectral methods that have also received recent attention [4-8]. The main advantage of SMs is their low complexity, but their disadvantage is, again, their sensitivity to noise. Umeyama [4] proposed an eigenvalue decomposition (EVD) approach for weighted graph matching. For two weighted graphs, eigenvalue decompositions were computed after which the product of the two eigenvector matrices were used as inputs to a linear assignment algorithm to find the optimal isomorphism. Scott et al.[5] adopted a method that maximized the inner product of two affinity matrices to find the correspondence between two images. This approach was extended by Shapiro et al.[6]. For two image feature sets, the singular value decomposition (SVD) of their feature distance matrix was computed and the distances between each pair of eigenvectors were used to determine vertex correspondences. Kosinov and Caelli[8] proposed an eigen-subspace projection clustering method for inexact graph matching. The main idea is that for a pair of graphs (having the same or different number of vertices) their eigenvalue/vector decompositions are obtained and the vertices are then projected into a normalized eigenspectral subspace that adjusts for differences in vertex numbers and spectral energies. They then perform clustering in this normalized subspace to determine correspondences. However, extensions of SMs to attributed graph isomorphisms have not been reported in the literature.

In this paper, these two algebraic methods (SM, LSM) for attributed graph matching are compared. In addition, we propose an extension to the IBKPGM method based on using a piece-wise graph decomposition approach (the GDKPGM method). The layout of this paper is as follows. In Section 2, some preliminaries of the IBKPGM method are introduced. Extensions to current algorithms are proposed in Section 3. Some issues related to the attributed graph spectral method are discussed in Section 4, experimental results and comparisons are presented in Section 5. Section 6 draws conclusions about such algebraic methods.

2 The IBKPGM Method

The focus of IBKPGM is a matching model for attributed graphs where an input graph, *G*:

$$G = (V, E, \{A_i\}_{i=1}^{r}, \{B_j\}_{j=1}^{s}) \tag{1}$$

is matched to a reference graph, G', where

$$G' = (V', E', \{A_i'\}_{i=1}^{r}, \{B_j'\}_{j=1}^{s}) \tag{2}$$

for $A_i \in R^{n \times n}$, $B_j \in R^{n \times 1}$, $A'_i \in R^{n' \times n'}$ and $B'_j \in R^{n' \times 1}$, representing the edge attribute matrices and vertex attribute vectors, respectively. The reference and input graphs each have *r* edge attributes and *s* vertex attributes while the number of vertices in *G'* and *G* are $n' := |V'|$ and $n := |V|$, respectively (see [1] for details).

Graph matching is then defined in terms of determining the permutation matrix, *P*, which minimizes

$$\min_P \left(\sum_{i=1}^{r} \left\| A_i - P A_i' P^T \right\|^q + \sum_{j=1}^{s} \left\| B_j - P B_j' \right\|^q \right) \tag{3}$$

where $\|\cdot\|$ denotes the Frobinius matrix norm with typical values for *q* of *1* or *2* and where P^T corresponds to the transpose of *P*. Although some proposed algorithms attempt to solve Eqn. (3) directly, Van Wyk et al [1] used the Kronecker Product form of Eqn. (3) to allow for a method that approximates *P* using an Interpolator-Based Kronecker Product formulation.

In this method, each edge attribute matrix of size of $n*n$ is transformed into a n^2*1 column vector and each vertex attribute vector is transformed into a diagonal matrix first, and then, also, a vector. Thus, Eqn. (3) is rewritten as

$$\min_{\Phi}\left(\sum_{i=1}^{r+s}\left\|vecA_i - \Phi vecA_i'\right\|^q\right) \tag{4}$$

where $\Phi = P \otimes P$ and $\otimes$ denotes the Kronecker product of matrices, *vecZ* is an operator that transforms a matrix *Z* into a vector. If the least squares method is applied to Eqn. (4) directly, its complexity is high, requiring the calculation of the pseudo-inverse of a $n^2*(r+s)$ matrix. To overcome this, van Wyk et. al.[1] used the interpolator method to derive an approximate optimal solution to Eqn. (4).

That is, if Eqn. (4) is viewed as the mapping $A_k = F(A_k')$, where $k = 1,...,r+s$, then the mapping of vertex i in A' to A can be approximated using an interpolator-based method described by

$$\tilde{F}_i(\cdot) = \sum_{l=1}^{r+s} C_{l|i} K(vecA_l', \cdot) \tag{5}$$

where $K(\cdot,\cdot)$ is the reproducing kernel. A common choice is a linear kernel:

$$K(vecX, vecY) = (vecX)^T vecY \tag{6}$$

In this case the interpolator coefficients, $C_{l|i}$, can be solved first by letting

$$A = CG \tag{7}$$

where

$$C = (C_1, C_2, ..., C_{r+s}) \tag{8}$$

for

$$C_l = (C_{l|i}) \tag{9}$$

$$A = (vecA_1, vecA_2, ..., vecA_{r+s}) \tag{10}$$

$$G = (G_{lk}) \tag{11}$$

and

$$G_{lk} = K(vecA_l', vecA_k') \tag{12}$$

Based on the estimated interpolator coefficients, $C_{l|i}$, the elements of Φ can be calculated by (see [1] for more details)

$$\hat{\Phi} = (\hat{\Phi}_1^T, ..., \hat{\Phi}_{n^2}^T)^T \text{ by } \hat{\Phi}_i^T = \left(\sum_{l=1}^{r+s} C_{l|i} vecA_l'\right)^T \text{, where, } i = 1,...,n^2 \tag{13}$$

Since, $\Phi = P \otimes P$, an approximation to the permutation matrix, P, namely, $\overline{P}$, can be derived from $\hat{\Phi}$:

$$\overline{P}_{ij} = \frac{\sum_{kl} \hat{\Phi}_{kl}}{n}\text{; where } i = 1,...,n\text{ ; } j = 1,..,n'\text{;}$$
$$k = (i-1)n+1,...,(i-1)n+n\text{ ; } l = (j-1)n'+1,...,(j-1)n'+n' \tag{14}$$

van Wyk et. al. [1] have shown that if an optimal assignment method is adopted, $\left\| P - \overline{P} \right\|_F$ is minimum in the mean-square-error sense, where F is the Frobenius norm (F-norm). Consequently, the permutation matrix, P , can be derived from $\overline{P}$ based on an assignment algorithm. Albeit, this method still lacks robustness to noise. For this reason we explore how to integrate the core concept of IBKPGM but over subgraphs of the graphs using the following simple decomposition model.

3 Graph Decompositions and GDKPGM Matching Algorithm

Here we have used the decomposition model proposed by El-Sonbaty and Ismail [2] where, for each vertex, *v*, a subgraph is defined consisting of all vertices that are directly connected (a path length of *1*) to *v*. Fig. 1 shows such a simple decomposition (trees rooted by each vertex, see [2] for details).

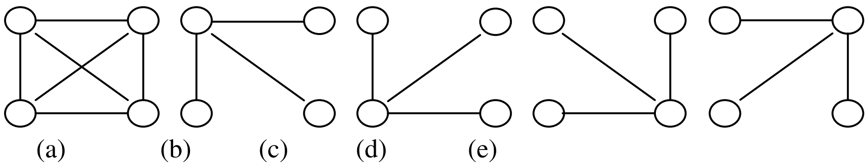


Fig. 1. Original graph (a) and all subgraphs (b)-(e) after decomposition.

Suppose that after decomposition the set of reference graph subgraphs consists of $\{SGR_j\}, j = 1,.., n'$ where n' corresponds to the number of vertices in the reference graph and SGR_j to the subgraph corresponding to vertex *j*. Similarly, $\{SGI_i\}, i = 1,.., n$, corresponds to the subgraphs in the comparison (input) graph, having *n* vertices.

3.1 Algorithm

1. Decompose both the input and reference graphs - as above.
2. For each pair of SGR_j and SGI_i, compute replicator coefficients, C_{ij} for $A_i = C_{ij} G_j$ where A_i represents the attribute matrix of subgraph SGI_i by Eqn. (8) of size, $n^2 * (r + s)$; G_j is derived from SGR_j by Eqn. (11).
3. Compute $\overline{P}$ from Eqns. (13) and (14) and use the matrix F-norm to obtain F_{ij}, $F_{ij} = \left\| A_i - PA'_j P^T \right\|_F$ - the "cost" of matching the two subgraphs(vertices).
4. Use an optimal assignment method to find optimal correspondence between subgraph SGR_j and SGI_i .

3.2 Complexity Analysis

Discussion is limited to undirected, fully connected graphs and, for analysis convenience, suppose the two graphs have the same number vertices, namely, $n = n'$. The complexity of the IBKPGM method is $O(n^4)$ being dominated by the extraction of the Kronecker match matrix when $n >> (r+s)$ from Eqns. (5)-(14). If we directly use their method, the complexity is high, $O(n^6)$. However, the proposed algorithm can reduce complexity. Based on the decomposition model, the edge attribute matrix of each subgraph has the following form: all diagonal elements are zero, and there is only one non-zero column and row - namely, the total number of non-zero elements is $2n-2$. If this matrix is transformed into a column vector, there are $2n-2$ non-zero rows and the vertex attribute matrix has only n non-zero diagonal elements. Hence, for each subgraph all attribute matrices are transformed into vectors and rearranged into an $n^2 * (r+s)$ matrix by Eqn. (8) having, at most, $3n-2$ non-zero rows, where $2n-2$ rows are from the edge attribute matrix and n rows from vertex attribute matrix.

According to Eqn. (12), the size of G_j is $(r+s)*(r+s)$, and the matrix A_i in Eqn. (12) has, at most, $3n-2$ non-zero rows; therefore C_{ij} has at most $3n-2$ non-zero rows. For Eqn. (13), since C and A'_i have, at most, $3n-2$ non-zero columns and rows their product has, at most, $3n-2$ non-zero elements, so the actual complexity is $O(n^2)$ for Eqn. (13). Thus, the total non-zero elements in Φ should be, at most, $n^2*(3n-2)$, and the complexity of Eqn. (14) should be $O(n^3)$.

van Wyk's method [1] uses the Kuhn–Munkres optimal assignment algorithm to determine the final optimal correspondence between vertices. It has a complexity of $O(n^3)$ [9]. However, Yamada [10] designed an optimal assignment algorithm whose complexity is $O(n^2)$ which can be used to replace the Kuhn–Munkres method. Using this, then, the complexity of each loop of the GDKPGM algorithm is $O(n^3)$. Thus, the total complexity of this method is $O(n^5)$. It is one order higher than IBKPGM. For El-Sonbaty and Ismail's method [2], its complexity is, best case, $O(n^3)$; worst case, $O(n^5)$.

4 Spectral Methods for Graph Matching (SMGM)

Spectral methods typically consist of the following simple steps:

1. For attributed graphs, compute their vertex-wise covariance matrix (column-wise: $A^T A$). For simple non-attributed and undirected graphs, this corresponds to their adjacency matrices.
2. For the above matrices, eigenvalue/vector decomposition is performed.
3. Normalize common subspace bases (including sign correction or sign-independent comparisons) are computed for projecting vertices from both graphs.

4. Correspondence between the two graphs can be derived by clustering vertices in normalized common subspace (see Kosinov and Caelli [8]) or by solving an assignment problem.

Umeyama's method [4] is a representative example of this latter approach for the restricted case of weighted graphs. Here we have generalized his method for fully attributed vertices and edges. This generalization results in the following algorithm.

1. For both input and reference graph, the vertex (n) and edge attribute values (r, s respectively) are defined as a single matrix where each vertex attribute vector is formatted as a diagonal submatrix, edge attributes as off-diagonal matrices, all being concatenated as a single $((r+s)\times n)\times n$ matrix, A.
2. The vertex-based covariance matrix, $A^T A$, is computed.
3. Eigenvalue/vector decomposition is performed on $A^T A$ and a subspace dimension selected in terms of the first w eigenvalues/vectors.
4. Correlations between the eigenvectors are computed as a measure of "distance" ("cost") of matching vertices in this subspace.

For eigenvectors of V1 and V2, which are from input graph and reference graph respectively, the product of $|V2| * |V1^{-1}|$ is calculated to get a cost matrix.

5. The best set of correspondences is selected by solving an optimal assignment problem based on the cost matrix at step 4.

We have used this method for comparison with the LSM approaches.

5 Experiments

5.1 Random Graphs

We have compared the results of the GDKPGM, IBKPGM and SMGM methods using full-connected random graphs as used by Van Wyk et al.[1]. The parameters n', n, r, s were fixed for each iteration. A reference graph was then randomly generated with all attribute values distributed between 0 and 1. An $n \times n'$ permutation matrix, P, was randomly generated to permute the row and columns. An independently generated noise matrix, having uniformly generated values within the [$-\varepsilon$, ε; $0< \varepsilon <1$] interval was then added to each vertex and edge attribute. The task was then to match the original graph with the graph corresponding to the permuted, perturbed vertices and corresponding edges. The estimated probability of a correct vertex-vertex correspondence was calculated as a function of ε for every 100 experiments.

To evaluate performance we focused on the following questions.

1) Does the decomposition method improve performance
2) Is LSM always superior to SM?
3) How do the parameters, $\{ n', n, r, s, \varepsilon \}$, influence the performance of above methods?

The following experiments were conducted to explore these issues. First, the attribute parameters, (r,s), were set to $(3,3)$, and (n', n) to (10,10), (30,30), (50,50) and (100,100). Second, $(r,s) = (1,0)$, and (n',n) were set to (10,10), (30,30), (50,50): a simple weighted graph. Third, $(r,s) = (1,1)$, and (n',n) were set to (10,10), (30,30), (50,50). Fourth, $(r,s) = (2,0)$, and (n',n) were set to (10,10), (30,30), (50,50). Fig. 2 shows results for the first experiment, and Tables 1-3 for the remaining.

5.2 Results

It can be seen in Fig. 2 that performance of the GDKPGM method is much better than that of IBKPGM. However, when ε is larger than 0.6, performance of GDKPGM also significantly decreases along with IBKPGM. Also, this decrease in performance increases with *n*, the number of vertices in the reference graph although, in all cases, GDKPGM performance is superior to IBKPGM. In this case, however, SM is inferior to LSM.

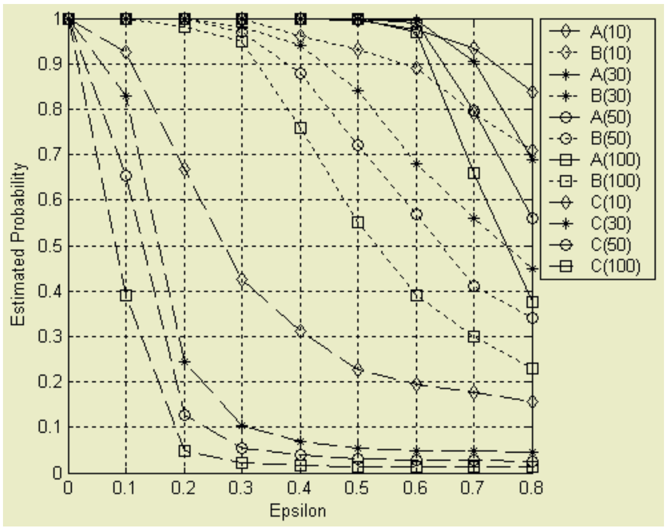


Fig. 2. Performance comparison of 3 methods, r=s=3, (n',n)=(10,10), (30,30), (50,50), (100,100). (r,s) correspond to numbers of edge and vertex attributes; (n',n) : to number of vertices in reference and input graphs. A, B, C represent the GDKPGM, IBKPGM and SMGM methods, respectively.

Table 1 shows that, for a weighted graph (*r*=1,*s*=0), when the noise level is not large ($\varepsilon < 0.3$), the SM is superior to LSM. When $\varepsilon = 0.1$ the performance of SM is much better than that of LSM. But as the noise increases performance of SM decreases significantly, and when $\varepsilon > 0.4$ it is quite inferior to the LSM methods.

Table 2 compares performance to that shown in Table 1 with an additional vertex attribute and demonstrates how additional constraints improved the performance of the LSM method but not the SM method, except for the low noise conditions: $\varepsilon = 0.1$. The same conclusion can be draw from results shown in Table 3 when compared to Table 1. The difference in the experimental conditions for Table 2 and 3, (both having 2 attributes) was that, for Table 2, there was one edge and one vertex attribute but there were only two edge attributes corresponding to Table 3. Overall, performance of all methods was better in Table 2 than that shown in Table 3.

Based on above experimental result, some conclusion can be drawn.

1) For low noise conditions, and with simple adjacency matrices or weighted graphs, SM is superior to LSM.
2) For attributed graph matching (r,s > 1) the LSM is superior to SM. Further, the decomposition method used in conjunction with LSM consistently improves performance under all conditions.
3) Increasing attributes of vertices is more helpful in improving performance than increasing the number of edge attributes.
4) When the additive noise is large all three methods have poor performance, suggesting the non-linear method [3], should be used.

Table 1. Performance comparison of LSM and SM, $(r,s) = (1,0)$, (n',n) =(10,10), (30,30), (50,50). See Fig. 2 for definitions of terms.

ε	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
LSM(10)	63.7	49.8	35.5	29.0	24.5	22.5	19.9	18.2
SM(10)	84.5	55.5	42.5	23.6	19.1	17.7	14.2	12.4
LSM(30)	33.7	18.5	13.6	10.8	9.6	8.3	7.9	6.3
SM(30)	72.8	37.2	15.9	8.9	5.9	5.6	5.3	4.6
LSM(50)	21.6	12.1	8.7	6.9	6.2	5.1	4.8	4.3
SM(50)	55.8	25.5	7.9	4.9	3.8	2.9	2.8	2.5

Table 2. Performance comparison of LSM and SM, $(r,s) = (1,1)$, (n',n) =(10,10), (30,30), (50,50). See Fig. 2 for definitions of terms.

ε	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
LSM(10)	94.9	84.2	69.9	58.1	50.0	41.5	34.5	33.5
SM(10)	93.0	68.5	48.2	30.9	23.8	21.4	19.6	16.2
LSM(30)	85.8	60.6	39.9	27.5	20.7	17.9	14.4	11.1
SM(30)	84.0	23.9	14.1	8.0	6.6	6.2	5.0	4.3
LSM(50)	77.7	45.4	28.1	19.1	14.1	11.1	8.9	8.3
SM(50)	69.8	13.3	8.6	5.0	3.5	3.1	2.8	2.0

Table 3. Performance comparison of LSM and SM, $(r,s) = (2,0)$, (n',n) =(10,10), (30,30), (50,50). (See Fig. 2 for definitions of terms).

ε	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
LSM(10)	90.0	74.1	56.1	48.0	39.1	30.8	26.1	25.1
SM(10)	91.3	56.4	31.4	25.0	18.1	16.0	15.0	13.5
LSM(30)	76.4	46.2	32.5	22.5	17.4	14.6	12.0	10.8
SM(30)	82.1	20.9	9.9	6.2	4.8	4.4	3.8	3.5
LSM(50)	66.3	33.9	21.9	14.5	11.9	9.3	7.4	6.4
SM(50)	65.4	11.9	4.7	3.4	3.04	2.8	2.7	2.1

6 Conclusions

In this paper, two main linear algebraic methods for attributed graph matching, LSM and SM have been compared along with a new graph decomposition model, combined with LSM, for attributed graph matching. For most cases, the LSM is superior to SM most probably due to the fact that the LSM uses more information than the SM. However, when there are few constraints such as in attributed or weighted graphs, LSM performance is inferior to the SM. For this reason SM would be the preferred algorithm for matching graphs defined only be their adjacency matrices – the more commonly used graph data model for recent SM methods. But, the SM is quite sensitive to noise. However, when there is little noise, its performance is quite good. In so far as LSM methods are concerned when a graph decomposition model is combined with the LSM, generally speaking, performance along all dimensions improves. However, when noise and/or vertices numbers are large even LSM performance is not satisfactory – suggesting that non-linear optimization, search methods, may provide better solutions.

References

1. van Wyk, B.J., Van Wyk, M. A.: Kronecker product graph matching. Pattern Recognition 36 9(2003), 2019-2030.
2. El-Sonbaty, Y., Ismail, M.A.: A new algorithm for subgraph optimal isomorphism. Pattern Recognition 31 2 (1998), 205–218
3. Gold, S., Rangarajan, A.: A graduated assignment algorithm for graph matching. IEEE TPAMI. 18 4 (1996), 377–388
4. Umeyama, S.: An Eigen Decomposition Approach to Weighted Graph Matching Problems. IEEE TPAMI 10 5 (1988), 695-703
5. Scott, G. L., Longuet-Higgins, H.C.: An algorithm for assoictaing the features for two patterns. Proc Roy Soc Lond, Vol B244 (1991), 21-26
6. Shapiro, L., Brady, J..M.: Feature-based Correspondence:an eigenvector approach, Image & Vision Computing , 10 5 (1992), 283-288
7. Luo, B., Wilson, R.C., Hancock, E.R.: Spectral embedding of graphs. Pattern Recognition 36 9 (2003), 2213–2230.
8. Kosinov, S. and Caelli, T.: Inexact Multisubgraph matching using Graph Eigenspace and Clustering Models. 9th International Workshop on Structural and Syntactic Pattern: SSPR2002. In: Caelli, T., Amin, A., Duin, R., Kamel, M., de Ritter, D. (Eds.), Structural, Syntactic and Statistical Pattern Recognition, Lecture Notes in Computer Science, LNCS 2396, Springer, 133-142.
9. Clark, J., Holton, D.A.: A First Look at Graph Theory. World Scientific Publishing, Singapore (1991).
10. Yamada, S., Hasai, T.: An efficient algorithm for the linear assignment problem. Elect. Comm. Japan 73 12 (1990), 28–36 Part 3.
11. Grewe, L., Kak, A.C.: Interactive learning of a multiple-attribute hash table classifier for fast object recognition. CVIU 61 3 (1995), 387-416.

An Auction Algorithm for Graph-Based Contextual Correspondence Matching

Barend J. van Wyk, Michaël A. van Wyk, and Guillaume Noel

French South-African Technical Institute in Electronics
at the Tshwane University of Technology
Staatsartillerie Road, Pretoria, South Africa
{ben.van.wyk,guillaume.noel,mavw}@fsatie.ac.za

Abstract. The *Auction Graph Matching* (AUGM) algorithm is presented. This algorithm is based on a novel joint probabilistic framework that transforms the graph matching problem into a linear assignment problem which is efficiently solved by the Bertsekas auction algorithm. A salient feature of this single-pass auction-based approach is that the inferred match probabilities are not only constrained over all objects in the reference image, but are also constrained over all objects in the input image.

1 Introduction

Representing the structural descriptions of objects by weighted graphs, reduces the problem of contextual correspondence matching to finding error-correcting graph or sub-graph isomorphisms, also referred to as the *Graph Matching* (GM). As pointed out in Refs. [1], [2] and [3] the graph matching problem is closely related to the *Quadratic Assignment Problem* which can be solved using a variety of neural, annealing, graduated assignment and other iterative methods. In general the *Quadratic Assignment Problem* is more difficult to solve than the *Linear Assignment Problem.* Several breakthroughs to efficiently solve the *Linear Assignment Problem* such as those detailed in Refs. [4] and [5] indicate that there is much to gain if the GM problem can be transformed into a *Linear Assignment Problem.* A method for achieving this transformation is presented in this paper.

Although *Quadratic Assignment* approaches are not in general directly interpretable using Bayesian frameworks such as those detailed in Refs. [6]–[15], the Auction Graph Matching (AUGM) algorithm is built on a Bayesian foundation. A *Joint Probabilistic Framework* is derived in the sequel that leads to a *Linear Assignment Problem* which can be efficiently solved by the Bertsekas auction algorithm [4]. The *Joint Probabilistic Framework* differs significantly from previous probabilistic approaches for contextual correspondence and graph matching, detailed in Refs. [6]–[15], in the following aspects: *(1)* Instead of directly inferring $P(\theta_i = \overline{\theta}_k)$, the probability of associating a vertex in the input graph with a vertex in the reference graph, our main focus is on the inference of the joint probabilities $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ from which $P(\theta_i = \overline{\theta}_k)$ are inferred. *(2)* Similar to the work of Kittler *et al.* [8], the AUGM is a single-pass technique. *(3)* Conventional probabilistic methods only constrain the probability of a vertex in the input graph being associated with a vertex in the reference graph over all the vertices in the reference graph. To further minimize the possibility of false matches the AUGM algorithm *also* constrains the probability of a vertex in the input graph being associated with a vertex in the reference graph, over all the vertices in the *input* graph. *(4)* The AUGM algorithm does not rely on the use of compatibility functions specified in terms of the face-units of the graph under match, sparse graph structures or Bayesian edit distances based on these notions.

2 Notation

Suppose an input graphs has n vertices represented by

$$\Omega = \{\theta_1, ..., \theta_n\} .$$

The objective is to calculate the probability of a vertex θ_i in the input graph, being associated with a vertex $\overline{\theta}_k$ in a reference graph having $\bar{n}$ vertices, represented by

$$\overline{\Omega} = \left\{\overline{\theta}_1, ..., \overline{\theta}_{\bar{n}}\right\} .$$

In our framework it is assumed that $\bar{n} \geq n$ and that the probability values $P(\theta_i = \overline{\theta}_k)$ are constraint by

$$\sum_k P(\theta_i = \overline{\theta}_k) = 1, \tag{1}$$

and

$$\sum_i P(\theta_i = \overline{\theta}_k) = 1, \qquad n = \bar{n} \tag{2}$$

$$0 \leq \sum_i P(\theta_i = \overline{\theta}_k) \leq 1, \qquad \bar{n} \geq n. \tag{3}$$

If, in addition it is required that

$$P(\theta_i = \overline{\theta}_k) \in \{0; 1\} \tag{4}$$

then equations 1 to 4 represent the enforcement of two way assignment constraints. Previous Bayesian frameworks were in general not able to enforce constraint 2 or 3.

For each pair of vertices θ_i and θ_j, $i \neq j$, we assume there are s binary measurements corresponding to the attributes of the edges of the input graph:

$$A_{ij} = \left\{A_{ij}^{(1)}, ..., A_{ij}^{(s)}\right\}, \qquad i \neq j, \quad i, j = 1, ..., n.$$

3 Bayesian Reasoning Framework

As usual we will assume that the conditional probability density function

$$p(A_{ij}|\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \tag{5}$$

corresponds to the compatibility coefficients calculated between edges of the input and references graphs using *edge* attributes. See for example Kittler *et al.* [8] , Christmas *et al.* [6] or Faugeras and Price [16]. We will now investigate how Eq. 5 relates to the joint probabilities $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$. Our approach drastically differs from previous approaches in the sense that we do not try to estimate or calculate the probabilities $P(\theta_i = \overline{\theta}_k|A_{ij})$ directly. Instead our main computational focus is on the inference of the joint probabilities $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ from which all $P(\theta_i = \overline{\theta}_k)$ are estimated using a simple weighted summation process. It is important to note that to infer $P(\theta_i = \overline{\theta}_{k,})$, the probabilities of associating a vertex in the input graph with a vertex in the reference graph, our framework *only relies on edge attributes.* Self edges (self arcs) and vertex attributes are not considered. As a consequence $p(A_{ij}|\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ and $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ where $i = j$ or $k = l$ are not considered in our framework and the independence assumption, i.e. $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = P(\theta_i = \overline{\theta}_k)P(\theta_j = \overline{\theta}_l)$, holds. Observe that according to Bayes' theorem

$$P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l|A_{ij}) = \frac{p(A_{ij}|\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)}{\sum_{k,l} p(A_{ij}|\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)}, \tag{6}$$

where $i \neq j$ or $k \neq l$. Since all $p(A_{ij}|\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ are fixed (via some compatibility calculation) the only way to maximize the *a posteriori probability* $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l|A_{ij})$ is by adjusting the joint probabilities $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$, i.e.

$$P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l|A_{ij}) = \max_{P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)} \left[\frac{p(A_{ij}|\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)}{\sum_{k,l} p(A_{ij}|\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)}\right] \tag{7}$$

where $i \neq j$ or $k \neq l$ subject to the constraints associated with $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ given by

$$\sum_{k,l,i\neq j,k\neq l} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = 1, \tag{8}$$

$$0 \leq \sum_{i,j,i\neq j,k\neq l} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \leq 1, \tag{9}$$

$$0 \leq P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \leq 1. \tag{10}$$

Constraints 8 to 10 were derived using Eqs. 1 to 3 and the fact that $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = P(\theta_i = \overline{\theta}_k)P(\theta_j = \overline{\theta}_l)$. Finding the constraint joint probabilities

$P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ which maximize $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l | A_{ij})$ over all i, j can be formulated as the following constrained optimization problem

$$\min \sum_{i,j,k,l} \left(p(A_{ij} | \theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) - P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \right)^2 \tag{11}$$

with respect to $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ where $i \neq j$ or $k \neq l$, subject to Eqs. 8 to 10.

3.1 Joint Probability Inference

Since $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = P(\theta_i = \overline{\theta}_l, \theta_j = \overline{\theta}_k)$ when our graphs are undirected and both these joint probabilities will belong to the same row of an assignment matrix we cannot directly impose the constraint $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \in \{0; 1\}$. However, the fact that $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = P(\theta_i = \overline{\theta}_l, \theta_j = \overline{\theta}_k)$ when our graphs are undirected implies that not all the joint probabilities $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ need to enter Eq. 11, conveniently reducing the dimension of the problem. In fact we only need to consider the indices in the set

$$\{i, j, k, l\}_{j=i+1,...,n \quad l=k+1,...,\bar{n}} \tag{12}$$

where $i = 1, ..., n$ and $k = 1, ..., \bar{n}$ provided that all final values for $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ are halved after solving the lower dimensional problem. It also implies that we can now impose the binary constraint $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \in \{0; 1\}$ on the lower dimensional problem and efficiently solve it using an optimal assignment algorithm. Figure 1 details the time, averaged over a 1000 runs, to execute the Bertsekas auction optimal assignment algorithm [4] written in C on a Pentium IV platform running Windows XP. The generation time of random input matrices is included in our time calculations. As expected the time required per assignment calculation is dependent on the number of undirected edges in the input and reference graphs. For undirected fully-connected graphs of the type considered in this paper the number edges are given by $\eta = \sum_{i=1}^{n-1} (n - i)$ where n represents the number of vertices in the graphs.

Note that the cardinality of the subset $\{i, j\}_{i,j=1,...,n}$ is $\eta = \sum_{i=1}^{n-1} (n - i)$ and that the cardinality of the subset $\{k, l\}_{k,l=1,...,\overline{n}}$ is $\overline{\eta} = \sum_{k=1}^{\overline{n}-1} (\overline{n} - k)$. The dimension of the matrix passed to the Auction algorithm will therefore be $\eta \times \overline{\eta}$. Consequently the rows of the assignment matrix are indexed by $\alpha = 1, ..., \eta$ where α is related to indices i and j by $\alpha = (j - i) + \sum_{s=1}^{i} |U_s|$where $j < i$, U_s is defined as the set $\{i = s, j\}_{j=i+1,...,n}$ and $|\cdot|$ denotes cardinality. Similarly the columns of the assignment matrix are indexed by $\beta = 1, ..., \overline{n}$ where β is related to indices k and l by $\beta = (l - k) + \sum_{s=1}^{k} |U_s|$ where $l < k$ and U_s is defined as the set $\{k = s, l\}_{l=i+1,...,\overline{n}}$.

4 Auction Assignment Algorithm for Graph Matching

Similar to the auction algorithm detailed in Ref. [4] we associate with each column of our assignment matrix (joint probability matrix) a price, say ρ_β. For our application we started by setting all the prices to zero. Let $p_{\alpha\beta} := p(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l | A_{ij})$ where α and β are related to indices i, j, k, l as described in the previous section. A row α is defined as almost happy with a column β assigned to it if

$$p_{\alpha\beta_\alpha} - \rho_{\beta_\alpha} \geq \max_{\beta=1,\ldots,\overline{n}} \{p_{\alpha\beta} - \rho_\beta\} - \delta,$$

where δ is a slack variable.

The auction process then proceeds by selecting a row, α, which has not been assigned a column β_α or is not almost happy. This row finds a column β_α which offers maximal value, i.e.

$$\beta_\alpha \in \arg \max_{\beta=1,\ldots,\overline{n}} \{p_{\alpha\beta} - \rho_\beta\}.$$

Then

1. Row α exchanges its previous β_α (if it had one) with the row its new β_α was assigned to at the beginning of the round.
2. The price ρ_β associated with β_α is set to $\rho_\beta = \rho_\beta + \max_\beta \{p_{\alpha\beta} - \rho_\beta\} - \max_{\beta \neq \beta_\alpha} \{p_{\alpha\beta} - \rho_\beta\} + \delta$.

The process is repeated in a sequence of rounds until all rows α are almost happy. The *slack* value, δ determines how fast the algorithm will converge and how optimal the final answer will be. For all our experiments we have set δ to a small fixed value that will guarantee convergence to an optimal solution as described in [4]. When the procedure terminates the joint probability values

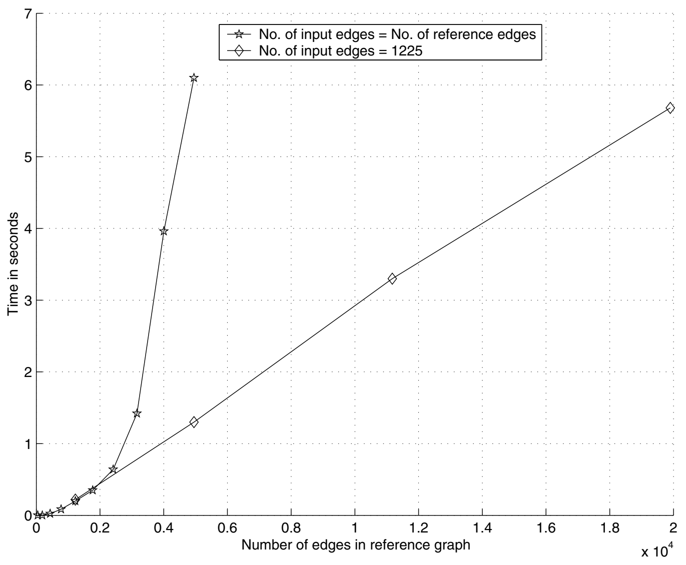


Fig. 1. Execution time of the Bertsekas auction routine.

$P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = P(\theta_j = \overline{\theta}_l, \theta_i = \overline{\theta}_k)$ corresponding to a row-column assignment are set to 0.5 (see previous section for reason why not set to one). The rest of the joint probability values are set to zero.

4.1 Marginal Probability Inference

After obtaining the constraint joint probabilities $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ the desired probabilities $P(\theta_i = \overline{\theta}_k)$ can be inferred using the following proposition:

Proposition 1. *The probabilities inferred by*

$$P(\theta_i = \overline{\theta}_k) = \frac{\sum_{j,l,j \neq i, l \neq k} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)}{n-1} \tag{13}$$

will satisfy the constraints given by Eqs. 1 to 3 if the joint probabilities $P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)$ satisfy the constraints given by Eqs. 8 to 10.

Proof: Observe that $\sum_{k,l,i \neq j, k \neq l} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = 1$ and that $\sum_{j,k,l,j \neq i, k \neq l} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) = n - 1$ which implies that $\sum_k \frac{\sum_{j,l,j \neq i, l \neq k} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)}{n-1} = 1$. Eq. 1 is therefore satisfied. Similarly $0 \leq \sum_{i,j,k \neq l, i \neq j} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \leq 1$ and $\sum_{l,i,j,l \neq k, i \neq j} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l) \leq \bar{n} - 1$ which implies that $0 \leq \sum_i \frac{\sum_{j,l,j \neq i, l \neq k} P(\theta_i = \overline{\theta}_k, \theta_j = \overline{\theta}_l)}{\bar{n}-1} \leq 1$. Eq. 3 is therefore satisfied since it is assumed that $\bar{n} \geq n$.

4.2 Final Assignment

Once all $P(\theta_i = \overline{\theta}_k)$ are inferred, the most appropriate $\overline{\theta}_k$ for a given θ_i is obtained by

$$\max_{\overline{\theta}_k} \{P\left(\theta_i = \overline{\theta}_k\right)\}_{k=1,\ldots,\bar{n}}. \tag{14}$$

It's easy to see that if all edge attributes are unique and no additive noise is present in the input graph that $\mathbf{P} := \left(P\left(\theta_i = \overline{\theta}_k\right)\right)$ will be an assignment matrix satisfying constraints 1 to 4.

5 Simulation Results

To test the performance of the AUGM algorithm dynamic random line matching experiments, similar to those proposed by Caelli and Caetano [18], were conducted. We prefer this type of experiment above the usual static applications such as stereo line matching, the matching of road networks or buildings as this allow us, through the addition of progressive noise and random line selection, to derive representative (application specific) performance curves for the algorithms included in our comparison. Although lines were used to represent the vertices of a graph, vertex features were not derived nor used. Binary relationships between lines were used as edge attributes. The differences in orientation between lines, line length ratios and distances between line midpoints were used. Refer to Li [19] for more information on the derivation of translation and orientation invariant line features.

For our experiments reference graphs having 50 vertices (derived from a randomly generated line image with line lengths uniformly distributed between 10 and 300 pixels) were used. For the first experiment input graphs were constructed by randomly rotating and translating all 50 lines of the reference line images. To test the robustness of the algorithms against line endpoint anomalies, uniform noise was added to the x and y coordinates of every line endpoint of the translated and rotated image. Noise values were obtained by multiplying a random variable – uniformly distributed on the interval $[-1/2, 1/2]$ – by the noise magnitude parameters given in figures 2 and 3. For the second experiment input graphs were constructed by randomly selecting 20 lines from the reference images before rotating, translating and adding noise.

We compared the performance of the AUGM algorithm to the performance of the non-iterative probabilistic method for contextual correspondence matching of Kittler, Petrou and Christmas (KPC) [8] since it is one of the most well-known single-pass methods available, and similar to the AUGM algorithm, has a probabilistic origin. The AUGM is also compared to Bayesian Successive Projection Graph Matching (BAYSPGM) algorithm described in [17] since it is a single-pass technique derived by the authors which preceded and inspired AUGM methodology. The estimated probability of correct vertex-vertex assignments are reported in figures 2 and 3. These results were calculated for a given value of noise magnitude by averaging the results from 200 trials. All algorithms were implemented using the Gaussian compatibility function described in Ref. [6] (with a diagonal covariance matrix) and the Faugeras-Price (FP) compatibility function described in [16]. Although both compatibility functions were implemented for all three algorithms only the best results obtained are reported. For the results reported in figure 2 the AUGM algorithm was implemented using a Gaussian compatibility function, and the KPC and BAYPGM algorithms using the FP compatibility function. For the results reported in figure 3 the KPC and AUGM algorithms were implemented using a Gaussian compatibility function and the BAYSPGM algorithm using the FP compatibility function. From our results we conclude that (for the application considered in this paper and the given compatibility functions) the AUGM algorithm performed significantly better than the BAYSPGM algorithm, and that it performed slightly better than the KPC algorithm for full-graph matching case.

6 Conclusion

A joint probabilistic framework was presented that transforms the GM problem into a *Linear Assignment Problem* that was solved in an efficient manner using the Bertsekas auction algorithm. From the derivation of the joint probabilistic framework it is clear that the formulation inherently makes provision for input graphs with missing vertices. A strategy for handling spurious edges in the input graphs and incorporating vertex features has been devised but due to

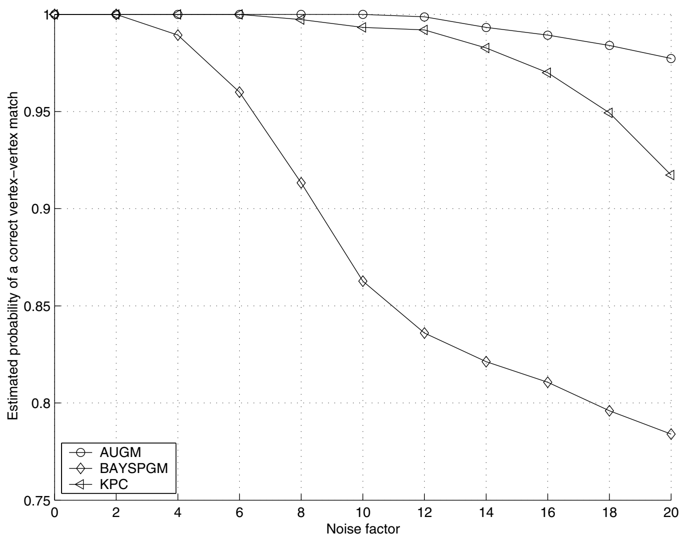


Fig. 2. Matching 50 translated and rotated input lines to 50 reference lines: Estimated probability of a correct vertex-vertex matching versus noise magnitude.

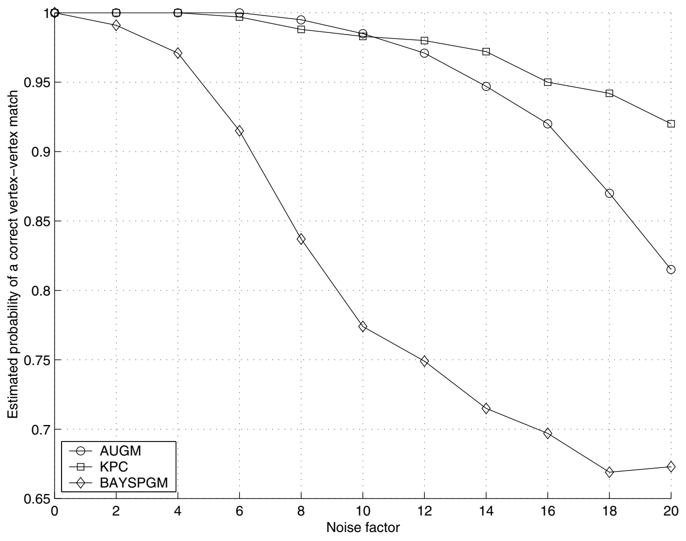


Fig. 3. Matching 20 translated and rotated input lines to 50 reference lines: Estimated probability of a correct vertex-vertex matching versus noise magnitude.

page constraints is beyond the scope of this paper. Work in progress include the derivation of a hybrid auction algorithm to directly solve the graph matching problem without first transforming it to a linear assignment problem.

References

1. Gold S. and Rangarajan A., A Graduated Assignment Algorithm for Graph Matching, IEEE Trans. Patt. Anal. Machine Intell, 18(4)(1996)377–388.
2. Simić P.D., Constrained Nets for Graph Matching and Other Quadratic Assignment Problems, Neural Computation, 3(1991)268–281.
3. Van Wyk B.J., Kronecker Product, Successive Projection, and Related Graph Matching Algorithms, PhD Thesis, University of the Witwatersrand, Johannesburg, 2003. [http://www.ee.wits.ac.za/comms/output/theses.htm]
4. Bertsekas D.P., The Auction Algorithm for Assignment and Other Network Flow Problems: A Tutorial, Interfaces, 20(1990) 133–149.
5. Jonker R. and Volgenant A., A Shortest Augmenting Path Algorithm for Dense and Sparse Linear Assignment Problems, Computing, 38(1987)325–340.
6. Christmas W.J., Kittler J. and Petrou M., Structural Matching in Computer Vision using Probabilistic Relaxation, IEEE Trans. Patt. Anal. Machine Intell., 17(8)(1995)749–764.
7. Cross A.D.J. and Hancock E.R., Graph Matching with a Dual Step EM Algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(11)(1998)1236–1253.
8. Kittler J., Petrou M. and Christmas W.J., A Non-iterative Probabilistic Method for Contextual Correspondence Matching, Pattern Recognition, 31(1998) 1455-1468.
9. Finch A.M., Wilson R.C. and Hancock R., Symbolic Matching with the EM Algorithm, Pattern Recognition, vol. 31(11)(1998)1777–1790.
10. Williams M.L., Wilson R.C. and Hancock E.R., Multiple Graph Matching with Bayesian Inference,Pattern Recognition Letters, 18(1997)1275–1281.
11. Cross A.D.J. and Hancock E.R., Graph Matching with a Dual Step EM Algorithm, IEEE Trans. Patt. Anal. Machine Intell., 20(11)(1998)1236–1253.
12. Wilson R.C. and Hancock E.R., A Bayesian Compatibility Model for Graph Matching, Pattern Recognition Letters, 17(1996)263–276.
13. Wilson R.C. and Hancock E.R., Structural Matching by Discrete Relaxation, IEEE Trans. Patt. Anal. Machine Intell., 19(6)(1997)634–648.
14. Finch A.M. , Wilson R.C. and Hancock E.R. , Matching Delauney Triangulations by Probabilistic Relaxation, Proceedings CAIP, (1995)351–358.
15. Meyers R.M., Wilson R.C. and Hancock E.R., Bayesian Graph Edit Distance,IEEE Trans. Patt. Anal. Machine Intell., 2(6)(2000)628–635.
16. Faugeras O.D. and Price K.E., Semantic Description of Aerial Images Using Stochastic Labeling, IEEE Transactions on Pattern Analysis and Machine Intelligence, 3(6)(1981)633-642.
17. Van Wyk B.J., Van Wyk M.A. and Botha J.J., A Matching Framework Based On Joint Probabilities, Proceedings of PRASA 2003, Langebaan, South Africa, 27-28 Nov. 2003, 125-130.
18. Caelli T. M. and Caetano T., Recent Developments in the Extraction and Matching of Image Structure and Syntax: From Relaxation to Junction Tree Models, Proceedings of PRASA 2003, Langebaan, South Africa, 27-28 Nov. 2003, 1-8.
19. Li S.Z., Matching: Invariant to Translations Rotations and Scale Changes, Pattern Recognition, 25(6)(1992)583-594.

Segmentation Graph Hierarchies*

Yll Haxhimusa and Walter Kropatsch

Pattern Recognition and Image Processing Group 183/2
Institute for Computer Aided Automation, Vienna University of Technology, Austria
{yll,krw}@prip.tuwien.ac.at

Abstract. The region's internal properties (color, texture, ...) help to identify them and their external relations (adjacency, inclusion, ...) are used to build groups of regions having a particular consistent meaning in a more abstract context. Low-level cue image segmentation in a bottom-up way, cannot and should not produce a complete final "good" segmentation. We present a hierarchical partitioning of images using a pairwise similarity function on a graph-based representation of an image. The aim of this paper is to build a minimum weight spanning tree (MST) of an image in order to find region borders quickly in a bottom-up 'stimulus-driven' way based on local differences in a specific feature.

1 Introduction

The authors in [16] asked: "How do we bridge the representational gap between image features and coarse model features?" They identify the 1-to-1 correspondence between salient image features (pixels, edges, corners,...) and salient model features (generalized cylinders, polyhedrons,...) as limiting assumption that makes prototypical or generic object recognition impossible. They suggested to bridge and not to eliminate the representational gap, and to focus efforts on: i) **region segmentation**, ii) **perceptual grouping**, and iii) image abstraction. In this paper, these goals are taken to consider multiresolution representations under the viewpoint of segmentation and grouping. The multiresolution is considered under the abstraction viewpoint in [18].

The union of regions forming the group is again a region with both internal and external properties and relations. The segmentation process results in "homogeneous" regions w.r.t the low-level cues using some similarity measures. Problems emerge because i) homogeneity of low-level cues will not map to the semantics [16] and ii) the degree of homogeneity of a region is in general quantified by threshold(s) for a given measure [6]. The low-level coherence of brightness, color, texture or motion attributes should be used to come up sequentially with hierarchical partitions [29]. It is important that a grouping method has following properties [5]: i) capture **perceptually important groupings** or regions, which reflect global aspects of the image, ii) be **highly efficient**, running in time linear in the number of pixels, and iii) creates **hierarchical partitions** [29].

* This paper has been supported by the Austrian Science Fund under grants P14445-MAT, P14662-INF and FSP-S9103-N04.

The aim is to build an MST of an image by combining the advantage of regular pyramids (logarithmic tapering) with the advantages of irregular graph pyramids (their purely local construction and shift invariance). The aim is reached by using the method for selecting contraction kernels to achieve logarithmic tapering, local construction and shift invariance [12]. Borůvka's algorithm [2] with dual graph contraction (DGC) algorithm [17] builds in a hierarchical way an MST (of the region) preserving the proper topology. After presenting related works and the pyramid representation, we recall the Borůvka's MST algorithm in Sec. 3. In Sec. 4 are given the definition of internal and external contrast, the merging criteria and building the hierarchy of an image. Sec. 5 reports results.

1.1 Related Works

A graph-theoretical clustering algorithm consists in searching for a certain combinatorial structure in the edge weighted graph, such as an MST [5, 8], a minimum cut [31, 29] and, the complete linkage clustering algorithm [19], reduces to a search for a complete subgraph i.e. the maximal clique [24]. Early graph-based methods [33] use fixed thresholds and local measures in finding a segmentation, i.e MST is computed. The segmentation criterion is to break the MST edges with the largest weight, which reflect the low-cost connection between two elements. To overcome the problem of fixed threshold, [30] attempts by normalizing the weight of an edge using the smallest weight incident on the vertices touching that edge. The methods in [5, 8] use an adaptive criterion that depends on local properties rather than global ones. The methods based on minimum cuts in a graph are designed to minimize the similarity between pixels that are being split [31, 29, 7]. A cut criterion in [31] is biased toward finding small components. The normalized cut criterion [29] is defined to avoid this problem, which takes into consideration self-similarity of regions. In contrast with the simple graph-based methods, such as breaking edges in the MST, cut-criterion methods capture the non-local properties of the image. It also produces a divisive hierarchical tree, the dendogram. However, they provide only a characterization of such cut rather than of final segmentation as provided in [5]. A minimum normalized cut approximation method [29] is computationally expensiv and the error in these approximation is not well understood. The clustering algorithms based on maximal clique work on unweighted graphs derived from the weighted graphs by means of some thresholding [15]. In [24] the concept of maximal clique is generalized to weighted graphs. Discrete replicator dynamics are used to find the maximal cliques, which is an instance of the relaxation labeling algorithm [27].

Gestalt grouping factors, such as proximity, similarity, continuity and symmetry, are encoded and combined in pairwise feature similarity measures [29, 26, 7, 32, 28]. Another method of segmentation is that of splitting and merging region based on how well the regions fulfill some criterion. Such methods [4, 25] use a measure of uniformity of a region. In contrast, [5, 8] uses a pairwise region comparison rather than applying a uniformity criterion to each individual region. Complex grouping phenomena can emerge from simple computation on these local cues [20].

Our method is related to the works in [5, 8] in the sense of pairwise comparison of region similarity. Rather than trying to have just one "good" segmentation [5], the method produces a stack of (dual) graphs (a graph pyramid), which down-projected on the base level will give a multi-level segmentation (a class of segmentation). The segmentation result of [5] is also included in our hierarchy.

2 Irregular Pyramid Representation

Hierarchical structures for description of the data for clustering purposes are studied very early in [19], or for image segmentation in [11]. In a regular image pyramid the number of pixels at any level k, is λ times higher than the number of pixels at the next reduced level $k+1$. The so called reduction factor λ is greater than one and it is the same for all levels k. If s denotes the number of pixels in an image I, the number of new levels on top of I amounts to $log_{\lambda}(s)$ (Figure 1a). Thus, the regular image pyramid is an efficient structure for fast grouping and access to image objects in top-down and bottom-up processes [14], because of the apriori known structure. However, the regular image pyramids are confined to globally defined sampling grids and lack shift invariance and have to be rejected as general-purpose segmentation algorithms [1]. These drawbacks are avoided by irregular image pyramids (adaptive pyramids) [23, 13], where the hierarchical structure (vertical network) of the pyramid is recursively built on the data. In [22] is shown that irregular pyramid can be used for segmentation.

In irregular pyramids, each level represents a partition of the pixel set into cells, i.e. connected subsets of pixels. The construction of an irregular image pyramid is iteratively local [21, 12]. This means that we use only local properties to build the hierarchy of the pyramid. On the base level (level 0) of an irregular image pyramid the cells represent single pixels and the neighborhood of the cells is defined by the 4 -connectivity of the pixels. A cell on level $k+1$ (parent) is a union of neighboring cells on level k (children). This union is controlled by so called contraction kernels (decimation parameters) [17]. Every parent computes its values independently of other cells on the same level. This implies that an image pyramid is built in $O[log(image\_diameter)]$ parallel steps. We represent the levels as dual pairs $(G_k, \overline{G_k})$ of plane primal graphs G_k and duals $\overline{G_k}$. The vertices of G_k represent the cells, and edges the neighborhood relations of the cells on level k. The edges of $\overline{G_k}$ represent the borders of the cells on level k and vertices meeting points of at least three edges from G_k. The sequence $(G_k, \overline{G_k})$, $0 \leq k \leq h$ is called (dual) **graph pyramid**. See [17] for the complete formalism. In order to simplify the paper presentation only graph G_k is used afterwards.

3 Minimum Weight Spanning Tree: Borůvka's Algorithm

Let $G_0(V, E, attr_v, attr_e)$ be a given undirected, connected, attributed plane graph consisting of the finite vertex set V and of the finite edge set E on the base level (level 0) of the pyramid, $attr_v : v \in V \rightarrow \mathbb{R}^+$ and $attr_e : e \in E \rightarrow \mathbb{R}^+$. Let each edge $e \in E$ be associated with a nonnegative **unique** real attribute

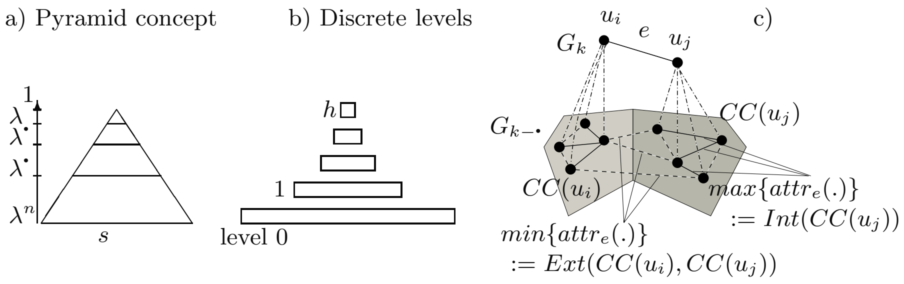


Fig. 1. (a,b) Multiresolution pyramid. (c) Internal and external contrast.

Algorithm 1 – Borůvka's Algorithm.

```
Input: Attributed graph $G(V, E)$.
1: $MST$ := empty edge list.
2: $\forall v \in V$ do { make a list of trees $L$ } .
3: while { there is more than one tree in $L$ } do
4:     each tree $T \in L$ finds the edge $e$ with the minimum weight which connects $T$ to
       $G \setminus T$ and add edge $e$ to $MST$.
5:     using edge $e$ merge pairs of trees in $L$.
Output: Minimum weight spanning tree - $MST$.
```

(weight). The problem is formulated as construction of an MST of G. A deterministic solution is proposed by Borůvka [2]. Borůvka algorithm is similar to Prim's algorithm but executed simulteonosly on the whole graph. We use Borůvka's algorithm to build MST in parallel [3]. The proof that Alg. 1 builds the MST is analogously with the MST Kruskal's proof [10].

Observation 1 *In 4^{th} step of the Algorithm 1, each tree $T \in L$ finds the edge with the minimal weight, and as trees become larger, the process of finding the edge with the minimal weight for each tree T takes longer.*

4 Hierarchy of Partitions

Hierarchies are a significant tool for image partitioning as they naturally mix with homogeneity criteria [11]. The goal is to find partitions $P_k := \{CC(u_1), CC(u_2), ..., CC(u_n)\}$ in k^{th} level of the pyramid such that these elements satisfy certain properties. We compare pairwise of neighboring vertices, i.e. partitions to check for similarities [5, 8]. In [5] a pairwise group merge criterion $Comp(CC(u_i), CC(u_j))$ is defined, that judges whether there is evidence for a boundary between two partitions $CC(u_i), CC(u_j) \in P_k$. This criterion measures the difference along the boundary of two components relative to a measure of differences of components' internal differences. This definition tries to encapsulate the intuitive notion of contrast: a contrasted zone is a region containing two connected components whose inner differences (**internal contrast**) are less than differences within it's context (**external contrast**).

4.1 Internal and Exernal Contrast

Let G_k be the graph on level k of the pyramid. Every vertex $u_i \in G_k$ is a representative of a **connected component** $CC(u_i)$ of the partition P_k. The equivalent contraction kernel (ECK) $N_{0,k}(u_i)$ [17] of a vertex $u_i \in G_k$, is a set of edges of the base level $e \in E_0$ that are contracted; i.e. applying ECK on the base level, one contracts the subgraph $G' \subseteq G_0$ onto the vertex u_i. The **internal contrast** of the $CC(u_i) \in P_k$ is the **largest dissimilarity** of component $CC(u_i)$, i.e. the largest edge weight of the $N_{0,k}(u_i)$ of vertex $u_i \in G_k$:

$$Int(CC(u_i)) := max\{attr_e(e), e \in N_{0,k}(u_i)\}. \qquad (1)$$

Let $u_i, u_j \in V_k$ be the end vertices of an edge $e \in E_k$. The **external contrast** between two components $CC(u_i), CC(u_j) \in P_k$ is the **smallest dissimilarity** between component $CC(u_i)$ and $CC(u_j)$ i.e. the smallest edge weight connecting $N_{0,k}(u_i)$ and $N_{0,k}(u_j)$ of vertices $u_i \in CC(u_i)$ and $u_j \in CC(u_j)$:

$$\begin{aligned} Ext(CC(u_i), CC(u_j)) := min\{&attr_e(e), \\ &e = (v, w) : v \in N_{0,k}(u_i) \wedge w \in N_{0,k}(u_j)\}. \end{aligned} \qquad (2)$$

In Fig. 1c an example of $Int(CC(u_j))$ and $Ext(CC(u_i), CC(u_j))$ is given. The $Int(CC(u_j))$ of the component $CC(u_j)$ is the *maximum* of weights of the solid line edges, whereas $Ext(CC(u_i), CC(u_j))$ is the *minimum* of weights of the dashed line edges (bridges) connecting component $CC(u_i)$ and $CC(u_j)$ on the base level G_0. By contracting the edges $N_{0,k}(u_j)$ one arrives to the vertex u_j. The pairwise merge criterion $Comp(CC(u_i), CC(u_j))$ between two connected components $CC(u_i)$ and $CC(u_j)$ can be defined as:

$$\begin{aligned} &Comp(CC(u_i), CC(u_j)) \\ &\quad := \begin{cases} \text{T if } Ext(CC(u_i), CC(u_j)) \le PInt(CC(u_i), CC(u_j)), \\ \text{F} \qquad\qquad\qquad \text{otherwise,} \end{cases} \\ &PInt(CC(u_i), CC(u_j)) := min(\, Int(CC(u_i)) + \tau(CC(u_i)), \\ &\qquad\qquad\qquad\qquad\qquad Int(CC(u_j)) + \tau(CC(u_j))\,)\,. \end{aligned} \qquad (3)$$

$PInt(\cdot, \cdot)$ is the minimum internal contrast difference between two components. For the merge criterion $Comp(\cdot, \cdot)$ to be false i.e. for the border to exist, the external contrast difference must be greater than the internal contrast differences. Any non-negative function $\tau(CC)$ of a single component CC can be used [5].

4.2 Construct Hierarchy of Partitions

A consequence of Obs. 1 is that a contraction of the edge e, which connects T and $G \backslash T$ in the 4^{th} step of Alg. 1 will speed up the search for minimum weight edges in Borůvka's algorithm. Since each tree (on level k) after edge contraction will be represented by a single vertex (in the level $k+1$), the edge with the minimum weight would be in a local neighborhood. The DGC algorithm [17] contracts edges and creates "super" vertices with father-son relations between vertices in

Algorithm 2 – Construct Hierarchy of Partitions.

```
Input: Attributed graph $G_{\bullet}$.
 1: $k := 0$
 2: repeat
 3:    for all { vertices $u \in G_k$ }  do
 4:       $E_{min}(u) := argmin\{attr_e(e) \,|\, e := (u,v) \in E_k$ or $e := (v,u) \in E_k\}$
 5:    for all { $e := (u_i, u_j) \in E_{min}$ with
       $Ext(CC(u_i), CC(u_j)) \leq PInt(CC(u_i), CC(u_j))$ }  do
 6:       include $e$ in contraction edges $N_{k,k^{\bullet}\,\bullet}$
 7:    contract graph $G_k$ with contraction kernels, $N_{k,k^{\bullet}\,\bullet}$: $G_{k^{\bullet}\,\bullet} = C[G_k, N_{k,k^{\bullet}\,\bullet}]$.
 8:    for all { $e_{k^{\bullet}\,\bullet} \in G_{k^{\bullet}\,\bullet}$ }  do
 9:       set edge attributes $attr_e(e_{k^{\bullet}\,\bullet}) := min\{attr_e(e_k) \,|\, e_{k^{\bullet}\,\bullet} := C[e_k, N_{k,k^{\bullet}\,\bullet}]\}$
10:    $k := k + 1$
11: until { $G_k = G_{k-\bullet}$ }
Output: A region adjacency graph ($RAG$) at each level of the pyramid.
```

subsequent levels, whereas Borůvka's algorithm is used to create son-son relation between vertices in the same level (horizontal relation).

The algorithm to build the hierarchy of partitions is shown in Alg. 2. Each vertex $u_i \in G_k$ defines a **connected region** $CC(u_i)$ on the base level of the pyramid. Since the presented algorithm is based on Borővka's algorithm [2], it builds an $MST(u_i)$ of each region, i.e $N_{0,k}(u_i) = MST(u_i)$ [9]. The idea is to collect the smallest weighted edges e (4^{th} step) that could be part of the MST, and then to check if the edge weight $attr_e(e)$ is smaller than the internal contrast of both of the components (MST of end vertices of e) (5^{th} step). If these conditions are fulfilled then these two components are merged (7^{th} step). Two regions will be merged if their internal contrast is larger than the external contrast, represented by the weight $attr_e(e)$ of the connecting edge. All the edges to be contracted form the contraction kernels $N_{k,k+1}$, which are then used to create the graph $G_{k+1} = C[G_k, N_{k,k+1}]$ [17]. In general $N_{k,k+1}$ is a forest. We update the attributes of those edges $e_{k+1} \in G_{k+1}$ with the minimum attribute of the edges $e_k \in E_k$ that are contracted into e_{k+1} (9^{th} step). This means the edge attributes are inherited. It can be shown, that Alg. 2 produces an MST [10]. At each level of the pyramid a region adjacency graph (RAG) is created, in an agglomerative way by topolgy preserving edge contraction. Each vertex of these RAGs is the representative of a sub-trees of MST. This greedy algorithm collects only the nearest neighbor with the minimum edge weights, known as *single linkage clustering*, and merges them if the pairwise comparison (Eq. 3) evaluates to “false”. See [9, 10, 5] for other properites of this algorithm.

5 Experiments on Image Graphs

We start with the trivial partition, where each pixel is a homogeneous region. The attributes of edges are defined as the difference of its end point vertices. The attributes of edges can be defined as the difference between features of end vertices, $attr_e(u_i, u_j) := |F(u_i) - F(u_j)|$, where F is some feature. Other attributes could be used as well e.g. [29] $attr_e(u_i, u_j) := exp\{\frac{-||F(u_i)-F(u_j)||_2^2}{\sigma_I}\}$, where F is some feature, and σ_I is a parameter, which controls the scale of proximity measures of F. F could be defined as $F(u_i) := I(u_i)$, for gray value intensity images, or $F(u_i) := [v_i, v_i \cdot s_i \cdot \sin(h_i), v_i \cdot s_i \cdot \cos(h_i)]$, for color images in HSV color distance [29]. However the choice of the definition of the weights and the features to be used is in general a hard problem, since the grouping cues could conflict each other [20]. For our experiments we use the difference between pixel intensities $F(u_i) := I(u_i)$, i.e. $attr_e(u_i, u_j) := |I(u_i) - I(u_j)|$. For color images we run the algorithm by computing the distances (weights) in RGB color space. We define threshold function $\tau(CC)$ to be function of the size of CC e.g. $\tau(CC) := \alpha/|CC|$, where $|CC|$ is the size of the component CC and α is a constant. This function controls the influence of the size of the components CC. The algorithm has one running parameter α. This constant is used to produce a kind of the over-segmented image. The influence of τ in Eq.3 decays after each level of the pyramid, since the $|CC|$ gets bigger. A larger α sets the preference for larger components. A complex definition of $\tau(CC)$, which is large for certain shapes would produce a partitioning which prefers certain shapes.

We use indoor and outdoor RGB images. We found that $\alpha := 300$ produces the best hierarchy of partitions of the images shown in *Monarch*[1], *Object*45[2] and *Object*11[2] Fig.2(1,3,4) and $\alpha := 1000$ for the image in Fig.2(2), after the average intensity attribute of vertices is down-projected onto the base grid. Fig. 2 show some of the partitions on different levels of pyramid and the number of components. In all images there are regions of large intensity variability and gradient. This algorithm copes with this kind of images.

In contrast to [5] the result is a hierarchy of partitions at multiple resolutions suitable for further goal driven, domain specific analysis. On lower levels of the pyramid the image is over-segmented whereas in higher levels it is under-segmented. Since the algorithm preserves details in low-variability regions, a noisy pixel would survive through the hierarchy. Image smoothing in low variability regions would overcome this problem, and it is not done, since this would introduce another parameter into the method. The hierarchy of partitions can also be built from an over-segmented image to overcome the problem of noisy pixels. For an over-segmented image, where the size of regions is large, the algorithm becomes parameterless.

6 Conclusion and Outlook

In this paper we introduced a method to build a hierarchy of partitions of an image by comparing in a pairwise manner the difference along the boundary of two components relative to the differences of components' internal differences. Even though the algorithm takes simple greedy decisions locally, it produces perceptually important partitions in a bottom-up 'stimulus-driven' way based only on local differences. It was shown that the algorithm can handle large variation

• 1)Waterloo image database and 2) Coil 100 image database

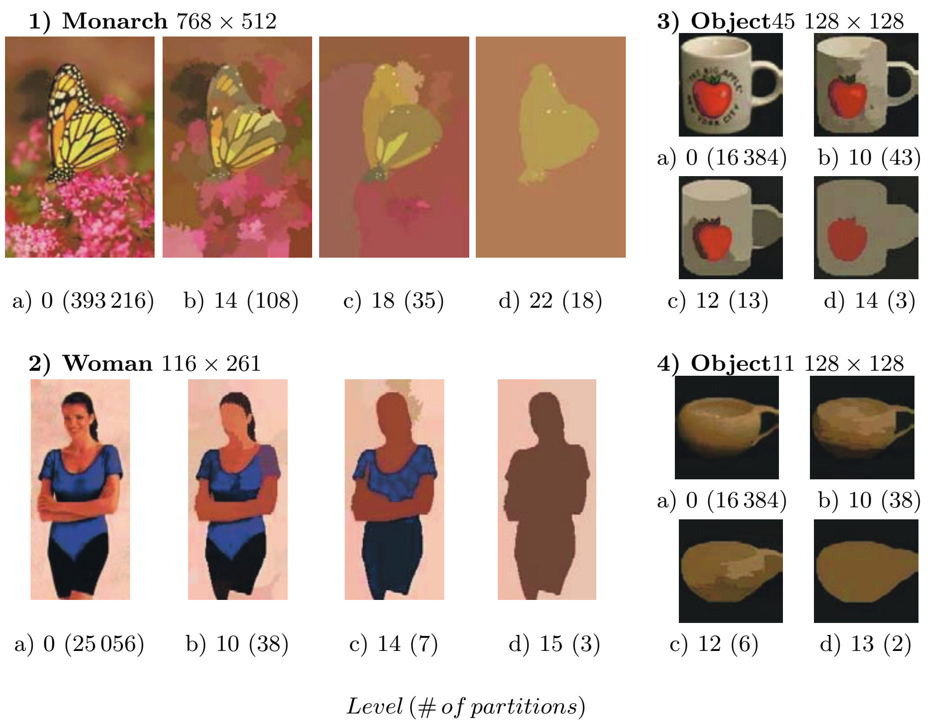


Fig. 2. Partitioning of images.

and gradient intensity in images. Since our framework is general enough, we can use *RAG*s of any over-segmented image and build the hierarchy of partitions. External knowledge can help in a top-down segmentation technique. A drawback is that the maximum and minimum criterion are very sensitive to noise, although in practice it has a small impact. Other criteria like median would lead to an *NP*-complete algorithm [5]. Our future work is to define different comparison functions which will prefer learned regions of specific shapes.

References

1. M. Bister, J. Cornelis, and A. Rosenfeld. A critical view of pyramid segmentation algorithms. *PRL*, 11(9):605–617, 1990.
2. O. Borůvka. O jistém problému minimálnim. *Práce Mor. Přírodvěd. Spol. v Brně (Acta Societ. Scienc. Natur. Moravicae)*, III(3):37–58, 1926.
3. R. Cole, K. P. N., and T. R. E. A Linear-Work Parallel Algorithm for Finding Minimum Spanning Trees. *Anual SPAA*, 11–15, 1994.
4. M. Cooper. The Tractibility of Segmentation and Scene Analysis. *IJCV*, 30(1):27–42, 1998.
5. P. F. Felzenszwalb and D. P. Huttenlocher. Image Segmentation Using Local Variation. *CVPR*, 98–104, 1998.
6. C.-S. Fu, W. Cho, S, and K. Essig. Hierarchical color image region segmentation for content-based image retrival system. *IP*, 9(1):156–162, 2001.

7. Y. Gdalyahu, D. Weinshall, and M. Werman. Self-Organization in Vision: Stochastic Clustering for Image Segmentation, Perceptual Grouping, and Image Database Organization. *PAMI*, 23(10):1053–1074, 2001.
8. L. Guigues, L. M. Herve, and J.-P. Cocquerez. The Hierarchy of the Cocoons of a Graph and its Application to Image Segmentation. *PRL*, 24(8):1059–1066, 2003.
9. Y. Haxhimusa and W. G. Kropatsch. Hierarchical Image Partitioning with Dual Graph Contraction. B. Milaelis and G. Krell, eds., *LNCS 2781*, 338–345, 2003.
10. Y. Haxhimusa and W. G. Kropatsch. Hierarchical Image Partitioning with Dual Graph Contraction. Technical Report No.81, PRIP, Vienna University of Technology, http://www.prip.tuwien.ac.at/ftp/pub/publications/trs/tr81.pdf, July 2003.
11. S. Horowitz and T. Pavlidis. Picture Segmentation by a Tree Traversal Algorithm. *J. Assoc. Compt. Math.*, 2(23):368–388, 1976.
12. Y. Haxhimusa, R. Glantz, M. Saib, G. Langs, and W. G. Kropatsch. Logarithmic Tapering Graph Pyramid. L. van Gool, ed., *LNCS 2449*, 117–124, 2002.
13. J.-M. Jolion and A. Montanvert. The adaptive pyramid, a framework for 2D image analysis. *CVGIP: Im. Under.*, 55(3):339–348, 1992.
14. J.-M. Jolion and A. Rosenfeld. *A Pyramid Framework for Early Vision.* 1994.
15. A. K. Jain and R. C. Dubes *Algorithms for Clustering Data.* 1988.
16. Y. Keselman and S. Dickinson. Generic model abstraction from examples. *CVPR*, 1:856–863, 2001.
17. W. G. Kropatsch. Building Irregular Pyramids by Dual Graph Contraction. *IEE-Proc. Vis. Im and Sig. Proc.*, 142(6):366–374, 1995.
18. W. G. Kropatsch. Abstract pyramid on discrete represtations. I. J. O. Lachaud, A. Braquelaire, and A. Vialard, eds., *LNCS 2301*, 1–21, 2002.
19. J. Lance and W. Williams. A General Theory of Classificatory Sorting Strategies. *Comput. J*, 9:51–60, 1967.
20. J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and Texture Analysis for Image Segmentation. *Int. J. on Com. Vis.*, 43(1):7–27, 2001.
21. P. Meer. Stochastic image pyramids. *CVGIP*, 45(3):269–294, 1989.
22. P. Meer, D. Mintz, A. Montanvert, and A. Rosenfeld. Consensus vision. *AAAI-90 Work. on Qual. Vis.*, 111–115, 1990.
23. A. Montanvert, P. Meer, and A. Rosenfeld. Hierarchical image analysis using irregular tesselations. *PAMI*, 13(4):307–316, 1991.
24. M. Pavan and M. Pelillo. Dominiant Sets and Hierarchical Clustering. *ICCV*, 2003.
25. T. Pavlidis. *Structural Pattern Recognition.* 1977.
26. P. Perona and W. Freeman. A Factorization Approach to Grouping. H. Burkhardt and B. Neumann, eds., *LNCS 1406*, 655–670, 1998.
27. A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *Sys. Man and Cyb.*, 6(6):420–433, 1976.
28. E. Sharon, A. Brandt, and R. Basri. Fast multiscale image segmentation. *CVPR*, 70–77, 2000.
29. J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *CVPR*, 731–737, 1997.
30. R. Urquhart. Graph Theoretical Clustering Based on Limited Neighborhood Sets. *Patt.Rec.*,13(3):173–187, 1982.
31. Z. Wu and R. Leahy. An Optimal Graph Theoretic Approach to Data Clustering: Theory and Its Application to Image Segmentation. *PAMI*, 15(11):1101–1113.
32. S. Yu and J. Shi. Segmentation with pairwise attraction and repulsion. *ICCV*, 52–58, 2001.
33. C. Zahn. Graph-theoretical methods for detecting and describing gestal clusters. *IEEE Comput.*, 20:68–86, 1971.

Clustering Variable Length Sequences by Eigenvector Decomposition Using HMM

Fatih Porikli

Mitsubishi Electric Research Laboratories, Cambridge MA 02139, USA
fatih@merl.com

Abstract. We present a novel clustering method using HMM parameter space and eigenvector decomposition. Unlike the existing methods, our algorithm can cluster both constant and variable length sequences without requiring normalization of data. We show that the number of clusters governs the number of eigenvectors used to span the feature similarity space. We are thus able to automatically compute the optimal number of clusters. We successfully show that the proposed method accurately clusters variable length sequences for various scenarios.

1 Motivation

Although many algorithms exist for unsupervised classification of patterns into clusters, most of these methods require the data space X consists of 'identical length' data points (feature vectors) $x_i = (x_{i1}, ..., x_{iN})$ where N is the dimension of the data space, i.e. $X : \mathcal{R}^N$. Such algorithms include the ordinary implementations of decision trees, neural nets, Bayesian classifiers, ML-estimators, support vector machines, Gaussian mixture models, k-means, and hierarchical approaches, self-organizing maps, etc [4].

However not all classification problems can be formulated into a data space that contains only equal length feature vectors. For instance, lets consider the following scenarios:

Example 1. A data space contains different shapes. We compute a sequence of boundary coordinates for each shape by starting from a certain point on the boundary. Then we obtain sequences such as $s_i = ((x_{i1}, y_{i1}), ..., (x_{iN_i}, y_{iN_i}))$ where (x_{ij}, y_{ij}) is the coordinate of the j^{th} boundary point for the i^{th} shape. In this case, the length of the sequences are not necessarily same since the length of the boundaries may be different, e.g. it is possible that $N_1 \neq N_2$.

Example 2. A basket contains unknown number of not necessarily identical balls labeled as a and b. At a random time instant we start drawing balls from this basket. We keep record of the symbols on the balls. Then we stop drawing balls at a random time instant. By repeating the experiment we obtain sequences as

$s_1 =$ abababababababababababababababa
$s_2 =$ ababababababababa
$s_3 =$ bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb, etc.

Note that the length of these sequences are different. This process is analogous to chopping a DNA into genes. Finding the common and divergent patterns from a big pool of protein genomes, which come in various sizes, is a challenge of current bioinformatics and requires clustering of variable length sequences.

We will refer the type of the data in these examples as variable length sequences. One way to adapt some of the above scenarios for ordinary classification, which processes constant length sequences, is to normalize the length of the feature vectors. Although commonly used due to its simplicity, normalization of the feature vector length (either by sampling or interpolation) causes severe degradation and aliasing. Besides, some clustering approaches assume that they can compute a centroid and then compare the data points with this centroid, as in k-means. It is not possible to obtain such a centroid for variable length data.

Thus, we propose a clustering algorithm that can classify variable length sequences. Our algorithm also estimates the optimum number of clusters and does not require normalization of the length of the feature vectors. Instead of working directly on the initial values, we transfer the sequences into a parameter space using Hidden Markov Models (HMM), which captures the probabilistic transition properties of sequential data.

In this work, we concentrate on the discrete label sequences. Using the parameter space representations, we compute an affinity matrix that shows the similarity of a given pair of sequences. Then we decompose the affinity matrix into a series of subspaces spanned by the eigenvectors corresponding to the largest eigenvalues. We threshold the decomposed values and partition clusters using simple connected component analysis. For each decomposition, we calculate a validity score indicating the fitness of the current clusters to the data. We determine the optimum number of clusters using the validity score. We give a flow diagram of the method in fig. 1.

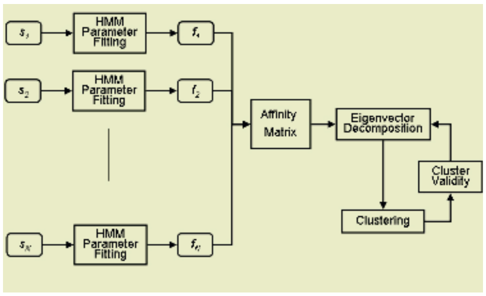


Fig. 1. Flow diagram of clustering sequences.

In the next section, we explain HMM's and affinity matrix. In section 3, we present eigenvector decomposition. In the following sections, we give details of the clustering and discuss simulations.

2 Parameter Space by HMM

We project each sequence s_i into the parameter space that is characterized by a set of HMM parameters. HMM's are richer representations of time series. An HMM is a probabilistic model composed of a number of interconnected states, each of which emits an observable output. A discrete hidden Markov model is defined by a set of states and an alphabet of output symbols [6]. Each state is characterized by two probability distributions: the transition distribution over states and the emission distribution over the output symbols. A random source described by such a model generates a sequence of output symbols as follows: at each time step the source is in one state, and after emitting an output symbol according to the emission distribution of the current state, the source jumps to a next state according to the transition distribution of its current state. Since the activity of the source is observed indirectly, through the sequence of output symbols, and the sequence of states is not directly observable, the states are said to be hidden.

An K-state $\{S_1, S_2, ..., S_K\}$, discrete HMM is represented by:

1. A set of prior probabilities $\pi = \{\pi_i\}$ where $\pi_i = P(q_1 = S_i)$,$1 \leq i \leq K$.
2. A set of state transition probabilities $H = \{h_{ij}\}$, where $h_{ij} = P(q_{t+1} = S_j | q_t = S_i)$,$1 \leq i, j \leq K$.
3. A set of output distributions $B = \{b_{ij}\}$, where $b_{ij}(y) = P(O_{t+1} = y | q_t = S_i, q_{t+1} = j)$,$1 \leq i, j \leq K$.

where q_t and O_t are the state and observation respectively at time t. It is common to denote the an M-mixture of HMM's by (H_m, B_m, π_m), $1 \leq m \leq M$. For discrete HMM, algorithms exist for: 1) computing the probability of observing a sequence, given a model, 2) finding the state sequence that maximizes the probability of the given sequence, when the model is known (the Viterbi algorithm), 3) inducing the HMM that maximizes (locally) the probability of the given sequence (the Baum-Welch algorithm, an expectation-maximization algorithm).

For each sequence, we fit a HMM (a discrete model in case the sequence components are labels, a continuous model in case the components are real numbers that reflect certain proportional properties, e.g. magnitude, coordinate, etc). The number of states K, number of models M, and the HMM topology (left-to-right) are assigned same for each sequence. This enables us to compute parameter space distances using the model state transition, observation, and prior matrix differences [8].

Definition 1. *A feature f_i is a set of real numbers that correspond the HMM parameters of a sequence s_i for a given number of states K, number of mixtures M, and left-to-right topology using the sequence as an observation ($f_i : s_i \rightarrow (H_m, B_m, \pi_m)_i$).*

The problem of estimating the correct number of clusters is a difficult one: a full Bayesian solution for obtaining the posterior probability on M, requires a complex integration over the HMM parameter space, as well as knowledge about the priors on the mixture parameters and about the priors on M itself. Often this integration cannot be solved in closed form, and Monte-Carlo methods and other approximation methods are used to evaluate it. However, these methods are computationally intensive [1].

Now, we can compute our affinity matrix. Given two sequences s_i, s_j, we determine the probability that feature set is generated by s_j and feature set f_j is generated by s_i. This probability indicates the mutual 'fitness' of the given sequences to corresponding HMM's. Thus, the affinity matrix represents the similarity of two sequences. The elements a_{ij} of A are equal to

$$a_{ij} = e^{-d(s_i,s_j)/2\sigma^2} \tag{1}$$

where the distance is defined as

$$d(s_i, s_j) = |P(s_i|f_i) + P(s_j|f_j) - P(s_i|f_j) - P(s_j|f_i)| \,. \tag{2}$$

and σ^2 is a scaler. The affinity matrix components will have values close to 1 if the corresponding sequences fit well to each other's models, and close to 0 otherwise. Note that similarity matrix $A \in \mathcal{R}^{n\times n}$ is a real semi-positive symmetric matrix, thus $A^T = A$.

Next, we explain the details of the eigenvector decomposition process.

3 Eigenvector Decomposition

The decomposition of a square matrix into eigenvalues and eigenvectors is known as eigenvector decomposition. For the affinity matrix A there are n eigenvalues λ with associated eigenvectors $\mathbf{v}$ which satisfy $A\mathbf{v} = \lambda\mathbf{v}$. To find these eigenvalues, we rewrite the previous equation as $(A - \lambda I)\mathbf{v} = 0$ and determinant is computed $det(A - \lambda I) = 0$.

Let $V \equiv [\mathbf{v}_1 \; \mathbf{v}_2 \; .. \; \mathbf{v}_n]$ be a matrix formed by the columns of the eigenvectors. Let D be a diagonal matrix $diag[\lambda_1, \lambda_2, .., \lambda_n]$. Lets also assume $\lambda_1 \geq \lambda_2 \geq ..\lambda_n$. Then the eigenvalue problem becomes

$$AV = [A\mathbf{v}_1 \; .. \; A\mathbf{v}_n] = [\lambda_1\mathbf{v}_1 \; .. \; \lambda_n\mathbf{v}_n] = VD \tag{3}$$

and $A = VDV^{-1}$. Since A is symmetric, the eigenvectors corresponding to distinct eigenvalues are real and orthogonal $VV^T = V^TV = I$, which implies $A = VDV^T$.

Iterative Eigenvector Computation. The main idea behind iterative computation is the following. Suppose we have some subspace $\mathcal{K}$ of dimension k, over which the projected matrix A has Ritz [7] value θ_k and a corresponding Ritz vector $\mathbf{u}_k$. Let us assume that an orthogonal basis for $\mathcal{K}$ is given by the vectors $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k$ (already determined eigenvectors).

Quite naturally the question arises how to expand the subspace in order to find a successful update for $\mathbf{u}_k$, which will become v_{k+1}. To that end we compute the defect $\mathbf{r} = A\mathbf{u}_k - \theta_k \mathbf{u}_k$. Then as in [3], we compute $\widetilde{\mathbf{z}}$ from $(D - \theta_k I)\widetilde{\mathbf{z}} = \mathbf{r}$, where D is the diagonal matrix of A as defined above. The vector $\widetilde{\mathbf{z}}$ is made orthogonal to $\mathcal{K}$, and the resulting vector is chosen as the new $\mathbf{v}_{k+1}$ by which $\mathcal{K}$ is expanded. This method find the largest eigenvalues in absolute value. The matrix $(D - \theta_k I)^{-1}$ can be viewed as a preconditioner for the vector $\mathbf{r}$. Although it is tempting to use this preconditioner as an approximation for $(A - \theta_k I)$, it would not lead to an expansion of our search space. To avoid this stagnation, we concentrate on the kth approximation $\mathbf{u}_k$ of the eigenvector $\mathbf{v}$, where $\mathbf{u}_k$ is normalized $||\mathbf{u}_k|| = 1$. The residual $\mathbf{r} = A\mathbf{u}_k - \theta_k \mathbf{u}_k$ is orthogonal to $\mathbf{u}_k$ because $\theta_k = \mathbf{u}_k^T A \mathbf{u}_k$ is the Ritz value associated with $\mathbf{u}_k$. We project the eigenvalue problem $Av = \lambda \mathbf{v}$ on span$(\mathbf{u}_k)$, and on its orthogonal complement. This leads to two coupled equations for λ and the complement $\mathbf{z}$ of $\mathbf{v}$ orthogonal to $\mathbf{u}_k$: $\lambda = u_k^T A(\mathbf{u}_k + \mathbf{z})$ and $\mathbf{z} \perp \mathbf{u}_k, \quad (I - \mathbf{u}_k \mathbf{u}_k^T)(A - \lambda I)(I - \mathbf{u}_k \mathbf{u}_k^T)\mathbf{z} = -\mathbf{r}$. Since λ is unknown, we cannot compute optimal update $\mathbf{z}$ from $\mathbf{u}_k$. However it is reasonable to replace λ by the current approximation θ_k. Thus we obtain $\mathbf{r} \perp \mathbf{u}_k, \quad (I - \mathbf{u}_k \mathbf{u}_k^T)(A - \theta_k I)(I - \mathbf{u}_k \mathbf{u}_k^T)\mathbf{z} = -\mathbf{r}$ as a good correction for $\mathbf{u}_k$. Similarly, we compute the approximate solution $\widetilde{\mathbf{z}}$ using this equation, and by making $\widetilde{\mathbf{z}}$ orthogonal to search space, we obtain $\mathbf{v}_{k+1}$. Briefly, we extract an approximate eigenvalue from the search subspace, project it, solve the projected eigenvalue problem, compute the corresponding Ritz value and residual, correct the approximate eigenvector u, and expand the search subspace with the correction vector.

The above iterative prediction is used at the following clustering stage.

4 Clustering

Although eigenvector based clustering [2], [9], [5] is addressed before in the literature, to our knowledge no one has established the relationship between the optimal clustering of the data distribution and the number of eigenvectors that should be used for spanning before. Here we show that the number of eigenvectors is proportional to the number of clusters.

Let a matrix P_k be a matrix in a subspace $\mathcal{K}$ that is spanned by the columns of V such as $P_k = [\mathbf{v}_1 \ \mathbf{v}_2 \ .. \ \mathbf{v}_k, \ 0]$ where V is the orthogonal basis satisfies $A = VDV^T$.

Now, we define vectors $\mathbf{p}_n$ as the rows of the truncated matrix P_k as

$$P_k = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_n \end{bmatrix} = \begin{bmatrix} v_{11} & \cdots & v_{1k} & 0 \cdots \\ v_{21} & \cdots & v_{2k} & 0 \cdots \\ \vdots & & & \vdots \\ v_{n1} & \cdots & v_{nk} & 0 \cdots \end{bmatrix} \tag{4}$$

We normalize each row of matrix P_k by $p_{ij} \leftarrow p_{ij} / \sqrt{\sum_j^k p_{ij}^2}$. Then a correlation matrix is computed using the normalized rows by $C_k = P_k P_k^T$. For a given P_k, the value of p_{ij} indicates the degree of similarity between the object i and object j. Values close to one correspond to a match whereas negative values and values close to zero suggest that objects are different. Let ϵ be a threshold that transfers values of matrix C_k to the binary quantized values of an association matrix W_k as

$$w_{ij} = \begin{cases} 1 \; c_{ij} \geq \epsilon \\ 0 \; c_{ij} < \epsilon \end{cases} \tag{5}$$

where $\epsilon \approx 0.5$. The clustering is then becomes grouping the objects that have association values equal to one $w_{ij} = 1$.

To explain why this works, remember that eigenvectors are the solution of the classical extremal problem $\max \mathbf{v}^T A \mathbf{v}$ constrained by $\mathbf{v}^T \mathbf{v} = 1$. That is, find the linear combination of variables having the largest variance, with the restriction that the sum of the squared weights is 1. Minimizing the usual Lagrangian expression $\mathbf{v}^T A \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - 1)$ implies that $A\mathbf{v} = \lambda \mathbf{v}$. Thus, $\mathbf{v}$ is the eigenvector with the largest eigenvalue.

When we project the affinity matrix columns on the eigenvector $\mathbf{v}_1$ with the largest eigenvalue and span $\mathcal{K}_1$, the distribution of the a_{ij} will have the maximum variance therefore the maximum separation. Keep in mind that a threshold operation will perform best if the separation is high. To this end, if the distribution of values have only two distinct classes then a balanced threshold passing through the center will divide the points into two separate clusters. With the same reasoning, the eigenvector $\mathbf{v}_2$ with the second largest eigenvalue, we will obtain the basis vector that gives the best separation after normalizing the projected space using the $\mathbf{v}_1$ since $\mathbf{v}_1 \perp \mathbf{v}_2$. It is important to note that, each additional eigenvector enables us to divide the space into an extra cluster. Thus, we conclude that;

Lemma 1. *The number of largest eigenvalues (in absolute value) to span the subspace is one less than the number of clusters.*

As opposed to using only the largest or first and second largest eigenvectors (also the generalized second minimum which is the ratio of the first and the second depending the definition of affinity), the correct number of eigenvectors should be selected with respect to the optimum cluster number.

After each eigenvalue computation of matrix A in the iterative algorithm, we compute a validity score α_k using the clustering results as

$$validity : \alpha_k = \sum_{c}^{k} \frac{1}{N_c} \sum_{i,j \in Z_c} p_{ij} \tag{6}$$

where Z_c is set of objects included in the cluster c, N_c number of objects in Z_c. The validity score gets higher values for the better fits. Thus, by evaluating the local maxima of this score we determine the correct cluster number automatically. Thus, we answer one important question of clustering; "what should be the total cluster number?"

The values of the thresholds should still be computed. We obtained projections that gives us the maximum separation but we did not determine the degree of separation i.e. maximum and minimum values of projected values on the basis vectors. For convenience, we normalize the projections i.e. the *rows* of current projection matrix (V_k) as $\mathbf{p}^T\mathbf{p} = 1$ and then compute the correlation $V_k^T V_k$. Correlation will make rows that their projections are similar to get values close to 1 (equal values will give exactly 1), and dissimilar values to 0. By maximizing the separation (distance) between the points in different clusters on an orthonormal basis, we pushed for the orthogonality of points depending their clusters; $\mathbf{p}_i\mathbf{p}_j \approx 1$ if they are in the same cluster, and $\mathbf{p}_i\mathbf{p}_j \approx 0$ if they are not in the same cluster.

As a summary, the clustering for a given maximum cluster number k^* includes

1. Compute A, approximate eigenvectors using Ritz values $\lambda_k \simeq \theta_k$, find eigenvectors v_k for $k = 1, .., k^*$,
2. Find $P_k = V_k V_k^T$ and Q_k for $k = 1, .., k^*$,
3. Determine clusters and calculate α_k,
4. Compute $\alpha' = d\alpha/dk$ and find local maxima.

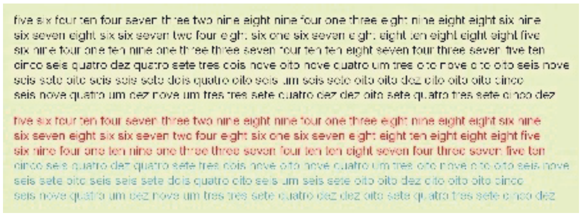
The maximum cluster number k^* does not affect the determination of the fittest cluster; it only limits the maximum number of possible clusters that will be searched.

5 Experiments and Discussion

We simulated the proposed method using several label sequences as given in fig. 2. We conducted the following evaluations:

Language Discrimination: We generated three sequences of random integers that are uniformly distributed in the range [1 : 10]. Then, we replaced each number in the sequence with its English and Portuguese spellings to obtain the letter sequences given in fig. 2-a (shown in black). Thus, each letter represents a label. We trained 4-states HMM's and applied eigenvector decomposition after computing the affinity matrix. The validity reached maximum for cluster number $k = 2$ and fig. 2-a shows the clustering results. As visible in the bottom part of fig. 2-a (blue and red clusters), the HMM's captured the dynamics of letter ordering that is intrinsic to each language and identified the language clusters accurately. We obtained similar results for different length sequences as well. In the future, we plan o represent each word using a separate label rather that using letters as labels for language related classification tasks.

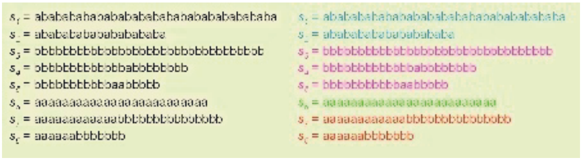
Pattern Matching: We generated a total of 8 random length sequences that can be partitioned into 4 patterns using two labels (a, b) as given in fig. 2-b, left. After computing 2-state HMM's we obtained the affinity matrix given in fig. 2-c. The validity scores after iterative clustering is shown in fig. 2-d, where it reaches maximum for the cluster number $k = 4$. The corresponding clusters after eigenvector decomposition is given in fig. 2-d. The results prove that the proposed method can accurately detect pattern similarities even though the length of the sequences may differ significantly.



five six four ten four seven three two nine eight nine four one three eight nine eight eight six nine
six seven eight six six seven two four eight six one six seven eight eight ten eight eight eight five
six nine four one ten nine one three three seven four ten ten eight seven four three seven five ten
cinco seis quatro dez quatro sete tres dois nove oito nove quatro um tres oito nove oito oito seis nove
seis sete oito seis seis sete dois quatro oito seis um seis sete oito oito dez oito oito oito cinco
seis nove quatro um dez nove um tres tres sete quatro dez dez oito sete quatro tres sete cinco dez

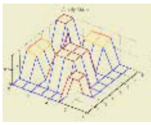
five six four ten four seven three two nine eight nine four one three eight nine eight eight six nine
six seven eight six six seven two four eight six one six seven eight eight ten eight eight eight five
six nine four one ten nine one three three seven four ten ten eight seven four three seven five ten
cinco seis quatro dez quatro sete tres dois nove oito nove quatro um tres oito nove oito oito seis nove
seis sete oito seis seis sete dois quatro oito seis um seis sete oito oito dez oito oito oito cinco
seis nove quatro um dez nove um tres tres sete quatro dez dez oito sete quatro tres sete cinco dez

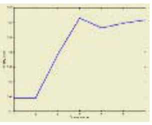
(a)



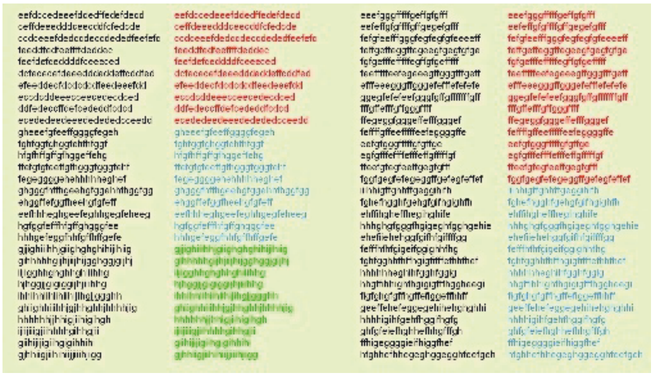
s_1 = ababababababababababababababababa	s_1 = ababababababababababababababababa
s_2 = abababababababababababa	s_2 = abababababababababababa
s_3 = bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb	s_3 = bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
s_4 = bbbbbbbbbbbbbbabbbbbbbbb	s_4 = bbbbbbbbbbbbbbabbbbbbbbb
s_5 = bbbbbbbbbbbbaabbbbb	s_5 = bbbbbbbbbbbbaabbbbb
s_6 = aaaaaaaaaaaaaaaaaaaaaaaaaa	s_6 = aaaaaaaaaaaaaaaaaaaaaaaaaa
s_7 = aaaaaaaaaaaaabbbbbbbbbbbbbb	s_7 = aaaaaaaaaaaaabbbbbbbbbbbbbb
s_8 = aaaaaabbbbbbb	s_8 = aaaaaabbbbbbb

(b)



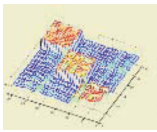


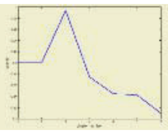
(c) (d)

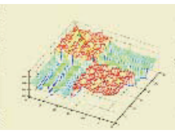


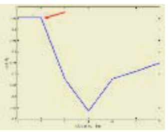
eefdccedeeefdcedffedefdecd	eefdccedeeefdcedffedefdecd	eeefgggffffgefgfgff	eeefgggffffgefgfgff
ceffdeeedddceecddfcfedcde	ceffdeeedddceecddfcfedcde	eefeffgfgffffgffgegefgfff	eefeffgfgffffgffgegefgfff
ccdceeefdedccdeccdededfeefefc	ccdceeefdedccdeccdededfeefefc	fefgfeeffgggfegfegfgfeeeeff	fefgfeeffgggfegfegfgfeeeeff
fecddfcdfceffffdcddec	fecddfcdfceffffdcddec	feffgeffegggffegeegfgegfgfge	feffgeffegggffegeegfgegfgfge
fecfdefcecddddfceeeced	fecfdefcecddddfceeeced	fgfgefffeffffegfgfgefffff	fgfgefffeffffegfgfgefffff
dcfeccecfdeeeddcdcddeffcddfcd	dcfeccecfdeeeddcdcddeffcddfcd	feeffffeeegeeegffgggfffgeff	feeffffeeegeeegffgggfffgeff
efeeddecfdcdcdcdfeedeeefcdd	efeeddecfdcdcdcdfeedeeefcdd	efffeeegggffggefefffefefefe	efffeeegggffggefefffefefefe
eccdcddeeeeccceecedecdced	eccdcddeeeeccceecedecdced	ggegfefefeefgggfgffgffffffgff	ggegfefefeefgggfgffgffffffgff
ddfedeccffcefcededdcfcdcd	ddfedeccffcefcededdcfcdcd	fffgffefffgfgggffff	fffgffefffgfgggffff
ecededeecdeeecededcdcceedc	ecededeecdeeecededcdcceedc	ffegeggfgggeffefffgggef	ffegeggfgggeffefffgggef
gheeefgfeeffgggcfegeh	gheeefgfeeffgggcfegeh	feffffgffeefffffeefegggggffe	feffffgffeefffffeefegggggffe
fghfggfghggfehfhfggf	fghfggfghggfehfhfggf	eefgfgggffffgfgffge	eefgfgggffffgfgffge
hfgfhffgffgfhggeffehg	hfgfhffgffgfhggeffehg	egfgfffefffeffefefgfffffgf	egfgfffefffeffefefgfffffgf
ffefgfgfeefgfhgggfgggfehf	ffefgfgfeefgfhgggfgggfehf	ffeefgfegfeeffgegfgff	ffeefgfegfeeffgegfgff
fegeggggehehhhhhheghef	fegeggggehehhhhhheghef	fggfgegfefegeggffgefegfeffef	fggfgegfefegeggffgefegfeffef
ghgggfhfffhgeehgfggehhfhggfgg	ghgggfhfffhgeehgfggehhfhggfgg	iihhigffghhffgagghfh	iihhigffghhffgagghfh
ehggffefggfheehgfgfeff	ehggffefggfheehgfgfeff	fghefhgghfgehgfgifhgighfh	fghefhgghfgehgfgifhgighfh
eefhhheghgeefeghhgegfeheeg	eefhhheghgeefeghhgegfeheeg	ehffihghefffhegihghife	ehffihghefffhegihghife
hgfggfeffhfgffghgggfee	hgfggfeffhfgffghgggfee	hhhghgfgggfhgigeghfgghgehie	hhhghgfgggfhgigeghfgghgehie
hhhgefeggfhhfgfhffgefe	hhhgefeggfhhfgfhffgefe	ehefiiehehggfgifhigiffffgg	ehefiiehehggfgifhigiffffgg
gjighiihhjgiighghghihijhig	gjighiihhjgiighghghihijhig	feffhfhfgigeifggighhfhg	feffhfhfgigeifggighhfhg
ghhhhhgjhjijhijgghggjgjhj	ghhhhhgjhjijhijgghggjgjhj	fghfgghhfhhhgigfffhefhhhhf	fghfgghhfhhhgigfffhefhhhhf
ijigghhghghhghihhhg	ijigghhghghhghihhhg	hhhhhheghhhfgghfggig	hhhhhheghhhfgghfggig
hjhggjggggjhjihhg	hjhggjggggjhjihhg	hhgfhhhigfhgigigfffhgghegi	hhgfhhhigfhgigigfffhgghegi
ihhihhihhihhjihgjggghh	ihhihhihhihhjihgjggghh	figfghgfgffhgffeffggeffhhff	figfghgfgffhgffeffggeffhhff
ghiighhiihhjgjhhghhjhhhhjig	ghiighhiihhjgjhhghhjhhhhjig	geefefehefeggegehihehghghhi	geefefehefeggegehihehghghhi
hhhhhhjjhhigjiihighgh	hhhhhhjjhhigjiihighgh	hhhhigihfgehfhgghhgfg	hhhhigihfgehfhgghhgfg
ijijiigjiihhhhgihhgii	ijijiigjiihhhhgihhgii	ghfgfeiefhghhefhhgffgh	ghfgfeiefhghhefhhgffgh
gihijjigihgjgihhih	gihijjigihgjgihhih	ffhigeggggieihiggfhef	ffhigeggggieihiggfhef
ghhigjihhijjiihjgg	ghhigjihhijjiihjgg	hghhcfhhegeghggeggffeefgeh	hghhcfhhegeghggeggffeefgeh

(e)









(f) (g) (h) (i)

Fig. 2. (a) Language discrimination set (upper) and the clustering results (lower). (b) Pattern detection set (left) the computed clusters (right), and (c) affinity matrix, (d) validity scores for this set. (e) Two random length, random distribution scenarios, (f-g) affinity matrix and validity scores for the 1^{st} column of (e), and (h-i) affinity matrix and validity scores for the 3^{rd} column of (e).

Random Letters: We generated random length, random distributed sequences using labels (c, d, e, f, g, h, i, j) as given in fig. 2-e (1^{st} and 3^{nd} columns). In the first column set, the first 10 sequences consist of uniformly distributed random drawings of from set (c, d, e, f), and similarly the second 10 sequences are made of (e, f, g, h), and the last 10 sequences are generated from (g, h, i, j) to allow partial overlap between each domain. The affinity matrix and validity scores are given in fig. 2-f and fig. 2-g, respectively. The maximum validity happens at

$k = 3$. The second column shows the clustering results for this set. The third row in fig. 2-e shows the random label sequences that are generated using the set (e, f, g) (first 15 sequences) and a bigger inclusive set of (e, f, g, h, i) (last 15 sequences). The affinity is depicted in fig. 2-h and the validity scores in fig. 2-i. As obvious in the clustering results, the optimum cluster number is estimated accurately and the eigenvector decomposition partitioned correctly at each time.

In conclusion, the main contributions of this paper are:

- We proposed a new method to compare the variable length sequences using the HMM parameter space.
- We showed that the number of largest eigenvalues (in absolute value) to span subspace is one less than the number of clusters.
- We used the above result as a quality assessment criterion for cluster fit.

References

1. J. Alon, S. Sclaroff, G. Kollios V .Pavlovic, "Discovering clusters in motion time-series data", *Proceedings of Computer Vision and Pattern Recognition*, 2003.
2. G.L. Scott and H. C. Longuet-Higgins, "Feature grouping by relocalisation of eigenvectors of the proxmity matrix" *In Proc. British Machine Vision Conference*, 103-108, 1990.
3. G. Sleijpen and H. Van Der Vorst, "A Jacobi-Davidson iteration method for linear eigenvalue problems", *SIAM J. Matrix Anal. Appl.*, vol. 17, 401–425, 1996.
4. A. K. Jain , M. N. Murty , P. J. Flynn, "Data clustering: a review", *ACM Computing Surveys (CSUR)*, 31(3), 264-323, 1999.
5. J. Shi and J. Malik. "Normalized cuts and image segmentation" *In Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 731-737, 1997.
6. L. Rabiner. "A tutorial on hidden markov models and selected applications in speech recognition", *Proceedings of IEEE*, 77(2), 257–285, 1989.
7. R. B. Morgan, "Computing interior eigenvalues of large matrices" *Linear Algebra Appl.*, 154/156, 289-309, 1991.
8. P. Smyth, "Clustering sequences with Hidden Markov Models", *Book: Advances in Neural Information Processing Systems, The MIT Press, M.C. Mozer, M.I. Jordan, T. Petsche*, 648, 1997.
9. Y. Weiss, "Segmentation using eigenvectors: a unifying view", *Proceedings IEEE International Conference on Computer Vision*, 975-982, 1999.

A Kernel View of Spectral Point Pattern Matching

Hongfang Wang and Edwin R. Hancock

Dept. of Computer Science, University of York
Heslington, York, YO10 5DD, UK
{hongfang,erh}@cs.york.ac.uk

Abstract. This paper investigates spectral approaches to the problem of point pattern matching. Specifically, kernel principle component analysis (kernel PCA) methods are studied and compared with Shapiro and Brady's approach and multidimensional scaling methods on both synthetic data and real world data. We demonstrate that kernel methods can be effectively used for solving the point correspondence matching problem with a performance that is comparable with other iterative-based algorithms in the literature under the existing of outliers and random position jitter. We also provide discussion of the theoretical support from kernel PCA to the earlier approach of Shapiro and Brady.

1 Introduction

The problem of point pattern matching is to find one-to-one correspondences among two given data-sets and serves as an important part in many computer vision tasks. Graph spectral methods have been used extensively for locating correspondences between feature point-sets, e.g. [8, 9]. In [8], Scott and Longuet-Higgins first use a Gaussian weighting function to build an inter-image proximity matrix between feature points in different images being matched and then perform singular value decomposition on the obtained matrix in order to get correspondences from the proximity matrix's singular values and vectors. This method fails when rotation or scaling between the images is too large. To overcome this problem, Shapiro and Brady [9] construct intra-image proximity matrices for the individual point-sets being matched with an aim to capturing relational image structure. The eigenvectors of the individual proximity matrices are used as the columns of a modal matrix. Correspondences are located by comparing the rows of the modal matrices for the point-sets under match. This method can be viewed as projecting the individual point-sets into an eigenspace, and seeking matches by looking for closest point correspondences. Carcassoni and Hancock have attempted to improve the robustness of this method to point-jitter using robust error kernels instead of the Gaussian [2] and have overcome problems due to differences in the structure of the point-sets by using spectral clusters [3]. Multidimensional scaling is also used to solve this problem by performing Procrustes alignment in the eigenspace [6]. However, these two latter approaches involve iterative computing which requires more computation than other approaches.

This motivates us to seek point matching algorithms that are both robust and without iteration. Kosibov and Caelli [1] have extended the Shapiro and Brady method of seeking correspondences by searching for matches that maximise the inner product of the truncated and re-normalised eignevactors.

The idea underpinning these spectral methods is to embed point-sets into a common eigenspace, and to find correspondences by performing alignment in this space. The key in this idea is that of finding the appropriate function which captures the essential properties of the given data-set which should also be robust under uncertainties such as outliers, random position jitter, occlusions, etc., and identifying the common eigenspace. Also the captured properties should be common in both data sets. The problem of how to select the best function, is a topic that has recently attracted considerable interest in kernel learning theory. The development of kernel PCA [7] provides us with a theoretically sound way of improving the existing spectral point pattern matching algorithms since it shares many features in common with spectral graph theory.

Our aim in this paper is to investigate the performance of kernel PCA for solving the point correspondence problem and provide a robust one-to-one point pattern matching algorithm which involves no iterations. We focus in detail the Gaussian and polynomial kernels which are invariant to similarity transformations and reflection, and compare their performance in point pattern matching with previous approaches. A common weakness with existing spectral methods is that they are particularly sensitive to structural variations in the point-sets. We demonstrate that the kernel approach is a feasible way for point pattern matching and with an appropriate kernel function, in this work the polynomial kernel, encouraging performance can be obtained and the results are less sensitive to these problems than the previous graph spectral methods.

2 Spectral Point Pattern Matching

The problem of point pattern matching can be described as given two feature point-sets $X_1 = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ and $X_2 = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ extracted from two different images, establish a one-to-one point correspondences between the two data-sets. Ideally, outliers can be removed from the data-sets during matching. In this paper, the feature points in each data-set are in the form of $\mathbf{x}_i = (x_i^{(1)}, x_i^{(2)})$ and $\mathbf{y}_i = (y_i^{(1)}, y_i^{(2)})$, respectively, where i are the indices and superscripts (1) and (2) represent each point's respective abscissa and ordinate. Our aim is to locate correspondences between the two point-sets.

The approaches of graph spectral methods for point pattern matching is to solve the point correspondence problem by first build a graph representation for each data-set where each graph node corresponds to an image feature point, and each edge between nodes corresponds to the relationships of the two feature points. After the graph construction, represent each graph by a matrix and find feature correspondences from the matrices' eigendecompositions. These methods aim to embed the dissimilarity (or similarity) properties of the original data into a common space in which correspondence matching can be performed. As mentioned in the last section, the two essential ingredients are the dissimilarity function and the embedding procedure. The dissimilarity properties are regarded as weights of the edges and are expressed in the form of a proximity matrix A with its elements A_{ij} represents the dissimilarity relationship between feature points $\mathbf{x}_i$ and $\mathbf{x}_j$.

The objects in image frames are usually subject to transformations such as translation, rotation, scaling, and reflection. Hence, it is desirable for the dissimilarity function to be invariant under these transformations and thus provide a basis for directly comparing images and for finding correspondences between the original feature points and their transformed counterparts. It is known from geometry that the Euclidean distance is invariant to any similarity transformation, so the dissimilarity functions underpinning many existing methods are related to the Euclidean distance between feature points (see for example, [9, 8]). Another example of similarity transformation-invariant property is the directional properties of feature points. This property can also be considered as a good candidate for constructing a suitable similarity function for spectral point matching.

When viewed from the perspective of kernel PCA, applying a dissimilarity or similarity function to the original data set is equivalent to the process of using a kernel function to map the data into a higher, possibly infinite, dimensional space. Moreover, this mapping interpolates the data in the new space according to their transformation invariant properties. From this perspective, we believe that kernel PCA provides us a sound theoretical explanation for spectral pattern matching, and by applying an appropriate kernel function, expected matching results should be obtained.

3 The Kernel Approach

Kernel PCA can be regarded as a generalization of PCA from a linear to a nonlinear transformation space. In the literature it has been shown to provide a better way of recovering the underlying principal components of the given data.

Conventional principal component analysis (PCA) provides an orthogonal transformation of the data from a high dimensional space to a low dimensional one which maximally preserves the variance of the original data. This is done by computing the eigenvalues and eigenvectors of the covariance matrix $\mathbf{C} = \frac{1}{M}\sum_{i=1}^{M}(\mathbf{x_i}-\overline{\mathbf{x}})(\mathbf{x}_i-\overline{\mathbf{x}})^T$, and then use the first N normalized eigenvectors ($N \leq M$, assume the eigenvalues are sorted in descending order) of the covariance matrix as the main projection axes for the training data. Since the method minimizes the residual covariance of the data points projected into the common eigen-subspace, it thus gives an optimum representation of the original data in the projection space.

The main difference between kernel PCA and conventional PCA is that kernel PCA first uses a function $\mathbf{T} : \mathbf{x} \mapsto \mathbf{\Phi}(\mathbf{x})$ to map the data from the low dimensional space into a new feature space F of higher dimension. Conventional PCA is then performed on the transformed data matrix. This gives kernel PCA the property of extracting nonlinear features from the data-set and makes it a powerful tool in many applications.

However, an explicit mapping $\mathbf{T}$ is not always exist. In real practices the mapping is implicitly done by choosing a suitable kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ for data points $\mathbf{x}_i$ and $\mathbf{x}_j$. However, there is a problem when choosing the function $K(\mathbf{x}_i, \mathbf{x}_j)$ since not every function is guaranteed to satisfy the requirements of a feature space. An approach of choosing a qualified kernel function is to use the properties described in the Mercer's theorem [10] which states that any continuous symmetric function $K(\mathbf{x}_i, \mathbf{x}_j)$ that satisfies the positive semidefinite condition $\int_{X\times X} K(\mathbf{x}_i, \mathbf{x}_j) f(\mathbf{x}_i) f(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \geq 0$ is ensured to be a kernel for some feature space. This provides a broad way of choosing the kernel mapping functions. In this paper, we study the Gaussian kernel and the polynomial kernels in more detail for reasons described in the last section.

To extract the principal components of the mapped data, first a covariance matrix needs to be constructed for the mapped data. Suppose that the data $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ in space F is centred, then the covariance matrix of the mapped data in this space is:

$$\overline{\mathbf{C}} = \frac{\mathbf{1}}{\mathbf{m-1}} \sum_{\mathbf{i=1}}^{\mathbf{m}} \mathbf{\Phi}(\mathbf{x_i}) \mathbf{\Phi}(\mathbf{x_i})^{\mathbf{T}}$$

Since the explicit mapping $\mathbf{T}$ is probably unknown, computing the covariance matrix directly is not feasible. Schölkopf, Smola, and Müller showed in [7] that by solving the eigen-equation $m\lambda\alpha = K\lambda$ in which the eigenvalues are $m\lambda$, the p_{th} feature vector, corresponding to the projection of the p_{th} feature point on the eigenspace, takes the form $< v^p, \Phi(\mathbf{x}) >= \frac{1}{\sqrt{\lambda^p}} \sum_{i=1}^{m} \alpha_i^p k(\mathbf{x}_i, \mathbf{x})$, which can be further simplified to ([5])

$$< v^p, \Phi(\mathbf{x}) >= \frac{1}{\sqrt{\lambda^p}} (K\alpha^p)_n = \sqrt{\lambda^p} \alpha_n^p \tag{1}$$

To generalize the method to non-centered data, the kernel function K becomes [7, 5] $K' = (I - ee^T) K (I - ee^T)$ where $e = M^{-1/2}(1, 1, \ldots, 1)^T$.

Based on the interesting transformation invariants, two kernel functions, the Gaussian kernel and the polynomial kernel, are of interest in this work. The polynomial kernel has the form $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + c)^d$, where c and d are constants ($d \neq 0$). It captures the directionality of the data which should be very important for correspondence matching. However, the dot product is not invariant under object's scaling so there is still a magnitude problem we should consider. To solve this problem, one way is to normalize the scaled and truncated eigenvectors. Another method is to scale both of the two eigenvector matrices by the eigenvalue matrix of the model data-set. In this work, the latter method is chosen to eliminate this problem.

The Gaussian kernel has the form $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\{-d_{ij}^2/\sigma\}$ where d_{ij}^2 is a dissimilarity metric between the points $\mathbf{x}_i$ and $\mathbf{x}_j$, which usually takes the form of a Euclidean distance between these two points as used in this work.

Having the kernel functions described, we now introduce our point matching algorithm using kernel PCA techniques. We expect that a suitable kernel function will capture the object's properties and feature points can be embedded into a lower dimensional feature space in terms of the extracted properties thus provide a basis for one-to-one correspondence matching. In short, the following procedures are taken for performing point matching by kernel PCA:

- Build a matrix representation A for each image, where the matrix elements are computed by $A_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$, and $\mathbf{x}_i$ and $\mathbf{x}_j$ are two feature points from the same image and also let the matrix A be centred;
- Perform the eigendecomposition of A: $A\lambda = \lambda\alpha$, with λ the eigenvalues and α the eigenvectors for each proximity matrix;
- Use equation (1) to compute the projection of each feature in the eigenspace spanned by α. Only the first two eigenvectors are used in the polynomial kernels, and only the first three are used in the Gaussian kernels;
- The correspondences between two feature point-sets are the pairs which have the smallest Euclidean distance between them.
- The σ values in the Gaussian function are chosen automatically using the heuristic formula $\sigma = \frac{1}{0.09}\sum_{i=1}^{m}(\frac{1}{m}\sum_{j=1}^{m} d_{ij}^2)^2$, where d_{ij} are the Euclidean distances between point pairs $\mathbf{x}_i$ and $\mathbf{x}_j$.

One may find the above algorithm somewhat similar to the approach by Shapiro and Brady [9]. In [9], a proximity matrix A is first built for each image with the matrix elements computed as $A_{ij} = \exp\{-d_{ij}^2/2\sigma^2\}$, where d_{ij}^2 is the Euclidean distance between points $\mathbf{x}_i$ and $\mathbf{x}_j$, and σ is an adjustable parameter. Shapiro and Brady explain this as the mapping of the original two dimensional data to a higher dimensional space and thus capture structural information from the feature points. They then perform eigendecomposition on matrix A to obtain its eigenvalues and eigenvectors and to get a new modal matrix which has the descendant-sorted eigenvectors as its columns for each data set. The rows of the matrix are then considered as the projections of the feature points into the eigenspace. When the data sets are of different size, only the first M leading eigenvectors from each data sets are used where M is the size of the smaller data set. To make the algorithm more robust, Shapiro and Brady also suggest to use the eigenvalues to scale their corresponding eigenvectors and put more emphasis on the more significant eigenvectors. This acts in a more similar way as the kernel PCA. Comparing with the kernel PCA approach described above, one can see that in this way, Shapiro and Brady's method can be regarded as a special case of the kernel PCA approach, which assumes the data in the mapped space has a mean zero and uses the Gaussian as the kernel function. Figure 3 shows the performance of Shapiro and Brady's method and the kernel PCA with a Gaussian kernel.

4 Experimental Results

Experiments are designed to compare the matching performance of the aforementioned spectral point matching algorithms. In addition to the algorithms described above, the multidimensional scaling is also included in this section. MDS is also a method commonly used for data dimension reduction which is based on eigenvalues and eigenvectors of a dissimilarity matrix [4]. It attempts to preserve the pairwise relationship between data points while mapping the data into a low dimensional space. The experiments here of matching using MDS is performed using the classical MDS in which the Euclidean distance is taken as the dissimilarity measure.

Focuses of the experiments are on the performance of the algorithms when the data are under transformations and contains uncertainties such as outliers and random position jitter. For this purpose, the data in the experiments are designed as in the following subsection.

4.1 The Data

The experiments are taken on both synthesized data and real data sets and have the following designs:

1. Synthetic data: Assume two dimensional affine transformation. Given $X = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}\}$, a synthetic dataset $Y = (X + 5 \times \mathbf{1}\mathbf{1}^T) \times 0.6 \times R$ is generated for testing the algorithms, where $\mathbf{1} = (1, \ldots, 1)^T$, $R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$ is the rotation matrix and $\theta = \frac{10}{180} \times \pi$.
2. Real data: Here we use the hands sequence shown in Figure (1) and the CMU house sequence ([3]) shown in Figure (2).
3. Noisy data: A Gaussian noise is added to the data set to test the robustness of the algorithm. First A 2-D Gaussian random matrix $D \sim N(\mu, \Sigma)$ is generated, and then the data are added to the matrix of the second feature point set X_2 using the equation $X_2 = X_2 + D$
4. Data sets with different size: To simulate structural errors we delete l consecutive points, where $l = 1, \ldots, 5$, from the second data set (the test data). Also feature point sets in the CMU house sequence have different sizes. For frame 01, 02, 03, 04, and 10 displayed in Figure (2), the sizes of each data-set are 30, 32, 32, 30, 30, respectively.

4.2 The Results

To compare the performances of the kernel approaches when deformations are present, experiments are performed on synthetically generated data where a 2D translation, rotation and isoscaling are added. The effect of missing points and random point position jitter in terms of the 2-D Gaussian random matrices with different covariance matrices as described above are also tested. The results are shown in Figure (3). The experimental results of random point jitter are averages of 100 runs for each covariance matrix. In the experiments of different data sizes, at the beginning, both data sets have 30 points. The results of missing points are averaged over all 30 runs. The testing method of missing data are as described above.

The results of the algorithms on real data sets are displayed in Figure (3), and Table (1). The experiments on missing points are performed in the same

Table 1. Matching results (*Total error numbers*)

	Hand data				CMU House			
Frames	08 - 25	09 - 11	09 - 25	11 - 25	01 - 02	01 - 03	01 - 04	01 - 10
KPCA,Gaussian	30	7	22	21	17	21	4	17
KPCA,Polynomial	5	7	6	12	11	11	3	16
MDS	35	5	26	27	17	21	25	29
Shapiro&Brady	30	7	22	21	26	28	3	25

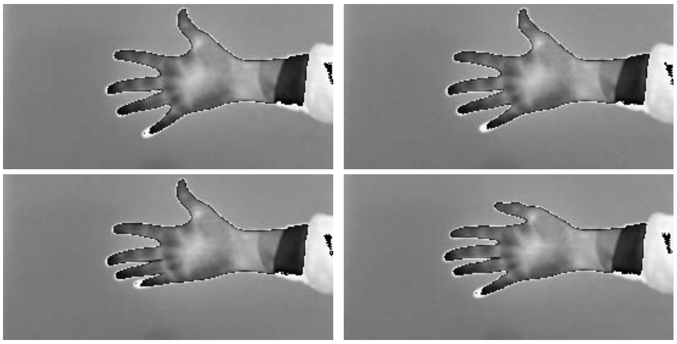


Fig. 1. The hand image data (*From left to right, up to down: frame 08, 09, 11, 25*)

way as for the synthetic data. The results are got from all 44 runs (since in this part, each data set has 44 points at the beginning).

In all the experiments using Shapiro and Brady's method, the eigenvalues are used to enhance their corresponding eigenvectors in order to improve the matching results.

From these experiments, we can see that the kernel PCA approach with a polynomial kernel gives the best results. Experiments on the CMU house data (table 1) also show that the polynomial kernel outperforms all the other algorithms. In all the experiments, the performance of kernel PCA with a Gaussian kernel and the Shapiro and Brady's method ([9]) similar due to their close relationship.

5 Discussion

In this paper we have explored the use of kernel PCA with a polynomial kernel function for finding correspondences between two feature point sets. A relationship with Shapiro and Brady's correspondence method [9] is also discussed. The experimental results reveal that the method offers performance advantages over a number of alternative methods. Besides of the plotted results, the polynomial kernel also shows a more stable performance in experiments of different data size. Even in worst cases it can still maintain a tolerable error rate. The performance of our algorithm is also comparable to the approaches in [2, 3, 6]. The relative

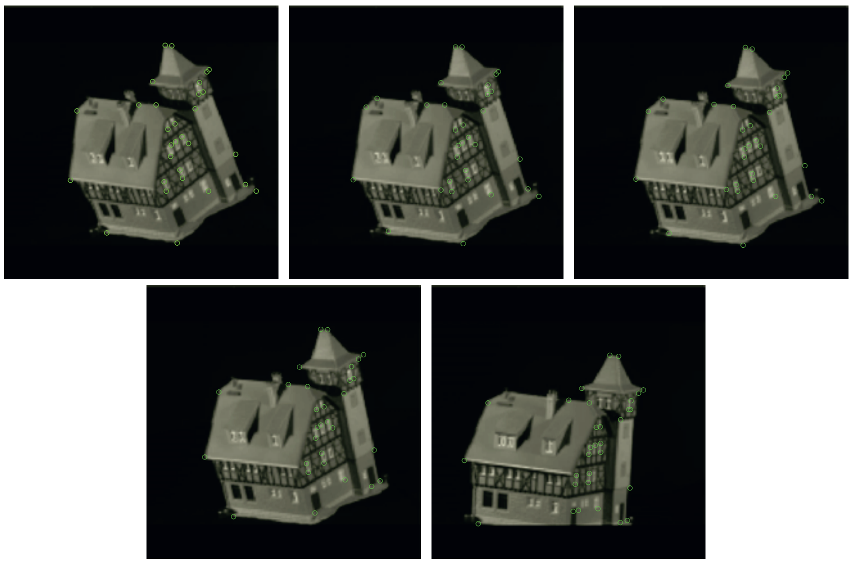


Fig. 2. The CMU house data (*From left to right, up to down: frame 01,02,03,04,10*)

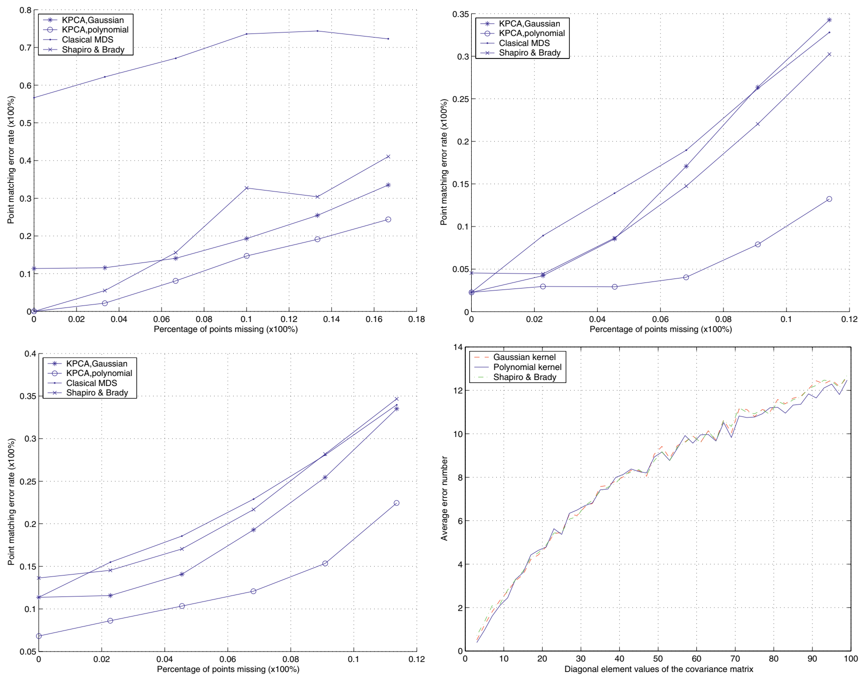


Fig. 3. Matching results (*From left to right, up to down: synthetic data; hand 08 and 09; hand 08 and 11; effects of Gaussian random position jitter*)

weakness of the polynomial kernel is that it requires slightly more computing time that other methods and this might be a problem when very big data sets are being matched and the time requirements is strictly restricted. However this is compensated by its non-iterative property. Comparing with previous iterative-based methods, its computing is more efficient.

In Gaussian kernels, the choose of the parameter σ is not an easy task. In [9], the value is chosen manually. In this paper, we use a heuristic formula based on each data-set's pairwise Euclidean distance matrix to compute the σ value automatically. In our experiments, this formula always chooses an appropriate σ value for different data sets.

Acknowledgements

This work was supported by the UK MOD Corporate Research Programme. The authors thank Dr. Andrew R. Webb for his encouragement and support.

References

1. Caelli,T. and Kosinov, S. "An eigenspace projection clustering method for inexact graph matching". *IEEE Tran. PAMI Vol.26, No.4, 2004*
2. M. Carcassoni and E. R. Hancock. "Spectral correspondence for point pattern matching". *Pattern Recognition. 36(2003) pp.193-204*
3. M. Carcassoni and E. R. Hancock. "Correspondence matching with modal clusters". *IEEE Tran. PAMI Vol.25 No.12, 2003*
4. T. F. Cox and M. A. A. Cox. "Multidimensional Scaling". *Chapman and Hall, London, 1994*
5. J. Ham, D. D. Lee, S. Mika, B. Schölkopf. "A kernel view of the dimensionality reduction of manifolds". *Max Planck Institute for Biological Cybernetics, Technical report TR-110, 2003*
6. B. Luo and E. R. Hancock. "Matching Point-sets using Procrustes alignment and the EM algorithm". *BMVC 1999*
7. B. Schölkopf, A. J. Smola, K. R. Müller. "Nonlinear component analysis as a kernel eigenvvalue problem." *Neural Computation*, 10:1299-1319, 1998.
8. G. L. Scott and H. C. Longuet-Higgins. An Algorithm for Associating the Features of Two Images. *Proc. Royal Soc. London Series B-Biological, vol.244, 1991.*
9. L. S. Shapiro and J. M. Brady. Feature-Based Correspondence - An Eigenvector Approach. *Image and Vision Computing, vol.10, pp.283-288, 1992*
10. V. N. Vapnik. "The nature of statistical learning theory". *Springer Verlag, New York, 1995*

Neural Networks for Improved Target Differentiation with Sonar

Naima Ait Oufroukh and Etienne Colle

Laboratoire Systèmes Complexes, 40, rue du Pelvoux
91020 Evry Cedex France
{Aitoufroukh.N,ecolle}@iup.univ-evry.fr

Abstract. This study investigates the processing of sonar signals with neural networks for robust recognition of indoor robot environment composed of simple objects (plane, corner, edge and cylinder). The neural networks can differentiate more targets with higher accuracy. It achieves this by exploiting the identifying features extracted from sonar signals. In this paper we compare two different architectures of neural networks (global and specialized structure) in term of classification rates, the best classifier obtained is used to recognize a robot environment. The results strengthen our claims that sonar can be used as a viable system for object recognition in robotics and other application domains.

1 Introduction

In contrast, typical robotics applications only use sonar as a range finder, measuring the time-of-flight of the leading edge of the ultrasonic echo to determine the distance of the object that generated the echo [1,2]. Target differentiation is of importance for intelligent systems that need to interact with and autonomously operate in their environment. Many works have studied ultrasonic sensors for this purpose [3,4]. Sonar is a very useful and cost-effective mode of sensing for mobile robots. The fact that sonar sensors are light, robust and inexpensive devices has led to their widespread use in applications such as map building [5], target tracking [6], and obstacle avoidance [7]. Although there are difficulties in the interpretation of sonar data due to poor angular resolution of sonar, multiple reflections and establishing correspondence between multiple echoes on different receivers [4], these difficulties can be overcome by employing an intelligent processing on sonar signals. This paper investigates the use of neural networks to process sonar signals encountered in target differentiation application for indoor environments. The motivation behind the use of neural network classifiers in sonar or radar systems is the desire to emulate the remarkable perception and pattern recognition capabilities of humans and animals [8], Carpenter used a Fuzzy ARTMAP neural network [9] to classify echoes from five objects. A comparison between neural networks and standard classifiers for radar-specific emitter identification is provided by Willson [10]. Neural networks have also been used in the classification of sonar returns from undersea targets [8]. The purpose of this paper is to recognize a real indoor environment by using intelligent processing and ultrasonic sensors. Before classification, we determine the most discriminant features set with several methods: sequential methods (Backward and Forward), optimal method (Branch and bound).

The classifier used is determined by a comparison of two neural networks. The networks considered are the Global Neural Network (GNN) and the Specialized Neural Networks (SNN).

2 Sonar System Description

The system we presented uses two readily available 6500-series Polaroid sonar with 80mm between centers. The sensors are mounted on the step by step motor (the step = 1.8°). It is contained in the copper box in order to avoid the interference between magnetic field and sonar signal. The sensors are employed as a transmitter/receiver (T/R) and a receiver (R). They are characterized by a 10m range and a beam aperture θ_0 of about $\theta_0 = \sin^{-1}(0.61\lambda/a)$. Where $\lambda = c/f_0$ is the wavelength of the acoustic signal, a = 14mm is the radius of the transducer and f_0 = 50kHz is the resonance frequency of the sensor. The most common sonar ranging system is based on the time-of-flight (TOF) which is the time elapsed between the transmission and the reception of a pulse. In commonly used TOF systems, an echo is produced when the transmitted pulse encounters an object and a range value $r = ct_0/2$ is produced with simple thresholding [5]. Here t_0 is TOF and c is the speed of sound in air (c=343,3 m/sec).

3 Target Differentiation

The target primitives modeled in this study are plane, corner, edge and cylinder with different dimension (Figure 1). Since the wavelength ($\lambda = 6.8$mm) is much larger than the typical roughness of used object surfaces, in this case, targets reflect acoustic beams specularly, like a mirror [5].

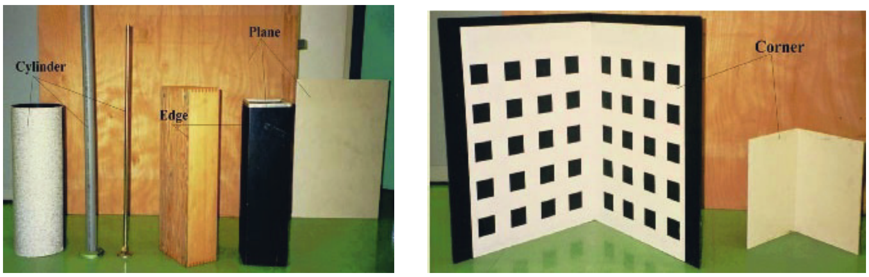


Fig. 1. Targets differentiated in this study.

4 Feature Vector

4.1 Dimensionality Reduction

There are two main reasons to keep the dimensionality of the pattern representation (i.e., the number of features) as small as possible: measurement cost and classification accuracy. A limited yet salient feature set simplifies both the pattern representation and the classifiers that are built on the selected representation. Consequently, the resulting classifier will be faster and will use less memory. Moreover, as stated earlier, a small number of features can alleviate the curse of dimensionality when the number of training samples is limited. On the other hand, a reduction of the number of features may lead to a loss in the discrimination power and thereby decreases the accuracy of the resulting recognition system. The classification depends strongly on the features selected to represent the sample. Feature extraction is achieved by an heuristic method. Good feature selection method is an essential step in a classification system [11], which permits to reduce the space of class representation. Among the existing criteria we use the following criteria:

$$J = \text{trace}\,(\hat{\Sigma}_W^{-1}\hat{\Sigma}_b) \tag{1}$$

Where $\hat{\Sigma}_w$ is the estimated intra group covariance matrix, which characterizes the dispersion of points in a class. $\hat{\Sigma}_b$ is the estimated inter group covariance matrix which characterizes the dispersion of classes between them.

We made comparison of two feature selection algorithms, with the rate of classification. Forward and Backward sequential selection (FSS, BSS) [12,13] are the most common sequential search algorithms. FSS begins with zero features, evaluates all subsets with exactly one feature and selects the one with largest criteria. BSS instead begins with all features and repeatedly removes a feature whose removal causes the least decrease of criteria. At the k step, we add or remove the feature ξ_j as:

$$J(\Xi_k \pm \xi_j) \geq J(\Xi_k \pm \xi_i) \quad \mathrm{i = 1, d - k} \quad \mathrm{i \neq j} \tag{2}$$

Where J is the criteria (3), Ξ_k is the whole set of features minus k features. The remaining parameters are most discriminative.

After a rough selection of parameters we apply the branch and bound method (BAB) to refine the result. This method examines all feature subsets. It will always find an optimal solution but with cost of computational time. We look for Ξ^* as:

$$\begin{cases} \Xi \subset \Xi^* = \left\{\xi^*, \ldots, \xi^*\right\} \\ J(\Xi^*) = Max(J(\Xi_{d'})) \end{cases} \tag{3}$$

The selected parameters contain the main features of the signal received by each sensor (Figure 3). We can regroup these parameters in four categories: the maximal amplitude, the shape, the energy and the difference of viewpoints between these two signals. This set is chosen among the 113 initial features extracted from the received signal, its envelope and its fast Fourier transformed (FFT) shown in Figure 2. The initial set of parameters component the maximum amplitude, time of flight (TOF), energy, length of the envelope and some Fourier transform coefficients. Because the device is composed of two sensors we also extract the differences between amplitudes, lengths and distances.

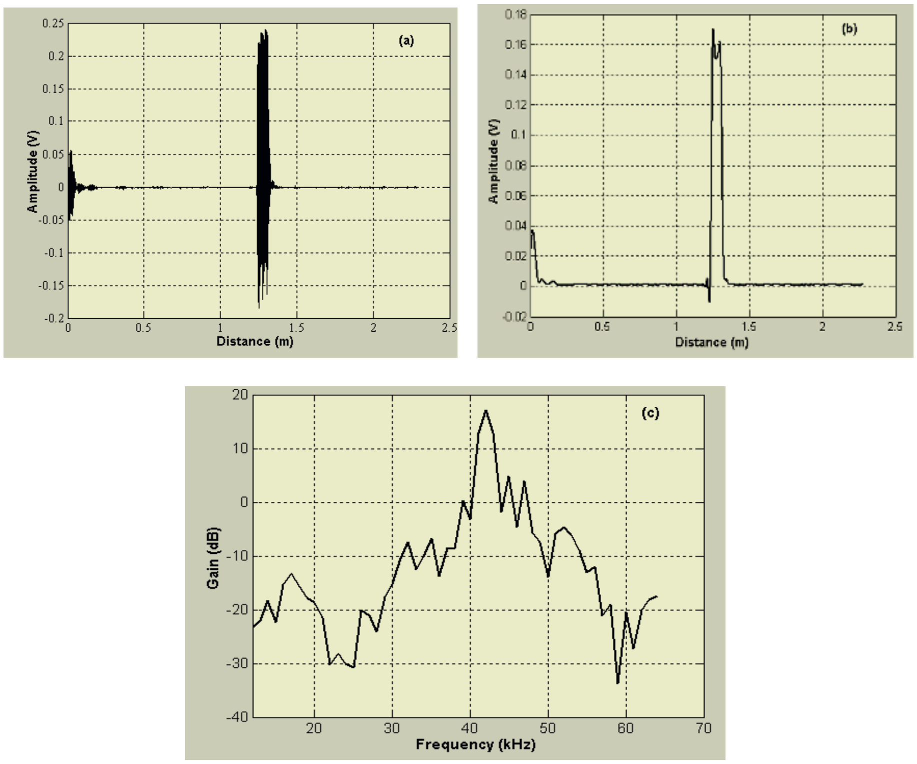


Fig. 2. Feature extraction : (a) real received signal, (b) the envelope of the echo and (c) fast Fourier transformed of the received signal.

5 Neural Network Classifier

We compare the classification rate of two supervised networks: Global Neural Network and Specialized Neural Networks. These networks used are a multilayer perceptron with the gradient back-propagation algorithm.

5.1 Global Neural Network (GNN)

The network employed has one hidden, one input and one output layer. The hidden layer comprises 18 neurons. This number is determined by the heuristic method, which consists to compare the generalization error for several networks architecture, we vary the hidden layer sizes and the input neurons. The number of output layer neurons is equal to the number of classes.

The output is represented by a posteriori probability (P_r) of neural classification evaluated by a Softmax function (Equation 1) [14]:

$$\Pr(k/x) = \frac{\exp(y_k)}{\sum_{j}^{K} \exp(y_j)} \tag{4}$$

y_j is the j output of the neural classifier,

K is a number of classes.

The maximal probability gives the membership class of example. The classifier is conceived after two phases: learning and generalization using ultrasonic data set.

5.2 Specialized Neural Networks (SNN)

This network is composed of several global neural networks (Figure 4). The number of networks is equal to the number of classes. Each network is specialized to recognize one class. The number of output layer neurons is equal to 2, the class to differentiate and the class "Others" which include all the other classes. Each network has a different architecture that we determine with the heuristic method.

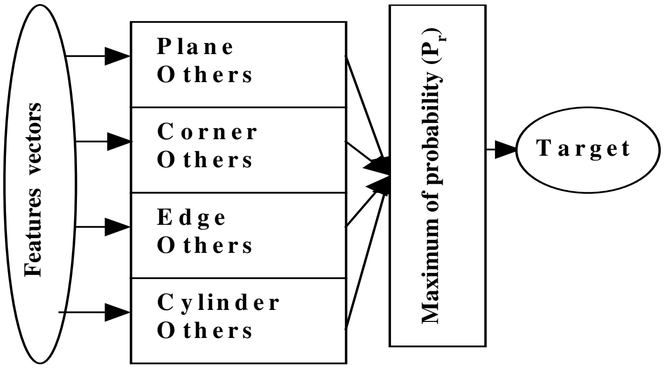


Fig. 3. Specialized Neural Networks architecture.

6 Results

6.1 Data Base

Two bases of measures have been done, one for the training and another for the generalization. Each one is composed of the four objects, plane, corner, cylinder and edge, placed at various distances from the sonar. The number of examples depends on the dimension of object. For the plane, we use two planes of 16cm and 122cm width, the corner 25cm and 48cm width, the edge 20cm and 43cm width and for the cylinder, we use three with different diameter: 16mm, 4cm and 20cm.

The range varies between 0.20m to 2.30m, with an increment of 20cm. The orientation of the sensor varies between ± 20° with an increment of 5° .

6.2 Determination of Discriminating Parameter

Choosing the feature vector dimension is usually a compromise between classification performance and computational time. We compare the FSS and BSS methods, according to the given rate classification by quadratic discriminant analysis (QDA). This method is based on the normal distribution:

$$f_k(X) = (2\pi)^{-p/2} |\Sigma_k|^{-1/2} e^{-\frac{1}{2}(X-\mu_k)^T \Sigma_k^{-1} (X-\mu_k)} \tag{5}$$

Where μ_k and Σ_k are the class k ($1 \le k \le K$) population mean vector and covariance matrix.

Entry Vector of GNN
We determine the features used like entry vector of GNN with different methods (FSS, BSS and BAB). After comparison between FSS and BSS, we find that FSS gives the best result in term of classification until 30 parameters. We apply the BAB method to refine the final set. We could not use them for more than 30 parameters because of the importance of the computation time.

Table 1. Features selection.

	FSS		**BAB**	
Parameters number	113	60	30	**5**
Mean rate of classification	94	90	80	**76**

The table 1 shows the mean rate of object classification with different number of parameter using QDA. We reduce the dimension of patterns to 5 parameters that are:

- Length of receiver echo, TOF of receiver (R), Difference between the two amplitudes,Two Fourier transform coefficients that correspond to resonance frequency : 50kHz of transmitter/receiver (T/R) and of receiver (R).

Entry Vector of SNN
The same methods are applied to obtain the entry vectors of each SNN network.

The SNN structure is illustrated in Table 2, where each target has a different number of features, selected from the initial features set (see section 4) with the different selection feature methods (FSS, BSS and BAB). We choose the features number that correspond to the best rate classification.

Table 2. SNN structures.

	Size of entry vector	**Classification rate (%)**
Plane	7	96
Corner	5	92
Edge	10	91
Cylinder	7	95

6.3 Classification

The GNN has one hidden layer which comprises 18 neurons. The number of output layer neurons is 4. The number of input layer neurons is equal to 5 illustrated in (§ 6.2). The SNN has a different number of hidden layer neurons for each object shown in Table 3. The number of output layer neurons is equal to 2.

Table 3. Hidden layer neurons.

	Hidden layer neurons
Plane	17
Corner	10
Edge	13
Cylinder	15

Table 4 shows the rate of good classification in generalization given by the two methods (GNN, SNN) with the different objects (plane, corner, edge and cylinder).

Table 4. Rate of good classification.

	GNN	**SNN**
Plane	71%	99%
Corner	73%	94%
Edge	56%	96%
Cylinder	60%	83%

The Specialized Neural Networks method (SNN) offers a best rate of classification for all the simple targets (plane, corner, edge and cylinder). This method gives better results because each network is specialized for one class and the estimation of the separation surface between two classes (target and others) is simpler than for the four classes.

6.4 Classification of Robot Environment

We tested SNN performances on a robot environment composed of simple targets set (plane, corner and cylinder) with different dimensions. There are three cylinders with different diameters (20cm, 4cm and 2cm).

After scanning the environment, the aim is to recognize the targets encountered by the sonar system. We classified the two echoes viewed on the sensor line of sight. We added an ambiguity reject class to the networks output in order to reduce the classification error. We reject an example in ambiguity when probabilities (given by equation 4) of the four networks outputs are equal. The result of classification is superposed on the environment and is shown in the Figure 4.

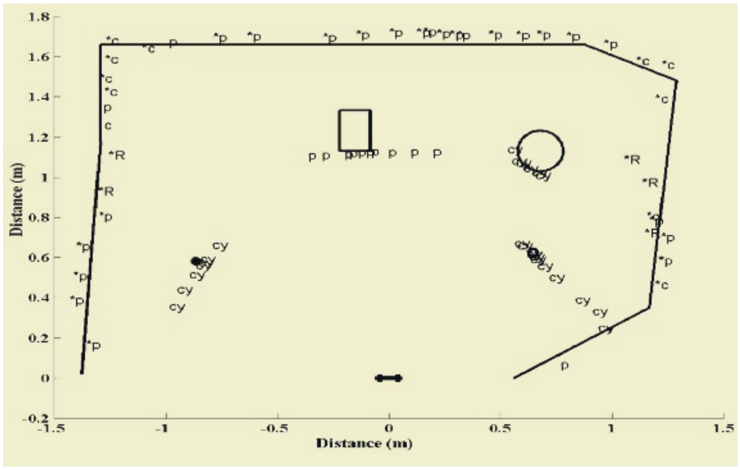


Fig. 4. Classification of a robot environment. Classification are represented by letters: c=corner, p=plane, cy=cylinder, *p: second plane (second echo encountered on the sensor line of sight) and R= rejected measure.

The expanded recognition areas are owed to the aperture of the emission signal (plane and cylinder). The error of classification is specially due to the occultation of some objects by others. In this case, the energy reflected by the object occulted is lower than the energy reflected by the first object located in front of the sonar. The classification error of the environment is 3.7%. The result showed in figure 6 verifies the performance of the SNN method to recognize the simple targets.

7 Conclusion

Neural networks are employed to process real sonar data after trained to learn identifying relations for the target primitives. In this study, The SNN method is compared to the GNN method to classify 2D indoor environment based on ultrasonic measures. The best result is obtained with the SNN method. The information included in the echo is sufficient to classify the environment in four classes (edge, corner, plane and cylinder) with good performances. The results confirm the value of sonar as a sensor for object recognition and suggested wider use of neural networks as robust pattern classifiers in sensor-based robotics. For future works it is important to consider the unsupervised learning algorithms to make the classification process more robust to changes in environmental conditions.

References

1. Borenstein, J., Everett, H., and Feng, L. Navigating mobile robots. A.K. Peters, Wellesley, MA. (1996).
2. Leonard, J. J.; and Durrant-Whyte, H. F. Directed sonar sensing for mobile robot navigation. The Kluwer international series in engineering and computer science. Kluwer Academic Publishers, Boston, (1992).
3. Barshan, B.;Kuc, R. Differentiating sonar reflections from corners and planes by employing an intelligent sensor. IEEE Trans. Pattern Anal. (1990).
4. Barshan, B., Ayrulu,B., Utete, S.W. Neural Network-Based Target Differentiation Using Sonar for Robotics Applications. IEEE Trans, Robotics and Automation, (2000), 435-442.
5. Bozma, Ö. ; Kuc, R. Building a sonar map in a specular environment using a single mobile sensor. IEEE Trans. Pattern Anal and Machine Intell, (1991).
6. Kuc, R. Three-dimensional tracking using qualitative bionic sonar. Robotics Autonomous Systems, (1993). 213-219.
7. Borenstein, J.; and Koren, Y Obstacle avoidance with ultrasonic sensors. IEEE Journal of Robotics and Automation, Vol. 4, N°.2, (1988). 213-218.
8. Roiblat, H. L., Au. W. W. L., Nachtigall. P. E., Shizumuru. R, and Moons. G., Sonar recognition of targets embedded in sediment, Neural Network, G. B. (1995) 1263-1273,
9. Carpenter et al. Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. IEEE Transactions on Neural Networks, 3(5), (1992), 698–712
10. Willson, Radar classification using a neural network, Proc. SPIE, optical Engineering and photonics in aerospace sensing, (1990). 200-210
11. Peremans, H., Audenaert, K., Van Campenhout, J. M.,. A high resolution sensor based en tri-aural perception. IEEE Trans on Robotic and Automation, CA, (1993)
12. Jain, K., Duin, P. W, Mao, J. Statistical pattern recognition : A review. IEEE Trans. Pattern Anal and Machine Intell, (2000).
13. Ait Oufroukh, N.; Colle, E. Pattern Recognition with Ultrasonic Sensor using Classification Methods. 1st Hybrid intelligent system (HIS), Chili, (2002), 673-680,
14. Bishop, C. Neural Networks for Pattern Recognition. Clarendon Press, Oxford, (1995).

An MCMC Feature Selection Technique for Characterizing and Classifying Spatial Region Data

Despina Kontos[1], Vasileios Megalooikonomou[1], Marc J. Sobel[2], and Qiang Wang[1]

[1] Department of Computer and Information Sciences
Temple University, 1805 N.Broad St., Philadelphia, PA 19122, USA
{dkontos,vasilis,wang32}@temple.edu
[2] Department of Statistics, Temple University, 1810 N.13th Street
Philadelphia, PA 19122, USA
sobel@sbm.temple.edu

Abstract. We focus on characterizing spatial region data when distinct classes of structural patterns are present. We propose a novel statistical approach based on a supervised framework for reducing the dimensionality of the initial feature space, selecting the most discriminative features. The method employs the statistical techniques of Bootstrapping simulation, Bayesian Inference and Markov Chain Monte Carlo (MCMC), to indicate the most informative features, according to their discriminative power across the distinct classes of data. The technique assigns to each feature a weight proportional to its significance. We evaluate the proposed technique with classification experiments, using both synthetic and real datasets of 2D and 3D spatial ROIs and established classifiers (Neural Networks). Finally, we compare our method with other dimensionality reduction techniques.

1 Introduction

Feature selection is a very important process for analyzing patterns in spatial data. In certain application domains, such as in geography, meteorology or medicine, we seek to focus on specific Regions of Interest (ROIs) that occupy a small portion of the data and extract informative features [1]. Examples of such ROIs are areas with high levels of precipitation in meteorological maps and brain regions of high activity in fMRI[1] (see Figure 1). A well-known characterization technique is to map data using the extracted features into points in a *K*-dimensional (*K*-d) space [2].
When dealing with spatial patterns, shape is one of the main characteristics that needs to be represented. Several approaches have been used for this purpose [3]. To obtain the initial characterization vectors here, we use an approach initially presented in [4] that considers properties of internal value of ROIs in addition to their shape. This method works particularly well for non-homogeneous as well as for homogeneous ROIs. It efficiently forms a *K*-d feature vector using concentric spheres in 3D (or circles in 2D) radiating out of the ROI's center of mass and extracting quantitative information regarding both its structure and content. In several cases though, the number of features extracted is too large to support efficient pattern analysis and classification.

[1] Functional-Magnetic Resonance Imaging: shows physiological activity in brain.

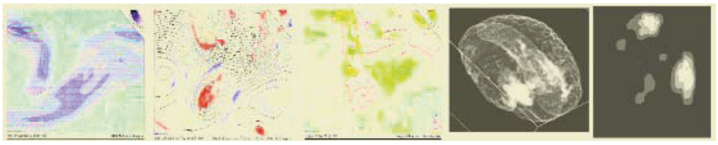


Fig. 1. Examples of geographical / meteorological and medical 2D and 3D spatial ROIs.

Several techniques have been proposed for reducing the dimensionality of data [5]. These approaches can be separated into two categories: (i) those having the property of transforming the initial features introducing a completely new subspace and, (ii) those that attempt to find an optimal subset of the initial features that are considered to be more significant.

Principal Component Analysis (PCA), also known as Singular Value Decomposition (SVD), is the most widely used technique from the first category due to its conceptual simplicity and efficient computation. It has been extensively used in many applications, such as medical image pattern analysis [6]. Multidimensional Scaling (MDS) is another dimensionality reduction technique with wide applicability [7]. The Discrete Fourier Transform (DFT) [8] and Wavelet Transform [9] have also been applied. Algorithms in the second class search for an optimal subset of the initial vector attributes, rather than a transformation. A well-known technique is forward feature selection that seeks to find an optimal subset of features [10]. Other approaches [11] combine the process of attribute selection with the induction algorithm used for classification. Statistical pattern recognition techniques have also been proposed [12]. Although proven effective, the first class of techniques fails to preserve the initial attribute values; the new feature vectors do not correspond to real data measurements. This introduces greater difficulty in interpreting the conceptual representation of the new feature space.

Our approach shares mostly characteristics of the second class of feature selection techniques and is based on a statistical framework that employs Bootstrapping, Bayesian inference and the Markov Chain Monte Carlo (MCMC) techniques. Our method applies to cases where distinct classes of data are present and a training set of labeled instances is available. In the particular case examined here, the level of the discriminatory significance of features varies across the classes of the observed data. These statistical techniques are used to select the most significant features, according to their discriminative power across the distinct classes of data, giving rise to a significant reduction in dimensionality.

2 Methodology

The general idea of the proposed feature selection technique is based on the assumption that the classes are generated by distinct structural pattern distributions reflected by the characterization vectors for each class. After learning a model/distribution for each class (in fact, a posterior over the models) using probabilistic modeling, bootstrapping and Bayesian inference, we find the features that are generated significantly differently under each model/class. We observe a training set T consisting of a number n_j of objects (ROIs) $O_{i,j}$, $i=\{1,\ldots, n_j\}$ of class j, $j=\{1,\ldots,M\}$. Each object is characterized by a feature vector of size K. That is, $O_{i,j}=(f_{i,j}[1],\ldots,f_{i,j}[K])$. We would

Table 1. Symbol Table.

$r_1,\ldots,r_K$	Radii
$O_{i,j}$	Object $i=1,\ldots,n_j$ of class $j=1,\ldots,M$
$f_{i,j}[k]$	Feature $k=1,\ldots,K$, of object $O_{i,j}$
$\Delta f_{i,j}=\left(\Delta f_{i,j}[2],\ldots,\Delta f_{i,j}[K]\right)$	Observed probability vectors for object $O_{i,j}$
$\Delta f_j=\frac{1}{n_j}\sum_{i=1}^{n_j}\Delta f_{i,j}$, $j=1,\ldots,M$	Observed probability vector for objects of class j
$p_j=(p_j[2],\ldots,p_j[K])$, $j=1,\ldots,M$	Parametric probability vector for objects of class j
$num_j=(num_j[2],\ldots,num_j[K])$, $j=1,\ldots,M$	Combined bootstrap count vector for objects of class j
N_j, $j=1,\ldots,M$	Combined bootstrap sample sizes for class j

like to use these features to determine appropriate ways to distinguish between different classes. In the presentation of the proposed feature selection technique we assume that these features are given by the characterization procedure [4] described briefly in Section 1, although with appropriate preprocessing any other characterization procedure can be used instead. In this case, feature $f_{i,j}[k]$ corresponds to the fraction of the object $O_{i,j}$ occupied by a sphere of radius r_k (the fraction of the sphere occupied by the object $O_{i,j}$ can be used as well). Let us consider consecutive features $f_{i,j}[k]$, $f_{i,j}[k\text{-}1]$ corresponding to radii r_k, r_{k-1} respectively. The difference between such features, calculated as a proportion of the total feature difference, is $\Delta f_{i,j}[k]=(f_{i,j}[k]-f_{i,j}[k-1])/(f_{i,j}[K]-f_{i,j}[1])$, where k=2,...,K. Note that after this normalization, the fractional proportions satisfy the relationship: $\sum_{k=2,..,K}\Delta f_{i,j}$ and the components of $\Delta f_{i,j}$ can be treated as probabilities. Let $\Delta f_{i,j}$ = $(\Delta f_{i,j}[2],\ldots,\Delta f_{i,j}[K])$ be the *observed probability vector* attached to object i of class j (i=1,…,n_j; j=1,…,M). We will consider these vectors to be the vectors representing (characterizing) the initial ROIs. We employ also the notation $\Delta f_j = {}^1\!/_{n_j}\sum_{i=1}^{n_j}\Delta f_{i,j}$, (j=1,.., M) for the *observed probability vector* for objects of class j.

We assume that the observed probability vectors Δf_j arising from class j are generated by a (parametric) probability vector: p_j=(p_j[2],…,p_j[K]), j=1,…,M. These are the "true" apriori values for the observed probabilities atattached to objects of class j, capturing the spatial pattern of the corresponding class. The procedure for selecting the most discriminative features is as follows:

1. Bootstrapping is done by 'sampling' a large number, B, of instances from each observed probability vector $\Delta f_{i,j}$ of class j. Each sample consists of different features, selected according to their component probabilities, which are actually equal to the

feature values after the normalization step. Under the assumptions that (i) the feature vectors for different objects are mutually independent, and (ii) the observed probability vectors $\Delta f_{i,j}$ arising from each class are well characterized by parameters, the count vectors obtained from this sampling are easily combined to form an approximate probability distribution for each class. We use the notation $num_j[k]$ to denote the number of times feature f_k, k=1,…,K is chosen by the sample for class j, j=1,…,M.

2. We also employ the notation N_j; $j=1,...,M$ for the total bootstrap sample size used for sampling from class j; $j=1,...,M$.
3. We assume that the components of the observed probability vectors $\Delta f_{i,j}$ (i=1,...,n_j; j=1,...,M) are mutually independent, conditional on the true probability vectors p_j=(p_j[2],…,p_j[K]), j=1,…,M i.e.,

$$P(\Delta f_j \mid p_j) \approx \prod_{i=1}^{n_j} P(\Delta f_{i,j} \mid p_j); \quad P(\Delta f \mid p) = \prod_{j=1}^{M} P(\Delta f_j \mid p_j) \tag{2.1}$$

We note that, although it is tempting to assume that the observed probability vectors have a multinomial (or multinomial-type) distribution in the parameters $p_j[k]$; k=2,...,K , j=1,…,M, this is not possible in view of the fact that the components of the observed probability vectors are not integers. An alternative approach, which we propose here, involves constructing an approximate multinomial likelihood for the observed probability vectors using the bootstrap counts, $num_j[k]$ (k=1,..,p; j=1,…,M) and basing inference on this likelihood. We make the assumption that the likelihood of the observed probability vectors takes this form. This assumption is tantamount to assuming that the bootstrap sample sizes N_j (j=1,..,M) are 'large enough' to have the property that the vector, $N_j * \Delta_j$, has components all of which are positive integers. The multinomial probability density function (see Equation 2.1) in the bootstrap count takes the form:

$$P(num_j \mid p) = \prod_{k=1,..,K} \binom{N_j}{num_j[1]...num_j[K]} p_j[k]^{num_j[k]} \quad ,j\text{=1,..,M} \ .$$

$$P(num \mid p) = \prod_{j=1}^{M} P(num_j \mid p) \, , \ \ j\text{=1,..,M} \ . \tag{2.2}$$

3. We postulate a parameter λ_j with (apriori) mean value $1/N_j$ having the following property: the observed probability vectors $\Delta f_{1,j},..,\Delta f_{n_j,j}$ are mutually independent with a likelihood similar to that of λ_j*num_j (j=1,…,M). This is easily done by assuming an exponential prior with density λ_j*exp{-λ_j*N_j}. After inserting this parameter, the multinomial likelihood is transformed into:

$$P(num_j \mid p, \lambda_j) = \left(1 \Big/ \sum_{j=1}^{M} p_j^{\lambda_j}[k] \right)^{N_j} \prod_{k=1,\dots,K} \binom{N_j}{num_j[1]\dots num_j[K]} p_j[k]^{\lambda_j * num_j[k]} \ . \tag{2.3}$$

where j=1,..,M. We assume therefore that :

$$P(\Delta f_j \mid p, \lambda_j) \approx P(num_j \mid p, \lambda_j)\,; \quad P(\Delta f \mid p) = \prod_{j=1}^{M} P(\Delta f_j \mid p) \ . \tag{2.4}$$

The posterior distribution of the parametric probability vectors $p_1,..,p_M$ and scaling parameters, $\lambda_1,..,\lambda_M$ (calculated using the approximate likelihood (2.2)) is complicated in this case. We have evaluated it using Markov Chain Monte Carlo (MCMC). We note that the variability in estimates of the true probability values arises in Equation (2.4) from the variability in the (normalized) bootstrap counts. This variability decreases as the bootstrap sample sizes increase (by the law of large numbers). Equation (2.4) is used in MCMC simulations to update the p's. We also note that as the bootstrap sample size increases, the distribution of the (scaled) combined count statistics approach that of the mean observed probability attribute vectors, making our model asymptotically correct.

4. We distinguish the best features (corresponding to radii) by evaluating a measure of variation $Var[k]$ for the p's at each radius k;

$$Var[k] = \sum_{j=1}^{M} (p_j[k] - \overline{p}[k])^2; \quad \overline{p}[k] = {}^{1}\!/\!{}_{M} \sum_{j=1}^{M} p_j[k] \ . \tag{2.5}$$

We distinguish the best one over each posterior simulation by choosing that feature $f_{k'}$ corresponding to radius $r_{k'}$ having the property that:

$$Var[k'] = \max\{Var[k]; k = 1,.., K\} \ . \tag{2.6}$$

The feature having the maximum variation over the greatest number of posterior simulations (calculated using MCMC) is deemed the best. This is justified by the fact that a high degree of variance for a specific attribute across the distinct labeled classes (inter-class variance) indicates a high degree of dissimilarity in the spatial pattern at the specific radius increment. Hence, the attribute can be considered to be highly informative with respect to class membership. Also, employing a large number of bootstrap samples B reduces high variance that might exist due to noise.

3 Experimental Results

All the experiments were implemented in Matlab using the Statistics v.3 toolbox of Mathworks. For classifiers we used Neural Networks implemented by the *PRTools v.3.1* toolbox for Matlab [13].

3.1 Artificial Data

We used artificial data sets that were generated using a parametric growth model for spatial ROIs introduced in [4]. The main idea is that the growth process begins with one initial voxel (or cell) at time t=0 and progresses using an "infection" procedure (see Figure 3), where each infected cell may infect its non-diagonal neighbors with some probability. The datasets are the following: (i) 2DHom: 100 2D Homogeneous ROIs, 50 spherical and 50 elongated to two opposite directions (north-south), with 14- feature characterization vectors (see Figure 2(b)-(c)), (ii) 2DNonHom: 100 2D Non-Homogeneous ROIs, 50 elongated to the one direction (north) and 50 elongated to two opposite directions (north-south), with 14 - feature characterization vectors (see Figure 2(d)-(e)), (iii) 3DHom: 100 3D Homogeneous ROIs, 50 spherical and 50 elongated to two opposite directions (north-south) , with 7 - feature characterization vectors and (iv) 3DNonHom: 100 3D Non-Homogeneous ROIs, 50 spherical and 50 elongated to two opposite directions (north-south), with 14 - feature characterization vectors. Each dataset consists of two distinct labeled classes of spatial pattern.

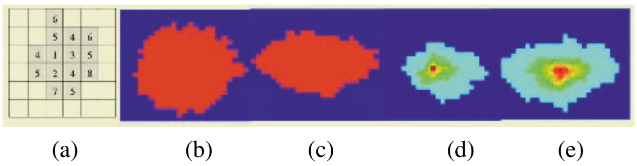


Fig. 2. (a) a sample of the growth process, 2D samples of (b),(c) Homogeneous, and (d),(e) Non-Homogeneous ROIs used in our experiments.

Using these artificial datasets we run a set of basic experiments. We tested 4 different combinations of bootstrapping sample size *B* and number of MCMC posterior simulations. For each of the combinations, we discovered the most discriminative attributes for the ROIs of each set and assigned a weight to them in the range [0...1] that indicates their discriminative power. Figure 3 (a)-(d) illustrates these results. A basic observation from this first set of experiments is that, by increasing the number of bootstrap sample *B* the method discovers less attributes each with a more significant weight. On the other hand, reducing the number of bootstrap sample *B* and increasing the number of MCMC simulations tends to spread the weights to more attributes, with a less significant weight factor to each individual attribute.

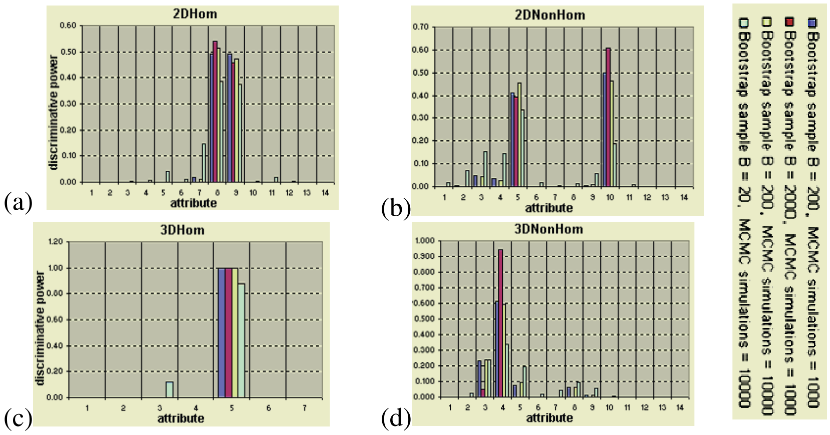
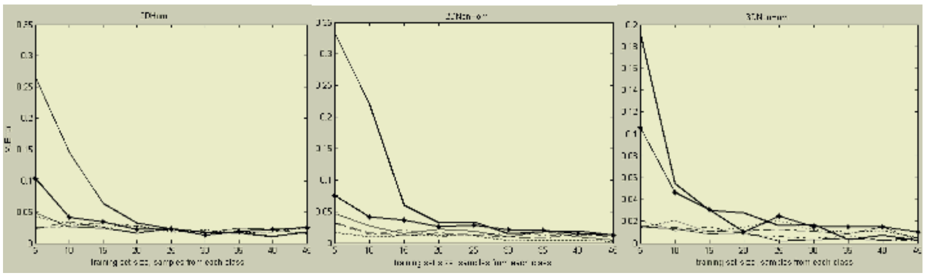


Fig. 3. Discriminative power of attributes discovered by the proposed method for (a) **2DHom**, (b) **2DNonHom**, (c) **3DHom** and (d) **3DNonHom** datasets and for different combinations of Bootstrap sample size B and number of MCMC simulations.

We continue with classification experiments using these selected discriminative features. The neural network consisted of one hidden layer with 5 neurons, number of inputs equal to the number of attributes used in each case, and one output indicating



(a) (b) (c)

━━━━	All features of the characterization vector.
—— ——	Selected features only, for B = 200, MCMC simulations = 1000.
- - - - - - -	Selected features only, for B = 2000, MCMC simulations = 1000.
— - — -	Selected features only, for B = 200, MCMC simulations = 10000.
————	Selected features only, for B = 20, MCMC simulations = 10000.
━━●━━	SVD features

Fig. 4. Neural network classification performance (mean error) when using all features, features obtained by SVD and features obtained by the proposed method for (a) 2DHom, (b) 2DNonHom and (c) 3DNonHom datasets.

the class. The training was performed using the Levenberg-Marquardt optimization and the training set size ranged from 5 to 45 samples from each class (two classes of 50 ROIs each in every set). We report the curves of mean classification error after 40 repetitions for all different sizes of the training data set. We also include the comparative classification performance using (i) all the attributes of the $\Delta f_{i,j}$ characterization vectors and (ii) the first 5 (2D data) and 3 (3D data) most significant components of the SVD transformation applied on the $\Delta f_{i,j}$ characterization vectors. Figure 4 (a)-(c) shows the classification performance for the various cases. A first observation is that in any case the classification error is very small. The classification performance when using only the discriminative features selected by the proposed approach is, in almost all cases, comparable to (or better than) that of using all the features of the characterization vectors or those features obtained by SVD.

3.2 Real Data

We experimented using ROIs extracted from 3D fMRI brain activation contrast maps. The fMRI scans were obtained from a study designed to explore neuroanatomical correlates of semantic processing in Alzheimer's disease [14]. For the experiments pre sented here, we focused only on a specific region of the brain that has been shown to be highly associated with the development of Alzheimer's disease [15]. The dataset consisted of 9 control and 9 patient 3D ROIs and the characterization vectors were constructed using 40 features. Figure 5 shows this ROI in consecutive 2D slices of the 3D volume.

We applied the proposed feature selection technique for sample size $B = 400$ and number of MCMC posterior simulations = 1000. We perform classification experiments, using the discriminative features selected. To avoid overfitting due to a small training dataset (9 controls vs. 9 patient samples) we applied one-layer perceptron networks trained by the Pocket algorithm and leave-one-out cross validation. We repeated the process of training and testing for 10 times and report the average accuracy. The first row of Table 2 includes the classification results when using all the discriminative features selected, as well as subsets of them based on the significance

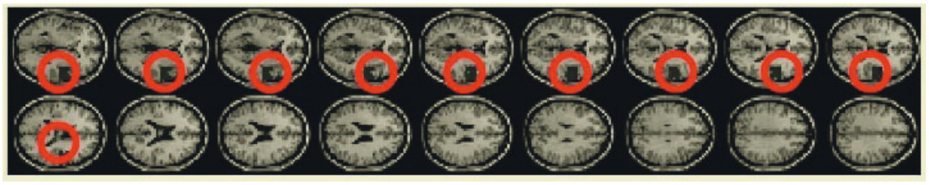


Fig. 5. The ROI used for applying the proposed feature selection technique. It is shown in consecutive 2D slices after being overlaid on a canonical brain atlas.

Table 2. Classification performances for the fMRI ROIs.

	The best features (13) ($w_k > 0$)	The best 7 features ($w_k > 0.005$)	The best 5 features ($w_k > 0.02$)	The best 2 features ($w_k > 0.2$)	INITIAL All 40 features
MCMC	85.56 %	**84.44 %**	**87.22 %**	**82.78 %**	82.22 %
FFS	**86.67** %	**84.44** %	82.22 %	81.11 %	

weights, w_k, obtained by the proposed approach. The second row of Table 2 shows the comparative results when using the forward feature selection (FFS) approach. In all the experiments we used the initial characterization attribute values. The accuracy obtained when using all the initial 40 features is also reported. It is interesting to observe that the proposed MCMC approach behaves better than the forward feature selection technique (greedy approach), especially when using only the highly discriminative features ($w_k > 0.02$); it is comparable to the greedy approach in all other cases.

4 Conclusions

We presented a novel dimensionality reduction technique which employs the statistical framework of Bootstrapping simulation, Bayesian inference and Markov Chain Monte Carlo (MCMC). The method applies when labeled distinct classes of spatial ROIs are available, aiming to select the most informative features with respect to class membership. The proposed approach assigns a weight to each selected feature revealing its discriminative power. We experimented both with synthetic and real data performing classification experiments using both all the initial characterization attributes and only the selected ones by the proposed method). We compared the proposed approach with SVD and forward feature selection. We concluded, on the data we experimented with, that the proposed approach always outperforms SVD. Also, it is better than forward feature selection as the number of selected features is reduced, making it a better alternative over the greedy approach. Finally, the proposed technique was shown to be effective when applied on real data. In this case the proposed technique performed better than the forward feature selection approach, especially when using highly discriminative attributes, while being comparable in other cases.

Acknowledgement

The authors would like to thank A. Saykin for providing the fMRI dataset and clinical expertise and H. Dutta for working on preliminary experiments. This work was supported, in part, by NSF research grants IIS-0083423 and IIS-0237921 and by NIH Research Grant #1 R01 MH68066-01A1 funded by the National Institute of Mental Health and the National Institute of Neurological Disorders and Stroke and the National Institute on Aging.

References

1. Megalooikonomou, V., Ford, J., Shen, L., Makedon, F., Saykin, A.: Data mining in brain imaging, Statistical Methods in Medical Research, Vol. 9 (4) (2000) 359-394
2. Guting, R.H.: An Introduction to Spatial Database Systems, VLDB Journal 3 (4) (1994) 357-399
3. Loncaric, S.: A Survey of Shape Analysis Techniques. Pattern Recognition, Vol. 31 (8) (1998) 983-1001
4. Megalooikonomou, V., Dutta, H., Kontos, D.: Fast and Effective Characterization of 3D Region Data, In Proc. of the IEEE International Conference on Image Processing (ICIP) 2002, Rochester, NY (2002) 421-424
5. Carrerira-Peripinan, M.A.: A review of Dimension Reduction Techniques, Technical Report CS-96-09, Dept. of Computer Science, University of Sheffield (1997)
6. Petrakis, E.G.M., Faloutsos, C.: Similarity Searching in Medical Image DataBases, IEEE Trans. on Knowledge and Data Engineering, Vol. 9 (3) (1997) 435-447
7. Kruskal, J.B., Wish, M.: Multidimensional scaling, SAGE publications, Beverly Hills (1978)
8. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases, In Proc. of the ACM SIGMOD Int.Conf. on Management of Data, Minneapolis, MN (1994) 419-429
9. Chan, K.P., Fu. A.C.: Efficient time series matching by wavelets, In Proc. of the Intl. Conference on Data Engineering ICDE , Sydney, Australia (1999) 126-133
10. Fukunaga, K.: Introduction to Statistical Pattern Recognition, 2nd Ed., Academic Press, New York (1990)
11. Kohavi, R., John, G.: Wrappers for Feature Subset Selection, Artificial Intelligence, Vol. 97 (1-2) (1997)273-324
12. Jain, A., Duin, P., Mao, J.: Statistical pattern recognition: A review, IEEE Transactions on PAMI, Vol. 22 (1) (2000) 4-37
13. Duin, R.P.W.: A Matlab Toolbox for Pattern Recognition, PRTools Version 3.0 (2000)
14. Saykin, A.J., Flashman, L.A., Frutiger, S.A., Johnson, S.C., Mamourian, A.C., Moritz, C.H., O'Jile, J.R., Riordan, H.J., Santulli, R.B., Smith, C.A., Weaver, J.B.: Neuroanatomic substrates of semantic memory impairment in Alzheimer's disease: Patterns of functional MRI activation, Journal of the International Neuropsychological Society, Vol. 5 (1999) 377-392
15. Megalooikonomou, V., Kontos, D., Pokrajac, D., Lazarevic, A., Obradovic, Z., Boyko, O., Saykin, A., Ford, J., Makedon, F.: Classification and Mining of Brain Image Data Using Adaptive Recursive Partitioning Methods: Application to Alzheimer Disease and Brain Activation Patterns, Human Brain Mapping Conference (OHBM'03), New York, NY (2003)

Human Action Recognition by Inference of Stochastic Regular Grammars

Kyungeun Cho, Hyungje Cho, and Kyhyun Um

Department of Computer and Multimedia Engineering
Dongguk University, Pildong 3 ga 26, Chunggu, Seoul, 100-715, Korea
{cke,chohj,khum}@dgu.ac.kr

Abstract. In this paper, we present a new method of recognizing human actions by inference of stochastic grammars for the purpose of automatic analysis of nonverbal actions of human beings. We applied the principle that a human action can be defined as a combination of multiple articulation movements. We measure and quantize each articulation movements in 3D and represent two sets of 4-connected chain code for xy and zy projection planes, so that they are appropriate for the stochastic grammar inference method. This recognition method is tested by using 900 actions of human upper body. The result shows a comparatively successful achievement of 93.8% recognition rate through the experiments of 8 action types of head and 84.9% recognition rate of 60 action types of upper body.

1 Introduction

Human action recognition is an active area of research in pattern recognition. Medical analysis of human gait movement, nonverbal communication in social psychology, VR using avatar control, automatic man-machine interaction, development of surveillance systems, sign-language recognizer, choreographic analysis of dance and ballet, and gymnastic movement - all belong to this application area of automatic human action recognition. In some areas, action recognition systems are already established such as the Chinese Sensei system analyzing Tai-Chai recognizing and translating sign language [1,2].

Practically, there are manifold phases in recognizing human actions in videos including tracking of human, separation of human bodies from the background, identification of body parts, and recognition of human actions [2]. There are various approaches to the recognition of human actions such as Dynamic Time Warping (DTW), template matching method, fuzzy method [2], Hidden Markov Model (HMM) and syntactic method [3,4] etc.

In a previous research, Stochastic Context Free Grammar as one of the syntactic method was used to the recognition system of human actions [3]. The system consists of an HMM bank and a probabilistic Earley-based parser. Grammar inference was referred to as the further study in their framework of stochastic parsing. In this paper, we show that a stochastic regular grammar inference method can resolve this problem because it has no much limit in inferring the grammar.

This paper presents a recognition scheme to analyze human actions on 3D temporal data of video where an unsupervised inference procedure is introduced to stochastic grammars. This scheme is based on the principle that a human action is defined as a combination of multiple articulation movements which is built up from multiple mutually-synchronized temporal data [5]. Human action is considered as a stochastically predictable sequence of states [3]. So, we apply a mixed statistical-syntactic approach to the recognition of human upper body action. In addition, we use a mechanism to infer stochastic grammars, which deals with learning the production probabilities for a set of given grammars. A grammar represents single human action.

The remainder of this paper is organized as follows: Section 2 recalls the theoretical concepts and notations of stochastic grammar inference method. Section 3 presents a data representation for human action sequence used in our work. Section 4 explains an overview of action recognizer and describes its main functions. Section 5 shows experimental results and analysis. Finally, in the last section we provide some discussion about our contribution and its future works.

2 Inference of Stochastic Grammar

In our work, we apply an inference of stochastic grammar scheme to recognize human actions. In this section we concisely describe the appropriateness of our method for human action recognition and the theoretical concepts and notations of stochastic grammar inference method.

2.1 Syntactic Pattern Recognition for Human Actions

Syntactic pattern recognition is a method focused on the structure of pattern. The recognition method dismantles an objective pattern into simple subpatterns to recognize and explain relations among the subpatterns with a grammar theory of formal language. In general a movement of one body part in a human action can be described as a sequence of subpatterns, such as left-left-left-right-right-right-up-down. In this case left, right, up, down are subpatterns that construct a movement of one body part. If a subpattern sequence includes transformations, noises, observatory errors, and incomplete feature extractions, etc., it is very difficult to express all actions in simple grammar although they have the same meaning. Stochastic grammar reflects these features most effectively, by applying probability to each grammar [6,7,8].

Previous researches have been conducted to recognize all actions after the composition of some patterns to produce each grammar class. They need intricately huge labors to extract features of every object for its recognition and to artificially grammaticize distinctive ingredients. However, computers can construct standard patterns and automatically accumulate knowledge using an existing technique. A research has been actively performed on how an unknown pattern is recognized through accumulated knowledge. Syntactic recognition through grammatical inference is applicable to this case [9]. An application of grammatical inference was implemented in the field of music processing for modeling musical style. The models were used to generate and classify music by style [10].

Stochastic grammar inference is a recognition method that satisfies grammatical inference and stochastic grammar. This has already been implemented in several studies, such as normal or abnormal chromosome recognition [7], digit and shape recognition [11], text and speech recognition [12, 13] etc. Our study is an attempt to apply stochastic grammar inference to human action recognition.

2.2 Theoretical Concepts and Notations

In this subsection we introduce the inference of stochastic grammar method to seek the probability value of each production with given grammar and learning patterns. The given learning patterns are composed of subpatterns that respectively produce different grammar and probability value of each grammar.

Let's consider an M-class problem characterized by the stochastic grammars G_{sk}□$(N_k, \sum_k, P_k, D_k, S_k)$ for k□1, 2, ..., M. N_k is a finite set of nonterminals, $\sum_k$ is a finite set of terminals, P_k is a finite set of productions, D_k is a set of probability values of the production to be assumed, and S_k means the starting symbol [6]. In this paper, we define grammars for the actions of human upper body such as follows.

G_{sk}□(N, Σ, P, D_k, S), $k \in \{1...M\}$, N□{S, R, L, U, D}, Σ □{right, left, up, down},

P = { S -> right R	R -> right R	L -> right R	U -> right R	D -> right R
S -> left L	R -> left L	L -> left L	U -> left L	D -> left L
S -> up U	R -> up U	L -> up U	U -> up U	D -> up U
S -> down D	R -> down D	L -> down D	U -> down D	D -> down D
	R -> right	L -> right	U -> right	D -> right
	R -> left	L -> left	U -> left	D -> left
	R -> up	L -> up	U -> up	D -> up
	R -> down	L -> down	U -> down	D -> down }

To estimate D_k, the probability P_{kij} associated with the production $A_i \rightarrow \beta_j$ in G_{sk} must be obtained for each learning pattern set X of same actions. It is approximated by the relation ;

$$estimated\ \ P_{kij} = {}^{\wedge}P_{kij} = \frac{n_{kij}}{\sum_r n_{kir}} \tag{1}$$

In equation (1) n_{kij} means the total average number of times when $A_i \rightarrow \beta_j$ in G_{sk} is used to all the learning patterns. It is obtained by equation (2).

$$n_{kij} = \sum_{x_h\ in\ X} n(x_h) \bullet\ p(G_{sk} / x_h) \bullet N_{kij}(x_h) \tag{2}$$

In equation (2) $n(x_h)$ is the frequency of all patterns occurred in X, $N_{kij}(x_h)$ is the number of times that $A_i \rightarrow \beta_j$ is used when a pattern of x_h is parsed. $p(G_{sk}/x_h)$ means the probability with which a pattern of x_h is produced from G_{sk}. $\sum_r n_{kir}$ in (1) is computed over all productions in G_{sk} that have the same A_i [6].

If D_k, for k=1…M, is obtained, the inferring step of the stochastic grammar as the learning step is completed, and the recognition step to recognize some arbitrary patterns can be performed.

3 Data Representation for Human Action Sequence

A representation for recognition of human action is developed. To satisfy the specification of stochastic regular grammar defined in this paper, raw data are converted to a suitable format using a preprocessing step.

3.1 Data and Data Aquisition

We utilized STABIL++ [14] to detect and track head and arm positions in video sequences. STABIL++ produces 3D positions of 11 color markers on each articulation, trunk, and head as in Fig.1. We call each color marker as a node. In our system, we use only 8 nodes such as 2 head nodes and 6 articulation nodes for 2 arms. Fig.2 shows an example of video data sequence: an action of left arm turning forward.

Data of node's movement is a sequence of the pair of x, y, z position, (x,y,z) on the world coordinates.

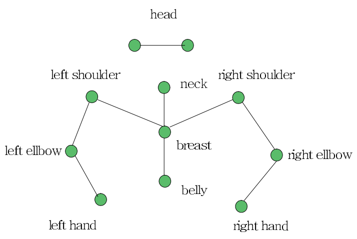


Fig. 1. Position of color markers.

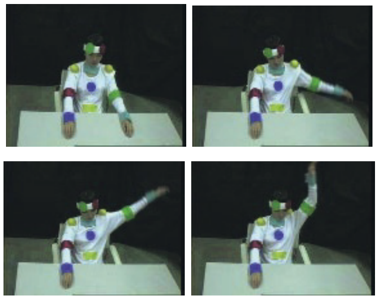


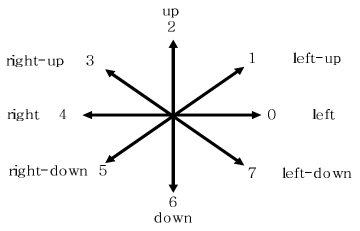
Fig. 2. Example of video data sequence.

3.2 Plane Projection of Quantized Data and 4-Connected Chain Coding

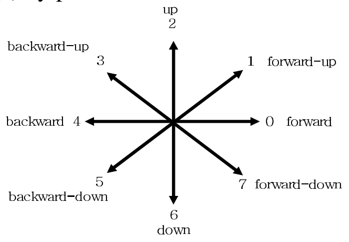
(x,y,z)s of articulation are used after their quantization at intervals of 3 cm so that very small motions in an action are disregarded. After converting quantized data into projection values in each xy, zy plane, we code them in 8-connected chain codes. The 8-connected chain coding of the projected data to xy plane has the meaning of left, left-up, up, right-up, right, right-down, down as in Fig.3(a). The 8-connected chain coding of the projected data to zy plane has the meaning of forward, forward-up, up, backward-up, backward, backward-down, down as in Fig.3(b). For example, the left window of Fig.3(c) shows the projection of a node movement to xy plane, and the right window is its projection to zy plane for an action in Fig.2.

After projection, a subpattern sequence in 8-connected chain codes are transformed into one in 4-connected chain codes. Too many productions are created by using 8-connected chain codes. It occurs high frequency of transition to next state and may decrease the recognition rate. To prevent it, we apply 4-connected chain coding. The 4-connected chain coding of the projected data to xy plane has the meaning of left, up, right, down. The 4-connected chain coding of the projected data to zy plane has the meaning of forward, up, backward, down. For example, forward-down in 8-

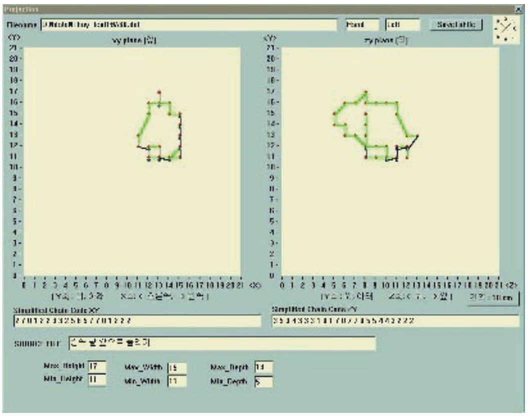
connected chain coding is transformed to forward and down in 4-connected chain code using the transformation rule [7]. The format of subpattern sequence will be indicated as (code code_count)s. "code" denotes each direction. "code count" refers to the repeating frequency of codes. An example of the subpattern sequence is shown in Table 1.



(a) xy plane : 8-connected chain code



(b) zy plane : 8-connected chain code



Filename [illegible] Hand Left

Simplified Chain Code XY [illegible]

Simplified Chain Code ZY [illegible]

Max_Height 17 Min_Height 11 Max_Width 15 Min_Width 11 Max_Depth 14 Min_Depth 5

(c) an example of xy, zy plane projection and 8-connected chain coding

Fig. 3. 8-connected chain code and result after plane projection and 8-connected chain coding.

Table 1. An example of subpattern sequences for an action in Fig.2.

node = left hand
plane = xy
	left 12 up 11 right 4 up 5 right 9 down 13 left 2
	down 1 left 5 down 2 left 3 up 13 left 1 up 1 #

node = left hand
plane = zy
	backward 23 up 16 forward 23 down 14 backward 6 down 2 backward 8
	up 7 backward 2 up 1 backward 6 down 2 backward 15 up 3 backward 2
	up 5 backward 2 up 9 forward 2 up 1 forward 2 up 1 forward 15 down 1
	forward 2 down 2 forward 4 down 10 backward 24 up 2 backward 2
	up 13 forward 3 up 2 forward 22 down 12 forward 2 down 5 #

3.3 Specification of Stochastic Regular Grammar

In our work, we represent 60 types of human actions using stochastic regular grammar. We estimate a stochastic value for each production of the regular grammar. Thus, M grammars that classify M human actions are expressed as follows. G_{sk_xy} is the grammar for the pattern projected to xy plane.

$$G_{sk\_xy} \square (N_{xy}, \Sigma_{xy}, P_{xy}, D_{k\_xy}, S),\ \ k \in \{1...M\},$$

where $N_{xy} \square \{S, R, L, U, D\}$, $\Sigma_{xy} \square \{\text{right, left, up, down}\}$

G_{sk_zy} is the grammar for the pattern projected to zy plane.

$$G_{sk\_zy} \square (N_{zy}, \Sigma_{zy}, P_{zy}, D_{k\_zy}, S),\ \ k \in \{1...M\},$$

where $N_{zy} \square \{S, B, F, U, D\}$, $\Sigma_{zy} \square \{\text{backward, forward, up, down}\}$

Fig.4 shows a state diagram of automata for the pattern projected to xy plane.

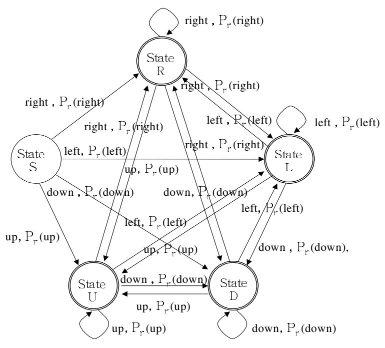


Fig. 4. State diagram of finite automata.

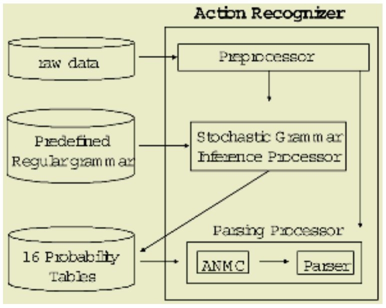


Fig. 5. The action recognizer.

4 Overview of Action Recognizer and Its Main Functions

Our action recognizer system is composed of a preprocessor, a stochastic grammar inference processor and a parsing processor. The preprocessor quantizes (x,y,z) data for input action and makes two sequences of 4-connected chain code for xy and zy projection planes. The stochastic grammar inference processor takes learning data and produces 16 probability tables by using the predefined regular grammar. Parsing processor takes subpattern sequences and searches a class with the highest stochastic value through the related stochastic information of the table. Fig.5 shows the structure of action recognizer. ANMC(Approximate Node Movement Classifier) is a preclassifier for searching action classes which have the same movement complexity. In this case movement complexity is defined as the combination of moving nodes.

4.1 Generation of Probability Tables

The inference step of stochastic regular grammar is applied to 8 nodes and for each projection plane. So, 16 probability tables are derived from the inference of the stochastic regular grammar.

The probability value for each production is obtained as shown in Table 2. It is an example of one node. The productions are replaced with the Confrontation Rules defined to follow the method of P_{kij}. The subscript "k" means the position of the grammar class, the subscript "i" means the subscript on the left hand side, and the subscript "j" means the subscript on the right hand side. Confrontation Rules are as follows.

A1 = S		
A2 = R	β1 = right R	β5 = right
A3 = L	β2 = left L	β6 = left
A4 = U	β3 = up U	β7 = up
A5 = D	β4 = down D	β8 = down

Some examples of the probability values practically estimated are shown in the column D_{44} of Table 2. D_{44} means the probability value of productions estimated after

Table 2. A probability table of left hand node for projection xy plane.

	Productions	Replaced Productions	D_1	D_2	……	D_{44}	……	D_m
1	S -> right R	A1 -> β1	P_{111}	P_{211}		0.12217		P_{m11}
2	S -> left L	A1 -> β2	P_{112}	P_{212}		0.56045		P_{m12}
3	S -> up U	A1 -> β3	P_{113}	P_{213}		0.30227		P_{m13}
4	S -> down D	A1 -> β4	P_{114}	P_{214}		0.01511		P_{m14}
5	R -> right R	A2 -> β1	P_{121}	P_{221}		0.69052		P_{m21}
…								
35	D -> up	A5 -> β7	P_{157}	P_{257}		0.00038		P_{m57}
36	D -> down	A5 -> β8	P_{158}	P_{258}		0.05257		P_{m58}

learning the action in Fig.2. This example shows the estimated probability value to the learning patterns of left hand node obtained through the projection on the xy plane.

4.2 Classification and Parsing for Recognition

When an action pattern is given, 16 subpattern sequences for 8 nodes on 2 projection planes are obtained after preprocessing. The parsing processor takes subpattern sequences as input, inspects moving nodes by ANMC and decides action classes that are compared. Then the nodes that are moved in action for each plane are calculated according to the "Multiplication Law of Probability", because all movements of nodes are statistically independent. The result probability for one action is calculated as follows. 'Node_i' denotes a node probabilitiy.

$$P(\text{Node}_i \cap \text{Node}_j \cap \ldots \cap \text{Node}_k) = P(\text{Node}_i) \times P(\text{Node}_j) \times \ldots \times P(\text{Node}_k),$$
$$\text{for any } i,j,k \in \{1 \ldots 16\} \text{ and } i \neq j \neq k$$

The parser estimates every probability for each action class that is comparable, and looks for an action class with the highest probability among them.

5 Experimentation and Results

In order to confirm the effectiveness of our method, three types of experiments are carried out. We show the experiment data, kind of experiments and recognition performance of each experiment in our system.

5.1 Experimental Data

900 human upper body actions of 60 types that 3 persons gestured were recorded in STABIL++ system. 490 action data are used for learning and 410 action data are used for testing. Practically, the data for recognition to analyze actions are composed of the movements of head, bodies, arms, and of other compound body movements as in Table 3. These kinds of actions are selected with the reference to the data for human action analysis studies [15]. For example, head movements are hang down head and to the former place, raise head upward, turn head to the right and to the former place.

Table 3. Summary of human upper body actions.

Complexity	Body Part	Movement type	Direction	No. of Actions
Primitive	Head	hang down, raise up, turn, bend, rotate	up, down, right, left, forward, backward	8
	Trunk	bend, turn, lean, shake	right, left, forward, backward	9
	Right hand or arm	contacting, turn, raise	up, down, right, left, forward, backward	14
	Left hand or arm	contacting, turn, raise etc	"	14
Combination	Both hands or arms	raise, turn, fold, cross	"	8
	Others		"	7

5.2 Experimental Result

Table 4 shows the results that have been obtained from 3 kinds of experiments with our action recognizer. Experiment 1 of 8 action types of head shows the recognition rate of 93.8%. In this experiment we don't use the ANMC because head actions have all same movement complexity. 40 action data are used for learning and 32 action data for testing. Experiment 2 of 60 action types of head and body without ANMC shows the recognition rate of 64.6%. Experiment 3 of 60 action types of head and body with ANMC shows the recognition rate of 84.9%. 490 action data are used for learning and 410 action data for testing in experiment 2 and 3. The recognition rate is estimated as Recognition Rate=Number of Correctly Recognized Actions/Total Number of Actions.

Table 4. Experimental results.

Experiment	Experimental Contents	Learning Data	Recognition Rate
1	8 action types of head without ANMC	40	93.8 % (30/32)
2	60 action types of head and body without ANMC	490	64.6 % (265/410)
3	60 action types of head and body with ANMC	490	84.9 % (348/410)

Our recognition method achieved comparatively high recognition rate for only one node, where the movement complexity is 1 such as in Experiment 1. Actions in Experiment 2 and 3 have the movement complexity from 2 to 8. Nevertheless, we achieved comparatively high recognition rate in Experiment 3. A pre-classification technique of ANMC causes the recognition rate to be improved.

6 Conclusions

In this paper we proposed a recognition model to understand human upper body actions using stochastic grammatical inference method. Grammatical inferring has been left unexplored and referred to as the further study [3].

3D data sequence of human actions are encoded into 4-connected chain codes, and projected to xy and zy plane. These sequences are processed as the input of the stochastic recognizer. They are used in the learning step for building the probability tables. Using these tables in the recognition step, each action is classified into the befitting class of human action.

Our scheme is suitable not only for simple actions composed of single articulation movement, but also for complex actions composed of several articulation movements. We have showed the possibility of autonomous learning from predefined human action patterns. In our experiments, 93.8% recognition rate of 8 action types of human head and 84.9% recognition rate of 60 action types of human upper body were achieved.

The contact of human hands to another body part has much importance for the analysis of nonverbal actions of human [5,15]. However, the recognition system through the general inference method by the stochastic grammar doesn't yet reflect the characteristics of nonverbal actions. The task for the future study may be the work to extend the inference method in this paper to reflect peculiarities of nonverbal actions.

References

1. Becker, D.A. 'Sensei: A Real-Time Recognition, Feedback and Training System for T'ai Chi Gestures,' MIT Media Lab Perceptual Computing Group TR 426 (1997)
2. Moeslund, T.B and Granum, E., 'A Survery of Computer Vision-Based Human Motion Capture,' Computer Vision and Image Understanding, vol. 81, No.3 (2001) 231-268 (2001)
3. Inanov, Y.A.: Application of stochastic grammars to understanding action, MIT thesis (1998)
4. Hong, P., Turk, M., and Huang, T., 'Gesture Modeling and Recognition Using Finite State Machines,' Int'l Conference on Gesture Recognition, Grenoble, France (2000)
5. Frey, S.: 'Das Berner System zur Untersuchung nonverbaler Interaktion,' Methoden der Analyse von Face-to-Face-Situationen, Stuttgart, Metzler, (1981) 203-236
6. Gonzalez, R.C. and Thomason, M.G.: Syntactic Pattern Recognition, Addison-Wesley Publishing Company (1978) 177-270
7. Fu, K.S.: Syntactic Methods in Pattern Recognition, Academic Press (1974) 54-55, 124-229
8. Doest, H.: 'Stochastic Grammars: Consistency and Inference', Beoorderlingscommissie, (1994), 53-84
9. Gronfors, T. and Juhola, M.: 'Experiments and comparison of inference methods of regular grammars', IEEE Trans. on Systems, Man and Cybernetics, Vol.22(4), (1992) 821-826
10. Cruz-Alcazar, P.P. and Vidal-Ruiz, E.: 'Modeling musical style using grammatical inference techniques: a tool for classifying and generating melodies', Third International Conference on Web Delivering of Music, (2003) 77-84
11. Vidal, E., 'Application of the error-correcting grammatical inference algorithm (ECGI) to planar shape recognition', IEE Colloquium on Grammatical Inference: Theory, Applications and Alternatives, (1993) 24/1 - 24/10
12. Atwell, E., et al, 'Multi-level disambiguation grammar inferred from English corpus, treebank, and dictionary', IEE Colloquium on Grammatical Inference: Theory, Applications and Alternatives, (1993) 9/1 - 9/7
13. Galiano, I. and Segarra, E., 'The application of k-testable languages in the strict sense to phone recognition in automatic speech recognition', IEE Colloquium on Grammatical Inference: Theory, Applications and Alternatives, (1993) 22/1 - 22/7
14. Munkelt, O., et al., 'A model driven 3D image interpretation system applied to person detection in video images,' 14th ICPR 98 (1998) 70-73
15. Bull, P.E.: Posture and Gesture, Pergamon Press (1990) 163-187

Optimizing Classification Ensembles via a Genetic Algorithm for a Web-Based Educational System[*]

Behrouz Minaei-Bidgoli[1], Gerd Kortemeyer[2], and William F. Punch[1]

[1] Genetic Algorithms Research and Applications Group (GARAGe)
Department of Computer Science & Engineering, Michigan State University
2340 Engineering Building, East Lansing, MI 48824, USA
{minaeibi,punch}@cse.msu.edu
http://garage.cse.msu.edu

[2] Division of Science and Math Education, Michigan State University
College of Natural Science, LITE lab, East Lansing, MI 48824, USA
korte@lite.msu.edu
http://www.lon-capa.org

Abstract. Classification fusion combines multiple classifications of data into a single classification solution of greater accuracy. Feature extraction aims to reduce the computational cost of feature measurement, increase classifier efficiency, and allow greater classification accuracy based on the process of deriving new features from the original features. This paper represents an approach for classifying students in order to predict their final grades based on features extracted from logged data in an educational web-based system. A combination of multiple classifiers leads to a significant improvement in classification performance. By weighing feature vectors representing feature importance using a Genetic Algorithm (GA) we can optimize the prediction accuracy and obtain a marked improvement over raw classification. We further show that when the number of features is few, feature weighting and transformation into a new space works efficiently compared to the feature subset selection. This approach is easily adaptable to different types of courses, different population sizes, and allows for different features to be analyzed.

1 Motivation

Several web-based educational systems with different capabilities and approaches have been developed to deliver online education in an academic setting. In particular, Michigan State University (MSU) has pioneered some of these systems to provide an infrastructure for online instruction. The research presented here was performed on a part of the latest online educational system developed at MSU, the *Learning Online Network with Computer-Assisted Personalized Approach* (*LON-CAPA*). LON-CAPA

[*] This work was partially supported by the National Science Foundation under ITR 0085921.

is involved with two kinds of large data sets: 1) educational resources such as web pages, demonstrations, simulations, and individualized problems designed for use on homework assignments, quizzes, and examinations; and 2) information about users who create, modify, assess, or use these resources. In other words, we have two ever-growing pools of data.

This paper investigates methods for extracting useful and interesting patterns from these large databases of students using online educational resources and their recorded paths within the system. We aim to answer the following research questions: Can we find *classes* of students? In other words, do there exist groups of students who use these online resources in a *similar* way? If so, can we predict a class for any individual student? With this information, can we then *help* a student use the resources better, based on the usage of the resource by other students in their groups?

We hope to find similar patterns of use in the data gathered from LON-CAPA, and eventually make predictions as to the most-beneficial course of studies for each learner based on their past and present usage. The system could then make suggestions to the learner as to how best to proceed.

2 Background on Using GAs for Feature Selection/Extraction

Genetic Algorithms (GA) have been shown to be an effective tool to use in data analysis and pattern recognition [1], [2], [3]. An important aspect of GAs in a learning context is their use in pattern recognition. There are two different approaches to applying GA in pattern recognition:

1. Apply a GA directly as a classifier. Bandyopadhyay and Murthy in [4] applied GA to find the decision boundary in N dimensional feature space.
2. Use a GA as an optimization tool for resetting the parameters in other classifiers. Most applications of GAs in pattern recognition optimize some parameters in the classification process. Many researchers have used GAs in feature selection [5], [6], [7], [8]. GAs have been applied to find an optimal set of feature weights that improve classification accuracy. First, a traditional feature extraction method such as Principal Component Analysis (PCA) is applied, and then a classifier such as k-NN is used to calculate the fitness function for GA [9], [10]. Combination of classifiers is another area that GAs have been used to optimize. Kuncheva and Jain in [11] used a GA for selecting the features as well as selecting the types of individual classifiers in their design of a Classifier Fusion System. GA is also used in selecting the prototypes in the case-based classification [12].

In this paper we focus on the second approach and use a GA to optimize a combination of classifiers. Our objective is to *predict* the students' final grades based on their web-use features, which are extracted from the homework data. We design, implement, and evaluate a series of pattern classifiers with various parameters in order to compare their performance on a dataset from LON-CAPA. Error rates for the individual classifiers, their combination and the GA optimized combination are presented.

Two approaches are proposed for the problem of feature extraction and selection. The *filter model* chooses features by heuristically determined "goodness/relevant" or knowledge, while the *wrapper model* does this by the feedback of classifier evaluation, or experiment. Research has shown the wrapper model outperforms the filter model comparing the predictive power on unseen data [13]. We propose a wrapper model for feature extraction through setting different weights for features and getting feedback from ensembles of classifiers.

3 Dataset and Class Labels

As test data we selected the student and course data of a single LON-CAPA course, BS111 (Biological Sciences), which was held at MSU in spring semester 2003. This course integrated 24 homework sets, including 229 problems, all of which are online. All 402 students used LON-CAPA for this course. Some students who dropped the course in mid-semester have initial homework scores, but no final grades. After removing those students, there remained 352 valid samples. The grade distribution of the students is shown in Fig 1.

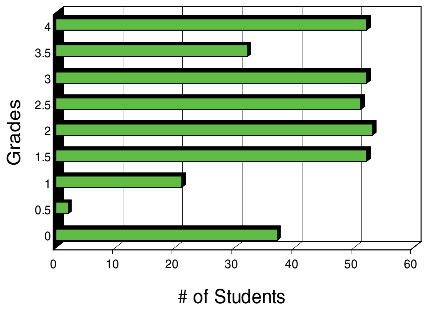


Table 1. Selecting 9-class labels.

Class	Grade	# of Std.	Percentage
1	0.0	37	10.51%
2	0.5	2	0.57%
3	1.0	21	5.97%
4	1.5	52	14.77%
5	2.0	53	15.06%
6	2.5	51	14.49%
7	3.0	52	14.77%
8	3.5	32	9.09%
9	4.0	52	14.77%

Fig. 1. LON-CAPA: BS111 SS03, Grades distribution.

We can group the students regarding their final grades in several ways, three of which are:

The nine possible labels can be the same as students' grades, as shown in Table 1

1. We can group them into three classes, "*high*" representing grades from 3.5 to 4.0, "*middle*" representing grades from 2.5 to 3, and "*low*" representing grades less than 2.5, as shown in Table 2.
2. We can also categorize students with one of two class labels: "*Passed*" for grades above 2.0, and "*Failed*" for grades less than or equal to 2.0, as shown in Table 3.

An essential step in performing classification is selecting the features used for classification. The BS111 course had an activity log with approximately 368 MB. After cleansing, we found 48 MB of useful data. We mined from these logged data 1,689,656 transactions from which the following features were extracted:

Table 2. Selecting 3-Classes labels regarding to students' grades in course BS111 SS03.

Class	Grade	# of Students	Percentage
High	Grade $\geq$ 3.5	84	23.86%
Middle	2.0 < Grade < 3.5	103	29.26%
Low	Grade $\leq$ 2.0	165	46.88%

Table 3. Selecting 2-Classes labels regarding to students' grades in course BS111 SS03.

Class	Grade	# of Students	Percentage
Passed	Grade > 2.0	187	53.13%
Failed	Grade $\leq$ 2.0	165	46.88%

1. Total number of tries for doing homework. (Number of attempts before correct answer is derived)
2. Total number of correct answers. (Success rate)
3. Getting the problem correct on the first try vs. those with high number of tries. (Success at the first try)
4. Getting the problem correct on the second try.
5. Getting the problem correct between 3 and 9 tries.
6. Getting the problem correct with high number of tries (10 or more tries).
7. Total time that passed from the first attempt, until the correct solution was demonstrated, regardless of the time spent logged in to the system.
8. Total time spent on the problem regardless of whether they got the correct answer or not.

4 Classification Ensembles

Pattern recognition has a wide variety of applications in many different fields, such that it is not possible to come up with a single classifier that can give good results in all cases. The optimal classifier in every case is highly dependent upon the problem domain. In practice, one might come across a case where no single classifier can achieve an acceptable level of accuracy. In such cases it would be better to pool the results of different classifiers to achieve the optimal accuracy. Every classifier operates well on different aspects of the training or test feature vector. As a result, assuming appropriate conditions, combining multiple classifiers may improve classification performance when compared with any single classifier.

The scope of this study is restricted to comparing some popular non-parametric pattern classifiers and a single parametric pattern classifier according to the error estimate. Four different classifiers using the LON-CAPA dataset are compared in this study. The classifiers used in this study include *Quadratic Bayesian classifier, 1-nearest neighbor (1-NN)*, *k-nearest neighbor (k-NN)*, *Parzen-window*[1]. These are some of the common classifiers used in most practical classification problems. After

[1] The classifiers are coded in MATLAB^{TM} 6.5.

some preprocessing operations the optimal *k*=3 is chosen for *kNN* algorithm. To improve classification performance, a fusion of classifiers is performed.

Normaliztion. Having assumed in Bayesian and Parzen-window classifiers that the features are normally distributed, it is necessary that the data for each feature be normalized. This ensures that each feature has the same weight in the decision process. Assuming that the given data is Gaussian, this normalization is performed using the mean and standard deviation of the training data. In order to normalize the training data, it is necessary first to calculate the sample mean μ , and the standard deviation σ of each feature in this dataset, and then normalize the data using the equation (1).

$$x_i = \frac{x_i - \mu}{\sigma} \tag{1}$$

This ensures that each feature of the training dataset has a normal distribution with a mean of zero and a standard deviation of unity. In addition, the *k*NN method requires normalization of all features into the same range.

Combination of Multiple Classifiers. Clearly, the data here suggest that in combining multiple classifiers we can improve classifier performance. There are different ways one can think of combining classifiers:

- The simplest way is to find the overall error rate of the classifiers and choose the one which has the least error rate on the given dataset. This is called an *offline classification fusion*. This may appear to be a classification fusion; however, in general, it has a better performance than individual classifiers.
- The second method, which is called *online classification fusion*, uses all the classifiers followed by a vote. The class getting the *maximum votes* from the individual classifiers will be assigned to the test sample.

Using the second method we show that classification fusion can achieve a significant accuracy improvement in all three cases of 2-, 3-, and 9-Classes. A GA is employed to determine whether classification fusion performance can be maximized.

5 GA-Optimized Ensembles of Classifications

Our goal is to find a population of best weights for every feature vector, which minimize the classification error rate. The feature vector for our predictors are the set of eight variables for every student: Number of attempts before correct answer is derived, Success rate, Success at the first try, Success at the second try, Success with number of tries between three and nine, Success with high number of tries, the time at which the student got the problem correct relative to the due date, and total time spent on the problem. We randomly initialized a population of eight dimensional weight vectors with values between 0 and 1, corresponding to the feature vector and experimented with different number of population sizes. We found good results using a population with 200 individuals. Real-valued populations may be initialized using the GA MATLAB Toolbox function *crtrp*. For example, to create a random population of

200 individuals with eight variables each: we define boundaries on the variables in *FieldD* which is a matrix containing the boundaries of each variable of an individual.

```
FieldD = [ 0 0 0 0 0 0 0 0 ;  % lower bound
           1 1 1 1 1 1 1 1];  % upper bound
```

We create an initial population with `Chrom = crtrp(200, FieldD)`, So we have for example:

```
Chrom = 0.23 0.17 0.95 0.38 0.06 0.26 0.31 0.52
        0.35 0.09 0.43 0.64 0.20 0.54 0.43 0.90
        0.50 0.10 0.09 0.65 0.68 0.46 0.29 0.67
        0.21 0.29 0.89 0.48 0.63 0.81 0.05 0.12
```

We used the simple genetic algorithm (SGA), which is described by Goldberg in [14]. The SGA uses common GA operators to find a population of solutions which optimize the fitness values. We used the *Stochastic Universal Sampling* [14] as our selection method, mainly due to its popularity and functionality. A form of stochastic universal sampling is implemented by obtaining a cumulative sum of the fitness vector, *FitnV*, and generating *N* equally spaced numbers between 0 and sum(FitnV). Thus, only one random number is generated, all the others used being equally spaced from that point. The index of the individuals selected is determined by comparing the generated numbers with the cumulative sum vector. The probability of an individual being selected is then given by

$$F(x_i) = \frac{f(x_i)}{\sum_{i=1}^{N_{ind}} f(x_i)} \tag{2}$$

where $f(x_i)$ is the fitness of individual x_i and $F(x_i)$ is the probability of that individual being selected.

The operation of *crossover* is not necessarily performed on all strings in the population. Instead, it is applied with a probability *Px* when the pairs are chosen for breeding. We selected *Px* = 0.7 since this would preserve a reasonably high level of the original population. Intermediate recombination combines parent values using the following formula [15]:

$$Offspring = parent1 + Alpha \times (parent2 - parent1) \tag{3}$$

where Alpha is a scaling factor chosen uniformly in the interval [-0.25, 1.25].

A further genetic operator, *mutation* is applied to the new chromosomes, with a set probability *Pm* as the rate of mutation. Mutation causes the individual genetic representation to be changed according to some probabilistic rule. Mutation is generally considered to be a background operator that ensures that the probability of searching a particular subspace of the problem space is never zero. This has the effect of tending to inhibit the possibility of converging to a local optimum, rather than the global optimum. We considered *Pm* = 1/800 as our mutation rate, due to its small value with respect to the population. The mutation of each variable is calculated as follows:

$$Mutated\ Var = Var + MutMx \times range \times MutOpt(2) \times delta \tag{4}$$

where delta is an internal matrix which specifies the normalized mutation step size; MutMx is an internal mask table; and MutOpt specifies the mutation rate and its shrinkage during the run.

During the reproduction phase, each individual is assigned a *fitness value* derived from its raw performance measure given by the objective function. This value is used in the selection to bias towards more fit individuals. Highly fit individuals, relative to the whole population, have a high probability of being selected for mating whereas less fit individuals have a correspondingly low probability of being selected. The error rate is measured in each round of cross validation by dividing "the total number of misclassified examples" into "total number of test examples". Therefore, our *fitness function* measures the accuracy rate achieved by classification fusion and our objective would be to maximize this performance (minimize the error rate).

6 Experimental Results

Without using GA, the overall results of classification performance on our dataset for four classifiers and classification fusion are shown in the Table 4. Regarding individual classifiers, 1NN and *k*NN have the best performance in the case of 2-, 3-, and 9-Classes, of approximately 62%, 50% and 35% accuracy, respectively. However, the classification fusion improved the classification accuracy significantly in all three cases. That is, it achieved 72% accuracy in the case of 2-Classes, 59% in the case of 3-Classes, and 43% in the case of 9-Classes.

Table 4. Comparing the average performance (%) of ten runs of classifiers on BS111 dataset for 2-, 3-, and 9-Classes, using 10-fold cross validation, without GA optimization.

Classifier	2-Classes	3-Classes	9-Classes
Bayes	52.6	38.8	22.1
1NN	**62.1**	45.3	29.0
*k*NN	55.0	**50.6**	**34.5**
Parzen	59.7	42.9	22.6
Classification Fusion	***72.2***	***58.8***	***43.1***

For GA optimization, we used 200 individuals (weight vectors) in our population, running the GA over 500 generations. We ran the program 10 times and obtained the averages, which are shown, in Table 5.

Table 5. Comparing the classification fusion performance on BS111 dataset Using-GA and without-GA in the cases of 2-, 3-, and 9-Classes, 95% confidence interval.

Classifier	2-Classes	3-Classes	9-Classes
Classification fusion of 4 Classifiers without GA optimization	71.19 ± 1.34	58.92 ± 1.36	42.94 ± 2.06
GA Optimized Classification Fusion, Mean individual (not best value)	81.09 ± 2.42	70.13 ± 0.89	55.25 ± 1.03
Improvement of Mean individual	***9.82 ± 1.33***	***11.06 ± 1.84***	***12.71 ± 0.75***

The results in Table 5 represent the mean performance with a two-tailed t-test with a 95% confidence interval. For the improvement of GA over non-GA result, a P-value indicating the probability of the Null-Hypothesis (There is no improvement) is also given, showing the significance of the GA optimization. All have p<0.001, indicating significant improvement. Therefore, using GA, in all the cases, we got more than a 10% mean individual performance improvement and about 11 to 16% best individual performance improvement. Fig. 2 shows the results of one of the ten runs in the case of 2-Classes. The dotted line represents the population mean, and the solid line shows the best individual at each generation and the best value yielded by the run (Due to the space limitation, only two graphs are shown).

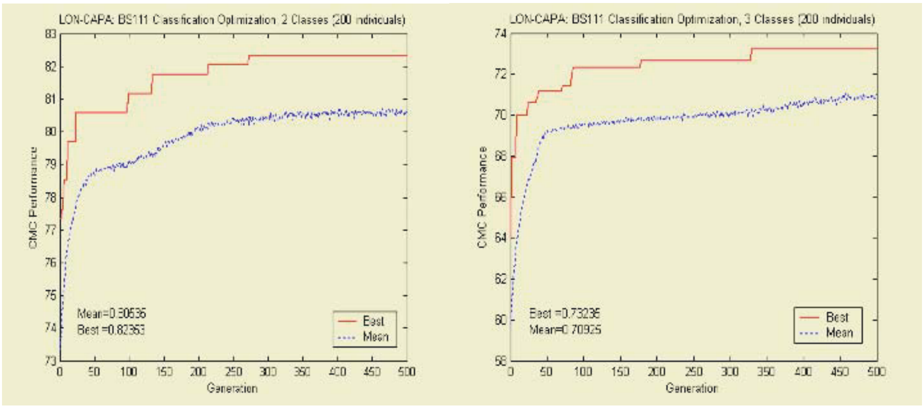


Fig. 2. GA-Optimized Combination of Multiple Classifiers' (CMC) performance in the case of 2- and 3-Classes, 200 weight vectors individuals, 500 Generations.

Finally, we can examine the individuals (weights) for features by which we obtained the improved results. This feature weighting indicates the *importance* of each feature for making the required classification. In most cases the results are similar to Multiple Linear Regressions or some tree-based software (like CART) that use statistical methods to measure feature importance. The GA feature weighting results, as shown in Table 6, state that the "Success with high number of tries" is the most important feature in all three cases. The "Total number of correct answers" feature is also important in some cases.

Table 6. Relative Feature Importance%, Using GA weighting.

Feature	2-Classes	3-Classes	9-Classes
Total Number of Tries	18.9	17.8	10.7
Total # of Correct Answers	**84.7**	**57.1**	27.4
# of Success at the First Try	14.4	**55.2**	34.2
# of Success at the Second Try	16.5	25.9	22.0
Got Correct with 3-9 Tries	21.2	38.8	11.1
Got Correct with # of Tries $\geq$ 10	**91.7**	**69.1**	**67.3**
Time Spent to Solve the Problems	32.1	14.1	28.3
Total Time Spent on the Problems	36.5	15.4	33.5

7 Conclusions and Future Work

We proposed a new approach to classifying student usage of web-based instruction. Four classifiers are used in grouping the students. A combination of multiple classifiers leads to a significant accuracy improvement in the 2-, 3- and 9-Class cases. Weighing the features and using a genetic algorithm to minimize the error rate improves the prediction accuracy by at least 10% in the all three test cases. In cases where the number of features is low, feature weighting worked much better than feature selection. The successful optimization of student classification in all three cases demonstrates the merits of using the LON-CAPA data to *predict* the students' final grades based on their features, which are extracted from the homework data. This approach is easily adaptable to different types of courses, different population sizes, and allows for different features to be analyzed. This work represents a rigorous application of known classifiers as a means of analyzing and comparing use and performance of students who have taken a technical course that was partially/completely administered via the web. In the present work, we propose an approach for predicting students' performance based on extracting the average of feature values over all of the problems in a course. For future work, we plan to implement such an optimized assessment tool for every student on any particular problem. Therefore, we can track students' behaviors on a particular problem over several semesters.

References

1. Raymer, M.L. Punch, W.F., Goodman, E.D., Kuhn, L.A., and Jain, A.K.: Dimensionality Reduction Using Genetic Algorithms. IEEE Transactions on Evolutionary Computation, Vol. 4, (2000) 164-171
2. Jain, A. K.; Zongker, D. Feature Selection: Evaluation, Application, and Small Sample Performance. IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 19, No. 2, February (1997)
3. De Jong K.A., Spears W.M. and Gordon D.F.: Using genetic algorithms for concept learning. Machine Learning 13, (1993) 161-188
4. Bandyopadhyay, S., and Muthy, C.A.: Pattern Classification Using Genetic Algorithms. Pattern Recognition Letters, Vol. 16, (1995) 801-808
5. Bala J., De Jong K., Huang J., Vafaie H., and Wechsler H.: Using learning to facilitate the evolution of features for recognizing visual concepts. Evolutionary Computation 4(3) - Special Issue on Evolution, Learning, and Instinct: 100 years of the Baldwin Effect (1997)
6. Guerra-Salcedo C. and Whitley D.: Feature Selection mechanisms for ensemble creation: a genetic search perspective. In: Freitas AA (Ed.) Data Mining with Evolutionary Algorithms: Research Directions – Papers from the AAAI Workshop, 13-17. Technical Report WS-99-06. AAAI Press (1999)
7. Vafaie, H. and De Jong, K.: Robust feature Selection algorithms. Proceeding of IEEE International Conference on Tools with AI, Boston, Mass., USA. Nov. (1993) 356-363

8. Martin-Bautista M.J., and Vila M.A.: A survey of genetic feature selection in mining issues. Proceeding Congress on Evolutionary Computation (CEC-99), Washington D.C., July (1999) 1314-1321
9. Pei, M., Goodman, E.D., and Punch, W.F.: Pattern Discovery from Data Using Genetic Algorithms. Proceeding of 1st Pacific-Asia Conference Knowledge Discovery & Data Mining (PAKDD-97) (1997)
10. Punch, W.F., Pei, M., Chia-Shun, L., Goodman, E.D., Hovland, P., and Enbody R.: Further research on Feature Selection and Classification Using Genetic Algorithms. In 5th International Conference on Genetic Algorithm, Champaign IL, (1993) 557-564
11. Kuncheva, L.I., and Jain, L.C.: Designing Classifier Fusion Systems by Genetic Algorithms. IEEE Transaction on Evolutionary Computation, Vol. 33 (2000) 351-373
12. Skalak D. B.: Using a Genetic Algorithm to Learn Prototypes for Case Retrieval an Classification. Proceeding of the AAAI-93 Case-Based Reasoning Workshop, Washigton, D.C., American Association for Artificial Intelligence, Menlo Park, CA, (1994) 64-69
13. John, G.H., Kohavi, R., Pfleger K.: Irrelevant Features and the Subset Selection Problem. Proceedings of the Eleventh International Conference of Machine Learning, Morgan Kaufmann Publishers, San Francisco, CA (1994) 121-129
14. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. MA, Addison-Wesley (1989)
15. Muhlenbein and Schlierkamp-Voosen D.: Predictive Models for the Breeder Genetic Algorithm: I. Continuous Parameter Optimization, Evolutionary Computation, Vol. 1, No. 1 (1993) 25-49

A Structural Approach to Adaptive Inverse Halftoning for Document Images

Hirobumi Nishida

Ricoh Image Appliance Lab, 1-1-17 Koishikawa, Bunkyo-ku, Tokyo 112-0002, Japan
hn@src.ricoh.co.jp

Abstract. This paper describes an efficient algorithm for inverse halftoning of scanned color document images to resolve problems with interference patterns such as moiré and graininess when the images are displayed or printed out. The algorithm is suitable for software implementation and useful for high quality printing or display of scanned document images delivered via networks from unknown scanners. A multi-resolution approach is used to achieve practical processing speed under software implementation. Through data-driven, adaptive, multi-scale processing, the algorithm can cope with a variety of input devices and requires no information on the halftoning method or properties (such as coefficients in dither matrices, filter coefficients of error diffusion kernels, screen angles, or dot frequencies). Effectiveness of the new algorithm is demonstrated through real examples of scanned color document images.

1 Introduction

When we display or print out scanned document images (obtained through image input devices) without applying any image processing operations to them or after scaling or rotating them, we often observe image distortions or degradations such as moiré phenomena and graininess. Halftone areas on printed document images consist of dot patterns generated by screening methods or error-diffusion algorithms. Such dot patterns often interact with image processing operations (halftoning, scaling, etc.) performed by output devices such as printers or displays. This problem of image degradation can be resolved if halftone areas composed of dot patterns are transformed to ideal continuous-tone representations. Such an operation of restoring ideal continuous-tone representations from halftone dot patterns is usually referred to as *inverse halftoning*. In designing inverse halftoning algorithms for scanned document images, we require that characters, graphics such as line drawings, and significant edges be preserved without being blurred.

Such processing for improving image quality has been embedded into imaging devices such as copiers. In particular, there have been a number of methods for inverse halftoning. Methods applicable to halftone patterns generated by screening methods and error-diffusion algorithms are nonlinear permutation filters [1], a combination of Gaussian filters, median filters, and edge enhancement [2], and a wavelet-based method [3]. Statistical approaches include a convex-projection method using knowledge on error-diffusion kernels and nonlinear optimization [4], MAP projection and estimation of error-diffusion kernels using a modified LMS method [5], Bayesian

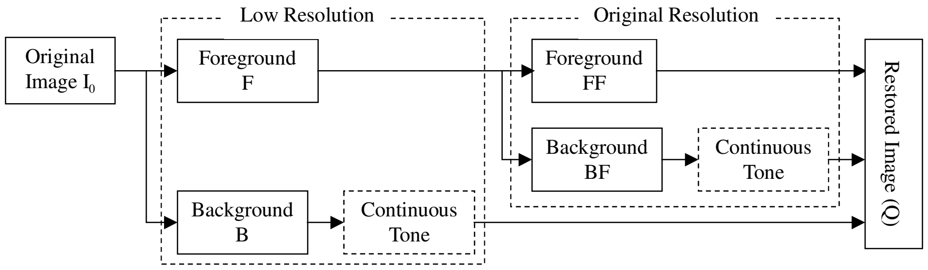


Fig. 1. Overview of the proposed algorithm.

estimation with nonlinear optimization [6], and MAP projection using wavelet [7]. However, they require expensive computation because iterative computations are involved. Kite et al. [8], Roetling [9,10], Miceli and Parker [11], and Chen and Hang [12] employ space-variant linear filters along with multi-scale gradient information. A hybrid LMS-MMSE method [13] and an LUT-based method [14] are computationally inexpensive.

In network and Internet environments, digital image data acquired through any input device can be transmitted to remote sites over networks, and the receiver can display or print out the image data delivered from unknown devices. Under these conditions, image data must be processed with software on individual PCs so that images can be printed or displayed in adequate quality without annoying distortions or degradations. Now, new technical requirements have emerged as follows:

- Practical processing speed must be achieved under software implementation.

Under network environments, image data delivered via networks from remotely located input devices are often processed on individual PCs for improving image quality when the data are displayed or printed out. Practical processing speed must be achieved under software implementation on PCs. If space-variant linear filters or wavelet-based inverse halftoning algorithms are implemented with software, computation time is not acceptable when they are applied to scanned color document images of A4 or letter size.

- Image processing systems must easily adapt to a variety of input devices.

In image processing systems embedded into stand-alone imaging devices where input and output media are integrated, algorithms and parameters are designed so that they fit the color characteristics, resolution, and frequency characteristics (MTF) of particular devices. However, image-processing systems optimal for a particular imaging device are not necessarily effective for other devices with different characteristics. Under network environments, there are large variations in characteristics of imaging devices, and the characteristics are even unknown when image data are transmitted from a remote site. Therefore, image-processing systems and algorithms must easily adapt to a variety of input devices. For instance, satisfactory image quality should be achieved for image data obtained through an unknown scanner just by setting a few, simple parameters.

- Information on the halftoning methods or properties is not required.

Specific methods for inverse halftoning can be employed if information is available on the halftoning method or properties (such as coefficients in dither matrices,

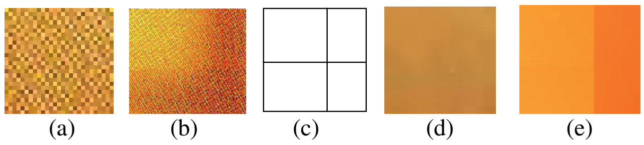


Fig. 2. (a) The background color is uniform or smoothly changes over the region. (b) The background consists of two or more distinct colors. (c) Location of windows. (d) Continuous tone representation for (a). (e) Continuous tone representation for (b).

filter coefficients of error diffusion kernels, screen angles, and dot frequencies). However, our targets are scanned images of books, magazines, journals, and newspapers that have been printed through unknown printing processes. Furthermore, characteristics of halftone dot patterns cannot be extracted when the input device has a poor response to high frequencies or the image resolution is inadequate. Therefore, blind methods are required without depending on the type or properties of halftoning processes.

In this paper, we describe an efficient algorithm for inverse halftoning of scanned color document images to resolve problems of interference patterns such as moiré and graininess when the images are displayed or printed out. The algorithm is suitable for software implementation and useful for high quality printing or display of scanned document images delivered via networks from unknown scanners. A multi-resolution approach is used to achieve practical processing speed under software implementation. Through data-driven, adaptive, multi-scale processing, the algorithm can cope with a variety of input devices and requires no information on the halftoning method or properties (such as coefficients in dither matrices, filter coefficients of error diffusion kernels, screen angles, or dot frequencies). Effectiveness of the new algorithm is demonstrated through real examples of scanned color document images.

This paper is organized as follows: An overview of the proposed algorithm is given in Sect. 2. The details of the algorithm are presented in Sect. 3. Examples and experimental results are shown in Sect. 4. Sect. 5 is the conclusion.

2 Outline

We outline the approach to resolving the three problems mentioned in Sect. 1. The overall architecture is shown in Fig. 1. A multi-resolution approach is used to achieve practical processing speed under software implementation. In each resolution, through data-driven, adaptive, multi-scale processing, the algorithm can cope with a variety of input devices and requires no information on the halftoning method or properties.

Multi-resolution Analysis

According to the Mixed Raster Content (MRC) Imaging Model, a color document image is composed of the following two components:

- Foreground: text, graphics (line drawings, in particular), and significant edges
- Background: other areas (continuous tone areas)

There are few significant edges in the background, which can be compressed to low resolution such as 100 dpi without perceptual degradation by human eyes, while

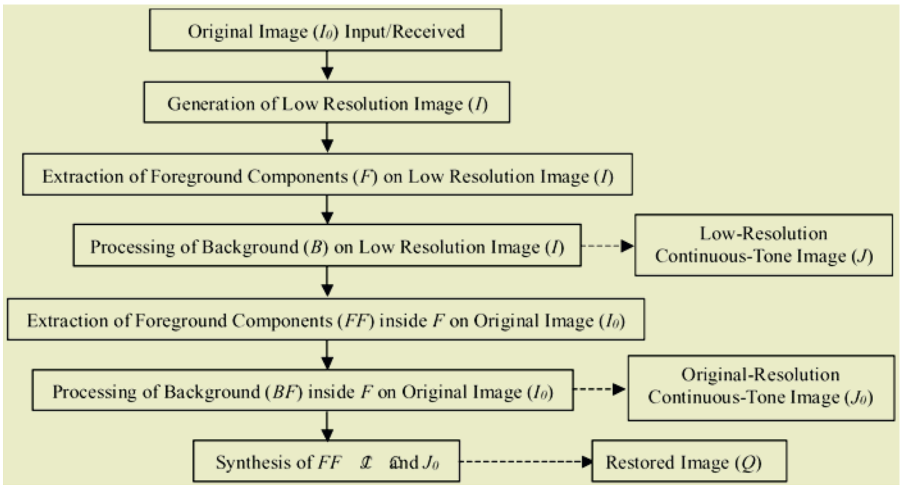

Original Image (I_0) Input/Received
Generation of Low Resolution Image (I)
Extraction of Foreground Components (F) on Low Resolution Image (I)
Processing of Background (B) on Low Resolution Image (I)
Low-Resolution Continuous-Tone Image (J)
Extraction of Foreground Components (FF) inside F on Original Image (I_0)
Processing of Background (BF) inside F on Original Image (I_0)
Original-Resolution Continuous-Tone Image (J_0)
Synthesis of FF J and J_0
Restored Image (Q)

Fig. 3. Flowchart for the proposed algorithm.

high resolution is required to preserve high frequency components for the foreground. In terms of image representation, quite different methods and viewpoints are required for these two components. Therefore, the background can be processed on a reduced image in low resolution, and consequently, the overall throughput can be improved significantly.

The multi-resolution analysis proceeds as follows: First, a reduced image *I* in low resolution is generated from the original image I_0. On the reduced image *I*, foreground components *F* are extracted based on local features. The remaining areas are set to the background *B*. The background *B* is transformed to a continuous-tone representation.

Next, the foreground components *F* are processed in the original resolution. Inside the area *F*, foreground components *FF* are further extracted based on local features, and the remaining areas are set to background *BF*. The background *BF* is also transformed to a continuous-tone representation.

Finally, the background components *B* represented as continuous tones, the foreground components *FF*, and the background components *BF* represented as continuous tones are synthesized together to obtain the restored image.

Processing of Background: Multi-scale Analysis from Coarse to Fine

The background components (*B* and *BF*) are transformed to continuous-tone representations. If the halftoning method and parameters for generating dot patterns are known, filter banks with appropriate support size can be prepared and space-variant linear filters can be applied to each pixel according to local image features. However, there is no information available to decide appropriate support size when characteristics of the input device are unknown. Therefore, multi-scale analysis from coarse to fine is employed.

In the background areas that remain after extracting the foreground, we consider a region *R* of size $W \times W$. There should be no significant edges, characters, or graphics in this region *R*. Therefore, if size *W* and location of *R* are appropriate, colors of pixels in region *R* can be replaced with the average color in *R*. For instance, for region R shown in Fig. 2(a), an appropriate continuous tone representation can be obtained by this operation as shown in Fig. 2(d).

However, distortions could be introduced when W is too large or location of R is inappropriate. For region R shown in Fig. 2(b), if windows are located as shown in Fig. 2(c), this operation leads to the representation shown in Fig. 2(e), where distortions are observed. To resolve this problem, we need to detect distortions, redoing the transformation with smaller W.

The edges on the transformed continuous-tone image should be a subset of edges on the original image, because smoothing or blurring has been applied. As far as the transformation process proceeds relevantly, there should not be edges on the transformed image that do not exist on the original. For each pixel, the edge magnitude on the transformed image should not exceed that on the original. As shown in Fig. 2(e), if there are some edges in the transformed image that do not exist on the original, it implies that W is too large or location of R is inappropriate. To correct the distortions, the continuous-tone transformation is redone around the pseudo-edges with smaller W. Since it is difficult to set the appropriate scale W adaptively for each pixel without a priori knowledge, a coarse-to-fine strategy is employed along with detection of anomalies on the transformed image based on the edge feature.

3 Algorithm

Following the overview presented in Sect. 2, we describe the algorithm in this section. A flowchart for the proposed algorithm is presented in Fig. 3. An example of the original image I_0 (400dpi) is shown in Fig. 4.

Foregroung Extraction

A reduced image I in low resolution (100 to 200dpi) is generated from the original image I_0 by a simple local averaging method. The reduced image I is smoothed with a linear filter for noise reduction. The selection of this filter depends on the frequency characteristics of the input image, but we have only to prepare just a few types of filters to deal with a wide range of characteristics. Furthermore, all pixels are initialized to background: $[i, j] \in B$ for all i, j. Foreground components F, as shown in Fig. 5, are extracted from the smoothed reduced image I by a locally adaptive thresholding technique, along with morphological dilation operations.

Transformation of Background to Continuous Tone Representation

The background areas B, as shown in white areas in Fig. 5, are transformed to a continuous tone representation through local, adaptive, multi-scale processing. Before the transformation, edge magnitude E_0 is calculated on the reduced image I.

Generation of Initial Transformed Images

The initial transformed image is generated for background components B. As shown in Fig. 6, for each horizontal scan line, runs of the maximum length wsz (typically 0.5mm) are constructed sequentially from left to right so that each run contains no foreground pixels inside. The average color is calculated for each run, and is set to

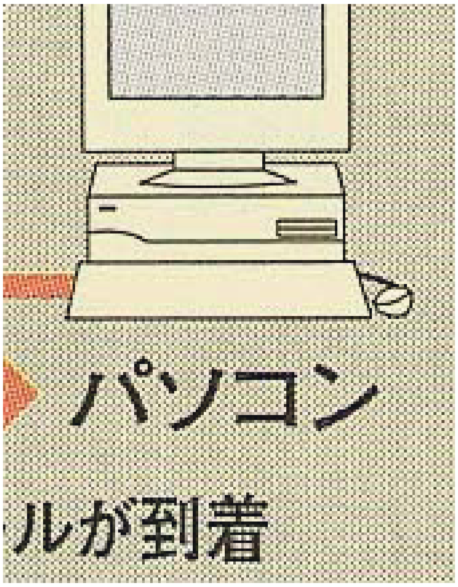


Fig. 4. Original image I_0.

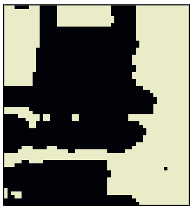


Fig. 5. Foreground components F extracted from the reduced image I.

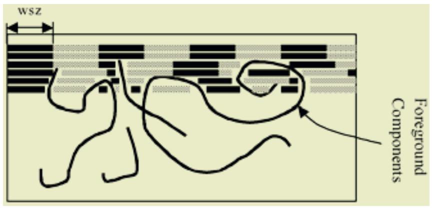


Fig. 6. Transformation into continuous tone representation.

each pixel on the run. In this way, the continuous tone representation C' is obtained based on horizontal runs. Furthermore, the continuous tone transformation is applied to vertical runs of the generated image C' in the same way as horizontal runs. The initial transformed image J is generated as a result of the continuous tone transformation for horizontal and vertical runs.

Correction to Transformed Images

The transformed image J is corrected through adaptive, multi-scale analysis of edges on the transformed images, based on the principle described in Sect. 2.2.

(1) Reduce *wsz* to the half. If *wsz* is less than a predetermined size, then go to (7).
(2) Compute the edge magnitude image E' for the transformed image J.
(3) Compute the pseudo edge image E_1 from the difference between E' and E_0. For pixel (i, j), if $E'(i, j) > E_0(i, j)$, set $E_1(i, j)$ to "ON," otherwise "OFF."
(4) Using the new value of *wsz*, the continuous tone transformation is redone locally around "ON" pixels of E_1. A new transformed image C is generated.
(5) On the transformed image J, colors of $(2 \cdot wsz - 1) \times (2 \cdot wsz - 1)$ pixels around each "ON" pixel on the pseudo edge image E_1 are replaced with those on the new transformed image C.
(6) Go to (1)
(7) End.

The correction is applied from a coarse scale to a fine scale by detecting distortions through the analysis of edge magnitude between the original and the transformed images. Fig. 7 shows the result of the series of processing embedded into the original image I_0.

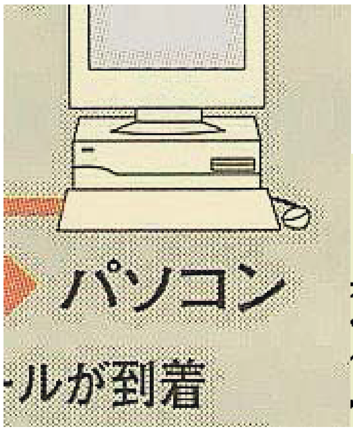


Fig. 7. Transformed image *J* embedded into the original image.

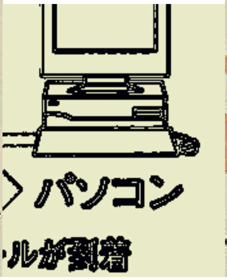


Fig. 8. Foreground components *FF*.

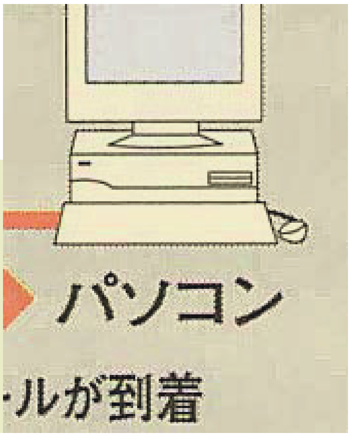


Fig. 9. Final result *Q* .

Processing in the Original Resolution and Synthesis

The foreground components *F* extracted from the reduced image *I* are processed in the original resolution in order to obtain a continuous tone transformation in fine detail. Note that, in Fig. 7, there are still some pixels that need to be transformed to a continuous tone representation, particularly around characters and line drawings.

Inside this area *F*, foreground components are extracted from the original image I_0 in the same way as described in Sect. 3.2. Let *FF* be the area extracted as foreground (Fig. 8), and *BF* be the remaining area. Furthermore, for the original image I_0, the area *BF* is transformed to a continuous tone representation in the same way as described in Sect. 3.3, and the transformed image J_0 is obtained.

The continuous tone representation *J* of the background components *B*, the foreground components *FF*, and the continuous tone representation J_0 of the background components *BF* are synthesized together to obtain to the final restored image *Q*. Fig. 9 shows the final result for the original image shown in Fig. 4.

4 Experimental Results

The proposed algorithm has been evaluated with real scanned document images to verify the effectiveness in terms of processing speed and image quality.

Computation time in a low-end environment (Pentium III, 933MHz□256MB) has been measured for 24bit full-color images of scanned documents of A4-size (210mm by 297mm), in comparison with space-variant linear filters with support size 5x5 and 7x7. Table 1 shows the quite promising results where remarkable acceleration can be observed for the proposed method. The result implies that practical processing speed can be achieved under software implementation.

Some real examples are presented for scanned color document images. Fig. 10(a) is the original image generated by the Floyd-Steinberg error-diffusion algorithm and

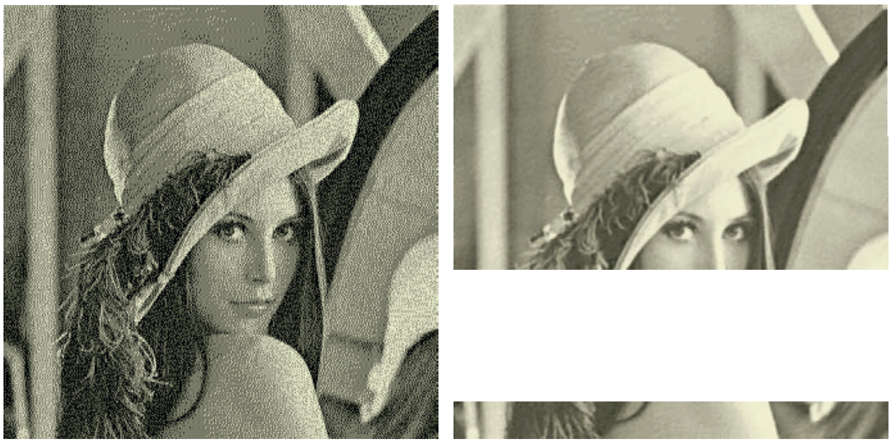


Fig. 10. Original image and result of the proposed algorithm.

Fig. 10(b) is the result of the continuous tone transformation. Because of the data-driven, adaptive, multi-scale processing, there is no perceivable distortion in the transformed image. Fig. 11(a) is a text image on top of grainy background and Fig. 11(b) is the result of the continuous tone transformation. The graininess of the background is completely resolved without blurring character strokes. Fig. 12(a) contains fine line drawings that are likely to be blurred by inverse halftoning and Fig. 12(b) is the result of the continuous tone transformation. Fine details are still preserved in the transformed image.

5 Conclusion

We have described an efficient algorithm for inverse halftoning of scanned color document images to resolve problems of interference patterns such as moiré and graininess when the images are displayed or printed out. The algorithm is suitable for software implementation and useful for high quality printing of scanned document images delivered via networks from unknown scanners. A multi-resolution approach is used to achieve practical processing speed under software implementation. Through data-driven, adaptive, multi-scale processing, the algorithm can cope with a variety of input devices and requires no information on the halftoning method or properties. Effectiveness of the new algorithm has been demonstrated through real examples of scanned color document images.

Table 1. Computation time in a low-end environment for full-color images of scanned documents of A4-size, in comparison with space-variant linear filters.

Resolution dpi	Proposed Method	Space-Variant Linear Filter	
		5x5	7x7
200	3sec.	11sec.	16sec.
400	7sec.	46sec.	67sec.

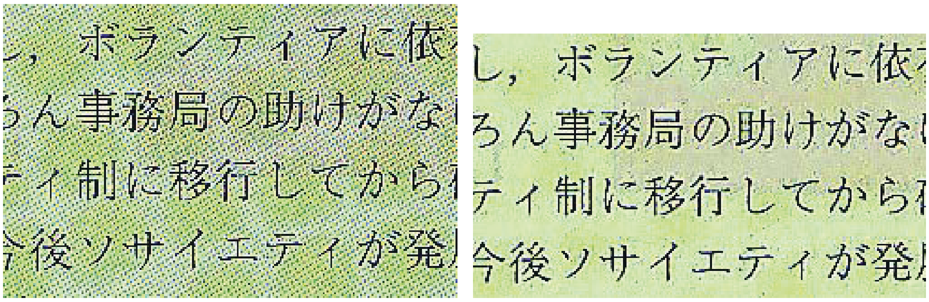

し，ボランティアに依
ろん事務局の助けがな
ティ制に移行してから
今後ソサイエティが発

Fig. 11. Original image and result of the proposed algorithm.

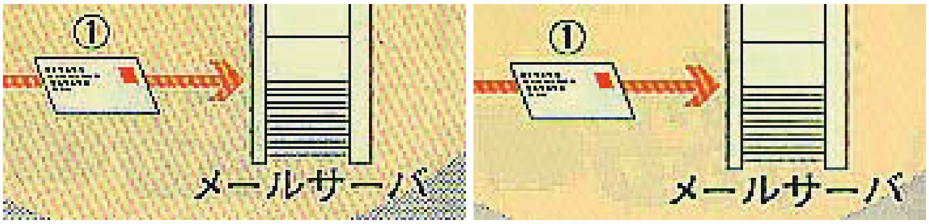


Fig. 12. Original image and result of the proposed algorithm.

References

1. Y. Kim, G. Arce, and N. Grabowski, Inverse halftoning using binary permutation filters, *IEEE Trans. Image Processing*, **4**, 1296 - 1311, 1995.
2. N. Damera-Venkata, T.D. Kite, M. Venkataraman, and B.L. Evans, Fast blind inverse halftoning, *Proc. 1998 IEEE International Conference on Image Processing*, 64 - 68, 1998.
3. J. Luo, R. de Querioz, and Z. Fan, A robust technique for image descreening based on the wavelet transform, *IEEE Trans. Signal Processing*, **46**, 1179 - 1194, 1998.
4. S. Hein and A. Zakhor, Halftone to continuous-tone conversion of error-diffusion coded images, *IEEE Trans. Image Processing*, **4**, 208 - 216, 1995.
5. P. Wong, Inverse halftoning and kernel estimation for error diffusion, *IEEE Trans. Image Processing*, **4**, 486 - 498, 1995.
6. R. Stevenson, Inverse halftoning via MAP estimation, *IEEE Trans. IP*, **6**, 574 - 583, 1997.
7. Z. Xiong, M. Orchard, and K. Ramchandran, Inverse halftoning using wavelet, *IEEE Trans. Image Processing*, **8**, 1479 - 1483, 1999.
8. T.D. Kite, N. Damera-Venkata, B.L. Evans, and A.C. Bovik, A fast, high-quality inverse halftoning algorithm for error diffused halftones, *IEEE Trans. IP*, **9**, 1583 - 1592, 2000.
9. P. Roetling, US Patent 5343309, August 1994.
10. P.G. Roetling, US Patent 4630125, 1986.
11. C. Miceli and K.J. Parker, Inverse halftoning, *Journal of Electronic Imaging*, vol. 1, pp. 143 - 151, April 1992.
12. L.-M. Chen and H.-M. Hang, An adaptive inverse halftoning algorithm, *IEEE Trans. Image Processing*, **6**, 1202 - 1209, 1997
13. P.-C. Chang, C.-S. Yu, and T.-H. Lee, Hybrid LMS-MMSE inverse halftoning technique, *IEEE Trans. Image Processing*, **10**, 95 - 103, 2001.
14. M. Mese and P.P. Vaidyanathan, Look-up table (LUT) method for inverse halftoning, *IEEE Trans. Image Processing*, **10**, 1566 - 1578, 2001.

Syntactic Modeling and Recognition of Document Images*

Ignacio Perea and Damián López

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 Valencia, Spain
{nperea,dlopez}@dsic.upv.es

Abstract. In this work, we propose a new scheme for the recognition of document images under a syntactic approach. We present a new method to model the layout of the document using a tree-like representation of the form. The syntactic representation of the documents are used to infer a tree automaton for each one of the classes involved in the task. An error-correcting analysis of tree languages allows us to carry out the classification. The experimentation carried out showed the good behaviour of the approach: error rate of 1.18%.

1 Introduction

Document processing [1],[20] can be decomposed into two phases, document analysis and document understanding. Where the former attempts to obtain a hierarchical representation of the document that embeds its geometric structure (i.e. [9]), the latter builds the logical structure of the document from the information obtained in the first analysis [4].

Another line of work related to paper documents is *Document Image Recognition*. The techniques applied to this task can be divided into two categories, those based on matching local features (i.e. [2][16]) and those that combine both local features and layout information (i.e. [17][19][21]).

In this work, we face the problem of form classification under a syntactic approach. A new procedure is presented to represent the layout of the document. Briefly, this method measures the importance of each of the components of the image and then represents the relative position of each component using tree-like components [7]. Once the documents are reduced to a tree-like structure, this work uses a tree language inference algorithm [10] which builds a tree automaton for each class of the problem using a minimum edition distance criterion.

In real applications, the documents to be classified usually do not fit exactly in any class. Therefore, it is necessary to obtain the class that is nearest to the document structure. In our work, to obtain that measure, we use an error

* Work partially supported by the Spanish CICYT under contracts TIC2000-1153 and TIC2003-09319-C03-02.

correcting analysis algorithm on tree automata [14]. This scheme and has shown good performance in handwritten digit recognition [11].

This work is structured as follows. First of all, the required notation is presented. Then, the approach to be used is explained: the tree-like representation procedure, the main features of the error-correcting analysis algorithm and the tree language inference method. The experimentation carried out shows the good performance of the approach (1.18% of error rate). Finally, the conclusions and the proposed future lines of work are presented.

2 Notation and Definitions

A *ranked alphabet* is defined as an association of a finite alphabet V with a finite relation $r \subseteq (V \times \mathbb{N})$. Let V_n denote the subset $\{\sigma \in V : (\sigma, n) \in r\}$.

Let V^T be the set of finite trees whose nodes are labeled with symbols in V, where a tree is defined inductively as follows:

$$\begin{array}{l} V_0 \subseteq V^T \\ \sigma(t_1, ..., t_n) \in V^T \ \forall t_1, ..., t_n \in V^T, \quad \sigma \in V_n \end{array}$$

Note that the set of symbols in V_n are the valid labels of the nodes with arity n; therefore V_0 is the set of valid leaf labels.

A *deterministic tree automaton* is defined as the four-tuple $A = (Q, V, \delta, F)$ where Q is a finite set of states; V is a ranked alphabet; $F \subseteq Q$ is a set of final states and $\delta = (\delta_0, \ldots, \delta_m)$ is a finite set of functions defined as:

$$\begin{array}{rl} \delta_n : (V_n \times (Q \cup V_0)^n) \to Q & \quad n = 1, \ldots, m \\ \delta_0(a) = a & \quad \forall a \in V_0 \end{array}$$

δ can be extended to operate on trees as follows:

$$\begin{array}{ll} \delta : V^T \to Q \cup V_0 & \\ \delta(a) = a & \forall a \in V_0 \\ \delta(\sigma_n(t_1, \ldots, t_n)) = \delta_n(\sigma_n, \delta(t_1), \ldots, \delta(t_n)) & \text{if } n > 0 \end{array}$$

A tree $t \subseteq V^T$ is accepted by A if $\delta(t) \in F$. This set of trees accepted by A is defined as $L(A) = \{t \in V^T | \delta(t) \in F\}$, this set is also referred to as the *tree language* accepted by A.

3 Document Image Classification

3.1 Tree-Like Representation

In this work, we used the database of structured forms *NIST Special Database 6 (SD6)*[3]. The database contains 12 different tax forms together with different schedules and form faces. Hence, 20 different form faces (classes) are represented in the database. The images have been automatically derived and synthesized and therefore contain no "real" tax data. Nevertheless, all the images appear to be real hand-printed forms prepared by individuals. There are 900 simulated tax submissions represented in the database averaging 6.22 form faces per submission. All the images are stored in a bitlevel black and white raster format.

Algorithm 3.1 Algorithm to obtain a tree-like model of the document images.

```
 1. Blur the image and reduce it by one eighth. /* result: 256 grey-level image */
 2. Detect the connected components (four connected pixels) of the image.
 3. For each connected component of the image:
      – Reduce it to its smallest rectangle.
      – Reduce each rectangle to its central point and assign a value that is proportional to the area of the rectangle to this point.
 4. Set $M$ to the maximum but two /* to smooth the value */ and $m$ to the minimum values in the image.
 5. Let $I = (M + m)/2$;
 6. Consider the following correspondence between intervals and symbols:
      – Interval $[0, \lfloor I/2 \rfloor]$; symbol 'A'
      – Interval $[\lceil I/2 \rceil, \lfloor I \rfloor]$; symbol 'B'
      – Interval $[\lceil I \rceil, \lfloor (3I)/2 \rfloor]$; symbol 'C'
      – Interval $[\lceil (3I)/2 \rceil, \lfloor (23I)/10 \rfloor]$; symbol 'D'
      – Interval $[\lceil (23I)/10 \rceil, \infty]$; symbol 'E'
 7. Create the root node of the tree.
 8. Set node to the root node of the tree
 9. Divide the image in four quadrants
10. Create a child node to node for each quadrant (from left to right and downwards).
11. For each node (and its corresponding quadrant):
      – If the quadrant contains more than one point, then label the (internal) node with symbol $\sigma$ and go to step 9.
      – If the quadrant has no points, then label the node as a leaf using the symbol empty.
      – Otherwise, assign a symbol to the node (see step 6).
12. Return the tree obtained.
```

Quad trees (q-tree) [7] has been broadly used to obtain tree-like representations. These representations have shown to be useful in pattern recognition tasks [10][11]. In order to capture only the more relevant features of the images, the algorithm we propose takes into account a reduced image.

The first step in modelling the layout of the documents was to perform a light blur of the images (we considered 5×5 pixels) and a reduction of the image by one eighth. Then, the resulting 256 grey-level images were processed to detect the connected-components. In this process, only four connected pixels were considered. Each connected-component was then substituted by the minimum rectangle that contains the component.

The image of the document was then simplified to an image where all the rectangles were reduced to a pixel in the middle of the rectangle with a weight that is proportional to its area.

In a standard q-tree representation, the root of the tree is associated to the whole image to model and any other internal node with one of the four quadrants of its parent image. The process is repeated recursively until a node represents a one-colour square, then, the node becomes a leaf labeled with the color of the region. The variation of the q-tree representation used in this work ends the recursion when the quadrant does not contain any point. The region is then labeled with the *empty* leaf. When the region contains only one point, the leaf is marked with a symbol that is proportional to the weight of the point (five different labels were considered). A scheme of the entire process is shown in Algorithm 3.1. An example of the run is shown in Figure 1. Note that no preprocessing of the image (slant correction, rescaled or similar) was performed.

3.2 Error-Correcting Analysis

The treatment of noisy or distorted samples has always been a problem in pattern recognition tasks, these distortions usually do appear as a side effect to the acquisition, preprocessing or primitive extraction phases.

This work makes use of an error-correcting analysis algorithm that was introduced in [14] and that gives a distance between a tree and a tree automaton.

To calculate the distance between a tree t to a tree automaton A, the algorithm explores the tree, and calculates the cost of reducing each subtree of the tree to each state of the automaton. To do this, the algorithm compares the successors of a node with the different transitions of the automaton.

The method visits the tree nodes in postorder. Therefore, when a tree is going to be analyzed, all the distances between its subtrees and the states of the automaton have already been calculated. These distances are stored in a matrix that is indexed by the nodes of the tree and the states of the automaton, thereby avoiding calculations carried out previously (for details see [14]). The whole process is carried out in polynomial time with respect to the size of the tree and the number of states of the automaton. The authors prove that the distance obtained is the minimum one according to the edit operations proposed.

3.3 Tree Language Inference

In our work, we model each class with a tree language. In order to obtain these languages, it is possible to use several algorithms (for instance [5][6][8][13][15]). In this work, we used a tree language inference algorithm that, in each step of the inference process, considers the automaton of the previous step and a new sample of the training set. Then, the inference algorithm performs an error-correcting analysis on the tree automaton. The modification of the automaton comes determined by the editing operations needed to force the automaton to accept the sample.

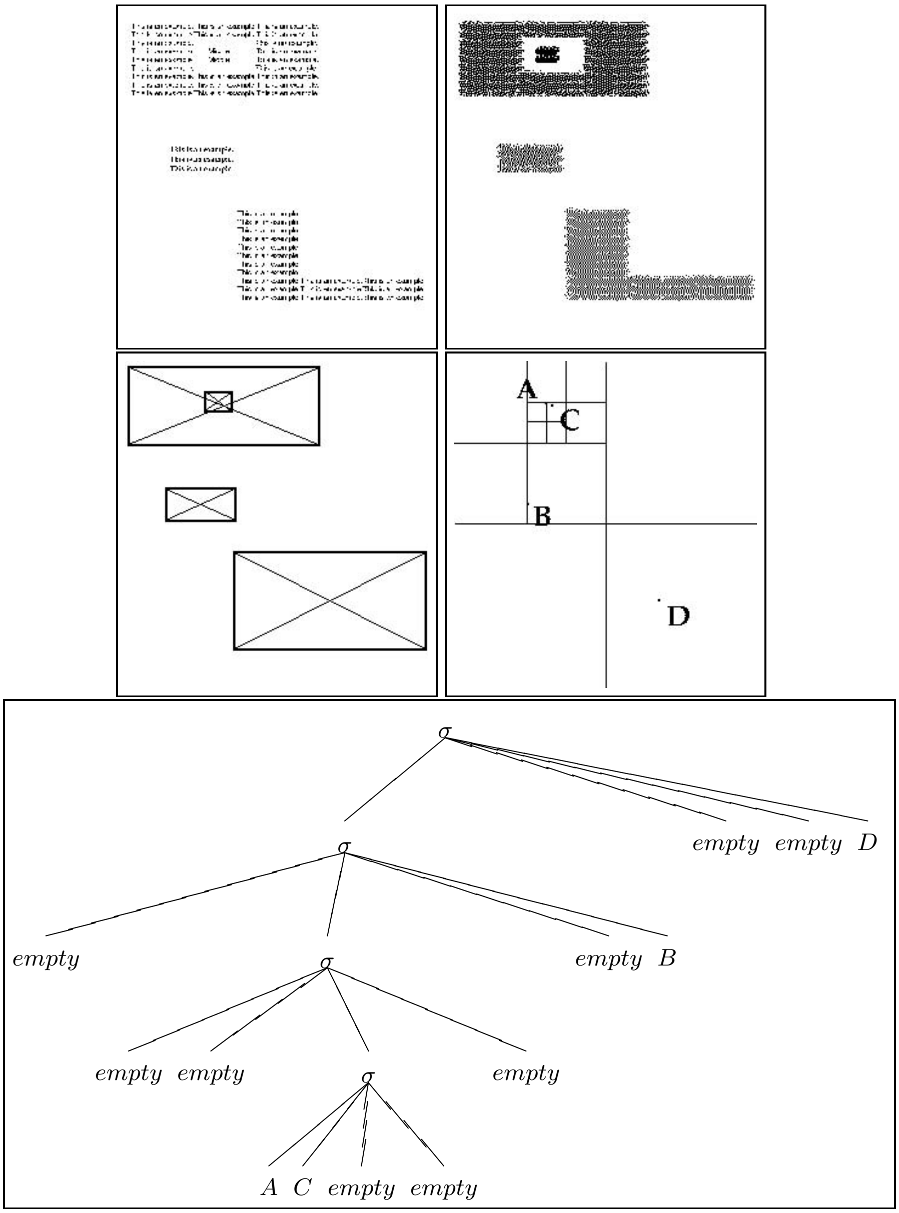


Fig. 1. Example of the run for Algorithm 3.1. From left to right and downwards: The original image (here reduced eight times), the blurred and reduced image, the smallest rectangles that contain a connected component, the final division in quadrants, and the resulting tree. A more compact notation for the tree is: $\sigma(\sigma(empty, \sigma(empty, empty, \sigma(A, C, empty, empty), empty), empty, B), empty, empty, D)$.

This algorithm has proven to be efficient when the differential features of the representation are based on the presence of substructures, the length of these substructures and their relative position in the sample representation.

There are two reasons why this new approach can be seriously considered for syntactic pattern recognition. On the one hand, the representation power of tree primitives is very high. On the other hand, this algorithm takes advantage of the features (substructures) that are already considered in the automaton, adding the new ones when they are presented in the inference process. As noted before, this approach has been used previously in pattern recognition tasks with good results.

4 Experimentation

Once the tree representation of the database was obtained, a tree automaton was inferred to represent each class. In order to infer these automata, we used the inference algorithm described above.

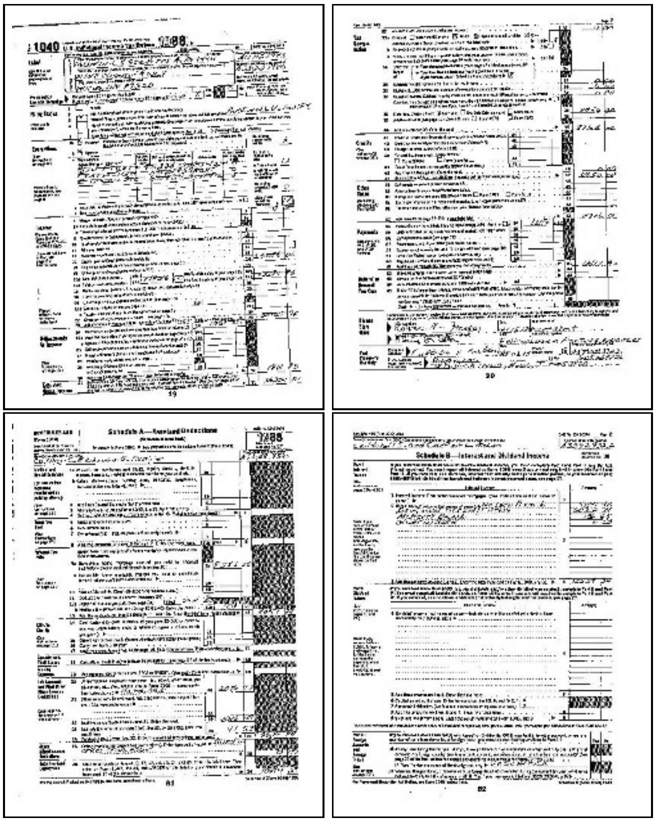


Fig. 2. Example images of each class in the dataset. Note the slant of the first image.

Once the automata were inferred, we used the error-correcting parser on tree automata to obtain the minimum distance of the test sample for each automaton. The document images were classified into the class with a minimum value criterion. This approach has previously been used in other pattern recognition tasks with good performance (e.g. [12][11]).

Our approach needs some samples from each class in order to train the models. Therefore, the four classes with greatest number of available images were selected from the database. These classes were: Form 1040 page 1, Form 1040 page 2, Form 1040 Schedules A and B. Example images are shown in Figure 2.

Two different experiments were performed. The models of the first experiment were inferred using 200 samples per class. In the second experiment, we used 300 samples per class. The results of the experimentation are shown in Table 1. Note that due to the number of samples available, the number of samples in the test set is not homogeneous.

Table 1. Results of the experimentation. The table on the left shows the confusion matrix when 200 training samples were used. The table on the right shows the confusion matrix when 300 samples were used to infer the automata.

class	0	1	2	3
0	399	1	0	0
1	1	399	0	0
2	1	17	268	4
3	1	0	2	377

class	0	1	2	3
0	398	2	0	0
1	0	399	1	0
2	1	7	178	4
3	0	0	0	280

The results show the excellent performance of our approach, an error rate of 1.84% was obtained when 200 samples were considered to infer the automata, and an error rate of 1.18% was obtained when 300 samples were used.

5 Conclusions

In this work, we tackle the classification of hand-filled forms using a syntactic approach. We propose a new procedure for modelling the layout of the forms which obtains a tree-like representation. This representation allows us to use a tree language inference algorithm to obtain a model for each class of the classification task. An error-correcting analysis is carried out to classify the test samples, taking into account the minimum editing-distance.

We used a subset of the *NIST Special Database 6* of structured forms and our syntactic approach to carry out the classification. The results of the experiments show the excellent behaviour of the approach: an error rate of 1.18%.

As a future line of work, the preprocessing of the image (for instance, applying methods to correct the slant of the image), should improve the results. Note that the approach does not consider the use of probabilities. Another line of work is the modification of the model to obtain a stochastic one which should also lead to better results.

Finally, the representation process could easily be modified for other pattern recognition tasks and should be explored further. Also, other methods for modelling the physical structure of forms [18] could be used in a syntactic approach to classification to produce good results.

References

1. R. Cattoni, T. Coianiz, S. Messelodi, and C.M. Modena. Geometric layout analysis techniques for document image understanding. Technical report, Instituto Trentino di Cultura, 1998.
2. F. Cesarini, M. Gori, S. Marinai, and G. Soda. Informys: a flexible invoice-line form-reader system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(7):730–745, 1995.
3. D.L. Dimmick and M.D. Garris. Nist special database 6. structured forms database 2. Technical report, National Institute od Standards and Technology. Advanced Systems Division. Image Recognition Group, 1992.
4. D. Dori, D. Doermann, C. Shin, R. Haralick, I. Phillips, M. Buchman, and D. Ross. *Handbook on Optical Character Recognition and Document Image Analysis. The representation of document structure: a generic object-process analysis.* World Scientific Pub. Co., 1996.
5. H. Fernau. Learning tree languages from text. *LNCS*, 2375:153–168, 2002. 15th Annual Conference on Computational Learning Theory (COLT 2002).
6. P. García. Learning k-testable tree sets from positive data. Technical Report DSIC/II/46/1993, Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, 1993. Available on: `http://www.dsic.upv.es/users/tlcc/tlcc.html`
7. G. M. Hunter and K. Steiglitz. Operations on images using quad trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2):145–153, 1979.
8. T. Knuutila. *Advances in Structural and Syntactic Pattern Recognition: Proc. of the International Workshop*, chapter Inference of k-Testable Tree Languages, pages 109–120. World Scientific, 1992.
9. S.W. Lee and D.S. Ryu. Parameter-free geometric document layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1240–1256, 2001.
10. D. López and S. España. Error correcting tree language inference. *Pattern Recognition Letters*, 23(1–3):1–12, 2002.
11. D. López. *Inferencia de lenguajes de árboles*. PhD thesis, Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, 2003. in spanish.
12. D. López and I. Piñaga. Syntactic pattern recognition by error correcting analysis on tree automata. *LNCS*, 1876:133–142, 2000. Joint IAPR International Workshops SSPR and SPR (S+SSPR 2000).
13. D. López, J. Ruiz, and P. García. *Pattern Recognition and String Matching*, chapter Inference of k-piecewise testable tree languages. Kluwer, 2003.
14. D. López, J. M. Sempere, and P. García. Error correcting analysis for tree languages. *International Journal on Pattern Recognition and Artificial Intelligence*, 14(3):357–368, 2000.
15. E. Mäkinen. On inferring linear single-tree languages. *Information Processing Letters*, 73:1–3, 2000.

16. H. Peng, F. Long, and Z. Chi. Document image recognition based on template matching of component block projections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1188–1192, 2003.
17. H. Peng, F. Long, Z. Chi, and W-C. Siu. Document image template matching based on component block list. *Pattern Recognition Letters*, 22:1033–1042, 2001.
18. S. Ramdane, B. Taconet, and A. Zahour. Classification of forms with handwritten fields by planar hiddenmarkov models. *Pattern Recognition*, 36:1045–1060, 2003.
19. R. Safari, N. Narasimhamurthi, M. Shridhar, and M. Ahmadi. Document registration using projective geometry. *IEEE Transactions on Image Processing*, 6(9):1337–1341, 1997.
20. Y.Y. Tang, M. Cheriet, J. Liu, J.N. Said, and C. Y. Suen. *Handbook of Pattern Recognition and Computer Vision, Document analysis and recognition by computers*. World Scientific Pub. Co., 1999.
21. T. Watanabe, Q. Luo, and N. Sugie. Layout recognition of multi-kinds of table-form documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):432–445, 1995.

Symbols Recognition System for Graphic Documents Combining Global Structural Approaches and Using a XML Representation of Data

Mathieu Delalandre[1], Éric Trupin[1], and Jean-Marc Ogier[2]

[1] Laboratory PSI, University of Rouen, 76 821 Mont Saint Aignan, France
{mathieu.delalandre,eric.trupin}@univ-rouen.fr

[2] Laboratory L3I, University of La Rochelle, 17042 La Rochelle, France
jmogier@univ-lr.fr

Abstract. In this paper we present a symbols recognition system for graphic documents, based on a combination of global structural approaches. Our system allows to extract components and their loops, with their inclusion and neighboring links. So, it is possible to construct different graph types according to the considered recognition problem. Our system uses an XML representation of data, which allows an easy manipulation of these ones. We present some results on a symbols set of GREC2003's recognition contest.

1 Introduction

This paper deals with the structural recognition of symbols applied to graphic documents. We present here a system based on a combination of global structural approaches. In the paper's follow-up, we present in section (2) the general problem of structural recognition of symbols on graphic documents. The two following sections are dedicated to two main system's parts: global structural analysis and images preprocessing in section (3), and structural classification in section (4). In section (5), we present the use of XML in the system. In section (6), we present some results and analysis on a symbols set of GREC2003's recognition contest. Finally, in section (7), we conclude and give some perspectives.

2 Structural Recognition of Symbols

Classically, a documents recognition system is decomposed into two main steps [7]: an images processing step and a recognition step. Two main approaches exist: statistical & connexionnist, and structural & syntactic[1]. This paper deals especially with the structural approach. This one uses graph representations of documents' objects.

[1] In the paper's follow-up, we talk about "structural" for "structural & syntactic".

Into graphic documents, many objects could be described by graphs, especially symbols [10]. Thus, in a structural recognition system of symbols, the images processing step extracts graphs from images corresponding to symbols, and the structural recognition step exploits these graphs.

The structural recognition step is generally used for symbols recognition, but also for other purposes like: learning, indexing, data structuring, and so on. It uses two main approaches: graph-matching [6], and graph-grammar [2]. The first one matches extracted graphs with model graphs. The second one applies different rules to transform extracted graphs into model graphs. A graph problem depends on two criteria: graph/subgraph, and exact-inexact. If extracted graphs correspond exactly to model graphs, the problem is known as exact. Unfortunately, in image applications, extracted graphs are noisy. So, it is an inexact graph/subgraph problem.

The images processing step extracts (or constructs) graphs from images corresponding to symbols. Into a related paper [5], we talk about "structural analysis" for this images processing step and propose a classification into local and global approaches. The boundary between the two approaches is the connected component. Fig. 1 gives an example. The local approach (b) decomposes the connected component (a) into arc, junction, and vector objects. The global approach (c) groups together three connected components (a) according to distance constraints.

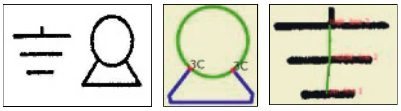


Fig. 1. (a) symbols (b) local structural analysis (c) global structural analysis.

This paper especially deals with the global structural analysis. This one extracts spatial links between connected components, in order to construct graph representations of symbols. Therefore, this analysis processes only the segmented symbols (not connected). Moreover, the connected components are graphical primitives of "low semantic". This allows to deal only with the few classes recognition problems. Three main approaches exist. The first one simply extracts connected components' gravity centers in order to use distance constraints between them with graphs based algorithms [11]. The next two extract more complete links from images between connected components: the inclusion [8] and the neighboring [3] links. The first one generally is based on blob coloring methods [8], and the second one on the generalized Voronoi diagrams methods [3]. The Fig. 2 gives an example of symbol with its inclusion graph (a), and the neighboring graph of its central part (b). Also, "hybrid" approaches exist, like global/local [8] or statistical/structural [4] approaches.

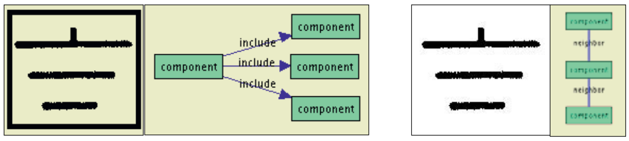


Fig. 2. (a) inclusion links (b) neighboring links.

In this paper we present a system for symbols recognition based on global structural analysis. This one exploits an approaches combination of global structural analysis. It allows to extract from images the connected components and their associated loops (see subsection 3.1), with their inclusion and neighboring links. So it is possible to construct different graph types according to the considered recognition problem. These graphs are next processed by a graph-matching algorithm. We present this system in the three next sections.

3 Global Structural Analysis and Images Pre-processing

We present in this section the first system's part for images processing. Methods presented in this section are grouped in a C++ library for connected components analysis, the CCLib[2]. We present first in subsection (3.1) the general method for connected components labelling and loops extraction. In subsection (3.2), we present the methods to construct connected components graphs. Finally in subsection (3.3), we present a pre-processing method based on connected components' surfaces analysis.

3.1 Connected Components Labelling and Loops Extraction

The central part of system's images processings is a connected components labelling method. Our labelling method is based on regions aggregation with an 8 connectivity analysis of black pixels. The Fig. 3 gives an example with the original image (a), and four successive aggregation steps (b). During these steps, the aggregated pixels are erased (steps 1-3: 3 pixels, step 4: 1 pixel).

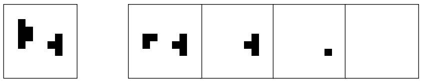


Fig. 3. (a) image (b) four successive aggregation steps.

This method is a few more complex than classical blob coloring methods [1], but labels and extracts in one step the connected components and their characteristics: gravity centers, areas "dx; dy", contour pixels, and so on. The Fig. 4 gives an example image (a) with its labelling result into SVG format (b) (see section 5).

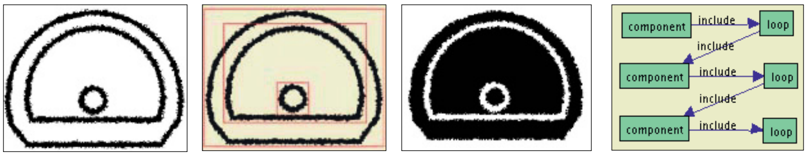


Fig. 4. (a) symbol (b) components labelling (c) loops extraction (d) inclusion graph.

[2] Connected Component Library: http://site.voila.fr/mdhws/

We use too this labelling method to extract loops images[3] (Fig. 4 (c)). The image's borders are first initialized into the image's background colour. Next, the image is inverted. This inverted image is next labelled. With the borders initialization, the first labelled connected component corresponds to image's background. This one is erased, and the others are used to create the loops image (Fig. 4 (c)).

3.2 Graphs Construction

Two main links exist between connected components (see section 2), the inclusion and the neighboring links. We present first two methods to extract these links. Next, we present a method to construct hybrid graphs combining these two links.

Firstly, we extract the inclusion links between connected components. The Fig. 4 (d) gives an example of inclusion graph into XGMML format (see section 5) of Fig. 4 (a). These inclusion links are directed, and give a tree description of symbol's components and their associated loops. Our method first extracts symbol's loops image (Fig. 4 (c)). Next, it labels components image and its associated loops image (Fig. 4 (a) and (c)). Finally in these two labelled images, the method analyses the contours' labels of components and loops in order to extract their inclusion links (Fig. 4 (d)).

Secondly, we extract the neighboring links between connected components. These links are undirected, and give a graph description of symbol's components or loops. Our method is based on the extension of connected components' contours. During this extension, the extended contours are labelled. The process is stopped when the extended contours meet other extended contours, connected components, or null zones "out of image". The Fig. 5 (b left) gives an example of four successive steps of contours extension of Fig. 5 (a). Then, the method extracts the boundary points from the obtained labelled map of extended contours. The Fig. 5 (b right) gives an example of image representation of extracted boundary points of Fig. 5 (a). Finally, these boundary points are next analyzed in order to find the neighboring links between connected components. The Fig. 5 (c) gives the neighboring graph into XGMML format (see section 5) of Fig. 5 (a). This method is more complex (θ^2 complexity)[4] than generalized Voronoi diagrams based methods (θ complexity) [3]. However, this method deals with the "high precision" problems. It allows to extract the exact boundary points (with a pixel precision), and gives in some cases more complete results than generalized Voronoi based methods.

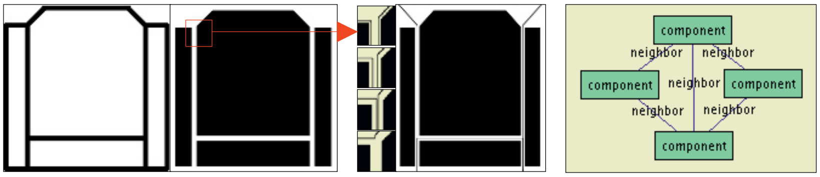


Fig. 5. (a) symbol (b) contours extension (c) neighboring graph.

[3] In the paper's follow-up, we talk about symbol's "components" and associated "loops".

[4] 25.1 seconds for a no compressed plan of 364 Kbytes (CPU 2 GHz, Windows System).

Finally, we use a last method in order to construct hybrid graphs, with the inclusion and neighboring links. The Fig. 6 (a) gives an example of hybrid graph into XGMML format (see section 5) of Fig. 5 (a), with the include links between connected components and loops, and the neighboring links between loops (Fig. 5 (c)). Our method uses the neighboring and inclusion graphs, with their associated connected components' characteristics provided by the labelling method (label, gravity centers, areas, and so on.). Next, the method analyses the two graphs and connected components' characteristics in order to construct the hybrid graphs. So, it is possible to construct three hybrid graph types, components based, loops based, loops and components based "both". The Fig. 6 (b), (c), and (d) give examples of three hybrid graph types into XGMML format (see section 5) of Fig. 4 (a).

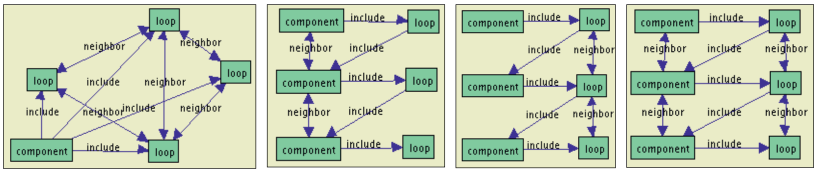


Fig. 6. (a) hybrid graph (b) components based (c) loops based (d) both based.

3.3 Images Pre-processing

The graph construction methods presented in the last subsection (3.2) are based on connected components extraction. Therefore, these methods are very sensitive to sparse noise (small components and loops adding). The Fig. 7 (b) gives an example of sparse noised image of Fig. 7 (a).

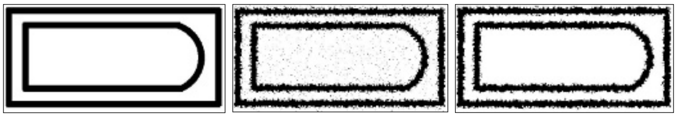


Fig. 7. (a) model image (b) sparse noised image (c) filtered image.

To solve this problem, we use a filtering method based on connected components' surfaces analysis (1). First, a table (*S*) of no null connected components' surfaces is created. This one can be created from components image, loops image, or to merge the both. From this surfaces table, a surface ratios table *(R)* is computed. The maximum ratio *(r)* is searched in order to find the surface threshold of image *(th)*. The Fig. 7 (c) gives an example of filtered image of Fig. 7 (b).

$$S = \bigcup_{i=1}^{n} s_i \; ; \; R = \bigcup_{i=1}^{n-1} \left(r_i = {}^{s_{i+1}}\!/_{s_i} \right) ; \; r_j = \max(R) \; ; \; th = s_j \; . \qquad (1)$$

4 Structural Classification

The extracted graphs (see section 3) are next exploited by an inexact graph-matching algorithm [9]. This algorithm allows to compute[5] similarity criterion between graphs (**2**), based on the overlap between a candidate graph *(g1)* and a model graph *(g2)*. This overlap corresponds to their common sub-graph *(gc)*. Two similarity criteria are computed (**3**) according to the common elements number *{cnn, cen}* on the nodes (δn), and on the edges (δe). The global similarity criterion is obtained by variance computation of (δn) and (δe). The classification's result corresponds to model graph's label of minimum global similarity criterion.

$$g1 = \{n_0,..,n_{nn1};e_0,..,e_{en1}\};\ g2 = \{n_0,..,n_{nn2};e_0,..,e_{en2}\};\ gc = \{n_0,..,n_{cnn};e_0,..,e_{cen}\}\,. \quad (2)$$

$$\delta e(g1,g2) = \frac{en1 \times en2}{cen^2} - 1\ ;\ \delta n(g1,g2) = \frac{nn1 \times nn2}{cnn^2} - 1\,. \quad (3)$$

The model graphs are learned through a graphics user interface: ojgBE[6]. The Fig. 6 gives examples of graphs' graphics visualizations. ojgBE allows to edit labelled graphs, directed and/or undirected. The edited graphs are included into a graphs base. The ojgBE user can browse into this graphs base and perform various actions like: graph labelling, graph copy, base sorting, graph orientation change, similarity analysis, and so on. The graphs base is stored into XGMML format (see section 5).

5 XML Use in System

The different system's parts (sections 3 and 4) generate output data into XML and its sub-languages. The use of this data representation language offers several enhancements in comparison with classical formats. XML is a meta-language because it is defined as a root language, which enables to define specialized sub-languages. We use SVG[7] for graphic data representations (Fig. 4 (b)). We also use XGMML[8] for graphs descriptions (see Fig. 6) in the global structural analysis tools (see section 3), the structural classifier (see section 4), and the graphics user interface ojgBE (see section 4). XML uses parsers and transforming processors. The parsers easily access to XML data, and the processors easily transform the XML data according to XSLT[9] scripts. This enables an easy data manipulation in the system, for data communication between system' parts (sections 3 and 4), and results evaluation (see section 6).

[5] We don't develop this algorithm here and report the reader to [9].
[6] Open java graph Base Editor: http://site.voila.fr/mdhws/
[7] Scalable Vector Graphics.
[8] eXtensible Graph Markup and Modeling Language.
[9] eXtensible Stylesheet Language Transform.

6 Experiment, Results, and Analysis

We present here some results obtained on a symbols set of GREC2003[10]'s recognition contest. This symbols recognition contest deals with the recognition of segmented architectural and electrical symbols (Fig. 8). Different sample tests are available[10] according the classes number (symbols set), binary or vectorial degradations[11], scale or orientation changes, and so on. These sample tests are available with their models files, in order to evaluate recognition results.

We have tested our system with a symbols set of 9 symbols (Fig. 8). We have used a hybrid graph representation based on loops' neighboring links (see subsection 3.2). Indeed, the inclusion and neighboring graphs aren't adapted for this recognition application. The inclusion graphs are similar for symbols s8 and s9, and the loops based neighboring graphs for symbols s1 and s7. Also, the components based hybrid graphs are similar for symbols s1 and s4. We have tested our system on 600 images, with 6 sample tests (of 100 images sized) of binary degradations (degrad-level2-m1 to degrad-level2-m6[12]). The Fig. 7 (a) gives an example of binary degradation of Fig. 7 (b).

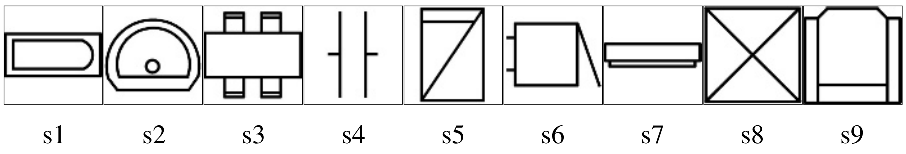


Fig. 8. Symbols set.

The Fig. 9 (a) gives the samples tests' results. We obtain perfect results on all tests, except on the test5 (89%). In order to complete these results, we have measured two noise types. The first one is an estimation of dilatation noise (**4**). This estimation *(Êd)* is computed between test images *(ti)*, and model images *(mi)*. It is based on the search of common black pixels number *(cbp)* between test and model images, and the black pixel number *(bp)* of test image. The second one is a structural noise. This structural noise gives the rate of noised graphs into the extracted graphs. These noised graphs are detected in regard to their model graphs. These two measures give noise rates before and after the global analysis step. We can see on Fig. 9 a partial correlation between the two noise curves. Indeed, loops and components are "bullet-proof" graphical primitives (the perfect results prove it). The main problem is the dilatation noise which created loops closures and components merges on symbols' noised images. This dilatation noise is the main problem area of structural noise.

$$\hat{E}_d(ti,mi) = 1 - cbp(ti,mi)/bp(ti) \ . \tag{4}$$

[10] International Conference on Graphics Recognition 2003: http://www.cvc.uab.es/grec2003/

[11] We report the reader to contest's web page[10] for information on the noise models used.

[12] In the section's follow-up, we talk about test1, test2, and so on.

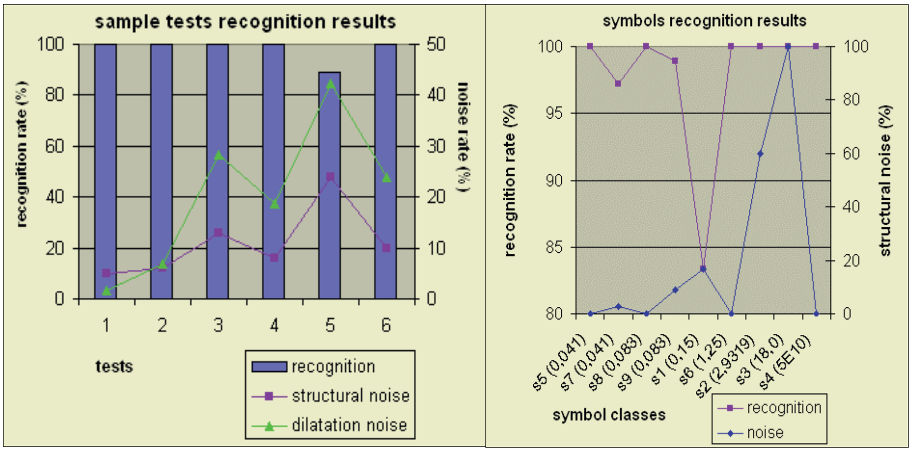


Fig. 9. (a) sample tests recognition results (b) symbols recognition results.

In order to analyze the structural noise distribution, we have computed the recognition rate and structural noise rate for each symbol class with all tests' results (Fig. 9 (b)). We can see a high level of noise (>15%) for symbols s1, s2 and s3. The Fig. 10 gives examples of the three main cases of structural noise met for these symbols. The first one is the components merge case (a). Indeed, the dilatation noise can merge the nearest components. It is a typical structural noise of symbol s1 and s2. The second one is the loops closure case (b). The dilatation noise can close the small size loops. This structural noise appears on symbols s2, s3, and s7. The last one is the neighboring links distortion case (c). The loops' distortions create and delete neighboring links in some difficult cases (like s3). The Fig. 9 (c) gives the model neighboring image (left), and an example of test neighboring image (right). The loops' distortions add diagonal neighboring links (with small boundaries) between symbol's loops. Still, the structural noise's effects depend of symbol classes. In the case of symbol s7 and s1, a low level of noise has important effects on the symbols recognition. In the case of symbols s2 and s3, a high level of noise has no effect on the symbols recognition. Indeed, the graph models used haven't the same "bullet-proof" criterion. In Fig. 9 (b) we have computed (and sorted) for each model graph the minimum similarity criterion with the other base's model graphs. This minimum criterion can be view as the "bullet-proof" criterion of considered model graph. Like this, we can see two main classes in the model graphs base: symbols of low minimum similarity criteria (≤0.15) (s1, s5, s7, s8, and s9), and symbols of high minimum similarity criteria (≥ 1.25) (s2, s3, s4, and s6).

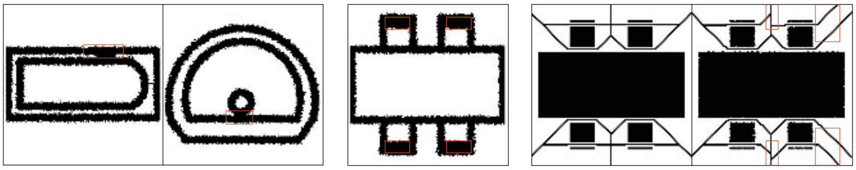


Fig. 10. (a) components merge (b) loops closure (c) neighboring links distortion.

7 Conclusion and Perspectives

In this paper we have presented a symbols recognition system applied to graphic documents. This system is based on a combination of global structural approaches. It allows to extract components and their loops, with their inclusion and neighboring links. It is possible to construct different graph types according to the considered recognition problems. These graphs are next processed by an inexact graph-matching algorithm. We present some recognition results and analysis on a noisy symbols set of GREC2003's contest. This system gives very good results. Still, it is sensitive to dilatation noise. Moreover, the global approaches allows to deal only with segmented symbols, and the few classes recognition problem.

For the perspectives, we want first realized contextual pre-processing step, in order to erode the dilated images. Next, we want to combine our global approach with statistical approach [4] in order to deal with the large classes recognition problems. Also, we want to extend our methods library with a generalized Voronoi method, in order to reduce the images' process times in the case of "low precision" problems. Finally, we want to combine our graph-matching algorithm with a graph-grammar tool, in order to correct some structural noise cases.

References

1. A.K. Aggarwal, A.V. Kulkarni. A Sequential Approach to the Extraction of Shape Features. Computer Graphics and Image Processing (CGIP), 6(6) : 538-557, 1977.
2. D. Blostein, H. Fahmy, A. Grbavec. Issues in the Practical Use of Graph Rewriting. Lecture Notes in Computer Sciences (LNCS), 1073 : 38-55, 1996.
3. M. Burge, G. Monagan. Using the Voronoi Tessellation for Grouping Words and Multi Part Symbols in Document. Vision Geometry IV, 1995.
4. M. Delalandre, P. Héroux, S. Adam, E. Trupin, J.M. Ogier. A Statistical and Structural Approach for Symbol Recognition Using XML Modelling. Structural and Syntactical Pattern Recognition (SSPR), 2002.
5. M. Delalandre, E. Trupin, J.M. Ogier. Local Structural Analysis: A Primer. Graphics Recognition (GREC), 2003.
6. E. Hancock, R. Wilson. Graph-Based Methods for Vision: A Yorkist Manifesto. Structural and Syntactical Pattern Recognition (SSPR), 2002.
7. R. Kasturi, L. O'Gorman, V. Govindaraju. Document Image Analysis: A Primer. Sadhana, 27(1) : 3-22, 2002.
8. O. El Badawy, M. Kamel. Shape Representation using Concavity Graphs. International Conference on Pattern Recognition (ICPR), 2002.
9. P. Héroux, S Diana, E. Trupin, Y. Lecourtier. A Structural Classification for Retrospective Conversion of Document. Structural and Syntactical Pattern Recognition (SSPR), 2000.
10. J. Lladós, E. Valveny, G. Sánchez, E. Martí. Symbol Recognition : Current Advances an Perspectives. Graphics Recognition (GREC), 2001.
11. P.K. Loo, C.L. Tan. Detection of Word Group Based on Irregular Pyramid. International Conference on Document Analysis And Recognition (ICDAR), 2001.

Sketch Recognition Based on Topological Spatial Relationship

Binbin Peng, Yin Liu, Liu Wenyin, and Guanglin Huang

Dept. of Computer Science, City University of Hong Kong, Hong Kong SAR, PR China
{cspengbb,liuyin,csliuwy}@cityu.edu.hk
hwanggl@cs.cityu.edu.hk

Abstract. In recognition of composite graphic objects, topological spatial relationships of their components play an important role. Although most researchers focus on binary topological relationships, they cannot carry all information of the internal structure of the compound objects. Therefore, we introduce the ternary relationship, which is a complement to the binary relationship, to describe composite graphic objects. Moreover, we provide a constrained partial permutation algorithm based on both the binary and ternary topological spatial relationships to recognize the sketchy objects input by users in an online manner. Experimental results show that this approach is both efficient and effective for online composite graphics recognition in our sketch-based graphics input system – SmartSketchpad.

1 Introduction

Sketching is a way of externalizing ideas, of turning internal thoughts public, of making fleeting thoughts more permanent [1]. People usually use sketches to express and record their ideas in many domains, including mechanical engineering, software design, information architecture [2] and map schematizing [3]. Although it may be easy for people to understand the designer's intention, which is presented in his sketches intangibly, it is not an easy case for computers. The ambiguity of sketches causes many problems in recognition.

The online graphics recognition problem can be specified into three levels: primitive shape recognition, composite object recognition, and document-level recognition and understanding [4]. For composite object recognition, existing approaches can be divided into two categories: the SPR (Statistical Pattern Recognition) approaches, such as Neural Networks (NN) [5] and Hidden Markov Model (HMM) [6], and the SSPR (Syntactical and Structural Pattern Recognition) approaches, such as Attributed Relation Graph (ARG) and Region Adjacent Graph (RAG) [7][8]. These methods mainly focus on unitary or binary relationships. In addition, contextual (top-down) knowledge has also been used to recognize freehand sketches of simple 2-D mechanical devices [9].

Sketches have the advantage of conveying elements and spatial relations in the real world with elements and spatial relations on paper and describing visuospatial ideas directly [1]. Users' sketchy objects are composed of both primitive shapes (e.g. line segments, arc segments, and ellipses) and their topological spatial relationships. These relationships among the primitive shapes play an important role in identifying composite objects. For example, when people distinguish one object from the others, they usually pay more attention to the structure of its components than to the absolute position, size, and orientation of each component. Topological relationship is a particular subset of geometric relations that are invariant under geometric transformations such as translation, rotation, and scaling. Traditionally, topological relationship is defined between two objects, such as regions, lines or points [10]. Zhan [11] provided eight binary topological relations between two fuzzy regions that are distinguished by the 9-intersection model [10], e.g. disjoint, contains, inside, equal, meet, covers, coveredby, and overlap. However, binary topological relationship itself cannot carry all information of the internal structure of composite objects. For example, it is impossible to distinguish the relative sequence of three lines, which are parallel, only with binary topological relationships. However, with the proposed ternary relationships, we are able to distinguish it easily. In this paper, two ternary relations are introduced, which, along with certain binary relationships, are used to describe the internal structure of composite graphic objects. A constrained partial permutation algorithm is also presented to recognize sketchy objects using these relationships. Experimental results show that this approach is both efficient and effective for online composite graphics recognition in our sketch-based graphics input system – SmartSketchpad [12].

The remainder of this paper is organized as follows: Section 2 introduces 5 classes of binary topological relationships and two ternary topological relationships that are used in our system. Section 3 presents a constrained partial permutation algorithm for recognition of sketchy objects using their topological relationships. Experiments and evaluations are presented in Section 4. Section 5 presents concluding remarks.

2 Topological Spatial Relationship

2.1 Notations

The point in 2D plane can be denoted as *Pt*. The primitive shape in a graphic object, as we consider, can be line segment (denoted as *Ls*), arc segment (denoted as *As*), or ellipse (denoted as *E*). The primitive shape is denoted as *P*, and CLASS(*P*) is the class of *P*. *L* is the line in which the segment (*Ls*) locates and *Ra* is the circle on which the arc segment (*As*) locates. The relationship between two primitive shapes (e.g. P_1 and P_2) can be denoted as $\mathrm{R}(P_1,P_2)$ and $\mathrm{R}'(P|P_1,P_2)$ is a ternary relationship among three primitive shapes (e.g. P, P_1, and P_2). Moreover, we use $\mathrm{R}(P,\bullet)$ or $\mathrm{R}'(P|\bullet,\bullet)$ to express all binary or ternary relationships which contain P and $\Phi(\mathrm{R}(P,\bullet))$ or $\Phi(\mathrm{R}'(P|\bullet,\bullet))$ is the number of relationships in $\mathrm{R}(P,\bullet)$ or $\mathrm{R}'(P|\bullet,\bullet)$.

In addition, $f(x,y)$ is the function of the primitive shape. For line, $f(x,y)= \mathrm{a}x+\mathrm{b}y+\mathrm{c}$, where b>=0 and if b=0, a>0. For ellipse, $f(x,y)=(x-x_0)^2/\mathrm{a}^2+(y-y_0)^2/\mathrm{b}^2-\mathrm{c}$, where c>0 and a,b≠0. For circle, $f(x,y)= (x-x_0)^2+(y-y_0)^2-\mathrm{c}$, where c>0.

2.2 Binary Topological Relationships

We first define five binary spatial relationships that are useful for sketch recognition as follows.

Definition 1. Binary Topological Relationships
Given two primitive shapes P_1 and P_2, the binary spatial relationship R(P_1,P_2) can be defined as follows, which are also illustrated in Fig. 1.

(i). Interconnection (R_{IC})**:** P_1 and P_2 have common end points, or two ellipses join together (Fig. 1(a)). Denoted as $R_{IC}(P_1,P_2)$.

(ii). Tangency (R_T)**:** The end points of P_1 are quite close to (or touching) some inner points of P_2, or a line segment is tangent to an ellipse (Fig. 1(b)). Denoted as $R_T(P_1,P_2)$.

(iii). Intersection (R_{IS})**:** If P_1 and P_2 have common inner points (Fig. 1(c)), we define it as $R_{IS}(P_1,P_2)$.

(iv). Parallelism (R_P)**:** P_1 and P_2 are line segments and are approximately parallel [13] within a sufficiently close distance (Fig. 1(d)). Denoted as $R_P(P_1,P_2)$.

(v). Concentric (R_C)**:** The centers of two ellipses or arc segments are sufficiently close (Fig. 1(e)). Denoted as $R_C(P_1,P_2)$.

According to above definition, the relationship R can be recognized even after geometric transformations. Therefore, these binary relationships are insensitive to the direction, the size and the rotation.

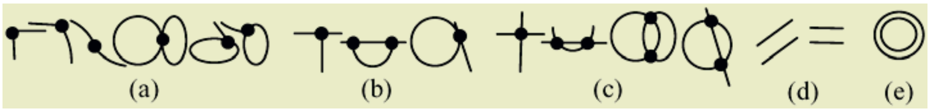


Fig. 1. Some examples of binary spatial relationship, with joint points drawn in thick black.

2.3 Ternary Topological Relationship

The above binary relationships are insufficient to distinguish all possible spatial relations among components. For instance, we cannot distinguish the relative sequence of three parallel lines or three concentric ellipses with only binary relationships. Therefore, the ternary relationship is introduced. Before we present the detailed definitions of the ternary relationship, we first define another two binary relationships: the *over* relationship and the *below* relationship. Given two primitive shapes P and P', which are not exactly the same in their geometric functions, and if the function of P is $f(x,y)$, we define:

Definition 2. The over relationship
If $\forall Pt$ in the primitive shape P', $f(\mathrm{Pt})\geq 0$ and there is at most one point Pt' at which $f(Pt')=0$, we can say P' is over P, denoted as $P'\uparrow P$.

Definition 3. The below relationship
If $\forall Pt$ in the primitive shape P', $f(Pt)\leq 0$ and there is at most one point Pt' at which $f(Pt')=0$, we can say P' is below P, denoted as $P'\downarrow P$.

Although the over/below relationships for circles or ellipses are rotation-invariant, the two relationships for lines are sensitive to rotation. Therefore, we define the following two ternary relationships based on the over/below relationships.

Definition 4. The middle relationship
Given three primitive shapes (P,P_1,P_2), if $P_1\uparrow P$ and $P_2\downarrow P$, we can say that P has the middle relationship with the other two primitive shapes (P_1 and P_2), denoted as $\mathrm{R'}_{\mathrm{M}}(P|P_1,P_2)$.

Every three primitive shapes may have more than one (at most two) middle relationship. For example, the object in Fig. 2 has three line segments and one ellipse. The three line segments have two middle ternary relationships: $\mathrm{R'}_{\mathrm{M}}(P_a|P_c,P_b)$ and $\mathrm{R'}_{\mathrm{M}}(P_b|P_a,P_c)$ which are drawn from the over/below relationships $P_c\uparrow P_a$, $P_c\downarrow P_b$, $P_a\uparrow P_b$ and $P_b\downarrow P_a$.

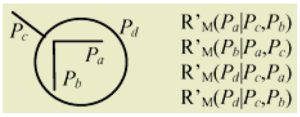

P_c
P_d
P_a
P_b
$\mathrm{R'}_{\mathrm{M}}(P_a|P_c,P_b)$
$\mathrm{R'}_{\mathrm{M}}(P_b|P_a,P_c)$
$\mathrm{R'}_{\mathrm{M}}(P_d|P_c,P_a)$
$\mathrm{R'}_{\mathrm{M}}(P_d|P_c,P_b)$

Fig. 2. An example object and its ternary relationships.

Definition 5. The parallel-middle relationship
If $\mathrm{R}_{\mathrm{P}}(P_1,P_2)$ or $\mathrm{R}_{\mathrm{C}}(P_1,P_2)$, and there is a primitive shape P, which can meet the conditions: $P\downarrow P_1$ and $P\uparrow P_2$, we say that these three primitive shapes have the parallel-middle relationship, denoted as $\mathrm{R'}_{\mathrm{P\text{-}M}}(P|P_1,P_2)$.

Apparently, the parallel-middle relationship can meet the rotation-invariance, that is, $\mathrm{R'}_{\mathrm{P\text{-}M}}(P|P_1,P_2)=\mathrm{R'}_{\mathrm{P\text{-}M}}(P|P_2,P_1)$. In addition, for the line segments, this ternary relationship is sensitive to the parallel relationship. Therefore, we cannot use the traditional parallel definition. For example, P_c and P_a in Fig. 3 may be considered as parallel under the traditional definition. Therefore, we use Revankar and Yegnanarayana's method [13] to judge the parallel relationship.

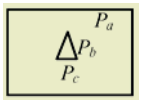


Fig. 3. The parallel-middle relationship is sensitive to the definition of the parallelism.

3 Sketch Recognition

The key problem in the composite object recognition process is how to match a sketchy object (SO) with a pre-defined object (PO). Obviously, enumeration of all possible cases is equivalent to a Partial Permutation problem, which is proved to be an NP-complete problem [14]. For an object composed of more than 8 components, it is not suitable to enumerate all cases in on-line recognition system. In fact, most permutations are illegal mappings due to violation of some constraints and can be neglected directly during the permutation process. Hence, this partial permutation is a constrained one and is referred to as the *Constrained Partial Permutation* by Xu et al. [15]. Xu et al.'s method [15] is based on binary relationships. In this paper, we improve it by handling with the ternary relationships as well as binary relationships.

3.1 Constrained Partial Permutation

Definition 6. Constrained Partial Permutation
Given that SO contains m primitive shapes (e.g. O_1, O_2, .. O_m) and PO contains n primitive shapes (e.g. P_1, P_2, .. P_n) where $m \leq n$, select m integers from [1..n], and then rank them in a list as enumeration sequence, written as $B(1)B(2)...B(m)$, where, 1, 2, ..., k, ...,m are the positions in the list and $B(1)$, $B(2)$, ..., $B(k)$, ...$B(m)$ are the values (1..n) at these positions in the list. We regard SO as a part of PO, only if we can find a possible enumeration sequence which satisfies all of the following three constraints:

Constraint 1: $\forall r \in [1,m]$, CLASS(O_r)=CLASS($P_{B(r)}$), $\Phi(\mathrm{R}_{\mathrm{IC,T,IS,P,or\ C}}(P_{B(r)},\bullet)) \geq \Phi(\mathrm{R}_{\mathrm{IC,\ T,IS,P,or\ C}}(O_r,\bullet))$, $\Phi(\mathrm{R'}_{\mathrm{M}}(P_{B(r)}|\bullet,\bullet)) \geq \Phi(\mathrm{R'}_{\mathrm{M}}(O_r|\bullet,\bullet))$ and $\Phi(\mathrm{R'}_{\mathrm{P\text{-}M}}(P_{B(r)}|\bullet,\bullet)) \geq \Phi(\mathrm{R'}_{\mathrm{P\text{-}M}}(O_r|\bullet,\ \bullet))$

Constraint 2: $\forall r,\ s \in [1,m]$, if there is a binary relationship $\mathrm{R}_{\mathrm{IC,T,IS,P,or\ C}}(O_r,O_s)$, there must be a relationship $\mathrm{R}_{\mathrm{IC,T,IS,P,or\ C}}(P_{B(r)},P_{B(s)})$ of the same type.

Constraint 3: $\forall r,\ s,\ t \in [1,m]$, if there is a ternary relationship $\mathrm{R'}_{\mathrm{M\ or\ P\text{-}M}}(O_r|\ O_s,\ O_t)$, there must be a relationship $\mathrm{R'}_{\mathrm{M\ or\ P\text{-}M}}(P_{B(r)}|P_{B(s)},\ P_{B(t)})$ of the same type.

To improve the efficiency of constrained partial permutation, we define four denying rules as follows:

Definition 7. Single Denying Rule:
Given $r \in$ [1..m] in SO and $i \in$ [1..n] in PO, P_i cannot be put into the place r, that is, $B_r \neq i$, written in a 2-tuple (i, r), if one of the following conditions can be met: (1) CLASS(O_r)≠CLASS(P_i); (2) $\Phi(\mathrm{R}_{\mathrm{IC,T,IS,P,or\ C}}(P_i,\bullet)) < \Phi(\mathrm{R}_{\mathrm{IC,T,IS,P,or\ C}}(O_r,\bullet))$; (3) $\Phi(\mathrm{R'}_{\mathrm{M}}(P_i|\bullet,\bullet)) < \Phi(\mathrm{R'}_{\mathrm{M}}(O_r|\bullet,\bullet))$; (4) $\Phi(\mathrm{R'}_{\mathrm{P\text{-}M}}(P_i|\bullet,\bullet)) < \Phi(\mathrm{R'}_{\mathrm{P\text{-}M}}(O_r|\bullet,\bullet))$.

Definition 8. Pair Denying Rule 1:
Given $r,\ s \in$ [1..m] in SO and $i,\ j \in$ [1..n] in PO, and P_j has been put into the place s (that is, $B_s = j$), if O_r has the binary relationship $\mathrm{R}_{\mathrm{IC,T,IS,P,or\ C}}(O_s,\ O_r)$ but P_i does not have the same binary relationship $\mathrm{R}_{\mathrm{IC,T,IS,P,or\ C}}(P_i,\ P_j)$, P_i cannot be put into the place r, that is, (i, r)-(j, s).

Definition 9. Pair Denying Rule 2:
Given $r,\ s \in$ [1..m] in SO and $i,\ j \in$ [1..n] in PO, and P_j has been put into the place s (that is, $B_s = j$), if O_r has the ternary relationship $\mathrm{R'}_{\mathrm{M,\ or\ P\text{-}M}}(O_r|O_s,\bullet)$ or $\mathrm{R'}_{\mathrm{M,\ or\ P\text{-}M}}(O_s|O_r,\bullet)$ but P_i does not have the same ternary relationship $\mathrm{R'}_{\mathrm{M,\ or\ P\text{-}M}}(P_i|P_j,\bullet)$ or $\mathrm{R'}_{\mathrm{M,\ or\ P\text{-}M}}(P_j|P_i,\bullet)$, P_i cannot be put into the place r, that is, (i, r)-(j, s).

Definition 10. Triangle Denying Rule:
Given r, s, $t\in[1..m]$ in SO and $i, j, k\in[1..n]$ in PO, P_j has been put into the place s and P_k has been put into the place t (that is, $B_s=j$ and $B_t=k$), if O_r has the ternary relationship $\text{R'}_{\text{M, or P-M}}(O_r|O_s,O_k)$ or $\text{R'}_{\text{M, or P-M}}(O_s|O_r,O_k)$ but P_i does not have the same ternary relationship $\text{R'}_{\text{M, or P-M}}(P_i|P_j,P_k)$ or $\text{R'}_{\text{M, or P-M}}(P_j|P_i,P_k)$, P_i cannot be put into the place r, that is (i, r)-(j, s)(t, k).

Therefore, given a permutation $B(1)B(2)...B(m)$, we can derive a function f: $[1..m]\rightarrow[1..n]$, where $f(i)=B(i)$ for any $i\in[1..m]$. We obtain four denying rules according to the following steps:

Step 1: if f is rejected by constraint 1, there must exist r ($r\in[1..m]$), which let O_r and $P_{B(r)}$ meet the single denying rule. Then we obtain a single denying rule ($B(r)$, r).

Step 2: if f is rejected by constraint 2, there must exist r, $s\in[1..m]$, which let O_r O_s and $P_{B(r)}$ $P_{B(s)}$ meet the pair denying rule 1. Then we obtain a pair denying rule ($B(r)$, r)-($B(s)$, s).

Step 3: if f is rejected by constraint 3, there must exist r, $s\in[1..m]$, which let O_r O_s and $P_{B(r)}$ $P_{B(s)}$ meet the pair denying rule 2, or there must exist r, s, $t\in[1..m]$, which let O_r, O_s, O_t and $P_{B(r)}$, $P_{B(s)}$, $P_{B(t)}$ meet the triangle denying rule. Then we obtain a pair denying rule ($B(r)$, r)-($B(s)$, s) or a triangle denying rule ($B(r)$, r)-($B(s)$, s)($B(t)$, t).

In order to acquire all possible permutations, we enumerate all such m-digit integers from the smallest, i.e., $123...m$, to the largest, i.e., n (n-1) (n-2) ... (n-m+1). Each time we give its next permutation by finding the smallest m-digit number that is not found before. The denying rules will be generated gradually. If the current permutation is $b_1b_2...b_{k-1}ib_{k+1}...b_m$, and it is rejected by the single denying rule (i,k), we directly skip all permutations with the form $b_1b_2...b_{k-1}iB_{k+1}...B_m$, where B_x ($x=k+1..m$) belongs to the set $\{1,2,3,...n\}-\{b_1,b_2,...b_{k-1},i\}$ and $B_p\neq B_q$ (p, $q=k+1..m$). If the current permutation is $b_1b_2...b_{k-1}ib_{k+1}...b_{t-1}jb_{t+1}...b_m$, where $t>k$, and it is same when it is rejected by the pair denying rule (i, k)-(j, t) or the triangle denying rule (i, r)-(j, s)(t, k). By doing so, many illegal mappings (permutations) can be pruned directly in the enumerating process.

3.2 Similarity

The similarity between SO and PO in a given match (legal mapping/permutation $B(1)B(2)..B(m)$) is denoted by $\text{Sim}_\text{B}(\text{SO},\text{PO})$ and defined as the weighted sum of the similarities of all pairs of matched primitives, as follows. ($L(P)$ is the length (for line/arc segments) or perimeters (for ellipses) of a primitive shape P.)

$$\text{Sim}_B(\text{SO},\text{PO})=\sum_{r=1}^{m}\omega_r\frac{\min(L(P_{B(r)}),L(O_r))}{\max(L(P_{B(r)}),L(O_r))}\ \text{where}\ \omega_r=L(O_r)/\sum_{j=1}^{m}L(O_j), r\in[1...m], \tag{1}$$

Then, the final similarity between SO and PO is defined as the maximal similarity under all possible mappings/permutations, as follows.

$$\text{Sim}(\text{SO},\text{PO})=\max_{B\in\psi(\text{SO},\text{PO})}\text{Sim}_B(\text{SO},\text{PO}) \tag{2}$$

where $\psi(\text{SO},\text{PO})$ is the set of all legal mappings between SO and PO.

4 Experiments

4.1 Experimental Environments

In this experiment, we simulate the sketch-based graphics input and recognition process in practical applications. Firstly, 349 composite graphic objects are created to form a database. Each object in the database contains no more than 15 components. Secondly, we use the method of Xu et al. [15] to generate queries from these objects randomly and match these queries with those objects in the database. For a given graphic object, which has m components, denote the width and height of the object as w and h respectively. A query is generated according to a noise rate τ_n, which is used to simulate the drawing noises, and a completion rate τ_C, which is used to simulate the incomplete form.

Algorithm 2: Query Generating. (Given a graphic object with n components and its completion rate τ_C)

```
Step 1: randomly select $n*\tau_C$ components.
Step 2: generate simulation noises for each component.
    Step 2.1: generate a horizontal shifting factor $\iota_H$ and a vertical shifting factor $\iota_V$
        between $-\tau_n$ and $+\tau_n$. Then shift this component by $\iota_H*w$ horizontally and by $\iota_V*h$
        vertically.
    Step 2.2: generate a random scaling factor $\iota_S$ between $1-\tau_n$ and $1+\tau_n$, and then rescale
        the component according to $\iota_S$.
    Step 2.3: for each component, generate a random rotation factor $\iota_R$ between $-\tau_n*\pi$
        and $\tau_n*\pi$, and then rotate the component according to $\iota_R$ counterclockwise.
Step 3: Combine these $n*\tau_C$ components as a whole group, then shift, rotate, and
    rescale this group, randomly, to form a query $q$.
```

Fig. 4(a) shows a regular object stored in the database and Fig. 4(b) is the generated query object for this regular object at $\tau_C = 0.9$ and $\tau_n = 0.1$.

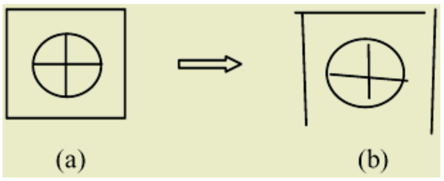


Fig. 4. An example of generated query object from a regular object. (a) The original model object stored in the database; (b) the generated query at $\tau_C = 0.9$ and $\tau_n = 0.1$.

For the *i-th* object in the database, we can generate a query q_i randomly according to τ_n and τ_C. Next, we rank all POs in the database according to their similarities to q_i in a descending order. Denote the position of the *i-th* object itself in the ranking list as $Rank_i$, e.g., $Rank_i$ equals to 2, that is, the *i-th* object has the second highest similarity to the query q_i. The recall rate R_n is defined as:

$$R_n = \sum_{i=1}^{Total\_Query} \frac{RANK_{\leq n}(q_i)}{Total\_Query}, \text{ where } RANK_{\leq n}(q_i) = \begin{cases} 1 & if\,(Rank_i \leq n) \\ 0 & if\,(Rank_i > n) \end{cases} \quad (3)$$

The experiment environment is Pentium III 1.3G CPU, 256MB memory, Windows XP, Visual C++ 6.0.

4.2 Performance Evaluation

In the experiment, τ_n is set to be 0, 0.1, and 0.2, respectively, to simulate different drawing styles. τ_n=0 is set to simulate a very formal drawing, which has few noise. τ_n=0.1 is set to simulate an ordinary user-sketched drawing, which has some noises. τ_n=0.2 is set to simulate a haste drawing, which has many noises when user draws freely. On the other hand, τ_C increases from 0.5 to 1.0 simulating different completion status. R_i is defined in Eq (3). R_i in percentage under different τ_n and τ_C are shown in Fig. 5.

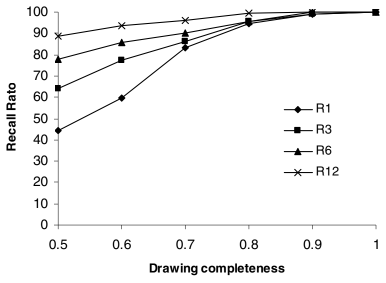
From Fig. 5 we can see that our algorithm is relatively sensitive to noises when a few components have been drawn. As more components have been drawn (i.e., as the object is more complete), higher recall rates can be achieved. We can also see that, when noises are relatively small (τ_n=0 or 0.1), the sketchy graphic object can be successfully recognized before it has been drawn completely. For instance, when a user has drawn 80% of his/her intended object with τ_n=0.1, the rate of successful recognition is above 80%. Hence, we can draw the conclusion that our approach can achieve good performance under noises for incomplete sketchy input.

Our algorithm uses not only the binary relationship but also the ternary relationship to recognize composite objects. This method can get a better performance than the method only using the binary relationships (that is, Xu et al.'s method [15]). Fig. 6 shows that the recall ratio of our algorithm is higher than Xu et al.'s algorithm when τ_n=0.1 and τ_n=0.2.

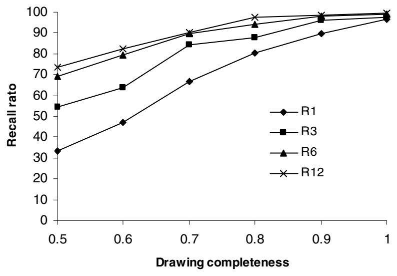
The time cost in the recognition process is also a much-concerned factor for evaluating our approach's performance. Especially, in our real-time interactive sketch recognition environment, the time cost should be as small as possible. The time cost comparison between our approach and Xu et al.'s approach is listed in Table 1. From this table, we can find that our algorithm is more efficient than Xu et al.'s algorithm. From all comparisons in our experiment, we find that our approach can save 56.3% time compared to Xu et al.'s algorithm in average and the recall ratio is also higher than Xu et al.'s approach.

Table 1. Comparison of time cost of our algorithm and Xu et al.'s algorithm for some shapes (Milliseconds).

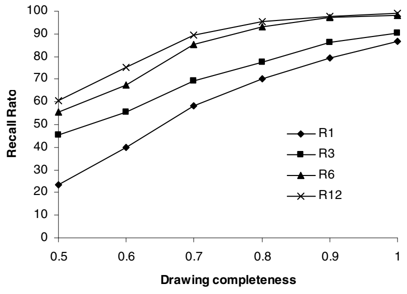
The number of components in the object	8	10	12	14
Time cost of Xu et al.'s algorithm	30	310	100	750
Time cost of our algorithm	10	50	70	400



(a) τ_n=0

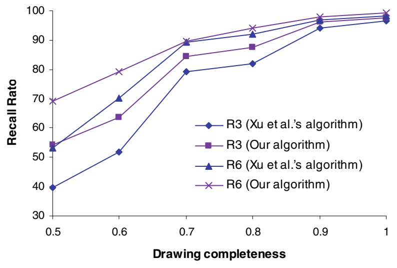


(b) τ_n=0.1

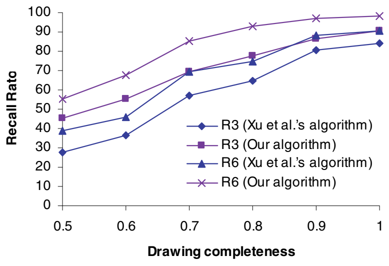


(c) τ_n=0.2

Fig. 5. Performance evaluation of the composite graphic object recognition.



(a) τ_n=0.1



(b) τ_n=0.2

Fig. 6. Comparison of recall ratio between our algorithm and Xu et al.'s algorithm [15].

5 Conclusion

In this paper, we have proposed two ternary spatial relationships among graphic components. The constrained partial permutation algorithm [15] has been improved to handle both binary and ternary topological spatial relationships for recognition of sketchy objects input by users in an online manner. From the experimental results, we can see that our improvement is more efficient and effective than Xu et al.'s method [15] and is practical in real-time sketch-based graphics input and recognition systems.

Acknowledgement

The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [Project No. CityU 1073/02E].

References

1. Tversky, B.: What do Sketches Say about Thinking? In: Proc. of AAAI Spring Symposium–Sketch Understanding. (2002) 148-151
2. Alvarado, C., Oltmans, M., Davis, R.: A Framework for Multi-Domain Sketch Recognition. In: Proc. of AAAI Spring Symposium–Sketch Understanding. (2002) 1-8
3. Skubic, M., Blisard, S., Carle, A., Matsakis, P.: Hand-Drawn Maps for Robot Navigation. In: Proc. of AAAI Spring Symposium–Sketch Understanding (2002) 140-147
4. Liu, W.: On-Line Graphics Recognition. To appear in Post-Proc. of GREC2003 (LNCS 2004)
5. Lee, S.W., Kim, Y.J.: A New Type of Recurrent Neural Network for Handwritten Character Recognition. In: Proc. of ICDAR95. Vol. 1 (1995) 38-42
6. Bicego, M., Murino, V.: 2D Shape Recognition by Hidden Markov Models. In: Proc. of 11th International Conference on Image Analysis and Processing. (1991) 20-25
7. Lladós, J., Martí, E., Josè, J.: Symbol Recognition by Error-Tolerant Subgraph Matching Between Region Adjacency Graphs. IEEE Trans. on PAMI. 23(10) (2001) 1137-1143
8. Xu, X., Liu, W., Jin, X., Sun Z.: Sketch-Based User Interface for Creative Tasks. In: Proc. of 5th Asia Pacific Conference on Computer Human Interaction. (2002) 560-570
9. Alvarado, C., Davis, R.: Resolving Ambiguities to Create a Natural Sketch Based Interface. In: Proc. of IJCAI-2001. (2001)
10. Egenhofer, M.J., Franzosa, R.: Point-Set Topological Spatial Relations. International Journal of Geographical Information Systems. 5(2) (1991) 161-174
11. Zhan, F.B.: Topological Relations Between Fuzzy Regions. In: Proc. of the 1997 ACM Symposium on Applied Computing. (1997) 192-196
12. Liu, W., Jin X., Sun, Z.: Sketch-Based User Interface for Inputting Graphic Objects on Small Screen Devices. Lecture Notes in Computer Science 2390 (2002) 67-80
13. Revankar, S., Yegnanarayana, B.: Machine Recognition and Correction of Freehand Geometric Line Sketches. In: Proc. of IEEE International Conference on Systems, Man, and Cybernetics. (1991) 87-92
14. Mehlhorn, K.: Graph Algorithm and NP-Completeness, Springer-Verlag. (1984)
15. Xu, X., Sun, Z., Peng, B., Liu, W.: An Online Composite Graphics Recognition Approach Based on Matching of Spatial Relation Graphs. To appear in International Journal on Document Analysis and Recognition.

A Study on the Consistency of Features for On-Line Signature Verification

Hansheng Lei and Venu Govindaraju

CUBS, Center for Unified Biometrics and Sensors
State University of New York at Buffalo, Amherst, NY 14260, USA
{hlei,govind}@cse.buffalo.edu

Abstract. A lot of different features have been proposed for on-line signature verification. By using these features, researchers implicitly believe they have high consistency as well as high discriminatory power. However, very little work has been done to measure the real consistency of these features. In this paper, we propose a model for consistency measure. Experiments were conducted to compare a comprehensive set of features commonly used for on-line signature verification.

1 Introduction and Motivation

Feature extraction and selection is the key for signature verification. A lot of work has been done on this [8, 9]. Different research groups use different features to discriminate genuine and forged signatures. The prerequisite for any feature is high *consistency*. That is, the feature from genuine signatures should be close to each other while the feature from forgeries should be far away. On-line signatures captured by digitizing device usually contains the information of the movement of pen (X-, Y-coordinates), pressure, altitude [8], etc. From the coordinate sequence, speed and acceleration can be derived. Among these information and potential features, which of them are reliable or consistent? Some researchers believe that the dynamic information such as the speed, acceleration or pressure are difficult to forge, thus they are able to distinguish skilled forgeries. However, the prerequisite is that the dynamic information from the authentic person should be consistent, otherwise false rejection will be incurred. Currently, we are lack of a solid support that the dynamic information are reliable or not. By using the extracted features, researchers implicitly believe they have high consistency as well as high discriminatory power. However, due to lack of consistency measure and benchmark databases, no experiment has been done to study the consistency of features proposed by a variety of research groups. Blind feature extraction can be avoided if there is a consistency measure model.

In this paper, we propose a consistency model and compare the consistencies of some commonly used features in on-line signature verification.

2 Consistency Measure

2.1 A Novel Consistency Model

Before we describe the consistency model, we need to know the basic requirements of on-line signature verification system.

Signature verification is a special two-category classification problem: true or false. We say it "special" because it is different from regular two-category problem. Fig. 1 shows the geometric difference. In the regular case, there are two clusters which can be discriminated by the decision boundary. The boundary is usually an *open* hyper-plane, as shown in fig. 1 a). However, in the case of signature verification, there is only one cluster, i.e., the set of genuine signatures, while forged signatures have no clustering characteristic because they have no reason to be close to each other. Therefore, the decision boundary must be a *closed* hyper-plane.

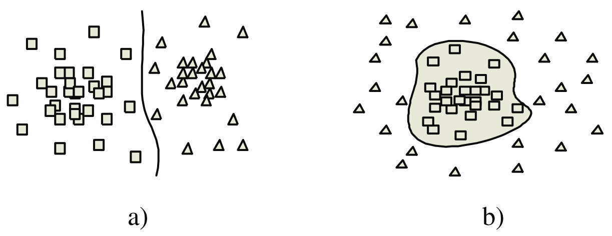


Fig. 1. Two different cases of two-category classification problem. a) Regular two-category problem has two clusterings with open decision boundary. b) Signature verification is a special two-category problem with closed decision boundary around the clustering genuine signatures.

Given few samples of genuine signatures(no more than 6 usually), it is very challenging to determine the decision boundary (threshold) no matter what kinds of features are used. Because of limited training samples, the consistency of feature becomes extremely important. There are many potential features to choose [7, 4, 1] and many new features are being invented. Thus, we are facing an increasing demand for a consistency model.

A simple consistency measure was defined by Lee etc [4] as:

$$d_i(a) = \frac{|m(a,i) - m(f,i)|}{\sqrt{\sigma^2(a,i) + \sigma^2(f,i)}}, \tag{1}$$

where $d_i(a)$ means the consistency of feature i for subject a, $m(a,i)$ is the mean of feature i from genuine signatures,$m(f,i)$ is the mean of feature i from corresponding forgeries and $\sigma^2(a,i)$ or $\sigma^2(f,i)$ means the variation of feature i from genuine or forged signatures.

While this model is appropriate for regular two-category classification, it faces severe problems for signature verification:

- $\sigma^2(f,i)$ does not make sense. As shown in fig. 2 b), signature verification is a special two-category classification problem where the forgeries have no reason to be close to each other.
- The mean $m(a,i)$ can not be calculated directly. For example, if we define the coordinate sequence itself (X, Y or $[X,Y]$) as feature, it is difficult to obtain the mean because the sequences are usually of different lengths. Even sequences can be re-sampled [1] to be of the same length, the mean of signature sequences actually does not mean anything.
- The calculation of distances between features are not limited to Euclidean norm. For example, if we define the sequence $[X,Y]$ as feature, we should use DTW (Dynamic Time Warping [3]) instead of Euclidean distance. Therefore, the calculation of $|m(a,i)-m(f,i)|$ and σ^2 should be generalized to incorporate different distance measures.

For above reasons, we need to tailor this model radically. Therefore, we define the consistency of feature i on subject a as:

$$Cons(a,i)=\frac{|M_{DM_i}(a,a)-M_{DM_i}(a,f)|}{\sqrt{\sigma^2_{DM_i}(a,a)+\sigma^2_{DM_i}(a,f)}}, \tag{2}$$

where the notions are described as follows:

DM_i: distance measure associated with feature i. We acknowledge that different features may have different distance measures, e.g., Euclidean norms, DTW, cross-correlation, etc.

$M_{DM_i}(C_1,C_2)$: the mean of the feature distances (by distance measure DM_i) between pairwise objects in class C_1 and class C_2. Formally,

$$M_{DM_i}(C_1,C_2)=\frac{1}{|C_1||C_2|}\sum_{c_1\in C_1,c_2\in C_2,c_1\neq c_2} DM_i(c_1,c_2), \tag{3}$$

where $DM_i(c_1,c_2)$ denotes the distance of feature i between object c_1 and c_2.

a: a set of genuine signatures. f: a set of corresponding forged signatures.

$\sigma^2_{DM_i}(a,a)$: The variation of the feature distances (by distance measure DM_i) within genuine signatures.

$\sigma^2_{DM_i}(a,f)$: The variation of the feature distances (by distance measure DM_i) between genuine and forged signatures.

2.2 Discussion

In the consistency model above, we calculate the mean of feature distances instead of the feature itself because some kinds of features have meaningless "mean" (for example, the coordinate sequence). Also, feature is usually associated with a certain kind of distance measure. Above model takes this into account. Therefore, the model is applicable for all kinds of features proposed for on-line signature verification.

Note that the consistency of feature i as $Cons(a,i)$ is for a particular subject a. The same feature could have different consistency value on different subjects.

Given a data set which consists of a set of subjects, we will calculate the mean and the standard deviation of the consistency value.

3 Commonly Used Features

We subjectively choose over 20 features and compare their consistencies on the SVC signature database[6]. We briefly discuss these features here. Details are referred to corresponding papers.

- *Coordinate sequences.* X, Y, $[X, Y]$ are the most straightforward features. The lengths of these features from different signatures are usually different. Therefore, DTW is used as distance measure [3]. There exist some variants of sequence here. Some researchers propose that the sequence be re-sampled so that they have equal arc-length [1]. We will compare their consistency with/without re-sampling.
- *Speed sequences.* Speed V, speed of X coordinate V_x and speed of Y coordinate V_y can be derived from sequence $[X, Y]$ directly by subtracting neighboring points. From the speed, acceleration V_a can be further derived.
- *Pressure, altitude, azimuth.* Pressure is one of the most common dynamic information of on-line signature. Some devices can capture additional information, such as azimuth (the clockwise rotation of cursor about the z-axis) and altitude(the angle upward toward the positive z-axis)[6].
- *Center of Mass $\overline{x}(l)$ and $\overline{y}(l)$, Torque $\overline{T}(l)$, Curvature-ellipse $s_1(l)$ and $s_2(l)$.* The five features were defined in [1]. Center of Mass is actually the smoothed coordinate sequence by Gaussian filter. Torque measures the area swept by the vector of pen position. $s_1(l)$ and $s_2(l)$ measure the curvature ellipse based on moments. The distance measure used here is cross-correlation (Pearson's r) weighted by the consistency of points.
- *Average speed $\overline{V}$, average positive speed on X-axis $\overline{V}_{x+}$, average positive speed on Y-axis $\overline{V}_{y+}$, total signing duration T_s.* Lee etc[4] lists two sets of features. These four features have the highest preference in the first set. The distance measure is Euclidean norm.
- $\cos(\alpha)$*,* $\sin(\alpha)$*, Curvature β.* α is the angle between the speed vector and the X-axis. The three features are proposed by Jain etc[7]. It also proposes coordinate sequence difference δx and δx. Actually, δx and δx are the same as feature # 5 and # 6 in table 1 respectively.

All above features and corresponding distance measures are summarized in table 1. We have to notice that these features are only a small portions of the proposed features for signature. We choose them because we believe they are among the most promising ones according to our experience.

4 Comparison of Consistency

We used the released SVC database [6] to calculate the consistencies of the features in table 1. SVC has two sets of signatures, namely task 1 and task 2. Each

Table 1. Commonly used features.

#	Feature	Dist. Measure
1	X-coordinate: X	DTW
2	Y-coordinate: Y	DTW
3	Coordinates: $[X, Y]$	DTW
4	Speed: V	DTW
5	Speed X: V_x	DTW
6	Speed Y: V_y	DTW
7	Pressure: P	DTW
8	Acceleration: V_a	DTW
9	Altitude: A_l	DTW
10	Azimuth: Z_u	DTW
11	Center of Mass X: $\overline{x}(l)$	Weighted r
12	Center of Mass Y: $\overline{y}(l)$	Weighted r
13	Torque: $T(l)$	Weighted r
14	Curvature-ellipse: $s.\,(l)$	Weighted r
15	Curvature-ellipse: $s.\,(l)$	Weighted r
16	Average speed: $\overline{V}$	Euclidean
17	Average positive V_x: $\overline{V}_{x+}$	Euclidean
18	Average positive V_y: $\overline{V}_{y+}$	Euclidean
19	Total signing time: Ts	Euclidean
20	Curvature: β	DTW
21	Angle: $\sin(\alpha)$	DTW
22	Angle: $\cos(\alpha)$	DTW

signature is represented as a sequence of point, which contains X *coordinate*, Y *coordinate*, *time stamp* and *pen status (pen-up or pen-down)*. In task 2, additional information like *azimuth*,*altitude* and *pressure* are available. There are 40 subjects in each task with 20 genuine signatures and 20 forgeries for each subject.

To compare the consistency of different features fairly, we need to normalize the raw signatures as well the feature distances. We normalized each signature by the same preprocessing: 1) smooth the raw sequence by Gaussian filter; 2) rotate if necessary [2]; 3) normalize the coordinate of each signature Sig_i by:

$$X_i = \frac{X_i - min(X_i)}{max(X_i) - min(X_i)}, Y_i = \frac{Y_i - min(Y_i)}{max(Y_i) - min(Y_i)} \tag{4}$$

The feature distance must be normalized because we have different distance measures here, such as DTW, Euclidean norm and weighted cross-correlation. Normalization of distances is done as follows. Suppose we apply DM_i to class a (genuine signatures) and class f (forgeries). We first calculate all pairwise feature distances within class a by DM_i. We find the maximum feature distance (denoted as D_{max}). Then, we calculate all pairwise feature distances between class a and f. For each *dist* among these distances, no matter within a or between

a and f, we normalize it by $e^{\frac{-dist}{2*D_{max}}}$. In this way, all distances are mapped to a value between 0 and 1. The larger the distance, the closer the value to 1.

Same feature may have different consistency on different subjects. Thus, we calculated the mean of consistency of each feature as well as its standard deviation cross subjects.

In addition, to show the relation between feature consistency and its discriminatory power in verification, we calculated the EER (Equal Error Rate) for each feature. We randomly chose 5 genuine signatures from each subject and used the left 35 signatures for verification. Given a testing signature, we calculated the feature distance with all the 5 genuine signatures one by one and returned the maximum normalized distance as similarity output (except the weighted cross-correlation distance, where we output the minimum normalized distance). To determine the EER, we varied the threshold from 0% to 100% and found the point where the FRR (False Rejection Rate) equals the FAR (False Acceptance Rate). By *universal* threshold, we chose the same threshold for all subjects to calculate the total error rate. By *user-dependent* threshold, we chose the optimal threshold for each subject and took the average error rate.

The results are summarized in table 2 in the increasing order of mean consistency value.

From the results as shown in table 2, we have the following observations:

– Although *azimuth* and *altitude* have relatively high mean consistency, they have high standard deviations, which means their discriminatory ability are not stable cross subjects. The corresponding high EER confirms this.
– Some features, like curvature-ellipse $s_1(l)$ and $s_2(l)$, torques $T(l)$, center of mass $\overline{x}(l)$ and $\overline{y}(l)$ are not good enough for skilled forgeries, although they are sophisticated and might be enough for random forgeries. This means, complex features are not necessary better than simpler ones.
– Features like # 17 through # 19 are too simple to carry enough discriminatory information. Thus, they have high EERs. They could be used to prune random forgeries but non-reliable for accepting genuine ones.
– EER has negative relation with the level of consistency, although the EERs do not strictly increase with the consistency decreasing. The variation of consistency also acts to affect the verification performance. This confirms the consistency of feature is directly related to its performance on verification.

Here we have to emphasize that all the EERs in table 2 are high because we used only one feature each time for verification. How to combine these features optimally is still an open problem. We also have to mention that further experiments on real and larger signature databases are necessary to claim the consistency of any given feature.

There is a belief in on-signature verification community that the curve of the signature should be re-sampled with uniform equal arc-length [7, 1, 2]. Is this necessarily true? We conducted experiments to answer this question. We re-sampled all the signature to be of length N and calculated the consistencies of features and corresponding EERs. The results are summarized in table 3 with $N = 200$ (we varied N and found the results had no much difference). Note that

Table 2. Consistencies of features (mean and standard deviation) and EERs (by universal threshold and user-dependent threshold).

Feature	Consistency		EER	
	Mean	Std.	Univ. T.	User. T.
Z_u	1.3789	3.2333	35.06%	26.63%
V_y	1.3255	0.3231	22.06%	11.38%
V_x	1.2644	0.3659	22.65%	16.81%
V	1.2374	0.4143	23.44%	17.63%
T_s	1.2025	0.8128	30.31%	28.18%
A_l	1.1829	4.578	37.63%	29.06%
$\cos(a)$	1.1199	0.2845	26.72%	16.19%
$[X, Y]$	1.1061	0.1795	22.91%	16.83%
$\sin(a)$	1.0997	0.2491	29.56%	20.63%
V_a	1.0966	0.3199	29.29%	22.50%
P	1.0647	0.375	36.86%	25.56%
β	0.985	0.1767	28.90%	20.81%
Y	0.9252	0.1921	25.86%	18.68%
X	0.7784	0.1313	29.59%	25.13%
$\overline{y}(l)$	0.637	0.1397	28.81%	19.19%
$\overline{x}(l)$	0.6023	0.122	29.59%	23.00%
$T(l)$	0.5291	0.1504	33.63%	25.68%
$\overline{V}$	0.4814	0.1761	34.50%	31.94%
$\overline{V}_{y+}$	0.4587	0.1662	36.69%	33.88%
$\overline{V}_{x+}$	0.3983	0.1717	36.69%	34.00%
$s_{\bullet}(l)$	0.3158	0.0295	43.86%	42.19%
$s_{\bullet}(l)$	0.3158	0.0292	43.96%	42.13%

Table 3. Consistencies of features and EER with uniform arc-length re-sampling.

Feature	Consistency		EER	
	Mean	Std.	Univ. T.	User. T.
V_y	0.8841	0.2425	32.36%	20.81%
V_x	0.8714	0.3025	30.44%	21.00%
V	1.1452	0.3066	27.94%	17.63%
$\cos(a)$	1.082	0.2352	29.41%	20.25%
$[X, Y]$	1.0546	0.5839	26.63%	20.94%
$\sin(a)$	0.8093	0.2661	32.56%	23.69%
β	0.8941	0.2360	32.19%	24.75%
Y	0.8683	0.2667	29.94%	21.56%
X	0.7893	0.4407	34.03%	29.63 %

some features in table 1 have no sense or difference with the signatures being re-sampled. Compared the consistencies and EERs with/without re-sampling, we can see that re-sampling does not necessarily improve performance. On the contrary, the performance is damaged to some degree.

5 Conclusion Remarks

A novel consistency model tailored for on-line signature verification is proposed in this paper. The consistency of feature is directly related to the feature's performance on signature verification. We conducted experiments to calculate the consistencies of a set of features. The results summarized in table 2 show that some features such as *speed*, *coordinate sequence*, angle α have relatively high consistency, while some others like *azimuth*, *altitude*, curvature-ellipse $s_1(l)$ and $s_1(l)$ are non-reliable. Also, we found that the re-sampling with uniform arc-length does not necessarily increase performance.

References

1. V. Nalwa. *Automatic on-line signature verification.* Proceedings of the IEEE, 85(2): pp. 213-239, Feb. 1997.
2. M. Munich, P. Perona. *Visual identification by signature tracking.* IEEE Trans. on Pattern Analysis and Machine Intelligence, 2003.
3. R. Martens, L. Claesen. *On-line signature verification by dynamic time-warping.* In Proc. 13th Int. Conf. Pattern Recognition, pp. 38-42, 1996.
4. L. Lee, T. Berger, E. Aviczer. *Reliable On-Line Human Signature Verification Systems.* IEEE Trans. on Pattern Analysis and Machine Intelligence , pp. 643-647, 1996.
5. T. Rhee, S. Cho, J. Kim. *On-Line Signature Verification Using Model-Guided Segmentation and Discriminative Feature Selection for Skilled Forgeries.* In the 6th Int. Conf. on Document Analysis and Recognition (ICDAR), 2001.
6. SVC. *The First International Signature Verification Competition. http://www.cs.ust.hk/svc2004/*, 2004
7. A.Jain, F. Griess, S. Connell. *On-line Signature Verification.* Pattern Recognition, 35(12): 2963–2972, 2002.
8. F. Leclerc, R. Plamondon. *Automatic signature verification: the state of the art 1989-1993.* International Journal of Pattern Recognition and Artificial Intelligence, 8(3): pp. 643-660, 1994.
9. R. Plamondon, F. Leclerc. *Automatic signature verification and writer identification the state of the art.* Pattern Recognition, 22(2): pp. 107-131, 1989.

Statistical Structure of Natural 4×4 Image Patches

Kostadin Koroutchev and José R. Dorronsoro*

Depto. de Ingeniería Informática and Instituto de Ingeniería del Conocimiento
Universidad Autónoma de Madrid, 28049 Madrid, Spain

Abstract. The direct computation of natural image block statistics is unfeasible due to the huge domain space. In this paper we shall propose a procedure to collect block statistics on compressed versions of natural 4×4 patches. If the reconstructed blocks are close enough to the original ones, these statistics can clearly be quite representative of the true natural patch statistics. We shall work with a fractal image compression–inspired codebook scheme, in which we will compute for each block B its contrast σ, brightness μ and a normalized codebook approximation D^B of $(B - \mu)/\sigma$. Entropy and mutual information estimates suggest a first order approximation $p(B) \simeq p(D^B)p(\mu)p(\sigma)$ of the probabibility $p(B)$ of a given natural block, while a more precise approximation can be written as $p(B) \simeq p(D^B)p(\mu)p(\sigma)\Phi(||\nabla B||)$. We shall also study the structure of $p(\sigma)$ and $p(D)$, the more relevant probability components. The first one presents an exponential behavior for non flat patches, while $p(D)$ behaves uniformly with respecto to volume in patch space.

1 Introduction

Natural images, that is, those derived from natural scenes, have a distinctive nature that makes them far from random. In particular, they convey "information" that allows their processing by the human visual system. In fact, natural image information is not distributed uniformly over the image: there are parts that are most relevant to the human visual system, while other are far less relevant. It is therefore clear that the undestanding of their statistical behavior it is extremely important, not only for basic human visual processing but also for a number of everyday visual information processing tasks such as for instance efficient static and dynamic image compression. A large effort has been undertaken in that direction [7, 2, 9]; a thorough and recent survey is in [8]. In any case, it can be quite easily seen that direct statitistics computation for 4×4 natural image blocks is not currently possible. In fact, state of the art lossless image compression (see e.g. [10]) can achieve for 8 bit gray level images compression rates down to 2.5 bits per pixel. Thus, a block of size 4×4 would requires in average about $16 \times 2.5 = 40$ bits. Assuming that this representation is close to the informational limit of the lossless representation of the image, it follows that

* With partial support of Spain's CICyT, TIC 01–572.

Fig. 1. Statistics computed from other codebooks are similar, provided the source image is "rich" enough, as exemplified here.

natural block statistics require at least 2^{40} 4×4 blocks or, in other words, about $2^{40-16} \simeq 16 \times 10^6$ natural 1024×1024 images. Direct statistics computation is clearly not possible today because of, among other things, the lack of so many machine readable raw images.

We shall work here with the equivalent representation $(\tilde{B}, \sigma, \mu)$ of a natural block B, with σ, μ the standard deviation and mean of B and $\tilde{B}$ its normalization $\tilde{B} = (B - \mu)/\sigma$. Direct statistics are clearly possible for the 1–dimensional σ and μ, while this is not so for $\tilde{B}$. To estimate them we shall approximate natural patches $\tilde{B}$ by normalized blocks D^B extracted from a given codebook. Since an approximation $\tilde{B} \simeq D^B$ immediately translates into the affine approximation $B \simeq \sigma D^B + \mu$, it is natural to try to derive D through fractal image compression (FIC) techniques [1]. If a good reconstruction quality is obtained, the (D_B, σ, μ) statistics should provide meaningful approximations to those of $(\tilde{B}, \sigma, \mu)$. This will be done in section 2, where we shall introduce a concrete FIC codebook, namely, 4×4 domains extracted from the well known Lena image, and shall use it to approximate about 280 million natural 4×4 patches extracted from the well known Van Hateren database [2]. Although the concrete codebook used certainly influences the resulting statistics, we have obtained similar results using codebooks derived from other images, provided they are "rich" enough. This is the case, for instance, of figure 1, derived from the Van Hateren database. That section also describes the approximating technique used to collect the raw frequency data for the approximations $B \simeq (D^B, \sigma, \mu)$, which are analyzed in section 3. The main results of that section are, first, several entropy and mutual information estimates for the joint (D^B, σ, μ) distribution and the marginals of D^B, σ and μ. These estimates suggest as a first approximation that the marginals may be taken to be independent, that is, that $p(D^B, \sigma, \mu) \simeq p(D^B)p(\sigma)p(\mu)$, decomposition that still leaves about 1.5 bits of mutual information between D^B and (σ, μ) to be explained. To do so we shall refine the previous first order approximation to a second one of the form $p(D^B, \sigma, \mu) \simeq p(D^B)p(\sigma)p(\mu)\Phi(||\nabla B||)$. The structure of the $p(D^B), p(\sigma)$ and $p(\mu)$ marginals is dealt with in section 3. While $p(\mu)$ does not carry significant information, we shall see that $p(\sigma)$ has an exponential structure and that $p(D^B)$ follows a nearly uniform behavior with respect to volume in image space. A final section contains a summary of the paper and pointers to further work.

2 Methods

As the natural patch source, we shall work with 4300 8 bit gray level images of size 1540×1024 from the Van Hateren database. We shall restrict ourselves to their 1024×1024 squared centers and take out flat blocks, that is, those B with $\sigma \leq 3$ (about 20% of all patches). As mentioned above, we shall approximate a normalized natural patch $\tilde{B} = \frac{B-\mu}{\sigma}$ by another normalized domain D^B taken from a codebook derived from a 256×256 version of the well known Lena image as follows: we will first extract all its 4×4 (overlapping) blocks. This gives $(256 - 4 + 1)^2 \simeq 2^{16}$ codebook domains, that become 2^{20} after adding for each block its 8 isometries and its negative (recall that we are reconstructing B using positive σ_B contrast factors). Again, we will exclude flat domains, about 25% of the initial Lena domains.

Thus, we will not estimate the direct distribution $p(B)$ but instead that of the B approximation $p(D^B, \sigma_B, \mu_B)$. To minimize the distortion that this approximation is bound to introduce, we shall take D^B as the codebook domain for which

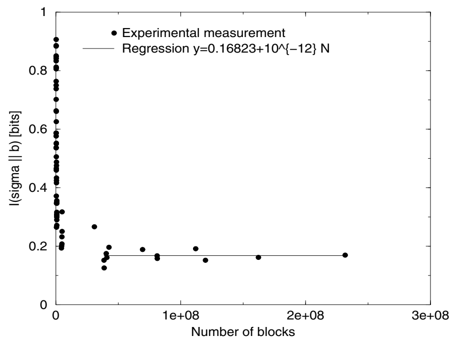
$$dist(B, D) = ||B - \sigma_B D - \mu_B||_\infty = \sup |B_{ij} - \sigma_B D_{ij} - \mu_B| \tag{1}$$

verifies $dist(B, D) \leq d_M$, taking in what follows $d_M = 8$. Reconstructing a full image $I = \{B_s\}$ by its patches' approximations $\hat{I} = \{D_s^B\}$, our choice of d_M should ensure that $||I - \hat{I}||_2 \leq 8$ and the PSNR of the reconstruction $\hat{I}$ verifies $PSNR(\hat{I}) = 20 \log_{10}(\frac{255}{||I-\hat{I}||_2}) \geq 20 \log_{10}(\frac{255}{8}) \simeq 30$. We shall discard those B for which a matching domain cannot be found. They are about a 1 per 1000 of all non flat domains, which results in a final number of about $232 \times 10^6 \simeq 2^{27.79}$ disjoint 4×4 patches.

Finding matching domains requires to perform at some point the costly full block comparisons in (1), that can make FIC very time consuming. To speed things up, we shall precede full block comparisons with a hash–like pre–comparison. We define a hash function (see [3] for further details)

$$h(D) = \sum_{h=1}^{H} \left(\left\lfloor \frac{D_{i_h j_h}}{\lambda} \right\rfloor \% C + \frac{C}{2} \right) C^{h-1} = \sum_{h=1}^{H} b_h C^{h-1}. \tag{2}$$

with $D_{i_h j_h}$, $1 \leq h \leq H$, adequately chosen points in D, and H, C and λ appropriately chosen parameters. In what follows we shall take the four corner pixels and an extra middle pixel and thus $H = 5$; as the base C we shall take $C = 16$.



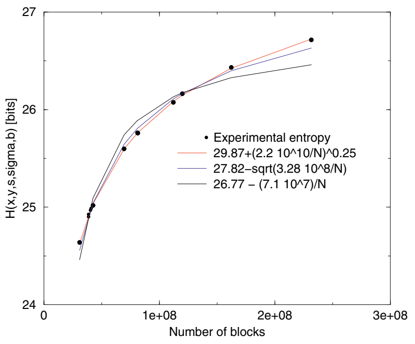


Fig. 2. Large sample behavior of the mutual information $I(\sigma||\mu)$ (left) and of the total entropy $H(i,j,s,\sigma,\mu)$ (right) estimates. While the left picture saturates, this is not the case for the $H(i,j,s,\sigma,\mu)$ estimate.

The choice of λ should ensure that $D_{i_h j_h}$ is distributed more or less uniformly on the interval $[-\lambda\frac{C}{2}, \lambda\frac{C}{2}]$, and, thus, that (2) defines a uniform base C expansion. A good choice for this would be to take λ in the interval 0.5–1, but to streamline our subsequent discussions, we shall take $\lambda = 2$. Once the values $h(D)$ have been computed for all codebook domains, those with the same value are stored in the same linked lists over a hash pointer table. Full block comparisons for a natural block B are performed only over the domains in the linked lists whose h index is contained in the B dependent set $H(B) = \{h_\delta(B)\}$, where

$$h_\delta(B) = \sum_{h=1}^{H} \left(\left\lfloor \frac{B_{i_h j_h} - \mu_B}{\lambda \sigma_B} + \delta_h \right\rfloor \%C + \frac{C}{2} \right) C^{h-1} = \sum_{h=1}^{H} r_h^\delta C^{h-1}, \qquad (3)$$

with the displacement vector $\delta = (\delta_1, \ldots, \delta_H)^t$ verifying $|\delta_h| \leq 1$. It is not difficult to show that this searching procedure will provide the optimal matching domain D^B. Our coding of a block B will then be

$$T(B) = (i, j, s, \sigma, \mu)$$

where (i,j) indicates the position in the Lena image of the left upper corner of the matching domain, and s is an index for the isometry and negative used (notice that the dilations in (1) are positive).

In order to obtain the statistics described in the next section, we shall compute first basic frequency value sets P_i over 9 image batches. All of them contain registers of the form $[i, j, s, \sigma, \mu, c]$, with c counting the number of patches in the file with σ, μ statistics and a matching i, j, s domain; they are sorted in lexicographic order. We then perform a first merge over 9 of these P_i sets to arrive to larger value sets Q_j and then a second merge over 9 of the Q_j sets to arrive to 6 large files S_k of statistical values, 5 of them corresponding to $729 = 9^3$ image batches and a smaller sixth one for the last 650 images. A final merge of these 6 files will give the statistics described in the next section.

Table 1. Different entropy measures (in bits) of image statistics (first column) and limit estimates for them.

Quantity	Value	Limit estimates
N	231511046	—
log. N	27.7865	
$H(i, j, s, \sigma, \mu)$	26.7141	29.87
$H(i, j, s)$	17.8156	17.82
$H(\sigma, \mu)$	10.4627	10.46
$I(i, j, s \| \| \sigma, \mu)$	1.5642	0.474
$I(\sigma \| \| \mu)$	0.1698	0.115
$I/H(i, j, s, \sigma, \mu)$	5.86 %	1.70 %
$I(\sigma, \mu)/H(\sigma, \mu)$	1.62 %	1.10 %

3 Entropy and Mutual Information Estimates

It is well known that the accuracy of any discrete probability entropy estimate depends on its sample regime, that is, the relationship between the sample number N and the number M of non–empty bins, i.e., of non–zero probabilities. In our case two very different regimes are to be considered. In the first one, we shall have $N \gg M$; this is the case for the joint (σ, μ) distribution and to a smaller extent for the (i, j, s) distribution. In this regime [6] we should expect for large N that the entropy and mutual information estimates closely approach saturation limit points that we can take as the true entropy and mutual information values. Figure 2, left, depicts this situation for the mutual information $I(\sigma||\mu)$. It has been computed for the full sample size range and shows a extremely fast drop for sample sizes below 10^6 followed by a nearly horizontal behavior afterwards. This can be interpreted as showing the sample information $\hat{I}_N$ estimates to saturate at a limiting value $I = 0.17$ which we can take as the actual value of $I(\sigma||\mu)$. On the other hand, the right picture shows a quite different situation for the joint entropy $H(i, j, s, \sigma, \mu)$. Clearly a saturation point has not been achieved, which shows that we are still far from an $N \gg M$ regime. Notice that while we have $\log N \simeq 28$ for the full database sample, the full sample estimate for $\hat{H}_N(i, j, s, \sigma, \mu)$ is 26.7, which is a lower bound for $\log M$. In other words $\log N/M \leq 1.3$, that is, we are in the $N \simeq M$ regime. A similar situation holds for $I(i, j, s||\sigma, \mu)$.

Table 3 collects these empirical entropy and mutual information estimates and also some estimates of possible limit values derived from functional approximations to the empirical data. In most cases they are close to the empirical estimates, while this is not the case for $H(i, j, s, \sigma, \mu)$ and $I(i, j, s||\sigma, \mu)$. We shall take these values as a starting point for our discussion. It can be seen from this table that the mutual information between the σ and μ is very small, about 0.17 bits, less than 2% of the joint entropy; therefore we can assume that σ and μ are independent. On the other hand, the mutual information $I(i, j, s||\sigma, \mu)$ is about 1.56, less than 10% of the joint entropy. Therefore, this suggests the first order approximation

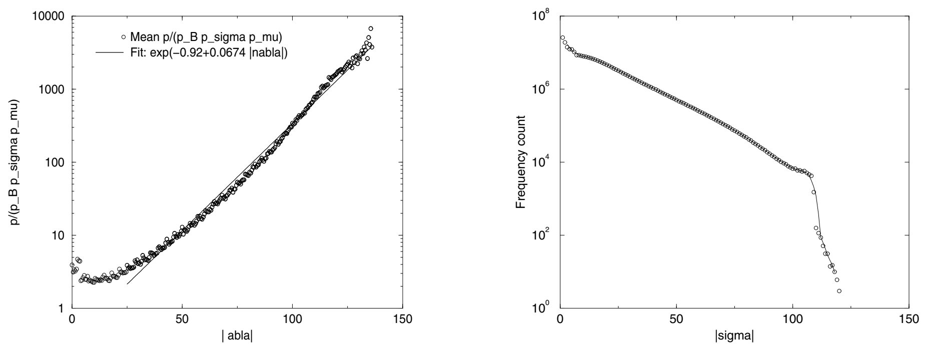


Fig. 3. Left: conditional expectation $\Phi(||\nabla B||)$ in (6) as a function of $||\nabla||$. For lower values of $||\nabla||$, that are the predominant in the distribution, no correction is possible. For larger $||\nabla||$ the approximation results to be approximately exponential. Right: relative frequencies of σ in a logarithmic scale. The linear central behavior suggests an exponential distribution.

$$p(B) = p(\tilde{B}, \sigma_B, \mu_B) \simeq p(i_B, j_B, s_B, \sigma_B, \mu_B) \simeq p(i_B, j_B, s_B)p(\sigma_B)p(\mu_B), \quad (4)$$

which nonetheless leaves more than one mutual information bit to be explained.

To do so, we shall study what we may call the divergence, that is, the average

$$d(i,j) = E_{s,\sigma,\mu}\left[\frac{p(i,j,s,\sigma,\mu)}{p(i,j,s)p(\sigma)p(\mu)},\right]$$

over the joint and (i,j,s) and σ and μ distributions. Values of d different from 1 indicate deviations from the independence assumption and projecting them back over the Lena image suggests that image borders are the main source of the divergence. Thus any correction to (4) should be significantly different from 1 over edge blocks. Other natural assumptions are that it be isotropic, translation invariant and not dependent on the block's absolute brightness, but only on the difference $B - \mu$. Moreover it is natural to look for a lowest order approximation. All this suggests to refine (4) to

$$p(B) \simeq p(D_B)p(\mu_B)p(\sigma_B)\Phi(||\nabla B||). \quad (5)$$

In order to find a reasonable Φ we have compared the conditional average

$$\Phi(||\nabla B||) = E_{||\nabla B||}\left[\log_2 \frac{p(i,j,s,\sigma,\mu)}{(p(i,j,s)p(\sigma)p(\mu)}\right] \quad (6)$$

with $E_{||\nabla B||}$ denoting the conditional expectation with respect to $||\nabla B||$. This is depicted in figure 3, left, which shows that while for the lower $||\nabla B||$ patches, the most predominant ones in natural images, no correction appears to be possible, $\Phi(||\nabla B||)$ can be quite well linearly approximated for the right side high $||\nabla B||$ values. In turn, this suggests that a natural approximation for $\Phi(s)$ is

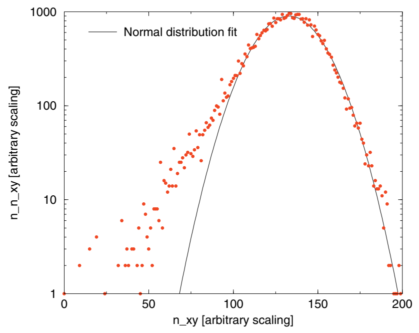
$$\begin{aligned}\Phi(s) &= e^{-0.923+0.0337s}, \quad s \geq 25\\ &= 0, \quad 0 < s < 25\end{aligned}$$

with the constant 0.923 being a normalization value so that the second order correction is actually a probability. A natural interpretation of the positive constant 0.0337 in Φ is to see it as a high–contrast patch distribution correction, a high information natural image component, as shown for instance in [4]. To check the impact of this second correction, we have re–computed now the mutual information between the experimentally obtained distribution and the above second order corrected distribution, which turns out to be 0.621 bits, that is, about 1 bit less than the previous estimate. Therefore, just 2.3% of the total information is not covered now by the second approximation. In any case, further alternatives for the correction have to be studied. For instance, there are indications that better edge detectors than simple gradient approximations could give better results.

4 Structure of the $p(i, j, s)$, $p(\sigma)$ and $p(\mu)$ Probabilities

The μ distribution depends on the camera's calibration and can be easily manipulated through, say, histogram equalization. It is thus largely irrelevant. The histogram of the σ distribution is depicted in figure 3, right, in vertical logarithmic scale. The figure shows a central linear behavior, suggesting an exponential distribution, with changes at the boundaries. The drop at around 100 is due to the limited range of brightness levels, with a theoretical maximum below 128 for 256 gray levels. Although seemingly hinting at some underlying structure, the cusp–like peak at 0 is mostly due to the layered structure of natural images, that produces many near flat, small deviation blocks. Anyway, the σ distribution makes clear that σ carries significant structural information. One way to visualize this is to project for each domain D with coordinates i, j in the Lena image the corresponding value of $-\log p(\sigma)$. When done, it shows a clear correlation between edges and large $-\log p(\sigma)$ values that makes clearer the significace of the σ component.

It is more complicated to visualize the structure of $p(i, j, s)$. A possibility is to fix our attention in the (i, j) distribution, and to consider the corresponding Lena domains as bins were the sample patches fall. Defining $N(m)$ as the number of such bins getting m sample patches, it can be seen that $\log N(m)$ shows a near parabolic structure, which is still more clear in figure 4, left, where $N(m)$ has been corrected taking into account the volume surrounding each codebook domain. To perform this correction, we may a priori assume that the hash linked lists cover regions with essentially the same volume, and also that all domains of a given list have the same volume. This implies that the a priori probability of a domain D is thus proportional to $\nu(h(D))$, with $\nu(h)$ the number of codebook domains D' such that $h(D') = h$. We then correct the direct counting estimate m' of the number of patches a certain domain D gets to $m = m' \times \nu(h(D))$. The corrected $N(m)$ values are depicted in figure 4, left. As it can be seen, the parabolic fitting is quite good for the higher patch count right part, which suggests that there is a volume uniform distribution of natural patches between codebook domains, that is, that the probability of a codebook region $\mathcal{R}$ receiving a patch is proportional to its volume $V(\mathcal{R})$.



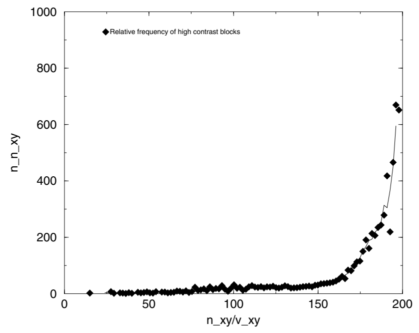


Fig. 4. Volume–corrected (left) values of $\log N(m)$, with $N(m)$ the relative number of domains gettin m natural blocks, that suggest a volume–uniform distribution of all normalized natural patches among codebook domains. This is not true, however, for high contrast patches: the right image shows a markedly higher proportion for them among high count domains.

We have also computed the distribution over m of the patches' average σ values, which shows that the left area corresponds mostly to low σ patches, with small denominators in (3) and, hence, larger hash values, that may cause them to be assigned to wrong matching domains. This would have a little effect in domains with a large block count, as they would lose about as many blocks as they gain, but a bigger one in domains getting fewer blocks. In any case, the natural interpretation of figure 4 is to assume a volume–uniform distribution for the (i, j) domain statistics, as the near gaussian behavior of $N(m)$ is easiest explained as a large sample aproximation of a binomial distribution. Apparently this may contradict recent results in [4], that show a marked structure of high contrast natural 3×3 blocks. However, notice that figure 4, right, depicting the proportion of the high-contrast codebook domains getting a (normalized) number m of patches, has a very sharp rise at the high patch count area. In other words, the high contrast blocks studied in [4] have an statistical behavior of their own, certainly not following the uniform behavior just described.

5 Summary and Future Directions

In this article we have shown how a representation $B \simeq (D_B, \sigma, \mu)$ of natural image 4×4 blocks B, with σ, μ the block's variance an mean and D_B a codebook approximation to the normalization of B, can be used to obtain significant natural image statistics. In fact, we have shown that these blocks' probabilities can be represented as a product of nearly independent factors. The analysis of these factors allows us to conclude that at least in the scale investigated (about one minute of angle), the information is essentially carried by a block's variance, that roughly correlates with the block's edges. This is in accordance with the well known Marr [5] hypothesis that natural image information is extracted from biological systems using its most singular points, e.g. its edges. However, it may be of some interest to point out that, here, this conclusion is drawn without any regard to the receiving system, but just by using information theoretical considerations; this may suggest that biological systems have adapted themselves to extract the part of a natural image most relevant in terms of information theory. On the other hand, the above results may have applications in, for instance, fast search engines in image databases, or in transmission of moderate quality images over low bit rate, noisy channels. All these matters are currently being researched.

References

1. Y. Fisher (ed.), Fractal Image Compression: Theory and Application, Springer Verlag, New York, 1995.
2. J.H. van Hateren and A. van der Schaaf, *Independent component filters of natural images compared with simple cells in primary visual cortex.* Proc.R.Soc.Lond. B, 265:359-366, 1998.
3. K. Koroutchev and J. Dorronsoro, *Hash–like Fractal Image Compression with Linear Execution Time*, Iberian Conference on Pattern Recognition and Image Analysis, IbPRIA 2003, Lecture Notes in Computer Science.
4. A.B. Lee, K.S. Pedersen and D. Mumford. *The Complex Statistics of High-Contrast Patches in Natural Images.* In WWW Proceedings of Second International IEEE Workshop on Statistical and Computational Theories of Vision. Vancouver, Canada, July 2001.
5. D. Marr, *Vision,* W.H. Freeman and Co., 1982.
6. L. Paninski, *Estimation of Entropy and Mutual Information*, Neural Computation, 15 (2003) 1191–1253.
7. D.L. Ruderman, *The statistics of natural images*, Network: Computation in Neural Systems , vol. 5, 517-548, 1994.
8. A. Srivastava, A.B. Lee, E.P. Simoncelli and S.C. Zhu, *On Advances in Statistical Modeling of Natural Images*, Journal of Mathematical Imaging and Vision 18(1), 17-33, 2003.
9. A Turiel, N Parga, D Ruderman and T Cronin, *Multiscaling and information content of natural color images,* Phys. Rev. E 62 , 1138-1148 (2000)
10. M. Weinberger, G. Seroussi and G. Sapiro, *LOCO-I: A Low Complexity, Context-Based, Lossless Image Compression Algorithm,* Proc. of the IEEE Data Compression Conference, Snowbird, Utah, March-April 1996.

Grey Scale Skeletonisation with Curvature Sensitive Noise Damping

Huaijun Qiu and Edwin R. Hancock

Department of Computer Science, University of York, York, YO10 5DD, UK
{jun,erh}@cs.york.ac.uk

Abstract. This paper describes a method for curvature dependent skeletonisation in grey-scale images. We commence from a magnetostatic analogy, where the tangential edge flow is intepretted as a current. A vector potential is constructed by integrating the current weighted by inverse distance over the image plane. The skeleton corresponds to the location of valley lines in the vector potential. To damp noise effects we damp the current with an exponential function of the local curvature. In addition, we describe a number of postprocessing steps that can be used to improve the quality of the detected skeletons. In the end, we compare the effects of two alternative ways for noise damping.

1 Introduction

Skeletal abstractions have been used to great effect in the representation and recognition of both 2D and 3D shapes. Some of the earliest work was performed by Blum [2], who showed how the skeleton could be used for the morphological analysis of biological forms. Most of the existing work of skeletonisation has focused on binary valued objects [1, 7–9, 12]. Here a number of methods have been investigated including the medial axis transform [1], the chordal axis transform [7], and the grassfire transform [2], More recent work has focused on the analysis of the skeleton as the singularities in the eikonal equation for inward boundary motion [8]. An analysis of this system using the Hamilton-Jacobi equations of classical mechanics has shown how the skeleton can be detected using the divergence of the distance map for the object boundary [9]. Recently, Torsello and Hancock [12] have shown how the Hamilton-Jacobi skeleton can be improved by modifying the eikonal equation to take into account curvature effects.

In addition there has been a limited effort directed at the analysis of grey-scale objects. For instance, Tari, Shah and Pien [10] have proposed a linear diffusion equation to smooth out noise and extract skeletons directly from the grey scale images. Tek et al [11] have shown that an orientation sensitive distance propagation function can be used to extract symmetries from fragmented contours by labelling skeletal points according to whether or not they represent the collision of consistently oriented boundary fronts. Cross and Hancock [3] have appealed to a magnetostatic analogy in which the edge tangent flow is regarded as a current density on the image plane. The differential structure of the resulting vector potential can be used to characterise symmetry lines, boundary-edges and corners [5]. By sampling the vector potential at various heights above the image plane, a scale-space representation is induced. One of the shortcomings with this method is that like the binary skeleton, there is no way of moderating the effects of high curvature effects, other than by smoothing. This can prove time consuming and has the effect of removing genuine boundary structure.

The aim in this paper is to return to the magnetostatic analogy, and to incorporate curvature dependent damping of the current, i.e. the edge tangent flow. This has the effect of controlling boundary noise and improving the shapeliness of the skeleton. To improve the postprocessing of the vector potential, we apply hysteresis linking to the candidate skeleton points. In the end, we demonstrate how can we control the degree of smoothing of high curvature boundary features.

2 Image Representation Using Vector Potential

We commence by convolving the raw image I with a Gaussian kernel of width σ. The kernel takes the following form

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left[-\frac{x^2+y^2}{2\sigma^2}\right] \tag{1}$$

With the filtered image to hand, the Canny edge map is recovered by computing the gradient

$$\underline{E} = \nabla G_\sigma * I \tag{2}$$

In order to compute a vector field representation of the edge-map, we will need to introduce an auxiliary z dimension to the original $x-y$ co-ordinate system of the plane image. In this augmented co-ordinate system, the components of the edge-map are confined to the image plane. In other words, the edge-vector at the point $(x,y,0)$ on the input image plane is given by

$$\underline{E}(x,y,0) = \begin{pmatrix} \frac{\partial G_\sigma * I(x,y)}{\partial x} \\ \frac{\partial G_\sigma * I(x,y)}{\partial y} \\ 0 \end{pmatrix} \tag{3}$$

For an ideal step-edge, the resulting image gradient will be directed along the boundary normal. In order to pursue our magneto-static analogy we would like to interpret the raw edge responses as elementary currents which flow around the boundaries and give rise to a vector potential. In other words, we would like to organise the elementary currents so that they are tangential to the boundaries of physical objects. Accordingly, we re-direct the edge-vectors to that they are tangential to the original planar shape by computing the cross-product with the normal to the image plane $\hat{z} = (0,0,1)^T$. The elementary current-vector at the point $(x,y,0)$ on the input image plane is defined to be

$$\underline{j}(x,y,0) = \hat{z} \wedge \nabla G_\sigma * I(x,y) \tag{4}$$

The key idea underlying the image representation is to characterise edges and symmetry lines using a vector potential. Edges corresponded to locations where the elementary current re-enforce one-another. In other words, the boundaries are identified as local maxima of the vector potential. Symmetry points are those at which there is cancellation between diametrically opposed elementary currents. Axes of symmetry are lines of local minimum in the vector potential. At the level of fine detail, intensity ridges or ravines (lines) give rise to local symmetry axes.

According to magneto-statics, the vector-potential associated with a field of elementary currents is found by integrating over volume and weighting the contributing currents according to inverse distance. In other words, the vector potential at the point $\underline{r} = (x, y, z)^T$ in the augmented space in which the original image plane is embedded is

$$\underline{A}(x, y, z) = \mu \int_{V'} \frac{\underline{j}(x', y', z')}{|\underline{r} - \underline{r}'|} dV' \tag{5}$$

where $\underline{r}' = (x', y', z')^T$ and μ is the permeability constant which we set equal to unity. Since the contributing currents are distributed only on the image plane, the volume integral reduces to an area integral over the image plane. As a result, the components of the vector potential are as follows

$$\underline{A}(x, y, z) = \begin{pmatrix} -\int\int \frac{\partial G_\sigma * I(x',y')}{\partial y'} \frac{1}{\sqrt{(x-x')^2+(y-y')^2+z^2}} dx'dy' \\ \int\int \frac{\partial G_\sigma * I(x',y')}{\partial x'} \frac{1}{\sqrt{(x-x')^2+(y-y')^2+z^2}} dx'dy' \\ 0 \end{pmatrix} \tag{6}$$

The structure of the vector-potential deserves further comment. In the first instance, the components are confined to the $x - y$ plane for all values of the auxiliary co-ordinate z. However, as we move away from the image plane the role of this auxiliary dimension is to average the generating currents over an increasingly large area of the original image plane. In other words, if we sample the vector-potential for various $x - y$ planes at increasing height above the image plane, we induce a scale-space representation. We exploit this property to produce a fine-to-coarse image representation as we sample the vector potential at increasing heights above the physical image plane.

3 Curvature Estimation

In this section we consider how to incorporate curvature dependent smoothing into the vector potential to control the effects of high curvature boundary noise. To measure the curvature we use the method developed by Harris [4] which is itself an extension of Moravec's [6] corner detector. We commence by approximating the Hessian matrix $H = \nabla\nabla^T I$ by the matrix $E = (\nabla I)(\nabla I)^T$, i.e.

$$E = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{7}$$

where I_x and I_y are the first-derivatives of the image I in the x and y directions. Suppose that α and β are the eigenvalues of the symmetric matrix E. The eigenvalues are proportional to the principal curvatures of the image intensity function, and the product of eigenvalues $\alpha\beta$ is hence proportional to the Gaussian curvature, while the sum of eigenvalues $\alpha + \beta$ is proportional to the mean curvature. According to Harris, the curvature response is the weighted sum of mean and Gaussian curvatures $R = \alpha\beta - k(\alpha+\beta)^2$. The quantity R can be used to characterise image features using the following tests

$$\begin{array}{lll} R > 0 & \textit{detected} & \textit{corners} \\ R < 0 & \textit{detected} & \textit{edges} \\ R \approx 0 & \textit{flat} & \textit{area} \end{array}$$

We use the curvature measure to damp the current density. We adopt a model in which the effect of the current decays exponentially with the curvature of the image. The modified current density is

$$\underline{j}(x,y,0) = \left[\hat{z} \wedge \nabla G_\sigma * I(x,y)\right] e^{-K \cdot R_{x,y}}$$

where K is a constant. Hence, the contribution from straight boundary segments is enhanced and the contributions from corners or places where the edge direction changes rapidly are suppressed.

By varying the constant K, we can control the degree of smoothing of high curvature boundary features. When K equals zero , then the original current density is recovered. When K is increased, the amount of curvature suppression is increased. As we will demonstrate later, the method of smoothing effect of boundary noise does not blur away genuine skeleton structures.

With the current density to hand, the vector potential is computed by performing the volume integration in Equation 6.

4 Skeletonisation and Noise Elimination

According to Cross and Hancock's [3] representation of image structure, symmetry lines follow the local minima of the vector potential and edge contours follow the local maxima. When viewed from the perspective of the differential structure of the vector potential, symmetry lines are locations where the component of the curl in the image plane vanish, i.e. $\hat{z} \wedge \nabla \wedge A(x,y,z) = 0$; edges are locations where the transverse component of the divergence vanishes, i.e. $\nabla \cdot (\hat{z} \wedge A(x,y,z)) = 0$. The main problem with this method for feature localisation is that it is subject to noise. In their work, Cross and Hancock fitted a prism surface to localise symmetry lines. Here we adopt a more sophisticated approach.

We commence by applying hysteresis linking to the edge (ridge) and symmetry (valley) lines in the vector potential to improve connectivity. For the edge (ridge) lines we perform connected components analysis. We dismember the web of ridge and valley lines at the locations of T-junctions. We then note the whereabouts of closed edge (ridge) lines. We remove those symmetry (valley) lines that fall outside the closed edge contours, and retain only those that remain in the interior.

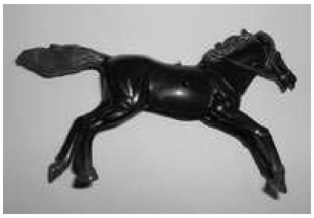
We illustrate the steps of our algorithm. In Figure 1(a) we show the raw image used in our study. Figures 1(b) and 1(c) show the principal curvatures. The maximum curvature is large along the edge contours of the image while the minimum curvature is large only at the locations of corners and noise. Figure 1(d) shows the magnitude $|\underline{j}|$ of the damped current density. White points correspond to locations where there is strongest damping, while the darker points correspond to strong current density. In Figure 1(e) we show the magnitude of the vector potential. Turning our attention to the postprocessing of the vector potential, the detected ridge and symmetry lines are shown in Figure 1(f). Figure 1(g) shows the result of applying hysteresis linking to the symmetry (valley) lines. After connected components analysis is performed, and the external symmetry lines have been removed then the resulting skeleton is shown in Figure 1(h). If the skeleton is dismembered at T-junctions, and branches with weal response are removed the result shown in Figure 1(i) is obtained. Finally, we fit straight lines to the detected symmetry lines and merge lines that are nearly parallel and close to each other, to obtain the result shown in Figure 1(j).

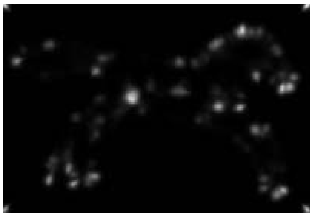
5 Experiment

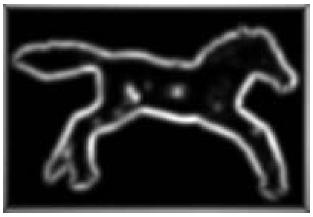
In this section, we provide some experimental evaluation of our noise-damped vector potential representation. The experimental work is divided into two parts. We commence with some examples on synthetic images to illustrate the effect of K in damping the noises compared with the similar effect from increasing height. Next we furnish some real-world examples.

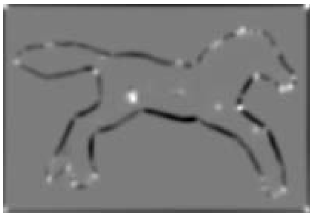
On the first row of Figure 2, we show three synthetic images with small spikes on the boundary and with a grey scale gradient on the interior. From the second row, we show in turn the magnitude of the vector potential displayed as a surface plot D, and, the correspondent detected ridges and ravines in black and white. For the first three rows of plots of vector potential, we increase the parameter K from 0 to 1, and then to 1.5. As we increase K, the effect is to damp-out noise while retaining the detail of the skeleton. As explained earlier, we can endow our image representation with a scale-space dimension by sampling the vector potential at increasing heights above the image plane. This is shown on the last two rows of vector potential plot by increasing the sampling height as we descend the columns. The effect of increasing the sampling height is also to smooth away noise, but this is as the expense of detail in the detected skeleton.

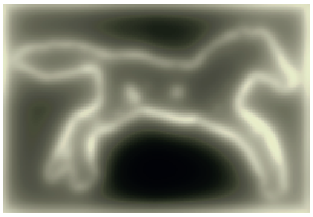
In Figure 3, we show some real-world images. There are four sets of images, grouped vertically for the first three with another one left at the bottom. For the vertical groups, the top panel shows the original image, the second panel shows the initial output, the third panel and the bottom panel show the output smoothed by increasing K and height respectively. While for the last group, the order is from top to bottom and from left to right.

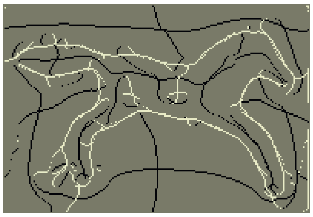

(a) Original image

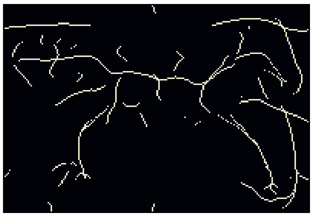

(b) First curvature matrix

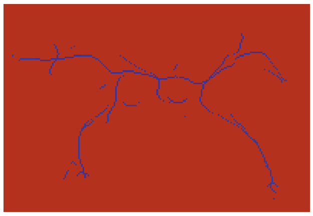

(c) Second curvature matrix

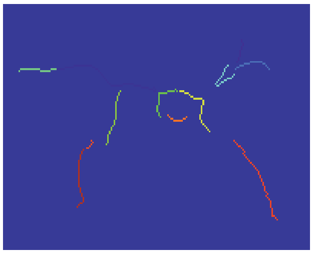

(d) Damped response from R function


(e) Vector potential magnitude


(f) Ridges and ravines


(g) Symmetry lines after hysteresis


(h) Skeleton lines


(i) Segmented skeleton lines

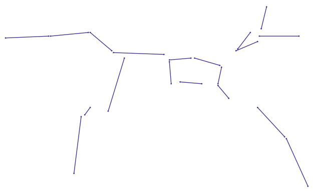

(j) Skeleton in straight lines

Fig. 1. Skeletonisation and noise elimination

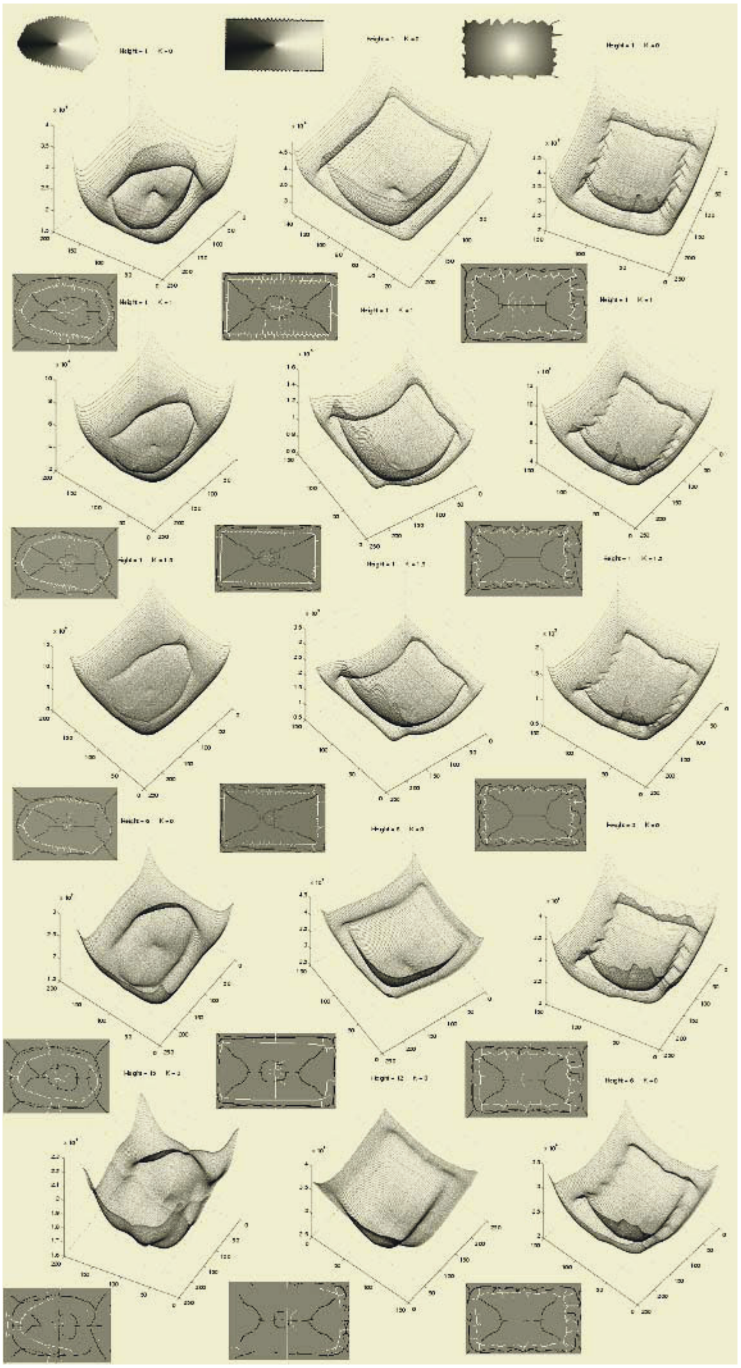


Fig. 2. Synthetic images with increasing K and height

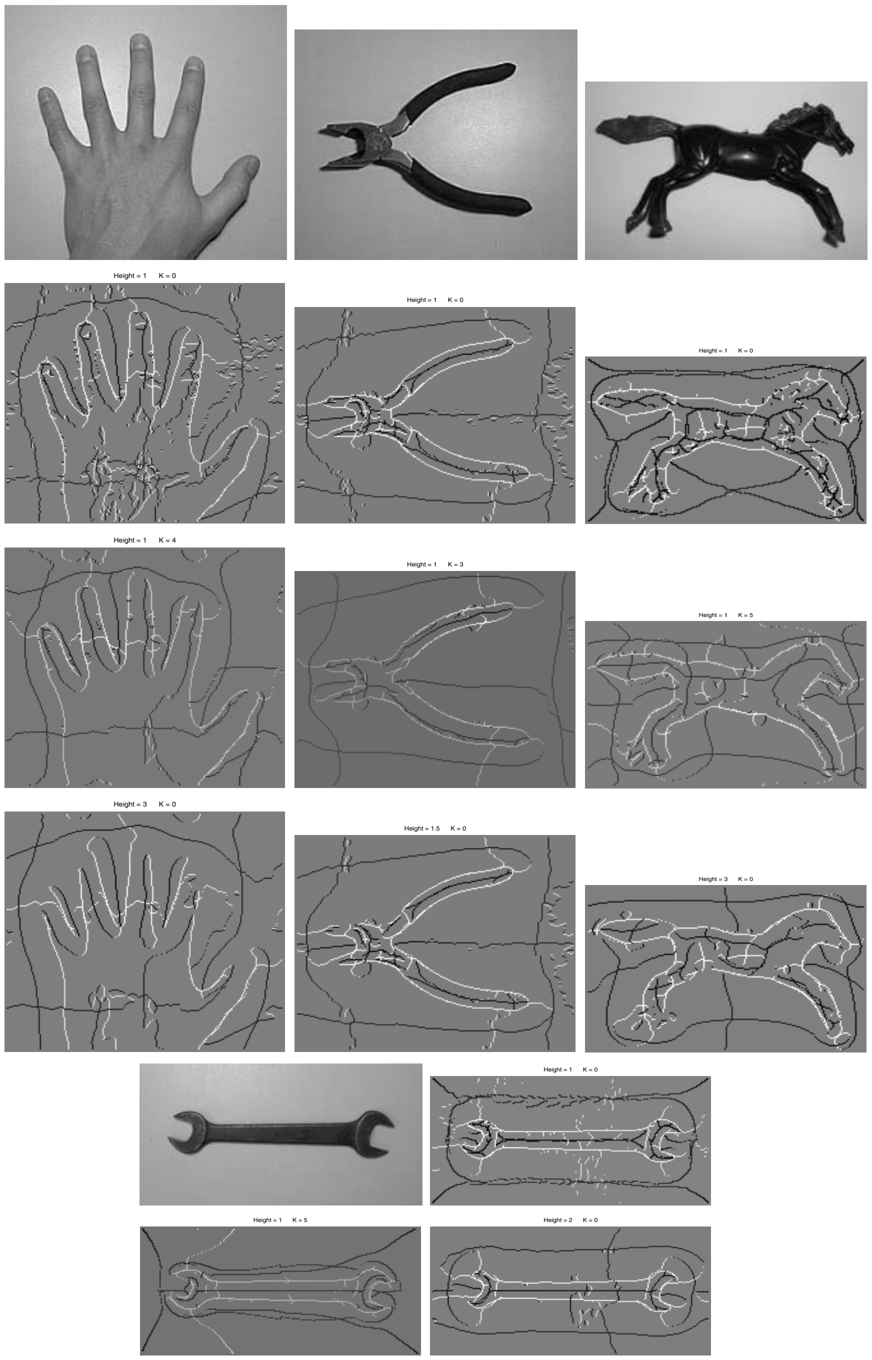


Fig. 3. Experimental results for real world objects

The main feature to note from these examples is that we can damp the noise by increasing parameter K as well as by increasing the height, but curvature damping outperforms height sampling in the following ways. First, the visual appearance of the results is more pleasing. Second, we preserve local curvature information. Third, as is the case with the synthetic images, when we increase the height, some useful ridges and ravines, especially ridges, begin to vanish rapidly as well as the noise.

6 Conclusion

In this paper, we incorporate curvature effects into the computation of grey-scale skeletons. The method builds on the magnetostatic analogy of Cross and Hancock, and employs a curvature dependent current damping. In addition, we have described a number of postprocessing steps that can be used to improve the quality of the detected skeletons. The advantages of the method are improved noise resilience of the detected skeleton, and better skeleton connectivity. In the end, we illustrate how to control high boundary noise by varying constant K rather than by increasing height. This results in good noise control and does not blur the details of the skeleton.

References

1. C. Arcelli and G.S Di Baja. A width-independent fast thinning algorithm. *IEEE PAMI*, 7(4):463–474, 1985.
2. H Blum. *A transformation for extracting new descriptors of shape*. MIT Press, W. Wathen-Dunn Ed. Models for the perception of speech and visual form, 1967.
3. A.D.J. Cross and E.R. Hancock. Scale space vector fields for symmetry detection. *Image and Vision Computing*, 17:337–345, 1999.
4. C.G. Harris and M.J. Stephens. A combined corner and edge detector. *Proceedings Fourth Alvey Vision Conference*, pages 147–151, 1988.
5. B. Luo, A. D. Cross, and E. R. Hancock. Corner detection via topographic analysis of vector potential. *Pattern Recognition Letters*, 20(6):635–650, 1999.
6. H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. doctoral dissertation, Robotics Institute, Carnegie Mellon University, 1980.
7. R.L. Ogniewicz and O. Kübler. Hierarchic voronoi skeletons. *Pattern Recognition*, 28(3):343–359, 1995.
8. K. Siddiqi, S. Bouix, A. Tannenbaum, and S.W Zucker. The hamilton-jacobi skeleton. *International Conference on Computer Vision*, pages 828 – 834, 1999.
9. K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1):13–32, 1999.
10. Z. S. G. Tari, J. Shah, and H. Pien. Extraction of shape skeletons from grayscale images. *Computer Vision and Image Understanding*, 66:133–146, 1997.
11. H. Tek, P. A. Stoll, and B. B. Kimia. Shocks from images: Propagation of orientation elements. *IEEE CVPR*, pages 839–845, 1997.
12. Andrea Torsello and Edwin R. Hancock. A skeletal measure of 2d shape similarity. *Lecture Notes in Computer Science*, 2059:260–271, 2001.

License Vehicle Plates Localization Using Maximum Correlation

Regis C.P. Marques, Fátima N.S. Medeiros,
Jilseph Lopes Silva, and Cassius M. Laprano

UFC- DETI
Campus do PICI, Bloco 705
60455-760 - Fortaleza, CE, Brazil
{regismarques,fsombra,jilseph,cassiusml}@deti.ufc.br
http://www.gpi.deti.ufc.br

Abstract. In this paper we propose an automatic method to locate Brazilian license plates in digital images acquired from a real monitoring traffic system. The approach is based on the correlation between lines of the preprocessed image and a square wave form that resembles lines in the region of the pattern plate. The maximum correlation points out the horizontal and vertical crossing axes over the plate. Prior to locating plates, the post processing Min/Max scheme is applied to the binary images in order to diffuse segmentation remnant noise.

1 Introduction

Many intelligent systems have been developed for traffic control and monitoring system in the last decade. Applications such as detection of irregular vehicles, parking and toll control, license plates location and reading are spread all over the world. These applications are based on automatic recognition of the license plates.

Comelli et al. [1] presented a system to recognize Italian license plates passing through a tollgate. The recognition system consisted of three main phases as follows: plate location, image preprocessing and characters recognition. Fig. 1 exhibits the modules that compounds this system. The preprocessing module consisted of conventional filtering and enhancement techniques and the characters within the plates were recognized by using a combination of operators based on pattern matching and templates. The algorithms were tested on more than three thousand real images with a recognition rate close to 91%. In [2] Naito and Tsukada proposed a robust plates recognition system. This system is capable of capturing fine image under bad illumination conditions, from twilight up to noon in the sunshine, capturing non-blurred moving vehicle images and recognizing plates even being inclined. In this system, even if the position of TV camera varied widely over 97% of the license plates could be recognized successfully [2]. Naito and Tsukada [2] evaluated the license plates location in binary images instead of using gray level images in order to simplify this task. Inspired in this idea we propose in this paper an algorithm to locate license plates in binary images. Our proposed method concerns the modules inside the dotted lines in Fig.1. It was developed to locate Brazilian license plates in digital images, although it is potentially applicable to other foreign plates.

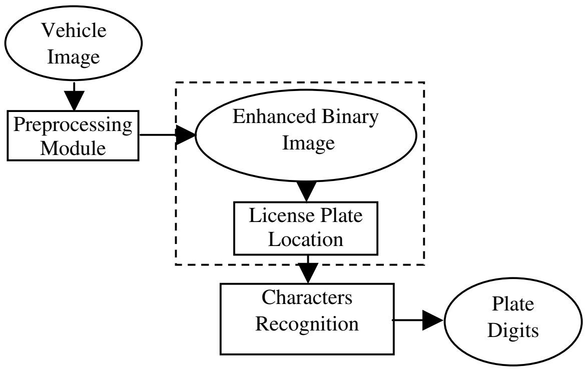


Fig. 1. Vehicle plate recognition system.

An image-based recognition system proposed in [6] applies neural network for automatic vehicle plate recognition. It consists mainly of two processes: the training process and the recognition process. In the former the database of coded plates characters is built and the neural networks are trained to recognize. The latter includes the vehicle plate localization, plate binarization, symbol segmentation, coding of segmented symbols and character recognition. The system was assessed according two aspects such as: the ability to estimate plate position and the plate characters recognition rate.

This paper is organized as follows. Section 2 describes the proposed methodology used to locate license plates. In Section 3 the experimental results are presented and Section 4 summarizes the conclusions.

2 The Proposed Methodology

This paper proposes an algorithm to locate license plates in moving or parked vehicles images. The method achieves the maximum correlation operator between the lines of the binary image and a wave form that resembles the pattern plate. In order to improve the segmentation process we apply the Min/Max method proposed by Malladi and Sethian [3] to remove the remnant noise of the segmentation process.

The background concepts on which the proposed method is based on such as segmentation, the Min/Max post-processing scheme and the correlation approach are described in the following.

2.1 Image Segmentation

Segmentation is an important task in systems based on digital image processing. Traditionally, thresholding segmentation methods [5] are very popular and computationally efficient in case of bimodal histograms. When these methods are extended to multimodal histograms images they tend to be computationally expensive and inaccurate. However, it is not the aim of this paper to develop or improve segmentation methods. Therefore a simple bilevel thresholding method is used to generate a binary image of the license plate pointing out it from the background.

2.2 The Min/Max Post-processing Method

Level set methods are interactive schemes for noise removal and image enhancement. Some algorithms based on these schemes require a stopping criterion due to Grayson's theorem that says the contour shrinks to zero and disappears [4].

In [3] was proposed a function called Min/Max that switches between removing noise and maintaining essential image properties. It means that the correct curvature flow is based on neighbourhood characteristics. The Min/Max function is presented in Equation (1), where *Ave* is defined as the average value of *I* in a disk of radius *R* centered around the point (*x*,*y*) [4].

$$F_{\min/\max} = \begin{cases} \min(k,0) & if \quad Ave^{R}_{I(x,y)} < 0 \\ \max(k,0) & if \quad Ave^{R}_{I(x,y)} \geq 0 \end{cases} . \tag{1}$$

The radius *R* implies that the *R*th pixels order structures are diffused into the background values. In the test images, the noise is characterized by small structures of first order. Due to this, the radius *R* =1 was chosen in order to diffuse remnant small structures from the segmentation process.

Fig. 2(a) displays a binary image of a vehicle plate and Fig. 2(b) shows its enhanced version applying the Min/Max scheme. As a consequence of the post-processing task it can be observed a visual improvement in the segmented image. It is worth noting that the segmentation remnant noise in Fig. 2(a) can affect the localization algorithm performance. Thus, the good quality of the images is assured by the post-processing task leading to accurate results.

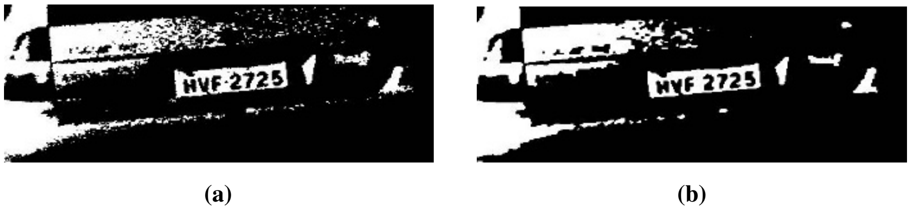


Fig. 2. (a) Segmented plate image and (b) the enhanced version.

2.3 The Proposed Method Based on Maximum Correlation

To locate plates the proposed method calculates the correlation between the locally enhanced binary image (*img*) and a 1-D square wave function, with period *p* and length *l* (*f*(*p*,*l*)). The use of a square wave form is justified by the amplitude patterns found in lines over the plate characters region in the binary image. These patterns are shown in Fig. 3, where the central region exhibits the plate patterns.

The Location Process Using the Square Wave Function

The period of the square wave function is defined as being the half of width of the plate character and the length of the convolution mask *l* is determined by the width of the plate (Fig. 4).

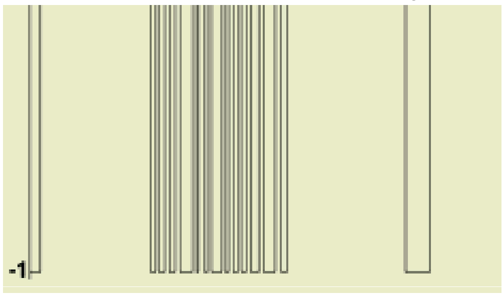


Fig. 3. The pattern line of a plate image.

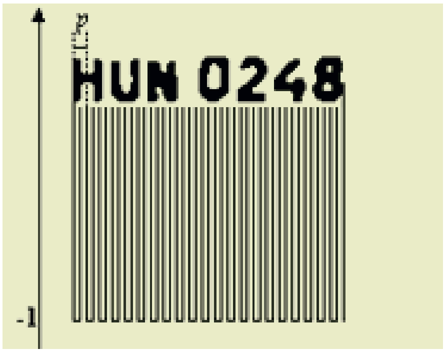


Fig. 4. The square wave function.

Once defined the wave form pattern, the correlation matrix is calculated and the maximum value in it points towards the plate location. The coordinates of the maximum correlation are used to indicate the center of the crossing axes in the located plate. The mathematical expression that defines it is given by:

$$PlateLocation = \max\{img * f(p,l)\}. \tag{2}$$

where max indicates the maximum correlation value in the correlation matrix, and $\{*\}$ indicates the convolution operator.

Fig. 5 presents the correlation result between a 1-D square wave function and the wave form in Fig. 6.

3 Experimental Results

The algorithm was applied to a set of 40 real test images. The database was divided into group 1 and group 2. The former consisting of 7 parked vehicles images and the latter consisting of 33 moving cars images taken by a real monitoring traffic system. Fig. 6 shows a parked vehicle image and Fig. 7 shows the located plate. Fig. 8 and Fig. 9 illustrate the original vehicle image and the incorrect plate location when applied to a moving car image. Fig. 11(a) and Fig. 13(a) illustrate the differences between the images that were post-processed and the ones that were not. The use of the post-processing Min/Max scheme improved the performance of the localization algorithm as display Fig. 11(a) and Fig. 13(a). The correct plate location corresponds to the horizontal and vertical axes crossing and it was obtained an overall error rate below 3% using the set of test images.

Different parameters *p* and *l* were set to parked and moving vehicles images, since these values depend on the distance that the pictures are taken by the acquisition system. For parked vehicles images *p*=10 and *l*=200 (pixels), for moving cars images *p*=15 and *l*=200 (pixels).

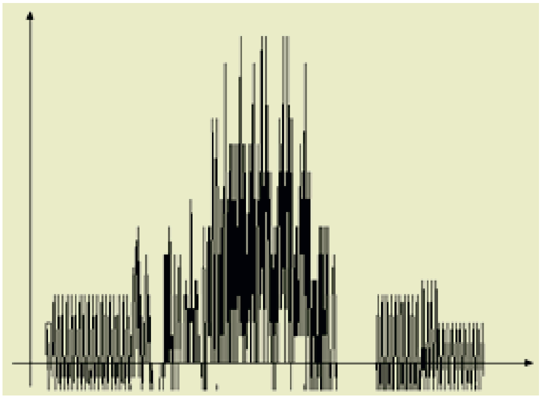


Fig. 5. The correlation result.

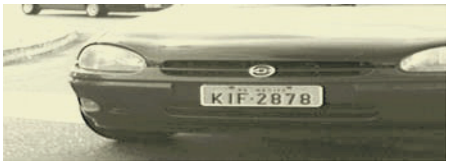


Fig. 6. Parked vehicle image.

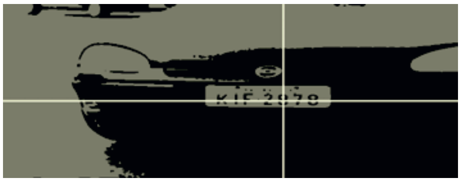


Fig. 7. The located plate.

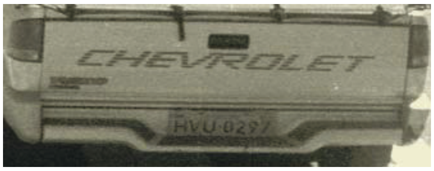


Fig. 8. Moving vehicle image.

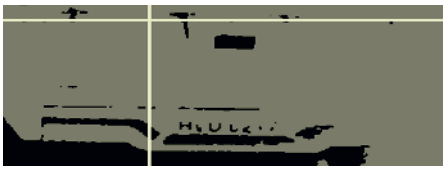


Fig. 9. Incorrect located plate.

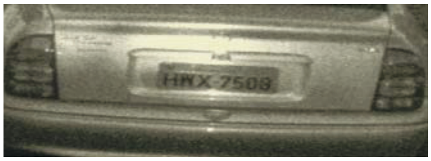
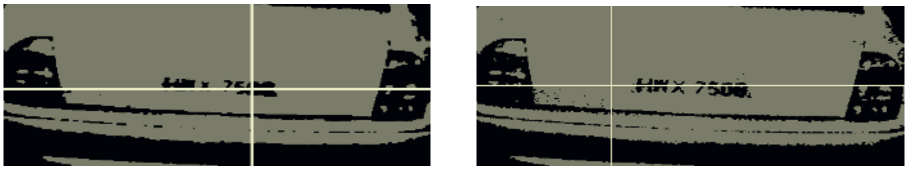


Fig. 10. Moving vehicle image. (Acquisition system in an indirect look).



(a) **(b)**

Fig. 11. (a) Located plate using the Min/Max scheme and (b) not using it.

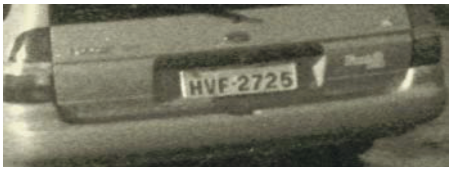
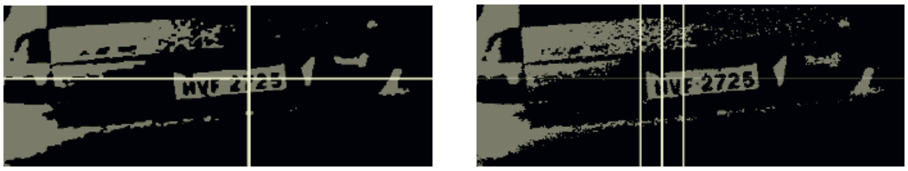


Fig. 12. Moving vehicle image. (Acquisition system in a direct look).



(a) **(b)**

Fig. 13. (a) Located plate using the Min/Max scheme and (b) not using it.

4 Concluding Remarks

The test images used in this paper were segmented simply employing a threshold segmentation method. The threshold value was not automatically adjusted for each image because it is difficult to set it for any kind of gray level image. To improve the proposed vehicle plates localization method, the binary images were enhanced by applying the suitable Min/Max scheme. This post-processing algorithm eliminated the remnant noise in the segmented vehicles images and provided reasonable localization results.

The preliminary tests of the algorithms were taken on the set of parked vehicles images producing good results as presents Fig. 7. Furthermore, the most relevant results of the algorithms concerns the moving car images taken by a real monitoring traffic system in different illumination conditions and car positions. Neither the parameters of the square wave function nor the image position were changed during the tests.

The critical point of the proposed method is the segmentation method. As displays Fig. 14 the plate localization result was influenced by the bilevel thresholding scheme used to segment the image. Thus, further developments will focuse segmentation improvements to overcome these kind of difficulties. It is worth noting that the algorithm was adjusted to the Brazilian plates, thus the wave form depends on the national plate pattern and the acquisition system of the license vehicle plates. Therefore, it can be adjusted to different plate patterns.

References

1. Comelli, P., Ferragina, P., Stabile, M. N., Stabile, F.: Optical Recognition of Motor Vehicle License Plates. IEEE Transactions on Vehicular Technology. Vol. 44(4), (1995) 790-799
2. Naito, T., Tsukada, T., Yamada, K., Kozuka, K., Yamamoto, S.: Robust License-Plate Recognition Method for Passing Vehicles Under Outside Environment. IEEE Transactions on Vehicular Technology. Vol. 49 (6) (2000) 2309-2319
3. Malladi, R., Sethian, J. A.: Image Procesing: Flows under Min/Max Curvature and Mean Curvature. Graphical Models and Image Processing. Vol. 59 (2) (1996) 127-141
4. Sethian, J. A.: Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry-Fluid Mechanics. 2nd edn. Computer Vision, and Materials Science. Cambridge University Press. New York (1996)
5. Pratt, W. K.: Digital Image Processing. 2nd edn. John Wiley & Sons. New York (1991)
6. Vázquez, M. Nakano, M., Pérez-Meana, H.: Automatic System for Localization and Recognition of Vehicle Plate Numbers. Journal of Applied Research and Technology. Vol. 1 (1) (2003) 63-77

Adaptive Context for a Discrete Universal Denoiser

Georgy Gimel'farb

Department of Computer Science, Tamaki Campus
The University of Auckland, Auckland, New Zealand
g.gimelfarb@auckland.ac.nz
http://www.tcs.auckland.ac.nz/~georgy

Abstract. Statistical analysis of spatially uniform signal contexts allows Discrete Universal Denoiser (DUDE) to effectively correct signal errors caused by a discrete symmetric memoryless transmission channel. The analysis sets no limits on a probability signal model apart from stationarity and ergodicity. Statistics of signal contexts are used first to learn the probability of errors and then to detect and correct the errors. Therefore a proper choice of context is an essential prerequisite to the practical use of DUDE. We propose to use the maximum likelihood estimate of context assuming the signals are modelled with a nonparametric generic Markov–Gibbs random chain or field. The model adds to stationarity and ergodicity only one more condition, namely, pairwise dependences between each signal and its context. Experiments with noisy binary images confirm a feasibility of such adaptive context, show some advantages of DUDE over more conventional median filtering, and relate the choice of a proper context size to the maximum entropy of the context statistics used for image denoising.

1 Introduction

Although denoising is one of the most extensively studied areas of signal and image processing, the variety of models and techniques involved is permanently growing (see, e.g., [1–3, 5] to cite a few). Discrete Universal Denoiser (DUDE) proposed recently in [6] recovers an original 1D sequence or 2D array of signals by analysing signal contexts in a noisy signal set corrupted by a memoryless symmetric transmission channel. In spite of simplicity, such signal model is of interest in many important practical applications where probability characteristics of signals are unknown, except for stationarity and ergodicity. The context consists of signals in a fixed (translation invariant) neighbourhood of each position in the sequence or array. Because marginal probability distributions of stationary and ergodic signals are translation invariant, DUDE uses relative frequencies of different signals with the same context first to learn the error probability for the channel and then to detect and correct errors.

Efficiency of DUDE essentially depends on context geometry, i.e. the number and relative positions of neighbours. Generally, most adequate geometry depends on signal sets to denoise. But typically the context is pre-defined by heuristic considerations, e.g., signals in a fixed rectangular window around each position. This paper attempts to directly estimate the best context assuming that the noisy signals are samples of a nonparametric generic Markov–Gibbs random chain or field [4]. This spatially uniform model is also stationary and ergodic, but it adds one more general restriction, namely, only pairwise dependencies between each signal and its context. This restriction allows for the approximate maximum likelihood estimate (MLE) of the context.

The paper is organised as follows. Section 2 describes DUDE in brief, compares it to a conventional median filter denoiser (MFDE), and discusses the MLE of context for the Markov–Gibbs signal model. Experiments with DUDE and MFDE and an entropy-based choice of the context size are presented in Section 3.

2 Maximum Likelihood Context

For brevity, we address only 2D binary images, although with obvious changes our consideration applies also to 2D arrays and 1D sequences of multi-level signals. Let $\mathbf{R} = \{(x,y) : x = 0,\dots,X-1; y = 0,\dots,Y-1\}$ denote an arithmetic 2D lattice supporting signal arrays $g : \mathbf{R} \rightarrow \mathbf{Q}$ where $\mathbf{Q} = \{0,\dots,Q-1\}$ is a finite set of scalar signals ($Q = 2$ for binary images). A translation invariant neighbourhood of size K for each site $(x,y) \in \mathbf{R}$ effects conditional probabilities of signal values $g(x,y) \in \mathbf{Q}$. Its geometry is specified by a set of (x,y)-increments $\mathbf{N} = \{(\xi_k, \eta_k) : k = 1,\dots,K\}$ such that each signal $g(x,y)$ has the context $\mathbf{C}_j = \{g(x+\xi, y+\eta) : (\xi,\eta) \in \mathbf{N}; (x+\xi, y+\eta) \in \mathbf{R}\}$. For binary images, there are 2^K different contexts $\mathbf{C}_j$; $j \in \{0,\dots,2^K-1\}$, such that j is the binary number $q_1 q_2 \dots q_K$ where $q_k \in \mathbf{Q}$.

Both DUDE and MFDE use contexts to decide whether a binary signal is true or corrupted. The passive MFDE follows the majority rule: each signal $g(x,y)$ is true if at least half of the context signals $(g(x+\xi, y+\eta) : (\xi,\eta) \in \mathbf{N})$ have the same value. The active DUDE finds 2^{K+1} conditional frequencies $f(q|\mathbf{C}_j)$; $q \in \mathbf{Q}$, of signals in a pixel, given its context $\mathbf{C}_j$ of size K; $f(0|\mathbf{C}_j) + f(1|\mathbf{C}_j) = 1$. If the minimum of these two frequencies is less than a certain threshold, θ, then such "less frequent" signal with this particular context is assumed to be noisy and will be reversed. The threshold derived in [6] depends on the noise probability estimated by the minimum conditional frequency of signals over all the contexts: $\mathrm{Pr}_{\text{noise}} \cong \min_{j,q}\{f(q|\mathbf{C}_j\}$.

We consider a noisy signal array $\mathbf{g} = (g(x,y) : (x,y) \in \mathbf{R})$ as a sample of a Markov–Gibbs random field with translation invariant geometric structure of pairwise dependencies between signals [4]. The model is specified with the Gibbs probability distribution

$$Pr(\mathbf{g}|\mathbf{N}) \propto \exp\left(\sum_{(x,y)\in\mathbf{R}}\left(V(g(x,y)) + \sum_{(\xi,\eta)\in\mathbf{N}} V_{\xi,\eta}(g(x,y), g(x+\xi, y+\eta))\right)\right)$$

Here, $V : \mathbf{Q} \rightarrow \mathbf{U}$ and $V_{\xi,\eta} : \mathbf{Q} \rightarrow \mathbf{U}$ are Gibbs potentials and $\mathbf{U}$ is the set of real numbers bounded from above. Both the potentials for and spatial geometry $\mathbf{N}$ of interdependent signal pairs are not predefined, may differ for different types of signals, and are estimated from a given signal array $\mathbf{g}$.

The log-likelihood $L(\mathbf{N}|\mathbf{g}) = \frac{1}{XY}\log \mathrm{Pr}(\mathbf{g}|\mathbf{N})$ maximised by the potentials and depending only on the neighbourhood $\mathbf{N}$ is analytically approximated as $L(\mathbf{N}|\mathbf{g}) = -\sum_{q=0}^{Q-1} f(q|\mathbf{g}) \log f(q|\mathbf{g}) + \sum_{(\xi,\eta)\in\mathbf{N}} d_{\xi,\eta}(\mathbf{g})$ by adapting the derivation in [4] to different marginal signal probabilities. Here, $d_{\xi,\eta}(\mathbf{g})$ is the squared distance between

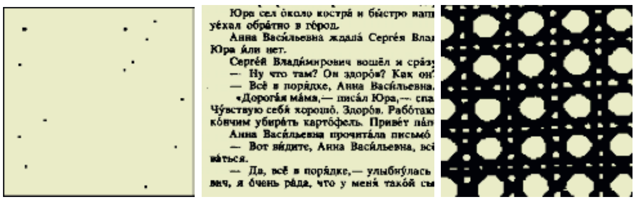


Fig. 1. Noiseless “Stars”, “Cyrillic Text”, and “Brodatz’s Cane”.

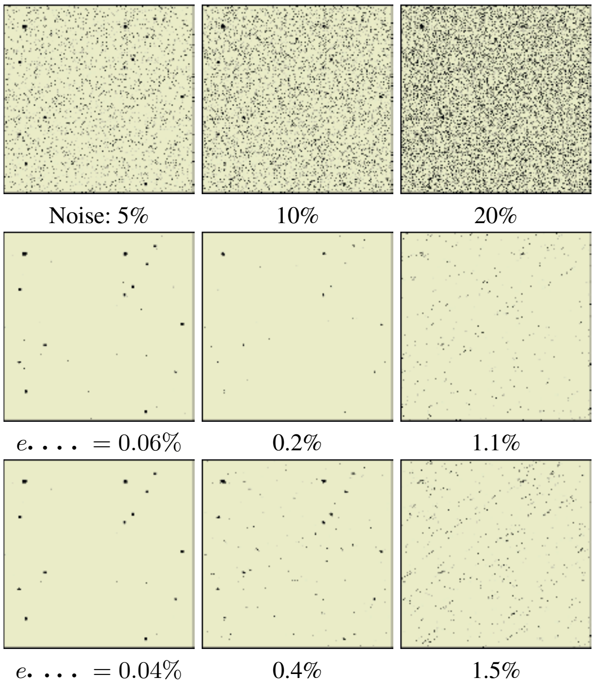


Fig. 2. Noisy and denoised “Stars”.

Table 1. Estimates of the noise probability Pr..... in DUDE.

Image:	“Stars”						“Cyrillic text”						“Brodatz’s Cane”					
Pr.....,%:	5		10		20		5		10		20		5		10		20	
Size K:	2	4	2	4	2	4	2	4	2	4	2	4	2	4	2	4	2	4
Estimate, %:	5.1	2.3	9.9	8.3	19.8	19.2	7.4	5.2	12.7	10.4	23.0	20.6	5.7	4.9	11.2	10.1	23.8	19.7

the frequency distribution $\mathbf{F}_{\xi,\eta}(\mathbf{g}) = \{f_{\xi,\eta}(q,s|\mathbf{g}) : q,s \in \mathbf{Q}^2\}$ of actual signal cooccurrences $(g(x,y) = q, q(x+xi, y+\eta) = s)$ in translation invariant pixel pairs $(x,y),(x+\xi,y+\eta) : (x,y) \in \mathbf{R}; (x+\xi,y+\eta) \in \mathbf{R}$ and the like distribution of independent signals with the same marginal probabilities $f(q|\mathbf{g})$; $q \in \mathbf{Q}$: $d_{\xi,\eta}(\mathbf{g}) = \sum_{(q,s)\in\mathbf{Q}} \left(f_{\xi,\eta}(q,s|\mathbf{g}) - f(q|\mathbf{g})f(s|\mathbf{g})\right)^2$. Therefore, the approximate MLE

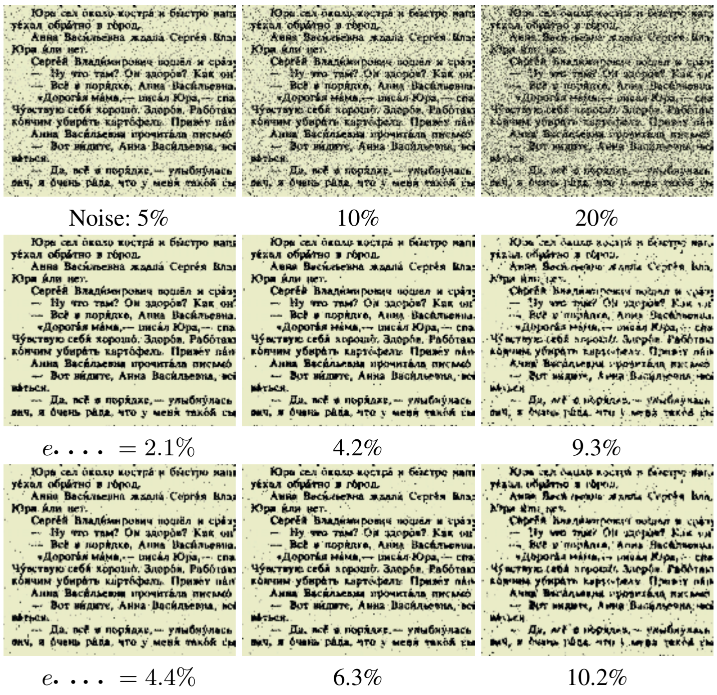


Fig. 3. Noisy and denoised "Cyrillic Text".

of the context of a given size K is specified for a particular signal array g by the K top-rank distances $d_{\xi,\eta}(\mathbf{g})$. But it is still necessary to quantitatively relate the context size K ensuring better performance to the context statistics used by DUDE.

3 Adaptive Maximum Entropy Contexts

Figure 1 shows initial binary images "Stars", "Cyrillic text", and "Brodatz's Cane" used in experiments and having 0.3%, 21%, and 66% of black pixels, respectively. The noisy versions obtained by modelling a symmetric transmission channel are presented in Figs. 2–4 together with results of denoising. The channel makes independent random bit inversions with a fixed probability $p = 0.05$, 0.10, or 0.20. Experiments below compare DUDE with the adaptively chosen contexts to MFDE with the same contexts.

For small context sizes, the minimum relative frequency of contexts is a reasonable probability estimate. But as mentioned in [6], it fails for larger contexts due to a large number of too rare or simply absent signal configurations in a given image. Thus we estimate the noise probability and the relevant threshold using only small radially symmetric contexts. Table 1 shows these latter estimates for the symmetric contexts of size 2 and 4 in the noisy images in Figs. 2–4. The estimate for size $K = 4$ fails for "Stars" because of a very small black area. To conduct all experiments in the same conditions, the probability estimates for the smallest symmetric context of size 2 is used below. For two other images the estimates for the size 4 are more adequate, but changes in quality

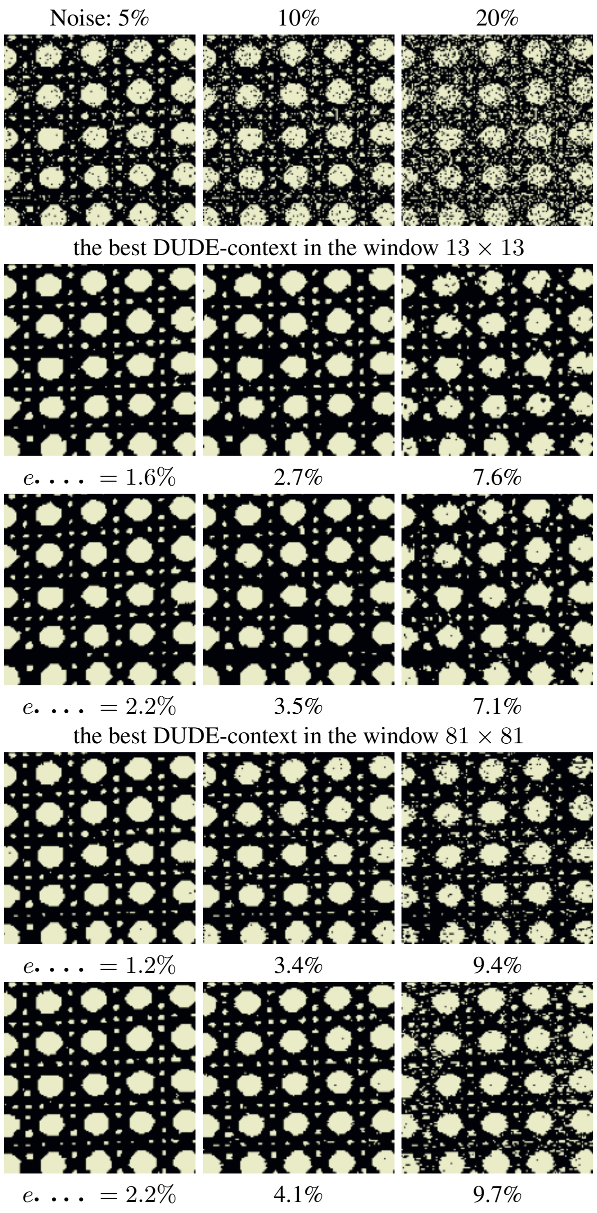


Fig. 4. Noisy and denoised "Brodatz's Cane".

of denoising are marginal. In all our experiments asymmetric neighbourhoods ranked below symmetric ones. Thus, we use below only these latter.

Tables 2–4 present residual errors after image DUDE and MFDE using radially symmetric adaptive contexts $\mathbf{N} = \{(\xi_i, \eta_i), (-\xi_i, -\eta_i) : i = 1, \ldots, K$ of size $K =$

Table 2. Denoising of "Stars": 0.3% black (B) and 99.7% white (W) pixels ($e_{\dots}$ denotes the total residual error after denoising; "B" and "W" indicate the individual residual errors for the black and white areas of the noiseless image, respectively; the maximum entropy and the minimum total errors are boldfaced; the individual errors are in italic and underlined).

Half-size i	1	2	3	4	5	6	7	8	9	10
Noise: 5% (actual: B 2.19%; W 5.09%); $\theta = 0.096$										
(ξ_i, η_i)	$(0,1)$	$(1,0)$	$(0,2)$	$(1,1)$	$(2,0)$	$(0,2)$	$(1,-2)$	$(0,3)$	$(1,-1)$	$(5,-1)$
Entropy	1.0	2.2	**2.5**	2.2	1.6	1.3	1.2	1.0	0.9	0.8
$e_{\bullet\bullet\bullet\bullet}$,%	0.2	0.08	**0.06**	0.08	0.1	0.2	0.3	0.4	0.5	0.8
B,%	53.6	4.9	5.5	4.4	3.8	*2.2*	2.2	2.2	2.2	2.2
W,%	0.01	0.06	*0.05*	0.07	0.1	0.2	0.3	0.4	0.5	0.7
$e_{\bullet\bullet\bullet}$,%	0.8	0.1	**0.04**	0.05	0.05	0.1	0.1	0.1	0.1	0.2
B,%	2.7	*2.2*	3.3	14.2	15.3	30.1	38.3	50.3	48.1	64.5
W,%	0.1	0.03	0.01	*0.0*	0.0	0.0	0.0	0.0	0.0	0.0
Noise: 10% (actual: B 10.9%; W 9.9%); $\theta = 0.178$										
(ξ_i, η_i)	$(0,1)$	$(1,1)$	$(0,2)$	$(2,0)$	$(2,-1)$	$(1,0)$	$(2,2)$	$(1,-2)$	$(1,2)$	$(4,2)$
Entropy	1.0	1.9	**2.9**	2.8	2.3	1.8	1.5	1.3	1.1	1.0
$e_{\bullet\bullet\bullet\bullet}$,%	0.3	**0.2**	0.2	0.2	0.4	0.5	0.9	1.3	1.9	2.5
B,%	99.5	65.6	61.2	42.6	27.9	12.0	*10.9*	10.9	10.9	10.9
W,%	*0.0*	0.03	0.05	0.1	0.3	0.5	0.9	1.3	1.8	2.5
$e_{\bullet\bullet\bullet}$,%	2.8	1.0	0.4	0.2	0.2	**0.1**	0.1	0.2	0.2	0.2
B,%	*13.1*	24.6	27.3	33.9	42.6	37.2	50.8	56.3	61.2	71.6
W,%	2.8	0.9	0.3	0.1	0.03	0.02	*0.0*	0.0	0.0	0.0
Noise: 20% (actual: B 21.9%; W 19.9%); $\theta = 0.317$										
(ξ_i, η_i)	$(0,2)$	$(2,-2)$	$(3,-5)$	$(1,-1)$	$(4,6)$	$(1,1)$	$(3,4)$	$(6,3)$	$(2,-1)$	$(2,4)$
Entropy	1.0	1.5	2.5	3.1	**3.1**	2.5	1.9	1.5	1.1	0.8
$e_{\bullet\bullet\bullet\bullet}$,%	0.3	**0.3**	0.3	0.5	1.1	2.3	4.5	7.4	10.2	13.1
B,%	99.5	99.5	99.5	92.8	80.3	64.5	51.4	43.7	32.2	*28.4*
W,%	*0.0*	0.0	0.02	0.2	0.4	2.1	4.3	7.3	10.1	13.0
$e_{\bullet\bullet\bullet}$,%	10.0	6.0	3.5	2.2	1.5	1.0	0.7	0.6	0.5	**0.4**
B,%	*32.8*	50.8	69.9	67.8	78.1	75.4	82.0	88.5	87.4	92.4
W,%	10.2	5.9	3.3	2.0	1.2	0.8	0.5	0.3	0.2	*0.1*

$2, 4, \dots, 20$. These results suggest that the entropy of distribution of the minimum relative frequencies of signal contexts roughly indicates the best choice of K. In the most cases, the maximum entropy either points directly to the context yielding the smallest residual error or to the adjacent variant. Apart from "Stars" under most intensive noise, DUDE always outperforms MFDE, but both the denoisers are opposite with respect to which areas are successfully denoised or additionally corrupted.

The channel with 5% noise had actually corrupted only 2.2% of the small black "Stars". The best DUDE result fot the context of size 3 is slightly worse than of MFDE with the same context. Both DUDE and MFDE additionally corrupt the black areas (to 5.5% and 3.3%, respectively) while almost completely clean the white background (0.05% and 0.01% of the residual noise, respectively, comparing to the initial 5.09%). When the context size increases, DUDE corrupts the black area less (down to 2.2%) but simultaneously leaves a bit more noisy white area (up to 0.7%). MFDU behaves in

Table 3. Denoising of “Cyrillic text”: 20.7% black (B) and 79.3% white (W) pixels (the same notation as in Table 2).

Half-size i	1	2	3	4	5	6	7	8	9	10
Noise: 5% (actual: B 4.98%; W 5.06%); $\theta = 0.137$										
(ξ_i, η_i)	$(1,0)$	$(0,1)$	$(1,-1)$	$(1,1)$	$(2,0)$	$(0,2)$	$(0,5)$	$(0,4)$	$(2,1)$	$(2,-1)$
Entropy	1.0	2.4	3.3	**3.7**	3.5	2.6	1.7	1.1	0.7	0.5
$e_{\bullet}$ . . . . ,%	3.5	2.8	2.3	2.1	2.1	**2.0**	2.1	2.2	2.3	2.4
B,%	12.5	8.2	5.6	5.0	*4.8*	5.0	5.2	5.3	5.3	5.3
W,%	*1.1*	1.4	1.4	1.3	1.4	1.2	1.2	1.3	1.5	1.6
$e_{\bullet}$ . . . . ,%	**4.4**	4.8	5.4	6.0	6.3	6.5	6.6	7.5	8.0	8.4
B,%	*10.1*	11.1	12.5	14.1	16.6	15.6	16.8	19.2	21.8	23.8
W,%	*3.0*	3.1	3.5	3.8	3.6	4.0	4.0	4.4	4.3	4.4
Noise: 10% (actual: B 9.83%; W 10.0%); $\theta = 0.222$										
(ξ_i, η_i)	$(1,0)$	$(0,1)$	$(1,-1)$	$(1,1)$	$(2,0)$	$(0,2)$	$(0,5)$	$(0,4)$	$(2,1)$	$(2,-1)$
Entropy	1.0	2.4	3.2	3.7	**3.7**	3.0	1.9	1.1	0.7	0.5
$e_{\bullet}$ . . . . ,%	6.1	5.0	4.5	4.2	4.2	**4.0**	4.2	4.5	4.9	5.4
B,%	18.3	17.6	11.9	11.4	10.9	10.5	10.5	10.5	10.4	*10.2*
W,%	2.9	*1.7*	2.5	2.3	2.4	2.3	2.5	2.9	3.5	4.2
$e_{\bullet}$ . . . . ,%	7.3	**6.3**	6.5	6.8	7.1	7.3	7.4	8.2	8.6	9.0
B,%	13.8	*13.7*	15.0	16.2	18.6	17.8	18.8	21.1	23.5	25.4
W,%	5.5	4.3	4.3	4.3	*4.0*	4.5	4.3	4.7	4.7	4.7
Noise: 20% (actual: B 19.8%; W 19.9%); $\theta = 0.355$										
(ξ_i, η_i)	$(1,0)$	$(0,1)$	$(1,-1)$	$(1,1)$	$(2,0)$	$(0,2)$	$(0,5)$	$(0,4)$	$(2,1)$	$(2,-1)$
Entropy	1.0	2.3	3.2	3.4	**3.5**	3.5	2.2	1.2	0.7	0.5
$e_{\bullet}$ . . . . ,%	13.0	10.4	9.7	9.4	**9.3**	9.4	10.7	11.6	13.1	15.0
B,%	29.9	28.2	30.0	27.8	26.2	25.6	25.6	23.5	21.3	*20.4*
W,%	8.5	5.7	*4.4*	4.6	4.8	5.1	6.7	8.5	10.9	13.5
$e_{\bullet}$ . . . . ,%	15.3	12.0	10.9	10.3	10.2	10.1	**9.9**	10.2	10.5	10.8
B,%	22.4	*21.1*	21.4	22.0	24.0	23.7	24.9	26.9	28.1	29.2
W,%	13.4	9.6	8.2	7.1	6.5	6.4	5.9	*5.7*	5.8	5.9

the opposite way because the larger contexts result in more corrupted black areas (up to 64.5%) while the white background becomes completely noiseless. For the larger noise level, the trends are even more transparent. With the small-size contexts ($K \leq 5$), the overall quality of DUDE is better. But DUDE considerably corrupts the black area and simultaneously reduces the noise in the white area while MFDU less corrupts the former but less clean the latter. For larger contexts ($K > 5$), MFDU becomes more efficient. Anyway, as follows from Table 5, heuristic choice of the context gives slighly worse results than the maximum entropy based one.

For two other images with more balanced ratios of black and white areas, DUDE always outperforms MFDE under the adaptive context. But the difference tends to become low when the noise level increases. Once again, in “Cyrillic text” DUDE starts from an additional corruption of the smaller black area and cleans the white one to the larger extent. Then the corruption of the former decreases (but no lesser than the noise level) whereas the level of noise cleaning in the latter varies only slightly up and down. MFDU gradually corrupts the black area more and more while slightly improves the

Table 4. Denoising of "Brodatz's Cane": 66% black (B) and 34% white (W) pixels (the same notation as in Table 2).

Half-size i	1	2	3	4	5	6	7	8	9	10
Noise: 5% (actual: B 4.98%; W 4.99%); $\theta = 0.108$										
(ξ_i, η_i)	$(0,1)$	$(1,0)$	$(1,-1)$	$(1,1)$	$(0,2)$	$(1,2)$	$(1,-2)$	$(2,0)$	$(2,-1)$	$(2,1)$
Entropy	1.0	2.4	**3.3**	2.9	2.0	1.4	1.1	0.9	0.7	0.6
$e_{\bullet\ldots}$,%	2.2	1.8	1.6	**1.4**	1.5	1.7	2.0	2.2	2. 5	2.7
B,%	1.7	1.6	*1.0*	1.1	1.2	1.4	1.6	1.8	2.1	2.2
W,%	3.1	2.1	2.7	*2.0*	2.1	2.2	2.6	2.9	3.2	3.4
$e_{\bullet\ldots}$,%	2.2	**1.9**	2.2	2.5	2.8	3.4	4.0	4.5	5.0	5.3
B,%	2.0	1.0	0.7	0.6	0.5	0.6	0.6	0.5	0.5	*0.5*
W,%	*2.7*	3.4	4.8	6.1	7.2	8.9	10.4	12.0	13.4	14.4
Noise: 10% (actual: B 9.76%; W 9.79%); $\theta = 0.200$										
(ξ_i, η_i)	$(0,1)$	$(1,1)$	$(1,-1)$	$(1,1)$	$(0,2)$	$(1,2)$	$(2,0)$	$(1,-2)$	$(2,-1)$	$(2,1)$
Entropy	1.0	2.5	3.3	**3.3**	2.4	1.6	1.1	0.8	0.6	0.5
$e_{\bullet\ldots}$,%	4.9	3.5	3.2	**2.7**	2.8	3.5	4.1	5.0	5.8	6.5
B,%	4.2	2.2	1.9	*1.6*	1.8	2.5	3.1	4.1	5.0	5.9
W,%	6.3	6.1	5.7	4.7	*4.5*	5.3	5.8	6.6	7.2	7.5
$e_{\bullet\ldots}$,%	4.9	3.5	**3.4**	3.5	3.9	4.4	4.8	5.1	5.6	5.9
B,%	4.6	2.4	1.7	1.6	1.3	1.2	1.2	1.0	1.1	*1.0*
W,%	5.6	*5.5*	6.6	7.3	8.8	10.4	11.6	12.9	14.2	15.3
Noise: 20% (actual: B 19.7%; W 20.1%); $\theta = 0.317$										
(ξ_i, η_i)	$(0,1)$	(1.0)	$(1,-1)$	$(1,1)$	$(0,2)$	$(1,2)$	$(1,-2)$	$(2,0)$	$(2,-1)$	$(2,1)$
Entropy	1.0	2.4	3.2	**3.5**	3.1	1.9	1.1	0.7	0.4	0.3
$e_{\bullet\ldots}$,%	13.2	9.6	8.6	**7.6**	8.0	9.7	12.1	14.8	16.8	18.3
B,%	11.8	7.7	5.0	*3.9*	5.0	6.7	9.6	13.1	15.8	17.7
W,%	15.5	13.0	15.3	14.5	*13.7*	15.1	16.6	17.8	18.3	19.1
$e_{\bullet\ldots}$,%	13.2	9.6	8.1	7.1	**6.7**	6.7	6.9	6.9	7.2	7.5
B,%	12.4	8.3	6.0	5.0	3.8	3.5	3.2	2.7	2.6	*2.4*
W,%	14.2	11.9	11.9	*11.0*	12.0	12.9	13.9	14.9	15.9	17.3

Table 5. Denoising of "Stars": the nearest symmetric 8-neighbourhood.

Noise	θ	Residual noise					
		DUDE			MFDE		
		$e_{\bullet\ldots}$,%	B-noise,%	W-noise,%	$e_{\bullet\ldots}$,%	B-noise,%	W-noise,%
5%	0.096	0.08	3.28	0.07	0.08	23.0	0.01
10%	0.178	0.26	29.0	0.18	0.21	29.0	0.12
20%	0.362	0.43	73.8	0.22	2.08	35.5	1.98

noise removal from the white area when the context size is growing up. If the sizes of the black and white areas are closer ("Brodatz's Cane"), these differences as well as the differences in the total quality of denoising decrease, and at lower noise levels DUDE outperforms MFDE while for the higher noise MFDE turns to be slightly better. Both the denoisers in this case reduce about 60–70% of the initial noise.

Comparisons to a heuristic context selection show that in the most cases the adaptive choice returns better contexts. It is worthy to mention that the best contexts are not necessarily continuous. For “Stars” and “Cyrillic text”, the window size to compare possible pixel pairs almost does not impact the choice of the very first characteristic neighbours. The characteristic context for the periodic texture “Brodatz’s Cane” estimated in the smaller window 13×13 differs from the context in the larger window 81×81 (see Fig. 4). It should be noted that under the manually selected best threshold θ, the latter disjoint context results in even slightly better denoising. But when the thresolds are estimated from the noisy image itself, the contexts for the smaller windows give better results.

4 Conclusions

These and other similar experiments show that, in principle, the contexts for DUDE may be chosen using the maximum likelihood structure of pairwise signal dependences in noisy binary images. If the noiseless black and white area are not considerably unbalanced, the active DUDE with the adaptively chosen context mostly outperforms the more conventional passive MFDE. But the higher the noise, the smaller the difference between both the denoisers. The size of the context can roughly be related to the maximum entropy of the distribution of the minimum relative frequencies of signals with the same contexts (these frequencies are used for DUDE itself, too).

When denoising an image, MFDE always additionally corrupts the non-dominant area in favour of cleaning the dominant one. DUDE is also biased to the latter area but corrupts it to the lesser extent.

References

1. M. Berman: Automated smoothing of image and other regularly spaced data. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, 1994, 460–648.
2. A. Blake and A. Zisserman: *Visual Reconstruction*. MIT Press: Cambridge, MA, 1987.
3. S. Geman and D. Geman, Stochastic relaxation, Gibbs distribution, and Bayesian restoration of images. *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 6, 1984, 721–741.
4. G. L. Gimel’farb: *Image Textures and Gibbs Random Fields*. Dordrecht: Kluwer Academic, 1999.
5. L. Rudin, S. Osher, and E. Fatemi: Nonlinear total variation based noise removal algorithm. *Physica*. vol. 60D, 1992, 259–268.
6. T. Weissman, E. Ordentlich, G. Seroussi, S. Verdu, and M. Weinberger: *Universal Discrete Denoising: Known Channel*. Technical Report HPL-2003-29. IT Research, HP Labs Palo Alto, February 10, 2003.

Pose Estimation from Airborne Video Sequences Using a Structural Approach for the Construction of Homographies and Fundamental Matrices

Eckart Michaelsen[1] and Uwe Stilla[2]

[1] FGAN / FOM, Gutleuthausstr. 1, 76275 Ettlingen, Germany
mich@fom.fgan.de
http://www.fom.fgan.de
[2] Photogrm. & Rem. Sens. / TU Munich, Arcisstr. 21, 80333 München, Germany
stilla@bv.tum.de
http://www.bv.tum.de

Abstract. A structural knowledge-based search method is utilized for the estimation of geometric transforms from airborne video sequences. Examples are projective planar homographies and constraints such as the fundamental matrix. These estimations are calculated from correspondences of interest points between two images. Different approaches are discussed to cope with the problem of outlier-correspondences. To ensure any-time performance the search process is implemented in a data-driven production system. The pose estimation from planar homographies is compared to estimations from fundamental matrices. A fusion of both approaches is proposed. The image processing is performed by bottom-up structural analysis using an assessment-driven control. Examples are from the thermal spectral domain.

1 Introduction

Pose trajectory estimation from moving cameras is an important task for scene reconstruction as well as navigation. Research in this field was stimulated by development of mobile autonomous robots. Particularly, methods using projective geometry were utilized [3][6][9]. Recently, unmanned aircraft equipped with video cameras are gaining increased attention for civil as well as military applications like traffic monitoring [16] or surveillance tasks. The appearance of a scene viewed from an aircraft depends on the flight altitude and the height of the sensed objects. If this ratio is large, the scene will appear flat. This implies a different approach than a spatial scene.

Flat scenes are treated by planar homographies. These may be estimated by e.g. minimizing the sum of absolute errors [1]. Given a Gaussian distribution on the displacements of the corresponding image positions it can be shown that the minimization of the sum of the squared errors is the optimal solution [9]. Actually, the direct linear transform (DLT) methods proposed today minimize an "algebraic" squared error sum that is not identical with the squared displacement error in the 2-d image coordinates. However, it has been shown that this error minimization approximates the Gaussian minimization very closely provided that the coordinates are normalized in a proper way [6]. The main disadvantage of minimization of squared error sums is the sensitivity to the inclusion of outliers into the calculation. An outlier is a correspondence that has been erroneously constructed. It does not follow the distribution assumptions underlying the estimation. Because of its possibly large displacement and particularly because of squaring, it may have a large weight in the computation where it should be neglected. Outliers cannot be avoided if automatic estimation is the task. Therefore so-called "robust" methods are proposed.

Section 2 presents and compares three robust estimation methods to solve the problem of planar homography estimation with DLT squared error sum minimization. The term "outlier" and its meaning in the context of homography estimation from airborne videos is further investigated in Section 3. Section 4 compares the pose estimation from planar homographies to estimations from fundamental matrices. A fusion of both approaches is proposed in Section 5. The image processing is performed by data-driven structural analysis and an assessment-driven control. All example data are taken from the thermal spectral domain to ensure independence of the daylight.

2 Robust Estimation of Planar Homographies

Robust estimation methods may be classified into approaches that assume the existence of mutually exclusive sets of inliers and outliers (Section 2.2) and others that assign weights to the correspondences (Section 2.1).

2.1 Iterative Re-weighting Least Squares (IRLS)

An example of the assigning of weights to the correspondences is iterative re-weighting least squares [7]. The inverse of the residual of the least squares solution of each correspondence of the complete sample is used to re-weight its influence. Correspondences yielding a large residual error will be punished and correspondences yielding a small error will gain more influence. If a large portion of the correspondences is expected to be wrong, a local minima problem may occur. The convergence of IRLS to the desired minimum is theoretically not guaranteed. It may end up with zero-error and thus infinite weight on an arbitrary minimal sample and random small weights on all other members. However, in our examples we found that it does converge slowly but robustly to a good solution. IRLS-estimation of 2D-homographies is available in public code libraries [17]. Proposals are made for the handling of occlusion outliers and lighting changes within the IRLS-method [8].

2.2 Random Sample Consensus

The standard method for inliers-outliers discrimination is the random sample consensus approach (RANSAC) [4]. The calculation is performed on minimal samples which are randomly picked from the complete sample of correspondences. The result of the calculation is tested on all the other correspondences giving a residual error. If this error is smaller than a threshold t_s, the correspondence will be termed to be in consensus with the actual sample. After repeating this procedure a sufficient number of times the search is terminated. The termination criterion bases on a minimum size of the current best consensus m and a maximal number of cycles n_c. There is an elaborate theory for the choice of these parameters (t_s, m, n_c) from the expected portion of outliers, a standard deviation of the error of the position of inliers, and a significance level [6]. The sample with the highest consensus is chosen and the corresponding consensus set is used to determine the estimation by mean squared error minimization.

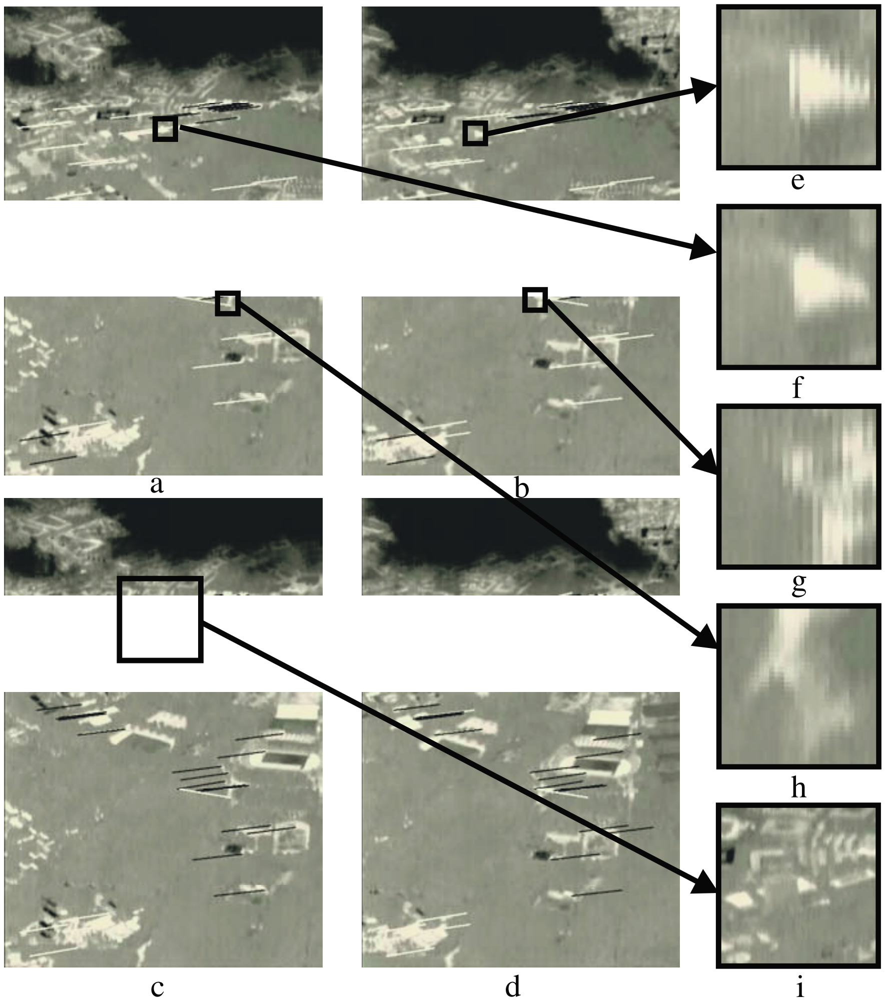


Fig. 1. Image pair of a thermal video sequence and corresponding points. a,b) Best GSAC-sample in black, other correspondences in white; c,d) RANSAC-sample with innlier (black) and outlier (white); g,h) incorrect correspondence excluded by both methods; e,f) incorrect correspondence found as RANSAC-inlier; i) greater section around that location showing that it results from partial occlusion.

2.3 Good Sample Consensus

Some authors proposed to modify random sampling by taking also the quality of the samples into account [11], [13]. This obvious idea is not new and has already been touched in the original paper of RANSAC [4]. For such improvements a criterion always has to be defined that assesses the suitability of a sample for the intended calculation. For example the position of the corresponding points within the images will be important for estimating homographies. They should cover as much area of the images as possible. Moreover, more than two collinear points should be avoided. Fig. 1 shows images with large homogenous regions and the structure concentrated in few regions. Because RANSAC only counts the number of mutually consistent correspondences, it may concentrate too much on densely structured regions and tend to under-estimate the importance of good but rather isolated correspondences elsewhere, e.g. the correspondences in the lower left corner that are missing in the inliers set of RANSAC.

In GSAC the assessment criterion is used to control the search for a good sample on which the solution is based. The correspondences in the lower left corner are now included. Section 5 explains how GSAC can be implemented by a structural method.

3 Classification of Correspondences

1) Correspondences Consistent with the Homography Model: A correspondence between structures in two different images will be called *correct* if the location on the object in the scene that caused them is the same. Additionally they must fulfill the model constraint. For homography estimation only those objects that are located on the assumed plane can cause correct correspondences. In urban terrain this plane will be at the average height of the buildings. Of course, we will have to tolerate small deviations from this constraint. The residual error will mainly result from the localization error of the 2d-structures and may be modeled as normally distributed.

2) Correspondences Consistent with a More General Geometric Model: In an urban area there may be tall buildings that are jutting out of the plane. Corresponding structures resulting from the roofs of such tall buildings will violate the homography constraint. Still, they may be correct in the sense that they come from the same physical property. Their deviation from the homography follows a different rationale: They will be located close to the epipolar line which goes through the point determined by the homography and through the epipole. They should be excluded from the estimation of the scene plane, but they may be included into the estimation of the camera rotation and epipole.

3) Correspondences from Moving Objects: Video sequences taken by a moving sensor yield image pairs that were obtained at different time instants. Moving objects in the scene may cause semantically correct correspondences that neither follow the planar homography nor the epipolar constraint. However, such correspondences from moving objects are required for applications like traffic monitoring. Fig 2 shows such a correspondence resulting from a moving vehicle. The correspondence is indicated as a white line, while other correspondences are drawn in black.

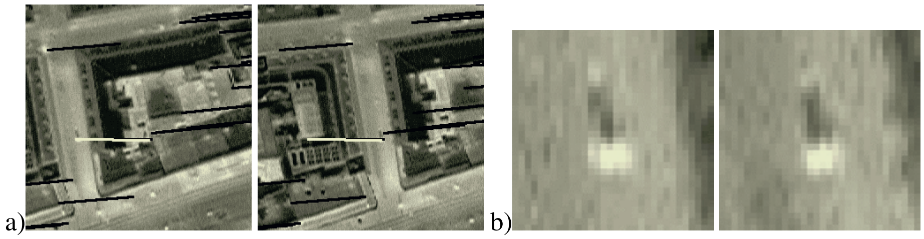


Fig. 2. Nadir looking sequence taken from urban terrain; example of a correct correspondence that is an outlier to the epipolar constraint estimation (a moving car on the ground); a) larger sections with surroundings, b) corresponding structures.

4) False Correspondences: Using an automatic method to construct the correspondences we cannot avoid the handling of semantically false correspondences. Most often these will result from occlusion phenomena. Fig. 1e and Fig. 1f show an example, where the outlines of a warm flat building roof (white) are partially occluded by a tall building in front of it (grey) – compare Fig. 1i. There is a structure correspondence located on the T-junctions caused by this occlusion. Such correspondences do not follow any predictable error function. They may accidentally be inside the error bounds of a homography estimation like the RANSAC-estimation displayed in Fig. 1a and Fig. 1b.

4 Robust Estimation of Epipolar Constraints

A central theorem of projective geometry states that from a pair of views of a scene the mutual orientation and translation of the cameras can be calculated from at least seven corresponding point-pairs *(x, x')* and that the position of the corresponding points in the 3d scene also follows from this reconstruction [6]. This is a constructive argument that is based on the inference of the fundamental matrix *F* from the correspondence data. This matrix formulates the epipolar constraint by stating $x^T F x' = 0$. It can also be estimated from this simple linear equation using at least eight correspondences. Such estimation is depicted in Fig. 3e and Fig. 3f. Problems with instability of the solution will occur, if all the correct correspondences are located in one plane. This happens in flat terrain. For testing this automatically, samples that are inconsistent with the homography model (see Section 3 class 2) are searched. Fig. 3e shows a sample of correspondences that gives the epipolar constraint depicted in Fig. 3f. But, this sample is smaller than the best GSAC-sample for the homography displayed in Fig. 3c. Moreover, in forward looking situations like the one presented in Fig. 3 the epipole (black/white cross) may be inside the frame, and scene reconstruction is impossible for the area around the epipole.

The effort for a random search for suitable minimal samples (containing seven or eight correspondences) is a rising polynomial with degree seven or eight with growing portion of outliers. Therefore, some authors introduced an intermediate part-of hierarchy into the samples [2]. Others propose a “plane plus parallax” approach [14]. First, one estimates a homography *H* that maps those points located on a dominant

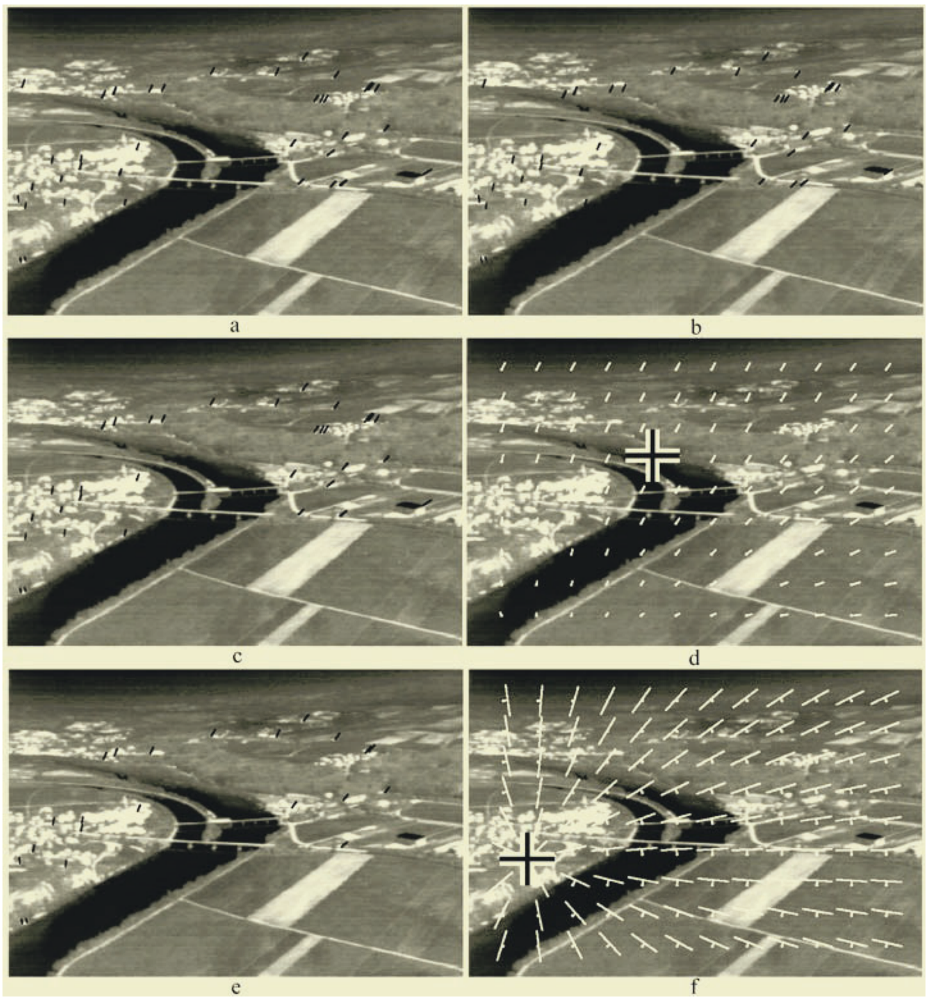


Fig. 3. Example of estimations from a forward looking oblique sequence; a) all correspondences on one frame b) same on other frame c) GSAC consensus correspondence set for homography estimation d) homography displayed as vectors on a grid array e) consensus correspondence set for epipolar estimation f) epipolar constraint; for each point on the grid the epipolar line is indicated and connected to the point.

plane from one image to the other *Hx=x'* (Section 2). Then the homography can be decomposed in the form $H=R-nt^T$. *R* is the camera rotation matrix and the outer product nt^T results from the plane normal n and the camera translation *t* [3].

If we use normalized camera coordinates the 3-d vector *t* can also be interpreted as epipole. Multiplying the skew-matrix constructed from this vector with the rotation matrix *R* will give the fundamental matrix *F* belonging to the image pair. This matrix estimation for the epipolar constraint may then be refined using additional correspondences of the type mentioned in Sect. 3 class 2). In Fig. 3d the estimated homography is presented as a white vector field and the calculated epipole. In spite of the considerably non-planar structure of the valley scene this turns out more stable than the direct epipolar constraint estimation. Particularly, the difference between the epipole estimations in Fig. 3d and Fig. 3f is considerable. A small rotation of the camera combined with such a displacement of the epipole give roughly the same point movements for forward looking geometries.

5 Production Nets for GSAC-Estimationof Geometric Entities

The pose estimation is partitioned into several steps and intermediate results. The overall structure of the process can be depicted by a so-called production net (Fig. 4). This bipartite graph contains productions and concepts (object types) as nodes. Arcs go from an object concept to a production whenever the objects are input to the production. Arcs go from a production to a concept whenever these concepts are constructed by the production. The productions contain constraints that incoming objects must fulfill to fit into the construction of the out-going objects of a higher concept. They also contain the functions that are necessary to construct these objects. The constructive part also contains an assessment part that evaluates the newly built object. Details of the control mechanism have been published in [15]. The assessment criteria used here are named in Fig. 4.

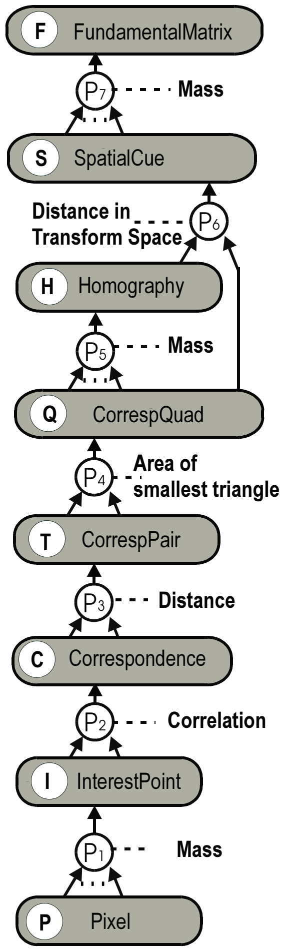


Fig. 3. Production-net with assessment criteria for bottom-up data-driven control.

The processing starts with the application of an interest operator on the images that marks locations where neither homogeneity nor an aperture problem is likely [5]. Pixels trespassing a threshold form the primitive objects **P** of the structural analysis. Production p_1 groups such primitive objects into interest objects **I** using vicinity as its constraint, center of gravity as its function and total mass for its assessment. The position of such an interesting location **I** is determined with sub-pixel-precision. Given such an object production p_2 will search the other image for corresponding partners. It will construct new objects correspondence **C** for all such objects and assess them by means of correlation. Each such object may vote for a translation transform. Production p_3 forms objects **T** of a pair of correspondence objects **C**. Since they may be used to vote for similarity transforms, these objects are assessed according to the distance between the two locations in the image. Large distances give more precision for such estimation. Production p_4 gathers two such objects **T** and forms a quadruple object **Q** from them.

From these a cue to the homography can be calculated. It is not only important that all four correspondences in such an object **Q** must be inliers of the type discussed in Section 3 class 1). Also no three of the four points are allowed to be collinear. They should cover as much area as possible. Therefore the area of the smallest of the four triangles in the quad is chosen as the assessment criterion. Production p_5 clusters the homography estimations from several consistent objects **Q**. The result is a new object **H** that is calculated via DLT squared error sum minimization from the sample of correspondences preceding the objects **Q** in the cluster. Thus Productions p_4 and p_5 implement the GSAC-rationale outlined in Section 2.3. An object **H** is assessed not only according to the number of correspondences in it, but also according to the assessment of the preceding objects **Q.** Production p_6 searches well directed for the outliers of the cluster process implemented by Production p_5. There may be correspondences in them that belong to the type described in Section 3 class 2). This results in spatial cue objects **S**. Such objects contain a fundamental matrix estimation. They are assessed according to the inconsistence of the homographies preceding them. Of course such cues need affirmation because it may result from correspondences of the types discussed in Section 3 classes 3) and 4). Production p_7 clusters consistent objects **S** into a well founded fundamental matrix estimation object **F** where the calculation is based again on DLT with the sample of the preceding correspondences.

The control scheme forms hypothesis of each newly constructed object and all the productions to which an arc goes from its type. These hypotheses get a priority according to the assessment of the object. All hypothesis compete for computational resources. In this manner homographies and fundamental matrices are already estimated from prominent and well positioned correspondences while other less important interest point objects still wait for an opportunity to search for correspondences in the other image. The process may be terminated at any time followed by choosing the best object **H** or **F** obtained up to this time instance according to the same assessment criteria.

6 Conclusion

Camera pose estimation using fundamental matrices as well as planar homographies can be obtained from the same images. The decision of which method should be preferred depends on the situation. Intermediate results give criteria for the choice. For a selected method the best sample of correspondences has to be searched. A structural knowledge-based approach combines both methods, uses well directed search and avoids early decisions. During the search run weights are assigned to entities like correspondences between structures in different images, pairs of such correspondences, quadruples and larger sub-sets. Each intermediate result is evaluated and the control of the whole system is based on these evaluations. Thus spurious calculations are avoided. Originally the production net approach has been invented for dealing with costly object recognition tasks in an accumulative way using affirmative intermediate results [10]. In pose estimation intermediate results may also be mutually competing. Thus, the assessments are a key issue in balancing such system. Tests comparing GSAC homography estimation performance to RANSAC and IRLS on a collection of example data are under way and will be published in [12].

References

1. Ben-Ezra, M., Peleg, S., Werman, M.: Real Time Motion Analysis with Linear Programming. CVIU, Vol.78, (1999) 32-52.
2. Chum, O., Matas, J., Obdrzalek, S. Epipolar Geometry from Three Correspondences. CVWW'03, Vatlice, Czech Republic, (2003).
3. Faugeras, O.: Three-Dimensional Computer Vision. MIT Press, Cambridge, Mass, (1993).
4. Fischler, M. A., Bolles, R. C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. Assoc. Comp. Mach., Vol. 24, (1981) 381-395.
5. Foerstner, W.: A Framework for Low Level Feature Extraction. In: Eklundh J.-O. (ed). Computer Vision – ECCV 94. Vol. II, B1, (1994) 383-394.
6. Hartley, R., Zisserman A.: Multiple View Geometry in Computer Vision. Proc. Cambridge University Press, Cambridge, (2000).
7. Holland, P. W., Welsch, R. E.: Robust regression using iteratively reweighted least-squares. Comm. Statist. Theor. Meth., Vol. 6 (1977) 813-827.
8. Jurie, F., Dhome, M.: Real Time Robust Template Matching. BMVC-2002 (2002) 123-132.
9. Ma, Y., Soatto, S., Kosecka, J., Sastry, S.: An Invitation to 3-D Vision. Springer, Berlin, (2000).
10. Michaelsen E., Stilla U.: Probabilistic Decisions in Production Nets: An Example from Vehicle Recognition. In: Caelli T., Amin A., Duin R. P. W., Kamel M. Ridder D. de (eds): Structural, Syntactic and Statistical Pattern Recognition SSPR/SPR 2002, LNCS 2396, Springer, Berlin (2002) 225-233.
11. Michaelsen E., Stilla U.: Good Sample Consensus Estimation of 2d-Homographies for Vehicle Movement Detection from Thermal Videos. In: Ebner H., Hiepke C., Mayer H., Pakzad K. (eds.): Photogrammetric Image Analysis PIA´03. Intern. Arch. of Photogr. and Rem. Sens., Vol. 34, Part 3/W8 (2003) 125-130.
12. Michaelsen E., Stilla U.: Sensor Pose Inference from Airborne Videos by Decomposing Homography Estimates. Accepted for ISPRS 2004, Commission III, WG III/I (2004)
13. Torr P. H. S., Davidson C.: IIMSAC: Synthesis of importance sampling and random sample consensus. IEEE – PAMI, Vol. 25, no. 3 (2003) 354-364.
14. Sawhney, H. S.: Simplifying motion and structure analysis using planar parallax and image warping. ICPR 94, IEEE-Press, Los Alamitos, Ca., Vol. I (1994) 403-407.
15. Stilla U., Michaelsen E., Lütjen K.: Automatic Extraction of Buildings from Aerial Images. In: F. Leberl, R. Kalliany, M. Gruber (eds.), Mapping Buildings, Roads and other Man-made Structures from Images, IAPR-TC7, Wien, Oldenburg, (1996) 229-244.
16. Sujew, S., Ernst, I.: LUMOS Airborne Traffic Monitoring. In: Int. Workshop on Airborne Traffic Measurement, DLR, Berlin, (2003).
17. Robust Estimation Library, rrel, university of Manchester (accessed 24 Dec. 2003), http://paine.wiau.man.ac.uk/pub/doc_vxl/contrib/rpl/rrel/html/

Temporal Post-processing of Decision Tree Outputs for Sports Video Categorisation

Edward Jaser, William Christmas, and Josef Kittler

Centre for Vision, Speech and Signal Processing, University of Surrey
Guildford GU2 7XH,UK
Tel.: +44 (0)1483 689294, Fax: +44 (0)1483 686031
{E.Jaser,J.Kittler,W.Christmas}@eim.surrey.ac.uk

Abstract. In this paper, we describe a multistage decision making system to deal with the problem of automatic sports video classification. The system is founded on the concept of cues, i.e. pieces of visual evidence, characteristic of certain categories of sports that are extracted from key frames. The main decision making mechanism is a decision tree which generates hypotheses concerning the semantics of the sports video content. The final stage of the decision making process is a Hidden Markov Model system which bridges the gap between the semantic content categorisation defined by the user and the actual visual content categories. The latter is often ambiguous, as the same visual content may be attributed to different sport categories, depending on the context. We tested the system using two setups of HMMs. In the first, we construct and train an HMM model for each sport. A post-processing step is needed in this setup to combine the outcomes of the individual HMMs. In the second setup, we eliminate the need for post-processing by constructing a single HMM with each node representing one of the sports we want to detect. Comparing the results obtained from both setups showed that a single HMM delivered the better performance.

1 Introduction

In this paper we consider the problem of automatic sports video categorisation. This problem arises during multidisciplinary events such as Olympic Games where huge volumes of video material are recorded, with the content randomly switching from one discipline to another. A coarse automatic annotation in terms of sport identity would aid the production of event summaries for news cast and other applications.

Much research in the field of multimedia analysis and retrieval is targeting the domain of sport videos. The reason is that most sport videos have a well-defined content structure and official rules and procedures compared to videos from other domains. Moreover, most sporting events take place in one location. That means only a limited number of cameras are needed to cover the play area and capture the event. Therefore, a set of characteristic views recorded by those cameras can be defined and associated with the events. Figure 1 gives an

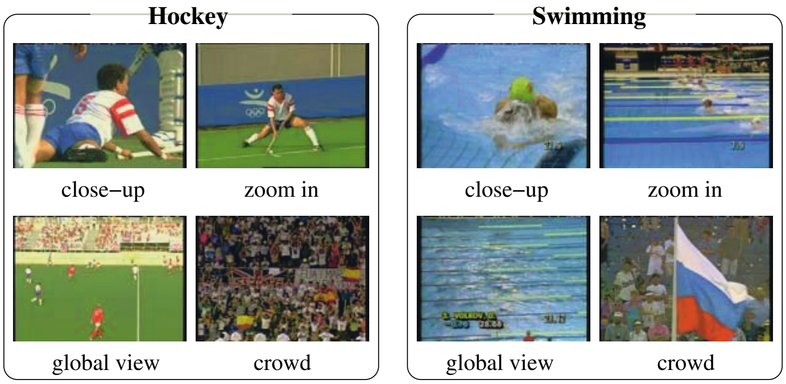


Fig. 1. Sport views

example of wide variety of views that exist in two sport disciplines, swimming and hockey.

Other work, specific to some form of sports annotation, include [3] in which the authors addressed the problem of parsing the content of football video programs. They used domain knowledge about football to construct an a priori model to aid the task of classifying key-frames of each shot to a predefined set of events. Xu et al [12] also addressed the problem of segmenting football videos into two basic semantic units "play" and "break". Addressing football videos analysis and summarisation as well, Ekin et al [2] proposed a fully automated system using both cinematic and object-based features. Chang et al [1] proposed a statistical method for the automatic extraction of predefined highlight segments in a baseball game video using an HMM built for each class of highlight. HMMs were also used by [5] for tennis scene classification and segmentation. An HMM was used to fuse audio and visual information. They also used HMMs to model tennis syntax and the hierarchical structure of a tennis match.

In this paper we propose a multistage decision making system that is founded on the concept of visual cues — pieces of visual evidence, characteristic of certain categories of sports that are extracted from key frames. The main decision-making mechanism is a decision tree which generates hypotheses concerning the semantics of the sports video content. The final stage of the decision making process is an HMM system which bridges the gap between the semantic content categorisation defined by the user and the actual visual content categories. Two setups of HMM are considered. In one setup, we constructed and trained an HMM for each sport investigated in our research. This setup is motivated by the fact that each HMM corresponding to a certain sport is constructed and trained independently. However, using this setup, a post-processing is needed to combine the outcomes of the individual HMMs to reach a final decision. In the other setup, a single HMM is constructed with each node representing one of the sports investigated. This setup has the advantage that it requires no post-processing. To reach a decision using this setup, we need to find the single best state sequence for the given observation sequence.

The paper is organised as follows. In Section 2 we give an overview of the system. We briefly describe the cue concept and how to generate cues deemed indicative of sport types in Section 3. Section 4 describes the process of generating sports video content hypotheses using decision trees. The post-processing of the decision tree outputs using two setups of HMMs is discussed in Section 5. The results of experiments designed to test and compare the two HMM setups is presented in Section 6. The paper is concluded in Section 7.

2 System Overview

In this section we give an overview of the system (Figure 2) and describe its various elements. Our goal in this paper is, given a video stream that contains sports material from one or more disciplines, to automatically segment the stream into sequences and label each sequence with the corresponding sport label.

First, the video stream is segmented to shots which are the basic temporal units in our system. For each shot, a number of key frames are extracted. The first stage of the decision-making process is the cue detection. Cue detectors operate on the key frames and generate judgement about the presence or the absence of the objects they try to detect. The shot after this stage is represented by the cues. This is distinct from the conventional approaches which are based on low level generic image features derived from colour and texture. Cues offer higher level representation which is application domain specific. Most importantly, they transform diverse input data structures into a standard form which facilitates the decision making process and promotes modularity (i.e. exploiting additional cues). Examples of cues could be: grass, sky, swimming pool lanes.

The second stage classifies each shot to one of the characteristic views, defined for each sport, using the information provided by the cue detectors. The functionality of this stage is realised by a decision tree classifier. The knowledge embodied in the decision tree is learnt from a set of labelled training samples covering all views the system is trying to detect. The decision tree is then used to classify each shot into one of the sport view categories.

The output of the decision tree may be subject to error due to either errors in the cue extraction or genuine ambiguity, i.e. the presence of cues that are characteristic of more than one discipline (e.g. crowd views). The third stage is designed to minimise this error by exploiting the temporal context using HMMs. HMMs, which process the sequence generated by the decision tree, bridge the gap between the semantic video content labelling by human observer and the data-driven hypotheses generated by automatic classification methods.

The individual stages of the system are described next in more detail.

3 Cue Detectors

In much of the previous work in automatic annotation of video material, the annotation consisted of the output of various feature detectors (i.e. MPEG7 descriptors). By itself, this information bears no semantic connection to the actual

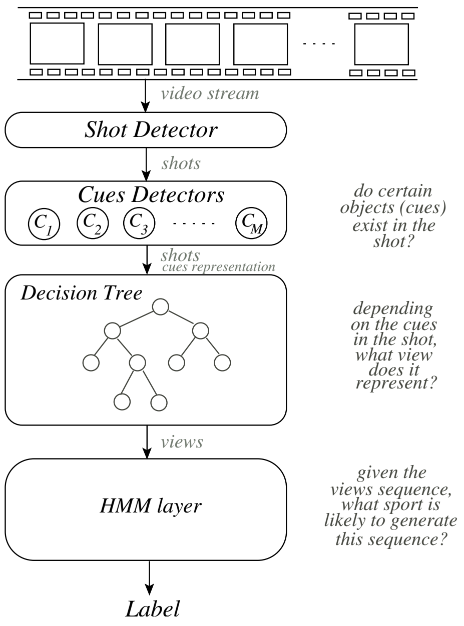

video stream
Shot Detector
shots
Cues Detectors
C_1 C_2 C_3 ..... C_M
do certain objects (cues) exist in the shot?
shots
cues representation
Decision Tree
depending on the cues in the shot, what view does it represent?
views
HMM layer
given the views sequence, what sport is likely to generate this sequence?
Label

Fig. 2. Proposed System

scene content — it is simply the output of some image processing algorithm. The cue detection approach [6] is taking the process one stage further. In this approach, the connection between low-level image data outputs and the semantics of the scene content can be defined by means of a set of training processes. Thus for example the system can be trained to associate the output of a texture feature detector with crowds of people in the scene. This mechanism can then be used to generate confidence values for the presence of a crowd cue in a scene, based on the scene texture. Different cues can then be combined to generate higher-level information, e.g. the type of sport being played. Figure 3 illustrates the cue generation process which involves three phases. Different cue detection methods have been developed [9, 8, 7]. Each method can be used to form a number of different cue-detectors provided that suitable training data is available.

Let us suppose that we have a set of M trained cue-detectors. Each cue detector operates on the key frame images and generates two pdf values $p(x|C)$ and $p(x|\bar{C})$, where C is the cue looked for by the cue detector and x denotes the measurement vector used. Assuming equal prior probabilities, we can estimate the a posterior probability $P(C|x)$ of an instance of a cue, C, existing in the image as follow:

$$P(C|x) = \frac{p(x|C)}{p(x|C) + p(x|\bar{C})} \tag{1}$$

Thus, for each key frame, we will obtain k values, one for each cue. A shot can then be represented by a vector $S = (\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}j, ..., \mathcal{C}_M)$ where $\mathcal{C}_j$ is the mean

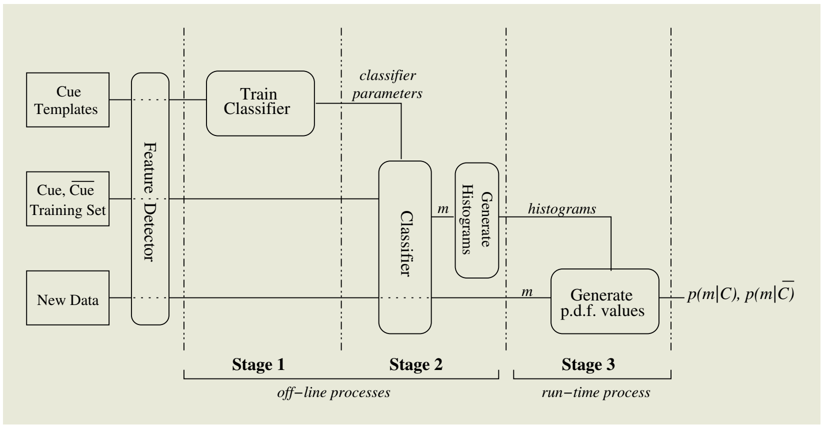


Fig. 3. Creating Cue Evidence

value of the posterior probabilities computed by the j^{th} cue-detector on the key frames that belong to the shot.

4 View Classification

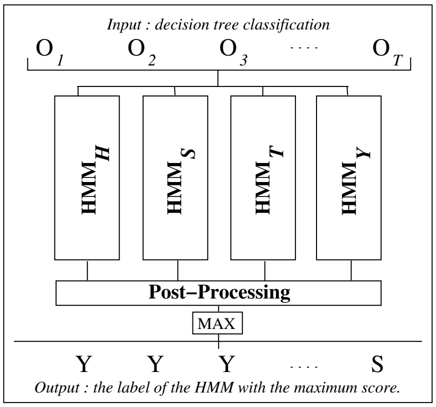
Based on earlier experience [4], we opted for a decision tree learning algorithm to build the model for solving the problem of classifying a shot to one of a set of predefined sport views. The “C4.5” algorithm [10] was adopted. The process of constructing a decision tree classifier requires a set of training examples. Each example is represented by the cue vector, S. A class label, which is typically a camera view, is attached to the examples. A splitting criterion, Information impurity, is used to recursively partition the training set in a way that increases the homogeneity of its partitions. The partitioning stops when one of the stopping rules is triggered at a node. This node becomes a class node and a label which represents the sport view with the largest number of shots is attached to it.

The classification of a shot using decision trees proceeds from top to bottom. Depending on its cue values, S navigates through the decision tree till it reaches a class node. The navigation is guided by the rules of the decision nodes visited. The shot is assigned the label attached to the class node at which its navigation terminates.

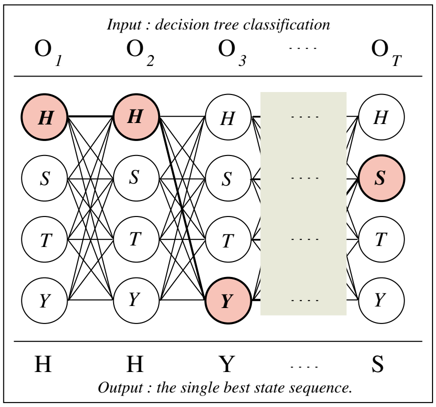
5 Post-processing Using HMMs

The HMM (described in detail in [11]) is a powerful tool, widely used in pattern recognition. In this paper, HMM is employed to minimise the ambiguity of classifying using a decision tree by exploiting the temporal context. HMMs bridge the gap between the semantic video content labelling by human observer and the data-driven hypotheses generated by automatic classification methods.

Two HMM setups are considered (Figure 4). In the first setup, we construct and train a separate HMM model for each sport we want to detect. In this setup, the forward algorithm can be used to compute the likelihood that an observation sequence is emitted by an HMM. Figure 5(a) shows the output of four HMMs operating on a sequence. Note that the output of HMM corresponding to the sport of the subsequence will exhibit the least change of all of the HMMs. To exploit this, we compute the discrete derivative of each HMM output and smooth the result with a Gaussian kernel (Figure 5(b)). The subsequence is labelled with the identity of this HMM.


Input : decision tree classification
O1 O2 O3 .... OT
HMM_H
HMM_S
HMM_T
HMM_Y
Post–Processing
MAX
Y Y Y .... S
Output : the label of the HMM with the maximum score.

(a) Setup 1 : four competing HMMs


Input : decision tree classification
O1 O2 O3 .... OT
H S T Y
H H Y .... S
Output : the single best state sequence.

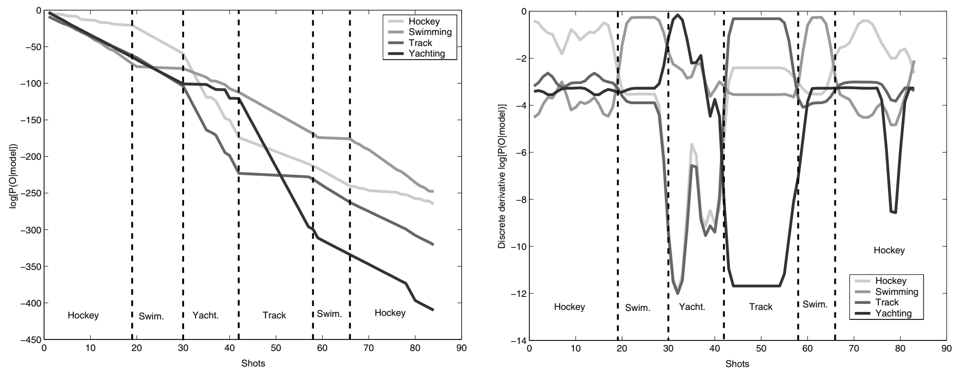
(b) Setup 2 : single HMM

Fig. 4. The two HMM setups considered in our system

In the second setup, we construct a single HMM with each node in this HMM representing one of the sports we want to detect. In this setup, the problem we need to solve is to find the single best state sequence given the observation sequence generated by classifying a sequence of shots using the decision tree classifier. The Viterbi algorithm is used to realise the most likely sequence state.

6 Experimental Results

In this section, we describe the experiments to evaluate the proposed system, and compare the results obtained from applying the two HMM setups. The experimental data is taken from video material from the 1992 Barcelona Olympic Games. The material includes four Olympic sport disciplines (hockey (H), swimming (S), track_events (T), yachting (Y)). The material was manually groundtruthed and split into three sets. One set was reserved for training the decision tree classifier; the second set was used for HMM training, and the remaining set was reserved for testing the system. Thirty-seven visual cues were identified, trained and used to generate cue evidence for the study.



(a) Output of the four competing HMM (b) After post-processing

Fig. 5. Output from four HMMs operating on a sequence generated by the decision tree classifier

Table 1. Confusion matrix for sports shot classification of the proposed system using four competing HMMs (Setup 1)

	H	S	T	Y	Recall	Precision
H	**522**	23	30	3	90.3%	87.0%
S	30	**909**	11	14	94.3%	93.4%
T	49	11	**469**	0	88.7%	92.0%
Y	0	30	0	**370**	92.5%	95.6%

Table 2. Confusion matrix for sports shot classification of the proposed system using a single HMM (Setup 2)

	H	S	T	Y	Recall	Precision
H	**551**	5	16	6	95.3%	91.4%
S	46	**903**	14	1	93.7%	98.8%
T	6	6	**516**	1	97.5%	94.5%
Y	0	0	0	**400**	100.0%	98.0%

We tested the proposed system on 52 sequences. Table 1, shows the results obtained from the experiments using Setup 1 in which four competing HMMs are used, one for each discipline. The results obtained from experimenting with the proposed system using single HMM, with each state in this HMM represent one of the sports investigated in this paper, are summarised in Table 2.

The proposed system performed well with both setups. The overall recognition rate is 91.87%(standard deviation = 5.08%) for Setup 1 compared to 95.91%(standard deviation = 3.18%) for Setup 2. We performed a t test and the test suggested that the difference between the performance of the two setups was significant statistically. Moreover, using setup 2 proved to be more convenient since it requires no post-processing on the obtained results. As far as the accuracy of classification for individual sports is concerned, we noticed that the hockey, track_events and yachting classification rate using Setup 2 were significantly better than when using Setup 1. Swimming performance, was almost the same in both setups.

One advantage of the proposed system is its ability to segment a sequence comprising more than one discipline and label the subsequences with the corresponding sport label. This information can be used to perform further analysis on any subsequence using specialised model once we know its coarse label.

Doing more analysis on the results, we noticed that just over 70% of the subsequences were correctly labelled and 3% were mislabelled. The boundaries of the remaining 27% of the subsequences, do not exactly correspond to the groundtruthed test data. However, in 59% of the latter cases, the errors in the boundaries are due to ambiguity in the material rather than the classification system, i.e. crowd shot at the beginning or the end of a sport event.

7 Conclusion and Future Work

In this paper, a multi-stage decision-making system for sports video classification was proposed. The first stage of the decision-making process detects application-specific cues. The second stage attaches a label, from a set of prototypical views of each sport, to each shot, using the information provided by the cue detection stage. The functionality of this stage is realised by a decision tree classifier. The third stage uses HMMs to process the sequence of view labels generated by the decision tree. The output of this stage is a final decision regarding the identity of the sport represented by the sequence, taking advantage of the temporal context. We experimented with the proposed system using two setups of HMM, four competing HMMs, one for each discipline, and a single HMM with each node representing a sport. It was noticed from the experiments that using single HMM delivered better results.

Our future plans include providing cues that deal with modalities other than the visual ones. They are expected not only to improve the sports video categorisation performance but also help to detect highlights such as hockey goal, etc. It is intended to use audio, speech and motion cues for this purpose.

Acknowledgements

This work was supported by the IST-2001-34401 VAMPIRE project funded by the European IST Programme.

References

1. P. Chang, M. Han, and Y. Gong. Extract Highlights From Baseball Game Video With Hidden Markov Models. In *IEEE International Conference on Image Processing (ICIP'02)*, 2002.

2. A. Ekin, A. M. Tekalp, and R. Mehrotra. Automatic Soccer Video Analysis and Summarization. *IEEE Transactions on Image Processing*, 12(8):796–807, July 2003.
3. Y. Gong, T.S Lim, and H.C. Chua. Automatic Parsing of TV Soccer Programs. In *IEEE International Conference on Multimedia Computing and Systems*, pages 167 – 174, May 1995.
4. E. Jaser, J. Kittler, and W. Christmas. Building Classifier Ensembles for Automatic Sports Classification. In Roli F Windeatt T, editor, *Proceedings of the 4th International Workshop on Multiple Clasifier Systems (MCS 2003)*, volume 2709 of *Lecture Notes in Computer Science*, pages 366–374. Springer-Verlag, June 2003.
5. E. Kijak, G. Gravier, P. Gros, L. Oisel, and F. Bimbot. HMM Based Structuring of Tennis Videos Using Visual and Audio Cues. In *IEEE International Conference on Multimedia and Expo (ICME)*, volume 3, pages 309–312, July 2003.
6. J. Kittler, K. Messer, W. Christmas, B Levienaise-Obadia, and D. Koubaroulis. Generation of Semantic Cues for Sports Video Annotation. In *Proceedings of the 2001 International Conference on Image Processing (ICIP 2001), Thessaloniki, Greece*, pages 26–29, October 2001.
7. B. Levienaise-Obadia, J. Kittler, and W. Christmas. Defining Quantisation Strategies and a Perceptual Similarity Measure for Texture-Based Aannotation and Retrieval. In *In IEEE, editor, ICPR'2000*, volume III, 2000.
8. J. Matas, D. Koubaroulis, and J. Kittler. Colour Image Retrieval and Object Recognition Using the Multimodal Neighbourhood Signature. In *D Vernon, editor, Proceedings of the European Conference on Computer Vision LNCS*, volume 1842, pages 48–64, 2000.
9. K. Messer and J. Kittler. A Region-Based Image Database System Using Colour and Texture. In *Pattern Recognition Letters*, page 1323 1330, 1999.
10. J. R. Quinlan. *C4.5 : Programs for machine learning*. Morgan Kaufmann, 1993.
11. Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *IEEE*, 77(2):257–286, 1989.
12. P. Xu, L. Xie, S. Chang, A. Divakaram, A. Vetro, and S. Sun. Algorithms and System for Segmentation and Structure Analysis in Soccer Video. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2001.

A New Method of Polyline Approximation

Alexander Gribov and Eugene Bodansky

Environmental System Research Institute (ESRI)
380 New York St., Redlands, CA 92373-8100, USA
{agribov, ebodansky@esri.com}

Abstract. Many methods of a raw vectorization produce lines with redundant vertices. Therefore the results of vectorization usually need to be compressed. Approximating methods based on throwing out inessential vertices are widely disseminated. The result of using any of these methods is a polyline, the vertices of which are a subset of source polyline vertices. When the vertices of the source polyline contain noise, vertices of the result polyline will have the same noise. Reduction of vertices without noise filtering can disfigure the shape of the source polyline. We suggested a new optimal method of the piecewise linear approximation that produces noise filtering. Our method divides the source polyline into clusters and approximates each cluster with a straight line. Our optimal method of dividing polylines into clusters guarantees that the functional, which is the integral square error of approximation plus the penalty for each cluster, will be the minimum one.

Keywords: vectorization, polyline compression, polyline approximation, shape analysis.

1 Introduction

Many methods of a raw vectorization produce lines with redundant vertices. Therefore the results of vectorization usually need to be compressed. In additional to compression, noise filtering usually should be done, because a noise caused by source document distortion, scanning, and binarization may be too big.

Compression methods based on throwing out inessential vertices are widely disseminated, maybe because of their simplicity [1-5]. The result of using any of these methods is a polyline, the vertices of which are a subset of source polyline vertices. If the vertices of the source polyline contain noise, vertices of the result polyline obtained with compression will have the same noise. Reduction of vertices without noise filtering can disfigure the shape of the source polyline. So as we have mentioned before [6], compression methods have to be used cautiously.

Pavlidis and Horwitz [7] suggested a method of the piecewise linear approximation that produces not only compression, but also noise filtering. The source polyline is divided by some vertices that we call "critical points" into polygonal sectors or clusters. The method of least squares is later used for approximating clusters with straight lines. The main problem consists of finding the critical points.

The «split and merge» method suggested by Pavlidis and Horwitz defines the minimum number of critical points such that the maximum deviation of approximation of each cluster with the straight line or the sum of integral square errors of each cluster is not more than a given threshold. The algorithm is suboptimal.

The vertices of the result polyline are intersections of adjacent approximating straight lines. These vertices do not necessary coincide with vertices of the source polyline. The threshold of approximation error controls the precision of approximation. The precision of the result polyline could be better than the precision of the noisy source line if a suitable threshold was selected.

Our algorithm divides the source polyline into clusters and approximates each cluster with a straight line similarly to the aforementioned algorithm. The technique of choosing critical points guarantees that the functional that is the integral square error of approximation plus the penalty for each critical point will be the minimum one.

2 Problem Statement

a. Let p_i, where $i = 0,\dots,n$, are vertices of the source polyline P. Let decomposition vertices p_{q_j} divide P into the set of non-overlapping polygonal sectors, where $q_j, (j = 0,\dots m)$, are the indices of source polyline vertices (see Fig. 1). The number of decompositions of P into m sectors is C_{n-1}^{m-1}. The number of all possible decompositions of P is 2^{n-1}.

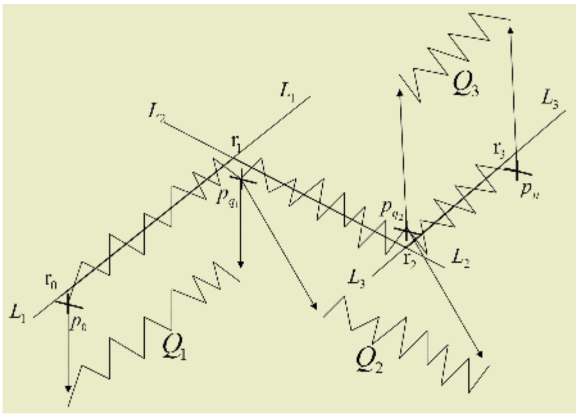


Fig. 1. Decomposition of the source polyline P into polygonal sectors.

b. Approximates each polygonal sector Q_j with a straight line L_j minimizing integral square error $\varepsilon_{q_{j-1},q_j}$ (see Appendix 1).

c. Let the measure of the error of the polyline approximation be a functional

$$F(m,\{q\},\Delta) = \sum_{j=1}^{m} \varepsilon_{q_{j-1},q_j} + m \cdot \Delta \ , \qquad (1)$$

where $\Delta \ (\Delta \geq 0)$ — is a penalty for each straight segment of the result polyline.

d. For a given Δ, let the optimal decomposition be the decomposition for which functional F is minimal. Approximation precision depends on the penalty Δ. The smaller Δ is, the more decomposition points will be found. If $\Delta = 0$, all vertices of the source polyline are decomposition points and $F = 0$. As $\Delta \to \infty$, the source polyline becomes one polygonal sector.
e. After finding the optimal decomposition or critical points, polygonal sectors have to be approximated by optimal straight lines. Then, the intersections of the lines approximating the adjusted polygonal sectors should be found. The new polyline goes through these intersections $r_j, (j = 0,\ldots,m)$. The beginning and the end of the new polyline are built as projections of the beginning and the end points of the source polyline to the first and last straight segments of the new polyline.

3 Iterative Algorithm for Obtaining an Optimal Solution

Let $P_i, i \le n$ be polylines defined by the first i straight segments of the source polyline P.

Given that the optimal decompositions of $P_k, P_{k-1},\ldots,P_1$ are known, we know the optimal (minimal) values of functional (1), the number of decomposition sectors $m_i,(i=1,\ldots,k)$, and the decomposition points $q_s^i,(i=1,\ldots,k; s=1,\ldots,m_i)$ for each of these polylines.

Find the optimal decomposition of P_{k+1}. This task may be solved with an exhaustive search.

Let M_{k+1}^j be a minimum value of the functional, when the last decomposition point of polyline P_{k+1} is p_j. Obviously, $M_{k+1}^j = M_j + \varepsilon_{j,k+1} + \Delta$ and $M_0 = 0$.

Let M_{k+1}^j be minimal when the value of $j,(j = k, k-1,\ldots,0)$ is equal to j^*, or $M_{k+1} = M_{k+1}^{j^*}$.

The optimal decomposition of the source polyline will be built when $k = n-1$.

The computational complexity of this algorithm is $O(n^2)$, because the complexity of the calculation of $\varepsilon_{j,k+1}$ is $O(1)$ [7].

4 Optimization

The described algorithm can be accelerated by the minimum estimation $M_{k+1}^{j_1,j_2}$ of the functional M_{k+1}^j for the case, when j is located in the half-open interval $[j_1, j_2)$. If $M_{k+1}^{j_3}$ was found for some j_3 which does not belong to $[j_1, j_2)$, and $M_{k+1}^{j_3}$ is less than $M_{k+1}^{j_1,j_2}$, then the beginning of the last polygonal sector for optimal decomposition cannot belong to $[j_1, j_2)$ and it is not necessary to analyze vertices located inside this half-interval.

In Appendix 2 two minimum estimations were found

$$S_{k+1}^{j_1,j_2} = M_{j_1} + \varepsilon_{j_2-1,k+1} + \Delta \text{ and } K_{k+1}^{j_1,j_2} = M_{j_2-1} + \varepsilon_{j_2-1,k+1}.$$

Therefore

$$M_{k+1}^{j_1,j_2} = \max\{M_{j_1} + \Delta, M_{j_2-1}\} + \varepsilon_{j_2-1,k+1} . \tag{2}$$

The minimum estimation (2) essentially accelerates the algorithm because the last point of the optimal decomposition of a long line is usually located closer to its end than to its beginning. The optimized algorithm is described in Appendix 3.

Fig. 2 compares the result of approximation obtained by suggested method and with a compression algorithm of Douglas-Peucker [1]. A regular decagon was used as ground-truth polyline. After densification, 1000 points were calculated. To obtain the source polyline, white noise was added to the coordinates of these points.

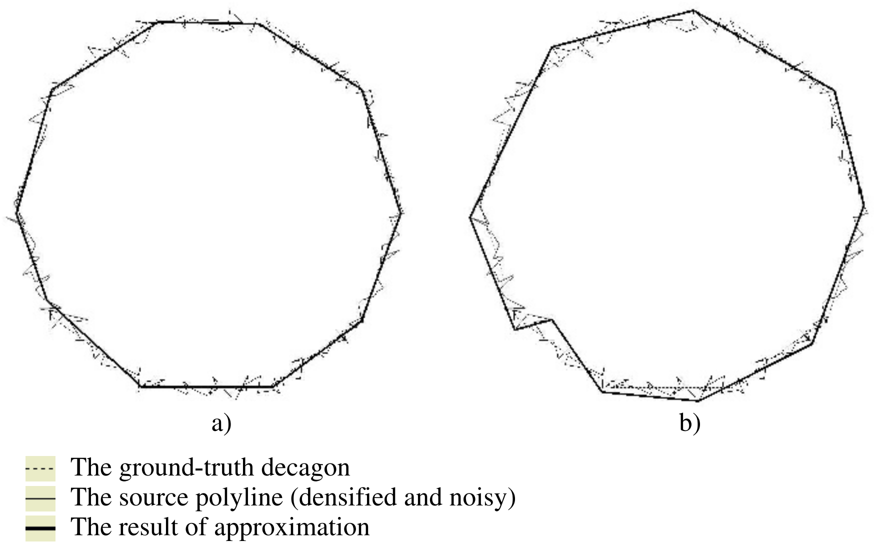

a)
b)
The ground-truth decagon
The source polyline (densified and noisy)
The result of approximation

Fig. 2. The result of approximation of the regular decagon with a) the suggested method and b) Douglas-Peucker compression [1].

Fig. 3 shows the dependence of the optimal value of functional (1) on penalty Δ, and Fig. 4 shows the dependence of the number of polygonal sectors on penalty Δ for the optimal decomposition of the source polyline. The horizontal segments in Fig. 4 show intervals of Δ, inside which the optimal number of segments are constant and decomposition vertices are not changed. The interval of Δ, corresponding to the result polyline with 10 segments, is the longest (in logarithmic scale) with the

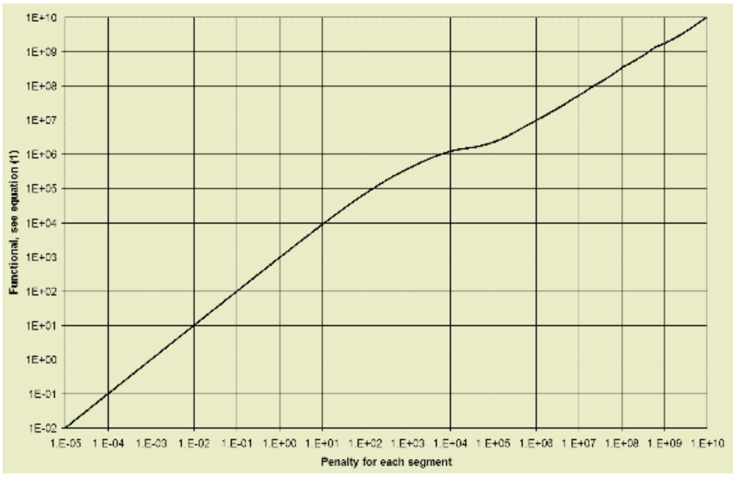


Fig. 3. Dependence a value of the functional (1) on the penalty Δ.

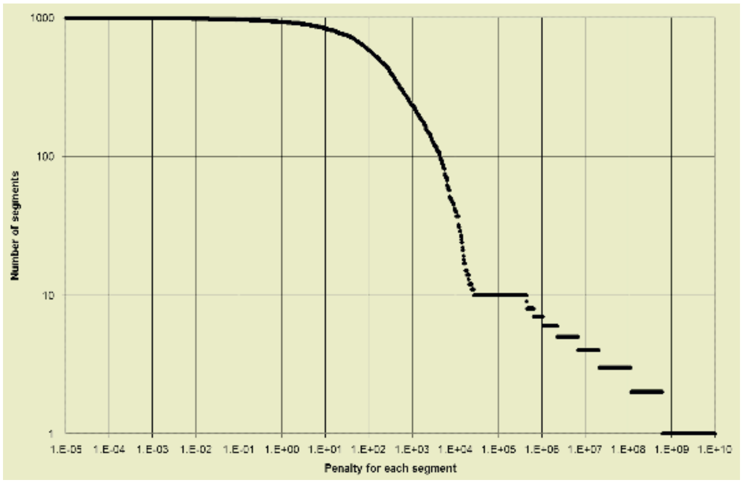


Fig. 4. Dependence the number of polygonal sectors on the penalty Δ.

exception of a singular case (the end interval). It is evidence of the stability of the algorithm to variations of Δ.

5 Conclusion

The goal of the article is to build an algorithm of filtering random errors of polylines. Suggested algorithm approximates the source noisy polyline with the new one. The vertices of the new polyline do not necessary coincide with vertices of the source polyline. The algorithm is based on the minimization of the functional that is the sum of the integral square error of approximation and a penalty Δ for each vertex of the result polyline.

Because the algorithm filters random noise the result polyline describes the shape of the ground-truth polyline better than the noisy source polyline. Therefore our algorithm is good for recognition of critical points.

The number of vertices of the result polyline usually is less than of the source polyline. Therefore the suggested algorithm can be used for compression. In the article we do not discuss the problem of stability of result polyline vertices found as intersections of straight lines. One more problem that was not analyzed here is selection of value of penalty Δ. Meanwhile correctness of selection of Δ can be evaluated only by the operator.

References

1. D.Douglas, Th.Peucker, "Algorithm for the reduction of the number of points required to represent a digitized line or its caricature", The Canadian Cartographer, Vol.10, #2 (1973), pp.112-122.
2. T.Lang, "Rules for robot draftsmen", Geographical Magazine, V.22, N.1, (1969), pp.50-51.
3. U.Ramer, "Extraction of Line Structures from Photographs of Curved Objects", Computer Graphics and Image Processing, Vol.4 (1975), pp.81 – 103.
4. J.Sklansky, V.Gonzalez, "Fast Polygonal Approximation of Digitized Curves", Pattern Recognition, Vol.12 (1980), pp.327-331.
5. D.G.Lowe, "Perceptual organization and visual recognition", 1985, Ch. 4, Kluwer Academic Publishers, Boston.
6. E.Bodansky, A.Gribov, M.Pilouk, "Smoothing and Compression of Lines Obtained by Raster-to-Vector Conversion", GREC 2002, LNCS 2390, D.Blostein, Y.-B.Kwon (Eds), 2002, pp.256-265.
7. Th.Pavlidis, S.L.Horwitz, "Segmentation of Plane Curves", IEEE Transactions on Computers, V.23, N.8, (1974), pp.860 – 870.

Appendix 1

The parametric description of the source line is $(x(t), y(t)), t \in [0, T]$ — where $(x(t), y(t))$ are the coordinates, t is the distance between the beginning and the current point of the polyline measured along the polyline, T is the polyline length.

Let $\varepsilon_{j_1,j_2}^{\alpha,C} = \int_{l_{j_1}}^{l_{j_2}} (x(t)\cos(\alpha) + y(t)\sin(\alpha) + C)^2 dt$ be an integral square error of approximation of the polygonal sector, located between j_1-th and j_2-th vertices of the source polyline, with straight line $\cos(\alpha)\cdot x + \sin(\alpha)\cdot y + C = 0$.

Let ε_{j_1,j_2} be the minimum integral square error, i.e. $\varepsilon_{j_1,j_2} = \min_{\alpha,C}\left\{\varepsilon_{j_1,j_2}^{\alpha,C}\right\}$

The optimal approximating straight line is defined with the next $\cos(\alpha)$, $\sin(\alpha)$, and C.

$$\cos(\alpha)=\begin{cases}\dfrac{2V_{xy}}{\sqrt{\omega^2+4V_{xy}^2}}, V_{xx}\ge V_{yy}\\ -\dfrac{\omega}{\sqrt{\omega^2+4V_{xy}^2}}, V_{xx}<V_{yy}\end{cases},$$

$$\sin(\alpha)=\begin{cases}-\dfrac{\omega}{\sqrt{\omega^2+4V_{xy}^2}}, V_{xx}\ge V_{yy}\\ \dfrac{2V_{xy}}{\sqrt{\omega^2+4V_{xy}^2}}, V_{xx}<V_{yy}\end{cases},$$

$$C=-\frac{V_x\cdot\cos(\alpha)+V_y\cdot\sin(\alpha)}{l_{j_2}-l_{j_1}},$$

where l_j is the distance between the beginning of the polyline and *j*-th vertex measured along the polyline,

$$\omega=\sqrt{(V_{xx}-V_{yy})^2+4V_{xy}^2}+\left|V_{xx}-V_{yy}\right|,$$

$$V_x=\int_{l_{j1}}^{l_{j2}}x(t)dt,\ V_y=\int_{l_{j1}}^{l_{j2}}y(t)dt,$$

$$V_{xx}=M_{xx}-V_x^2,\ V_{xy}=M_{xy}-V_x\cdot V_y,\ V_{yy}=M_{yy}-V_y^2,$$

$$M_{xx}=\int_{l_{j1}}^{l_{j2}}x^2(t)dt,\ M_{xy}=\int_{l_{j1}}^{l_{j2}}x(t)y(t)dt,\ M_{yy}=\int_{l_{j1}}^{l_{j2}}y^2(t)dt.$$

The minimum integral square error equals [7]:

$$\varepsilon_{j_1,j_2}=\tfrac{1}{2}\left(V_{xx}+V_{yy}-\sqrt{(V_{xx}-V_{yy})^2+4V_{xy}^2}\right).$$

Appendix 2

$$\min_{j_1\le j^*<j_2}\left\{M_{j^*}+\varepsilon_{j^*,k+1}+\Delta\right\}\underset{M_{j^*}\ge M_{j_1},\varepsilon_{j^*,k+1}\ge\varepsilon_{j_2-1,k+1}}{\ge}M_{j_1}+\varepsilon_{j_2-1,k+1}+\Delta,$$

$$\min_{j_1\le j^*<j_2}\left\{M_{j^*}+\varepsilon_{j^*,k+1}+\Delta\right\}\underset{\varepsilon_{j^*,k+1}\ge\varepsilon_{j^*,j_2-1}+\varepsilon_{j_2-1,k+1}}{\ge}$$

$$\ge\min_{j_1\le j^*<j_2}\left\{M_{j^*}+\varepsilon_{j^*,j_2-1}+\varepsilon_{j_2-1,k+1}+\Delta\right\}\underset{M_{j^*}+\varepsilon_{j^*,j_2-1}+\Delta\ge M_{j_2-1}}{\ge}M_{j_2-1}+\varepsilon_{j_2-1,k+1}.$$

These inequalities can be proven with two obvious expressions:
$\varepsilon_{j_1,j_2} + \varepsilon_{j_2,j_3} \le \varepsilon_{j_1,j_3} \forall 0 \le j_1 \le j_2 \le j_3 \le n$ and $M_{j_1} \le M_{j_2} \forall 0 \le j_1 \le j_2 \le n$.

Appendix 3: The Fast Algorithm of Optimal Decomposition of the Polyline P_{k+1}

1. Calculates the functional $M_{k+1}^{current} = M_k + \varepsilon_{k,k+1} + \Delta = M_k + \Delta$ supposing that a new decomposition point is the vertex of the source polyline with index $j^{current} = k$.
2. Calculates with expression (2) the minimum evaluation of the functional $M_{k+1}^{0,k}$ if the decomposition point is located inside the half-interval $[0,k)$.
3. If $M_{k+1}^{0,k} < M_{k+1}^{current}$, then $M_{k+1}^{0,k}$ and $[0,k)$ are sent to the priority queue.
4. While the priority queue is not empty
   a. Takes from the priority queue the request $M_{k+1}^{b,e}, [b,e)$ with the minimum value.
   b. If $M_{k+1}^{b,e} \ge M_{k+1}^{current}$, the request is not processed and end of algorithm.
   c. If $e - b < N$ (in our case $N = 8$)
   then a loop: for j from b till $e-1$.
   - Calculates M_{k+1}^{j}.
   - If $M_{k+1}^{j} < M_{k+1}^{current}$, then $j^{current} = j$, $M_{k+1}^{current} = M_{k+1}^{j}$.

   Else splits $[b,e)$ into two half-intervals $[b,j)$ and $[j,e)$.
   - Calculates with expression (2) the minimum evaluations of the functionals $M_{k+1}^{b,j}$ and $M_{k+1}^{j,e}$, suggesting that the new decomposition point is located first in $[b,j)$ and then in $[j,e)$.
   - If $M_{k+1}^{b,j} < M_{k+1}^{current}$, then $M_{k+1}^{b,j}$ and $[b,j)$ are sent to the priority queue.
   - If $M_{k+1}^{j,e} < M_{k+1}^{current}$, then $M_{k+1}^{j,e}$ and $[j,e)$ are sent to priority queue.

At the end of the algorithm, $j^* = j^{current}$ and $M_{k+1} = M_{k+1}^{current}$.

Splitting a half-interval can be done in different ways. One of them is dividing by two.

On Extending Symmetry Sets for 2D Shapes[⋆]

Arjan Kuijper and Ole Fogh Olsen

Image group, IT-University of Copenhagen
Glentevej 67, DK-2400 Copenhagen, Denmark

Abstract. Many attempts have been made to represent families of 2D shapes in a simpler way. These approaches lead to so-called structures as the Symmetry Set ($\mathcal{SS}$) and a subset of it, the Medial Axes ($\mathcal{MA}$). While the latter is commonly used, the former is still in the mathematical research stage. One reason for this is that in contrast to the $\mathcal{SS}$, the $\mathcal{MA}$ can be computed efficiently and fast, and yields one connected component for a closed shape.
In this paper a natural complement of the symmetry set, called the Anti-Symmetry Set ($\mathcal{ASS}$), is used to connect components bearing the full richness of the symmetry set. Secondly, new ways are presented to visualize these sets. One uses the radius of the describing circle as extra dimension, the other, the so-called pre-Symmetry Set (pre-$\mathcal{SS}$), uses the parameter space. Example shapes show the extra information carried in the $\mathcal{ASS}$ and the pre-$\mathcal{SS}$ in determining the special points on the $\mathcal{SS}$ as well as revealing the structure of the $\mathcal{SS}$ in more detail. They are also capable of distinguishing between different shapes where the $\mathcal{SS}$ and the $\mathcal{MA}$ in some cases fail.

1 Introduction

In 2D shape analysis the simplification of shapes into a skeleton-like structure is widely investigated. The Medial Axis ($\mathcal{MA}$) skeleton [4] is commonly used, since it can be calculated in a fast and robust way. Many resuls on simplification, reconstruction and database search are reported. The $\mathcal{MA}$ is a member of a larger family, the Symmetry Set ($\mathcal{SS}$), exhibiting nice mathematical properties, but difficult to compute and yielding distinct branches [6, 7]. To overcome the latter we introduce the anti-symmetry set ($\mathcal{ASS}$), resulting in a connected set.

Secondly, while the 2D visualization of the $\mathcal{MA}$ skeleton is unambiguous due to its limiting nature, the $\mathcal{SS}$ may give rise to intersecting curves that sometimes occur due to projections. Together with the need for augmenting the skeleton with information of the radius / scale / distance from the boundary at which it occurs, the need for a representation in a space with an extra dimension is evident - something that was already known for the skeleton [4] and used in the so-called Shock Graph method [15]. It has been mentioned by Wright et al. [17], but not been used afterwards. Another way of representation is obtained

⋆ This work is part of the DSSCV project supported by the IST Programme of the European Union (IST-2001-35443).

when the $\mathcal{SS}$ is considered in the parameter space, yielding the so-called Pre-Symmetry Set. A reason for using the $\mathcal{SS}$ instead of the $\mathcal{MA}$ is that the $\mathcal{SS}$ can be caught in a linear data structure, as presented elsewhere[12], in contrast to the graphs needed for representation of the $\mathcal{MA}$.

We will first give a short overview on the $\mathcal{MA}$ and the $\mathcal{SS}$ and then introduce the $\mathcal{ASS}$ and new ways for visualization.

2 Some Background Theory on Shapes

In this section we give the necessary background regarding properties of shapes, the Medial Axis, the Symmetry Set, labelling points on these sets and give an example to clarify the definitions.

Let $\mathcal{S}(x,y) = \{(x,y)|L(x,y)=0\}$ denote a closed 2D shape. Then $\mathcal{N}(x,y) = (L_x, L_y)(L_x^2+L_y^2)^{-1/2}$ denotes its unit normal vector, and $\kappa(x,y) = -(L_x^2 L_{yy} - 2L_xL_yL_{xy} + L_y^2 L_{xx})(L_x^2+L_y^2)^{-3/2}$ its curvature. The evolute $\mathcal{E}(t)$ is given by the set $\mathcal{S}+\mathcal{N}/\kappa$. Even if the curve is smooth and differentiable, the evolute contains non-smooth and non-differentiable points, viz. those where the curvature is zero or takes a local extremum, respectively.

2.1 The Medial Axis and Symmetry Set

The Medial Axis ($\mathcal{MA}$) is defined as the closure of the set of centers of circles that are tangent to the shape at least two points and that contain no other tangent circles: the are so-called maximal circles. The Symmetry Set $\mathcal{SS}$ is defined as the closure of the set of centers of circles that are tangent to the shape at least two points [6, 5, 8, 7]. Obviously, the $\mathcal{MA}$ is a subset of the $\mathcal{SS}$ [7].

To calculate these sets from above definition, the following procedure can be used: Let a circle with unknown location be tangent to the shape at two points. Then its center can be found by using the normal vectors at these points: it is located at the position of each point minus the radius of the circle times the normal vector at each point. To find these two points, the location of the center and the radius, do the following: Given two vectors p_i and p_j (right, with $i=1$ and $j=2$) pointing at two locations at the shape, construct the difference vector $p_i - p_j$. Given the two unit normal vectors N_i and N_j at these locations, construct the vector $N_i + N_j$. If the two constructed vectors are non-zero and perpendicular,

$$(p_i - p_j).(N_i + N_j) = 0, \tag{1}$$

the two locations give rise to a tangent circle. The radius r and the center of the circle are given by

$$p_i - rN_i = p_j - rN_j. \tag{2}$$

2.2 Labelling Points

It is known that the $\mathcal{MA}$ in itself carries insufficient information for representing and reconstruction a shape, since different shapes can yield the same $\mathcal{MA}$.

Therefore additional information can be added to the $\mathcal{MA}$ and used, as proposed by various authors [13–16]. It has been shown by Bruce et al. [6] that only five distinct types of points can occur for the $\mathcal{SS}$, and by Giblin et al. [8, 7] that they are inherited by the $\mathcal{MA}$. An A_1^2 point is the "common" midpoint of a circle tangent at two distinct points of the shape. An A_1A_2 point is the midpoint of a circle tangent at two distinct points of the shape but located at the evolute. An $A_1^2A_1^2$ point is the midpoint of two circles tangent at two pairs of distinct points of the shape with different radii. An A_1^3 point is the midpoint of one circle tangent at three distinct points of the shape. An A_3 point is the midpoint of a circle located at the evolute and tangent at the point of the shape with the local extremal curvature.

Based on the behaviour of the radii one can impose vectors on the $\mathcal{MA}$ denoting increasing radius, for example. This has been done in the so-called Shock Graph approach by Siddiqi et al. [13–16]. The $\mathcal{MA}$ or the $\mathcal{SS}$ together with the radius function is sufficient to reconstruct the shape [7].

3 Beyond the Symmetry Set

In this section we present three ways to add extra information to the $\mathcal{SS}$ based on Eqs. (1-2). The first is straightforward: use the radius of tangent circle to each set, as described in the previous section as an extra dimension. The second method is an extension of the $\mathcal{SS}$, called the anti-symmetry set ($\mathcal{ASS}$). Thirdly, the pre-symmetry set (pre-$\mathcal{SS}$) will be introduced as analysis and visualization tool. We focus on the $\mathcal{SS}$, since the $\mathcal{MA}$ is only a subset of it.

3.1 The $\mathcal{SS}$ Radius Space

Having incorporated the radius function upon the $\mathcal{SS}$ one has extra information. This information can be exploited in much more detail when the radius is considered as an extra dimension. First attempts were reported in [17], albeit only on a spline approximation of the shape. Using this dimension, the 1D curves in the 2D plane become 1D curves in 3D space. In this 3D space the $\mathcal{SS}$ curves reveal information that does not appear in a trivial manner in the 2D plane. For example, at an $A_1^2A_1^2$ point of the $\mathcal{SS}$, two curves are intersecting in 2D, but obviously not in 3D, since two different radii were involved. At an A_1^3 point, in contrast, three curves still intersect in the 3D space.

On the other hand, points at the $\mathcal{SS}$ that arise from locally minimal or maximal circles are clearly visible as local extrema of the 3D curve with respect to the radius.

3.2 The Anti-symmetry Set

Another extension is the Anti-Symmetry Set ($\mathcal{ASS}$). It is defined as the set of points satisfying Eq. 1, but not being part of the symmetry set. Figure 1b clarifies this.

The points p_1 to p_4 have in common that they give rise to points on the $\mathcal{MA}$ or the $\mathcal{SS}$ in specific pairs. For example, $p_1$and p_2 define a $\mathcal{MA}$ (and $\mathcal{SS}$) point and $p_1$and p_4 a $\mathcal{SS}$ point. The other combination p_1 and p_3 satisfies Eq. 1, but is not part of the $\mathcal{SS}$: they are part of the $\mathcal{ASS}$.

The anti-symmetry set appeared in the early 1990's due to Blake et al. [3, 2] in the field of robotics. There they considered this set in order to find an optimal finger position for a two finger grasp. Another – perhaps less striking – name is due to Giblin [11]: the Mid Parallel Tangent Locus, describing exactly what it is.

3.3 The Pre-symmetry Set

One way to visualize the locations the sign changes of Eq. 1 is by taking all points on the shape pair wise and plot these sign changes in a diagram. This was used by Holtom [11] and Giblin and Sapiro [9, 10, 1] in a different context (affine symmetry sets). Following their line of reasoning this diagram should be called the pre-$\mathcal{SS}$. Since it consists of non-intersecting lines for the $\mathcal{SS}$ it can act as a linear data structure describing the shape. In combination with the $\mathcal{ASS}$, same intersections occur as with the $\mathcal{SS}$ [12]. In the next section we will discuss the pre-$\mathcal{SS}$ and its properties in more detail.

4 Example Shapes

In this section we deal with some example shapes, staring with the simplest one: an ellipse. Since the ellipse is highly symmetric, some results may seem trivial, while others give a false intuition. We therefore continue with two other examples: a "cubic oval" [5] and a concave shape.

4.1 Ellipse

The ellipse $L(x, y) = x^2 + 4y^2 - 4 = 0$ has $\mathcal{N}(x, y) = (x, 4y)(x^2 + 16y^2)^{-1/2}$, and $\kappa(x, y) = -32(x^2 + 4y^2)(4x^2 + 64y^2)^{-3/2}$, with extremal values 1/4 and 2 at the four locations $(0, \pm 1)$ and $(\pm 2, 0)$, respectively. The evolute consists of four parts, joined at cusps at the locations $(0, \pm 3)$ and $(\pm 3/2, 0)$.

To find the $\mathcal{MA}$ with Eq. 2 we have two points on the shape with $y_1 = -y_2$ and $x_1 = x_2$ and $\tilde{y_1} = \tilde{y_2} = 0$ and $\tilde{x_1} = \tilde{x_2}$ for $(\tilde{x}, \tilde{y})$ a point on the $\mathcal{MA}$. Then $y_1 - 4ry_1/\sqrt{x^2 + 16y_1^2} = y_2 - 4ry_2/\sqrt{x^2 + 16y_2^2}$ yields $r = -\sqrt{x^2 + 16y_2^2}/4$. Together with $y_{1,2} = \pm\sqrt{4 - x^2}$ this gives $r = -\sqrt{16 - 3x^2}/4$. Then the points on the $\mathcal{MA}$ are given by $(\tilde{x}, \tilde{y}, r) = (x + rN, y + rN, r) = (x - x/4, y - 4y/4, -\sqrt{16 - 3x^2}/4) = (3x/4, 0, -\sqrt{16 - 3x^2}/4)$. Note that $x \in [-2, 2]$, so $\tilde{x} \in [-3/2, 3/2]$. The radius varies from $-1/2$ at the endpoints to -1 at the origin. So it has a $\mathcal{MA}$ formed by a straight line along the x-axis with its endpoints at the cusps of the evolute within the shape: $(\pm 3/2, 0)$.

Similarly, one can find the expression for the $\mathcal{SS}$ to be the curve above combined with the curve $(\tilde{x}, \tilde{y}, r) = (0, 3y, -\sqrt{4 + 12y_2^2})$, which varies from $(0, -3, -4)$ via $(0, 0, -2)$ to $(0, 3, -4)$, since $y \in [-1, 1]$, as shown in Figure 1c.

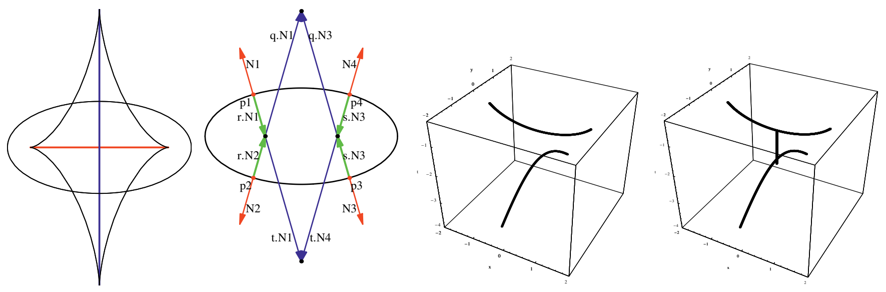


Fig. 1. a) The ellipse, its evolute and its $\mathcal{MA}$ (horizontal line) and $\mathcal{SS}$ (both lines) in 2D. b) Combinations of points contribute to the $\mathcal{MA}$, $\mathcal{SS}$, or $\mathcal{ASS}$. c) 3D representation of the $\mathcal{SS}$. d) The $\mathcal{SS}$ and $\mathcal{ASS}$ in 3D.

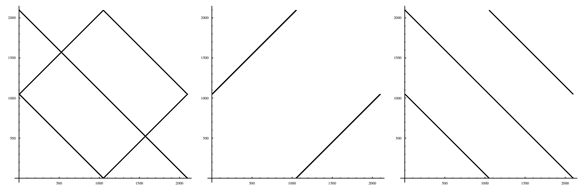


Fig. 2. The pre-$\mathcal{SS}$. Left to right: all zerocrossings, the $\mathcal{ASS}$ part and the $\mathcal{SS}$ part.

It therefore has a $\mathcal{SS}$ formed by the $\mathcal{MA}$, and a straight line along the y-axis with its endpoints at the cusps of the evolute outside the shape: $(0, \pm 3)$. So at the origin in 2D we have an $A_1^2 A_1^2$ point, with radii 2 and 1, see Figure 1a.

The $\mathcal{ASS}$ points are found as those points with $(x_1, y_1) = (-x_2, -y_2)$ and with radii $r = \sqrt{x_1^2 + y_1^2})$, varying between 1 and 2. This is identical to the two radii of the $\mathcal{SS}$ at the planar origin. The pre-$\mathcal{SS}$ and the parts of it determining the $\mathcal{SS}$ and the $\mathcal{ASS}$ are shown in Figure 2 from left to right. The diagrams are symmetric in the diagonal p_i, p_i. The dark lines represent the zero crossings of Eq. 1. At intersections of the diagonal and a zero crossing, a branch of the $\mathcal{SS}$ starts in an A_3 point. Since the shape is closed, the lines continue through the boundaries: the square represents a torus, since the starting point is arbitrary. Therefore the $\mathcal{SS}$ part contains four intersections with the diagonal, corresponding to the four cusps of the evolute. The two lines forming the $\mathcal{SS}$ of the square in the left image constitute one branch of the $\mathcal{SS}$. The long line together with the point at the origin (being a A_3 point) forms the second part of the $\mathcal{SS}$. The $\mathcal{ASS}$ image connects the curves. The intersections of the $\mathcal{SS}$ and the $\mathcal{ASS}$ are local extrema w.r.t. radius of the $\mathcal{SS}$, see Figure 1d.

4.2 The Cubic Oval

Firstly the closed part of a cubic oval given by $y^2 = 2bxy + a^2(x - x^3)$, with $a = 1.025$ and $b = 0.09$ is taken. Then six extrema of the curvature occur, while the curvature doesn't change sign and the shape is thus convex [6]. The

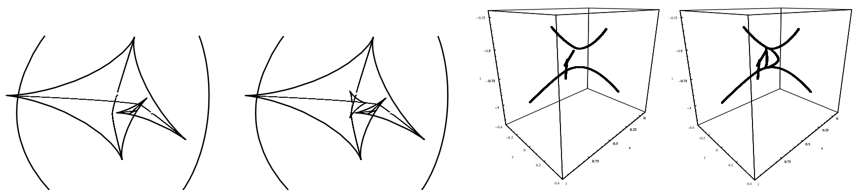


Fig. 3. Evolute of the cubic oval with the $\mathcal{SS}$ (a)and the $\mathcal{ASS}$ and the $\mathcal{SS}$ (b) in 2D. Radius space with the $\mathcal{SS}$ (c) and the $\mathcal{ASS}$ and the $\mathcal{SS}$ (d).

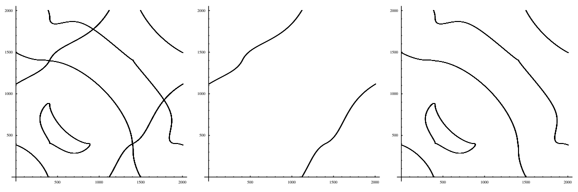


Fig. 4. The pre-$\mathcal{SS}$ of the cubic oval. Left to right: all zerocrossings, the $\mathcal{ASS}$ part and the $\mathcal{SS}$ part.

extra extrema of the curvature arise from a perturbation of the shape [5]. Since they alternate along the shape, a maximum and a minimum are created. As a consequence, the evolute is self-intersecting. Furthermore, the evolute consists six cusps. A direct consequence of this in its turn is that a new branch of the $\mathcal{SS}$ is created, since these branches always start in the cusps. This branch must be essentially different from the two other branches, since the original branches start in cusps that both arise from either local maxima of the curvature, or minima. These $\mathcal{SS}$ curves essentially need to have a local extremum with respect to the radius in 3D, as in Figure 1c. The newly created branch, however, has a minimum and a maximum as endpoints, so its behaviour in 3D must be different. The behaviour of the pre-$\mathcal{SS}$ is also different: a new branch implies a new "line" in the pre-$\mathcal{SS}$. Since the perturbation is a local effect, not all points on the shape are involved in creation the new branch. The "line" in the pre-$\mathcal{SS}$ thus must be a closed loop. This is indeed what occurs in Figure 4.

The projection of the $\mathcal{SS}$ and the $\mathcal{ASS}$ with the $\mathcal{SS}$, Figure 3, shows that again three curves joining pair wise in cusps for the ASS. The newly created branch shows some extra behaviour, as shown in Figure 5a. As clearly visible, the new branch bounces twice to the evolute, thus containing two A_1A_2 points. In the pre-$\mathcal{SS}$ (Figure 5b) these points are visible as the local extrema in vertical or horizontal direction of the closed loop (which has four, but two are due to the symmetry along the diagonal) due to the same bouncing.

Taking a closer look at the pre-$\mathcal{SS}$, one can see that the curve starting top-left also contains two pairs of local extrema. Consequently, one of the two original curves also contains two A_1A_2 points. This could also have been seen from the $\mathcal{SS}$ shown in Figure 5a: one curve traverses the evolute, which can only occur

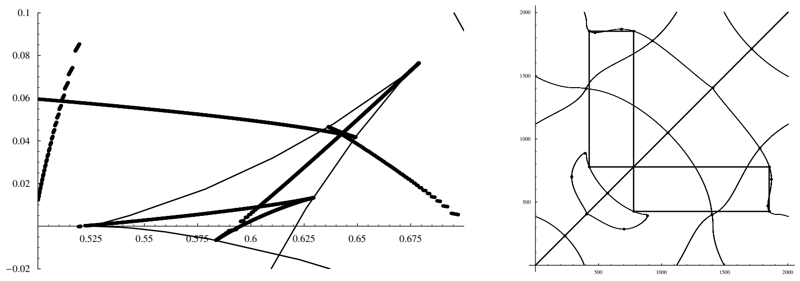


Fig. 5. Close up of the cubic oval with its evolute and symmetry set. Pre-$\mathcal{SS}$ of the cubic oval, with special points (see text).

by a double A_1A_2 combination. A way to get a grip on this is by the following: Consider the manifold making the swallowtail, with the A_3's on the corners. The symmetry sets walks along the manifold, starting from one A_3. Then it comes to the "end" of the manifold and continues its walk along the down-under part until it comes again to an end, where it goes on along the upper part until it reaches the second A_3 corner point. The same holds for the part of the symmetry set from which this new part originated[1].

As a result of the creation the new and old curve intersect in a A_1^3 point, where a circle is located tangent to three different points of the shape. Here three branches of the $\mathcal{SS}$ intersect. This A_1^3 point is also visible, albeit a bit hidden, in the pre-$\mathcal{SS}$. Figure 5b shows the pre-$\mathcal{SS}$ with all the special points: the A_1A_2 points as the local extrema in horizontal or vertical direction, the A_3 points at the intersection of a curve with the diagonal, the local extrema on the 3D $\mathcal{SS}$ curves as the intersections of the $\mathcal{SS}$ and $\mathcal{ASS}$ zero crossing curves, and the A_1^3 as the set of points linked by the lines. Only at a A_1^3 point there are three points in the pre-$\mathcal{SS}$ with the combination (p_1, p_2), (p_1, p_3), and (p_2, p_3) (and, of course, its diagonal symmetric counterpart). The 3D visualization of the $\mathcal{ASS}$ and the $\mathcal{SS}$, Figure 3d, shows again this all in one plot.

4.3 The Concave Case

As a more complicated concave shapes the one given by the equation $(x^2 + y^2 + a^2)^2 = b^2 + 4a^2x^2$, with $a = 1.99$ and $b = 4$ is taken . However, this example shows that the pre-$\mathcal{SS}$ for solely the $\mathcal{SS}$ part fails in determining the shape, see Figure 7: it has the same structure as the ellipse. This difference becomes clearer in Figure 6: The $\mathcal{SS}$ contains two straight lines, just as for the ellipse, albeit in this case one line "goes via infinity", since $\kappa = 0$, causing the two crosses. In 3D the distinction between the two branches on each side going to infinity is clearly visible (Figure 6c). The upper one corresponds to a 'negative' radius, i.e.

[1] This is the result of unfolding [5] the $A_\bullet$ that appeared when the perturbation held one double (non-generic) extremum, i.e. the transition from 0 to 2 new extremal values of the curvature of the shape.

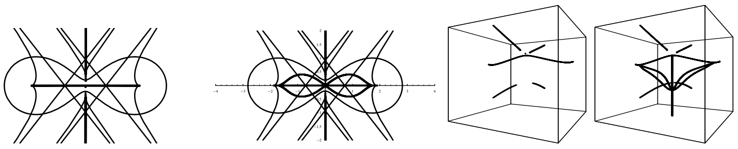


Fig. 6. Evolute of the concave shape with the $\mathcal{SS}$ (a) and the $\mathcal{ASS}$ and the $\mathcal{SS}$ (b). Radius space with the $\mathcal{SS}$ (c) and the $\mathcal{ASS}$ and the $\mathcal{SS}$ (d).

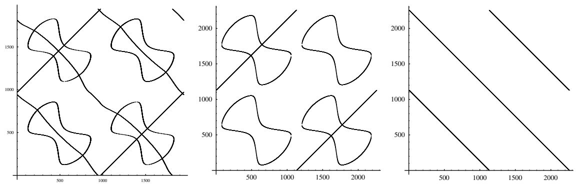


Fig. 7. Left to right: The pre-$\mathcal{SS}$, the $\mathcal{ASS}$ part and the $\mathcal{SS}$ part.

the radius related to a normal pointing outwardly. So again in the $\mathcal{SS}$ and $\mathcal{ASS}$ combination, the local extrema of the positive radii branches are connected. Note that the part along the x-axis contains three extrema.

5 Summary

In order to gain sufficient information from the $\mathcal{MA}$ or the $\mathcal{SS}$ additional information is needed. In this paper we extended the $\mathcal{SS}$ with information of the $\mathcal{ASS}$, the pre-$\mathcal{SS}$ and the visualization using the describing circle as an extra dimension. The representations in parameter space and in 3D space carry more information than the commonly used 2D visualization. In 3D possible ambiguities are avoided, while the use of the $\mathcal{ASS}$ guarantees a connection between the local extrema of the main positive radii branches, which are the main curves of symmetry. Next, we showed that special points along the $\mathcal{SS}$ can also be found in the pre-$\mathcal{SS}$, that can be used as a good indicator of the complexity of the $\mathcal{SS}$ and as localization tool for determining special points on it. Using these extensions of the $\mathcal{SS}$, one is able to catch the $\mathcal{SS}$ in a linear data structure [12], dependent on the 1D curves in the pre-$\mathcal{SS}$, in contrast to the graphs needed for representation of the $\mathcal{MA}$.

References

1. S. Betelu, G. Sapiro, A. Tannenbaum, and P. Giblin. Noise-resistant affine skeletons of planar curves. In *Proceedings of the 6th European Conference on Computer Vision (2000)*, volume 1842, pages 742–754, 2000. LNCS 1842.
2. A. Blake and M. Taylor. Planning planar grasps of smooth contours. *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 834–839 vol.2, 1993.

3. A. Blake, M. Taylor, and A. Cox. Grasping visual symmetry. *Computer Vision, 1993. Proceedings., Fourth International Conference on*, pages 724–733, 1993.
4. H. Blum. Biological shape and visual science (part i). *Journal of Theoretical Biology*, 38:205–287, 1973.
5. J. W. Bruce and P. J. Giblin. Growth, motion and 1-parameter families of symmetry sets. *Proceedings of the Royal Society of Edinburgh*, 104(A):179–204, 1986.
6. J. W. Bruce, P. J. Giblin, and C. Gibson. Symmetry sets. *Proceedings of the Royal Society of Edinburgh*, 101(A):163–186, 1985.
7. P. J. Giblin and B. B. Kimia. On the intrinsic reconstruction of shape from its symmetries. In *IEEE Conference on Computer Vision and Pattern Recognition, 1999*, pages 79–84, 1999.
8. P. J. Giblin and B. B. Kimia. On the local form and transitions of symmetry sets, medial axes, and shocks. In *Proceedings of the 7th International Conference on Computer Vision (1999)*, pages 385–391, 1999.
9. P.J. Giblin and G. Sapiro. Affine-invariant distances, envelopes and symmetry sets. *Geometriae Dedicata*, 71(3):237–262, 1998.
10. P.J. Giblin and G. Sapiro. Affine invariant medial axis and skew symmetry. *Computer Vision, 1998. Sixth International Conference on*, pages 833–838, 1998.
11. P. A. Holtom. *Affine-Invariant Symmetry Sets*. PhD thesis, University of Liverpool, 2000.
12. A. Kuijper. Computing symmetry sets from 2d shapes, 2003. Technical report, IT University of Copenhagen, Accepted ECCV 2004.
13. M. Pelillo, K. Siddiqi, and S. Zucker. Matching hierarchical structures using association graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(11):1105–1120, 1999.
14. T.B. Sebastian, P.N. Klein, and B. B. Kimia. Recognition of shapes by editing shock graphs. In *Proceedings of the 8th International Conference on Computer Vision (2001)*, pages 755–762, 2001.
15. K. Siddiqi and B.B. Kimia. A shock grammar for recognition. *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, pages 507–513, 1996.
16. K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 30:1–22, 1999.
17. M.W. Wright, R. Cipolla, and P.J. Giblin. Skeletonization using an extended Euclidean distance transform. *Image and Vision Computing*, 13(5):367–375, 1995.

Structural Analysis and Matching of Shape by Logical Property*

Nanhyo Bang and Kyhyun Um

Department of Computer and Multimedia Engineering, Dongguk University
Pil-dong 3 Ga 26, Chung-gu, Seoul, 100-715, Korea
{jjang,khum}@dgu.ac.kr

Abstract. The shape is one of most important feature for characterising an object. However, most shapes that are expressed with primitive uniform features have difficulty reflecting their logical and structural properties. In this paper, we propose a structural analysis scheme for the shape feature structured by logical properties, as well as a similar retrieval method. A shape is represented as a set of curve segments with a specific pattern. As a fundamental unit, a curve segment has adaptive features based on the logical property of its pattern. The relationship information of curve segments is expressed as a structural feature. We also use it as a feature for "coarse-fine" matching because our shape features have global characteristics as a structural feature and local characteristics as an adaptive feature of shape. Our experiments show that structural-adaptive features through logical analysis result in effectively classifying shapes according to their cognitive characteristics. Various experiments show that our approach reduces computational complexity and retrieval cost.

1 Introduction

Many information retrieval systems use images are faced with the challenge of representing and retrieving images. These systems have domain-dependent features in terms of retrieval or show improper results. This is because most researches have represented image as a primitive feature. Most primitive features express image as a low-level feature and numerically define the correlation between points. Researchers who have studied content-based retrieval systems define image as data with rich content. They classify features that compose image into a primitive feature and a logical feature[1][2]. A logical feature connotes a complex content, including semantic information. In [3], content is defined as "an understanding of the semantics of objects." Therefore, features that represent rich contents of an image must have a logical property in order to properly reflect semantic contents. In this aspect, the shape feature among many features that represent image has been widely used by virtue of its ability to recognize similar images.

* This research has been supported by Korean Science and Engineering Foundation No. R01-2002-000-00298-0

In this paper, we analyze shape according to the composition of its structural unit with logical properties. Fundamental units with adaptive features and structural relations between these units represent the logical contents of shape. We use a curve segment as a fundamental element to express the appearance of a shape. Structural and adaptive features are extracted through a logical and structural analysis using a curve segment. The structural feature as a global feature is used as a coarse-matching feature because it contains pattern and relationship information of curve segments comprising the whole shape. The adaptive feature, on the other hand, is a local feature. Using this feature produces correct results because of its ability to reflect specific geometrical properties of each pattern segment. Our experiments show how retrieval cost is reduced when conducting a search using these structural-adaptive features.

This paper is organized as follows: In chapter 2, we will explain a logical analysis scheme of the shape. In chapter 3, we will explain extraction methods of structural and adaptive features. A matching process and similarity function is explained in chapter 4. Experiment results and performance are shown in chapter 5. Related works are summarized in chapter 6, and the conclusion is drawn in chapter 7.

2 Structural Analysis of Shape

We define a shape as a set of curve segments with a pattern. Through preprocessing, we extract the necessary primitive feature in order to define the logical property and fundamental unit. Using this, we structure shape.

Preprocessing

We extract contour points from image's object to represent shape. However, because the size of contour points is too large, we use an adequate number of interest points that reflect a characteristic of a shape that is extracted from contour points. Interest points are aligned clockwise and described as a set $S = \{p_1, p_2, ..., p_n\}$, where an interest point is denoted by p_i and subscript n is the index number of the aligned interest point. A p_c is the gravity point of a shape. Two features are extracted from these interest points: θ and α feature.

- $f\theta(p_i) = \theta_i = \angle p_1 p_c p_k$, $1 < k \leq \mathrm{n}$
- $fIA(p_i) = \alpha_i = \angle p_{i-1} p_i p_{i+1}$, $0 \langle \alpha_i \langle 360$

In our work, a curve is separated using α feature of interest points that comprises a concave part. The separated segment is defined as curve segment, $CS = \{p_i \mid is \leq i \leq ie, \quad fIA(p_{is}), fIA(p_{ie}) > 180\}$

If CS includes two interest points, it becomes a concave segment. CS merges with the next CS into the new CS, if the next CS is also a concave segment. CS is a fundamental unit for the structured analysis of a shape.

Structure of Shape

Curve segments have specific patterns according to logical property. Logical properties are characteristics that logically define the curve pattern of CS. We redefine CS with logical properties as PS(Pattern Segment), and we generate PSS(Pattern Segment Sequence) as our shape feature to represent the structure and characteristic of shape. The logical properties of CS are expressed as adaptive features. The relationship between them is defined as follows:

- Shape S = PSS
- PSS = {PS} + structural features
- PS = CS + pattern type (with adaptive features that reflect logical properties by pattern type)

A CS has two logical properties: convex/concave property and turning property.

Definition 1. Convex/Concave Property. If interest points included in CS are all concave interest points or convex interest points, we define this CS with convex/concave property. Therefore, it is $\forall fIA(p_k) > 180$ or $\forall fIA(p_k) < 180$, when $p_k \in CS$ and $is < k < ie$.

Turning property means that CS changes direction or goes around a specific center.

Definition 2. Turning Property. Let CS be a sequence of two CS: $CS_1 = \{p_{is},..., p_m\}$, $CS_2 = \{p_m,...p_{ie}\}$, where $p_m \in CS$. One of them is CS with conver property and the other is CS with concave property. We define this CS as one with turning property if MBR(Minimal Boundary Rectangle) generated from convex CS overlap with MBR generated from concave CS.

PS is CS that has a specific pattern. First, CS is classified into either a concave curve segment or convex curve segment according to definition 1. Then, a turning curve segment is classified according to definition 2. The PSS is thus defined as $PSS=\{PS_i | PS_i = <pt, sflist, aflist>, \quad 1 \le i \le s\} \quad 1 \le i \le s\}$ where, pt is a pattern type, $sflist$ is structural featuress, and $aflist$ represents adaptive features of PS. PSS is stored in a database where information of interest points are saved.

3 Features Extraction of Pattern Segment

PSS has structural and adaptive features. PS that is an element of PSS shows a geometrical property of the partial curve to compose a shape through adaptive features.

Structural Features

Related information between PSs is used in order to compare the whole structure of shape. The α feature of interest point that divides adjacent PSs is simply used as a structural feature in our work. However, another shape is made by case even if they have the same α feature. The (a)(b) of Figure 1 shows such an example.

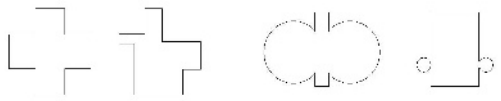


Fig. 1. (a) original shape, (b) an inner angle is the same as (a), but is a different shape. (c) original shape, (d) appearance of the curve is the same (c), but has a different scale ratio.

When a set of curves is compared, its structural feature will be reflected because the difference in the shape depends on the size difference even if the adjacent curves are similar to each other (Fig.1 (c)(d)). As explained above, when the pattern type of PSS of two shapes being compared is the same one, its structural features are used to calculate the structural difference of the shape.

- *α feature of a PS's last interest point, $\alpha dist$* - Because this feature is most simple feature to represent relationship between adjacent PSs, we use it when measuring structural similarity.
- *Pattern range, pr* -The start base angle (sa) and end base angle (ea) features are appended in order to solve the problem that α feature of two shapes is the same but appear to be different (Fig.1 (a)(b)). The start base angle is $\angle p_{is+1} p_{is} p_{ie}$ and the end base angle is $\angle p_{ie-1} p_{ie} p_{is}$.
- *Size ratio of PS, sr* - To solve problems such as (c) and (d) of Fig. 1, we extract the proportion of a PS to whole shape.

Adaptive Features

Adaptive features are features that reflect the logical and geometrical properties of each PS. They are called "adaptive features" because each of the different features adequately reflects the property of PS and is used in each of them. Representative adaptive features are as follows:

- *Average of α feature, $cc\alpha$* - This feature represents the degree of circularity.
- *Eccentricity e* - This represents the eccentricity of the polygonal segment.

$$\begin{aligned} &lleng = \overline{p_{mm} p_{md}},\ \ p_{mm} = (p_{is} + p_{ie})/2,\ \ p_{md} = Max(\underset{k=is+1}{\overset{ie-1}{\forall}} dp(p_{is} p_{ie}, p_k)) \\ &sleng = Max(\underset{si=is}{\overset{md-1}{\forall}} dp(p_{mm} p_{md}, p_{si})) + Max(\underset{ei=md+1}{\overset{ie}{\forall}} dp(p_{mm} p_{md}, p_{ei})) \\ &e = sleng/(sleng + lleng) \end{aligned} \tag{1}$$

A function $dp(L, p)$ return the length of perpendicular line from point p to line L. If e = 0.5, the MBR of the segment is almost a regular polygon. If e > 0.5, it will appear as an elongated segment pointing right to left. Otherwise, it will appear as an elongated segment pointing upwards.

- *Turning angle ta* - A ta is calculated from PS with partial or global turning properties. This feature extracts MBR from the concave segment of PS with a turning property. The center points of turn and turning angle are extracted from this MBR.

4 Matching

Similarity retrieval is executed using PSS. First of all, we filtered shapes that are most similar according to the structural information of PSS in order to reduce search cost. Similarity costs of structural and adaptive features are combined to calculate an integrated similarity cost.

Pattern Segment Matrix
For structural matching, the PSS feature of shape D in the database is compared with query shape Q. A Pattern Segment Matrix (PSM) is generated in order to find a comparably similar segment sequence. PSM expresses a value to represent the identity of pattern types between the PSSs of two shapes as elements of the matrix:

$$M_{rc}\text{'s value} = \begin{cases} 1, & QPS_r.pt = DPS_c.pt \\ 0, & QPS_r.pt \neq DPS_c.pt \end{cases} \qquad (QPS_r \in QPSS, DPS_c \in DPSS)$$

PSS of Q (QPSS) and PSS of D (DPSS) are used to represent the row and column axis of the matrix, M. After generating the matrix, we test whether the value of 1 continuously appears in a clockwise direction if M's value is 1. Again, we test the PSM made with QPSS, which was generated in reverse to find the symmetric shape. CSP(Candidate Similar Pattern sequence) are sequential segments with a similar pattern type between Q and D.CSP represent PSs with similar pattern in a diagonal direction in PSM. The CSP is as $CSP = \{CSP_i \mid CSP_i =< sq, sd, sleng >\}$ where, sq stands for index r of QPSS, the starting point of sequence, and sd is c index of DPSS, and $sleng$ is length of the sequence.

Similarity Cost
The $rCSP$, the segment sequence most similar to the shape of query and shape of database, is selected from PSM. The $rCSP$ is a CSP that is selected due to structural matching.

$$rCSP = MAX((\alpha dist_i + pr_i + sr_i)/3), \quad (\alpha dist_i, pr_i, sr_i \in \forall CSP_i) \tag{2}$$

Among the CSP, we select $rCSP$ with a maximum similarity value between structural features. The adaptive features of two PSs included in $rCSP$ are compared. $PtDist$ is the distance between adaptive features of two PSs. A similarity of adaptive features extracted according to segment pattern is calculated using Euclidian distance. The total similarity cost is defined as

$$\begin{aligned} SimCost = {} & \frac{(Qr + Dr)}{2} \times Dist(Structural\ features) \\ & \times Dist(Adaptivefeatures) \end{aligned} \tag{3}$$

In (3), if value of $Dist(Structural\ features)$ exceeds 0.8, then its value is 1. The Qr and Dr are the ratios of PSs belonging to $rCSP$ in the whole shape.

5 Experiments

The feasibility of structured analysis based on the logical characteristic of a shape and the utility of its adaptive features can be verified through experiments. For these, we use two test databases: 1,100 shapes from SQUID and 216 shapes selected from MPEG-7 test database. A search is then carried out with QBE.

5.1 Experimental Setup and Results

We have developed a prototype system with MS Visual C++ 6.0, and Oracle 8i. Figure 2 shows a block diagram of our prototype system.

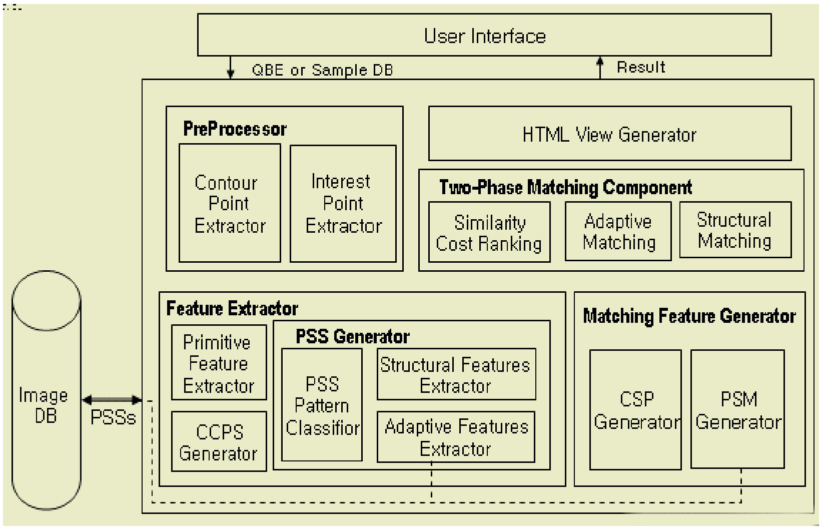
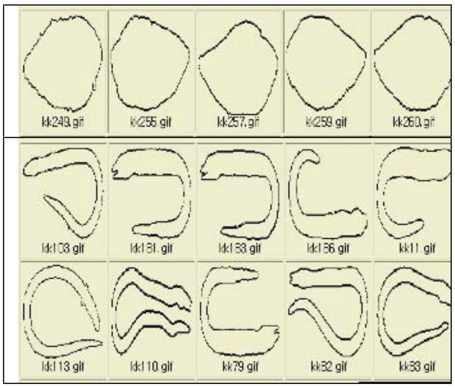


Fig. 2. Block Diagram of our Prototype System.



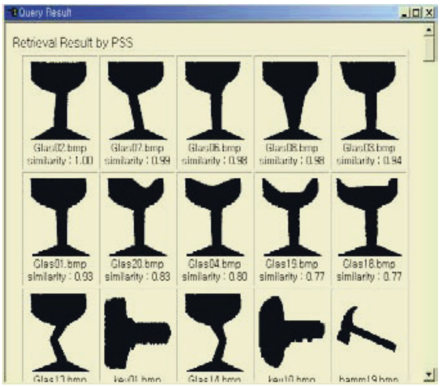

Query Result
Retrieval Result by PSS
Glas02.bmp similarity : 1.00
Glas07.bmp similarity : 0.99
Glas06.bmp similarity : 0.98
Glas08.bmp similarity : 0.98
Glas03.bmp similarity : 0.94
Glas01.bmp similarity : 0.93
Glas20.bmp similarity : 0.83
Glas04.bmp similarity : 0.80
Glas19.bmp similarity : 0.77
Glas18.bmp similarity : 0.77

Fig. 3. Classified Shapes.

Fig. 4. Query Results.

Figure 3 shows a retrieval result using only the PSS and logical properties from SQUID data. The global turning feature of the shapes in the second row of Figure 3 is applied. This retrieval result is the same as results produced by humans. Figure 4 is a result from the second test database, which has similar candidate shapes of partially various transformations that are highly similar. On an average, the extracted features are as follows: For SQUID, contour point is 693, interest points are 22 and PS is 8. For the second database, contour point is 878, interest points are 23 and PS is 7.

5.2 Performances

We complete a performance evaluation of shape features with features that are used, such as storage cost and feature extraction cost. We then compare our scheme with some representative methods such as the Grid Based Method and the Fourier Descriptors Method in [4].

Table 1. Storage Cost and Feature Extraction Cost Comparison.

Criteria	Grid Based Method	Fourier Descriptors Method	Ours
Storage cost	2*((mj/gn)2)/8+8	8 * fn	13 * m
Extraction cost	To find the major axis O(N2) To find the minor axis O(N) To rotate and scale O(N) To generate a binary number O(N3)	To find the centroid O(NlogN) To find all radii and compute FD coefficients O(r2) To generate signature O(r)	To extract the interest point O(NlogN) To segment PSS O(m) To extract structural-adaptive features O(m)

In Table 1, *mj* is the major axis size and *gn* is the grid cell size for the grid-based method. The *fn* is the number of Fourier coefficients and *r* is the number of radii for the Fourier Descriptors Method. In our scheme, *m* is the number of interest points. Our scheme shows better performance from the point of view of feature extraction cost. This is almost the same as the generation time of TPVAS in [4]. Tables 2 and 3 show computation cost to prepare internal data for searching and the space requirement to use in each searching step.

Table 2. Computation Cost.

Step	Cost
To make Pattern Matrix	$O(s^2)$
To make CSPs	$O(s^2)$
To generate rCSP	O(cs)

Table 3. Used Features and Size.

Search step	Used Features and Size
Structural	Pattern type feature : 6 bit * s Structural feature : 26 bit * s
Adaptive	Adaptive features : average 9 byte * s

In Tables 2 and 3, *s* is the number of a curve, and *cs* is the number of CSP. On the average, *s* is such a small number that it is about 1/3 of interest points. Our scheme has a merit to search with only a small quantity of structural features. We analyzed how search cost can be decreased on several representative query set for the second database.

Fig. 5. Search Space Comparison Graph.

Query set Q1 is queries that use shape data, which are then transformed by rotation, symmetry and scaling to the same shape. Query set Q2 is queries that use shapes with 2 or 3 PSs. Query set Q3 is queries that use shapes with more than 7 PSs. We then sort the data results according to similarity in value. The digit numbers on the y-axis refer to the number of images. Q1 approximates results with only structural features. In Q2, with a few PSs, shapes searching all appear as a result of the comparatively high similarity value. However, in Q3, there are more objects to search. This is because the partial matching value is included in the high ranking of results while some parts of PSs, with a query match with shapes, have fewer PSs.

6 Related Works

Contour- and region-based features are part of the representation method (Curvature Scale Space and Zernike Moment) of shape information adopted by the MPEG-7. Main topics covered by contour-based researches include matching strategies with contour points or additional information such as curvature of shape as well as methods for interest point extraction that preserves the original shape contour while reducing dimensionality [5]. In [6], Curvature Scale Space (CSS) information is taken at multi-resolution. In [7], e-envelope characteristics are represented by contour points. In [8], an object's shape is presented with line segments, and features derived from the line segments are used for matching. Using contour points, the multi-level resolution pyramid structure is designed and used for retrieval [9]. As an effort to solve these problems, concave/convex segment is extracted from the contour as shown in [10], and similarity search is conducted using dynamic programming. However, poor search performance is a common problem among researches that use these primitive features.

7 Conclusion

In this paper, we suggested methods to analyze a shape logically and to structure designed properties. A shape is composed of a curve segment as a fundamental unit. A curve segment has specific logical properties according to geometrical pattern. Through this, pattern segment is generated from curve segment. Pattern segment sequence possesses structural features between adjacent pattern segments. We use adaptive features extracted according to the segment's geometrical features. It is possible to increase the accuracy of search in comparison with ordinary point features. Since pattern segment reflects local features, it can be used for searching partial or overlapped objects. Experimental results show that our shape feature adequately represents object shape; and that it is possible for the retrieval method to improve accuracy and efficiency. Further researches for the proper classification of PS's pattern are needed so that it can represent the semantic feature of shape and enhance search performance. The designed logical properties are applied to various shapes of closed polygons. These logical features extend meaningful semantic features by integrating with other features.

References

1. Venkat N. Gudivada and Vijay V. Raghavan, "Content-Based Image Retrieval Systems," IEEE Computer, September 1995, pp. 18-22.
2. C. H. C. Leung and Z. J. Zheng, "Image Data Modeling for Efficient Content Indexing," Proc. Intl. Workshop on Multi-Media Database Management Systems, August 28-30, 1995, pp. 143-150.
3. A. Desai Narasimhalu, "Special Section on Content-based Retrieval," ACM Multimedia Systems, No. 3, 1995, pp.1-2.
4. Maytham Safar, Cyrus Shahabi and Xiaoming Sun, "Image Retrieval By Shape: A Comparative Study," IEEE Intl. Conference on Multimedia and Expo (I),2000, pp.141-154
5. D. Geiger, T. L. Liu and R. V. Kohn, "Representation and Self-similarity of Shapes," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, Issue No. 1, 2003, pp. 86-99.
6. F. Mokhtarian, "Silhouette-based Isolated Object Recognition through Curvature Scale Space," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.17, 1995, pp. 539-544.
7. I. Fudos , L. Palios and E. Pitoura, "Geometric-similarity Retrieval in Large Image Bases," Proc. of 18th International Conference on Data Engineering, 2002, pp. 441-450.
8. P. N. Suganthan, "Shape Indexing Using Self-organizing Maps," IEEE Transactions on Neural Networks, Vol. 13, Issue No. 4, July 2002, pp. 835-840.
9. M. Mignotte, "A new and simple shape descriptor based on a non-parametric multi-scale model," Proc. of. International Conference on Image Processing, Vol. 1, 2002 , pp. 445-448.
10. E. Milios and E.G.M. Petrakis, "Shape Retrieval Based on Dynamic Programming," Trans. Image Proc., Vol. 9, no. 1, 2000, pp. 141-146

Scale Descriptors through Phase Unwrapping

Alex Hughes and Richard C. Wilson

Department of Computer Science
University of York, York YO10 5DD, UK
{alex.hughes,wilson}@cs.york.ac.uk

Abstract. Shape matching typically relies on the candidate shapes being normalised for scale before the matching takes place. Many methods rely on finding the boundary or normalising the area of the shape. This is problematic when the object contains holes, or two objects have different underlying shapes. The advantage of our method is that the scale can be normalised separately from the shape.

1 Introduction

The shape of an object is typically defined as the configuration of pixels within the object, minus the effects of scale, rotation or translation. Shape matching techniques therefore need a way to cancel out, correct, or ignore these effects. In the case of scale, the objects can be resized to some nominal scale. However, this relies on generating a scale value for the object under examination, not always an easy task.

One technique is to detect the boundary of a shape and then use that information to determine a scale factor for the whole object. In the simplest case of a solid object with a clear unbroken outline (i.e. a silhouette), this is quite straightforward. However, there are a number of complications that may arise. For instance, the boundary of the object may not be clearly defined - it may be blurred or otherwise hard to determine. The object may also have multiple potential boundaries, due either to holes in the object, or to the object itself being composed of a number of separate parts. Even a single clear boundary may cause problems depending on its configuration, for example, if it doubles back on itself.

As such, it would appear that one of the major difficulties facing such feature-based derivations of scale is the resolution of ambiguities, a task which must largely be performed in the design phase, and by which they are later constrained. Additionally, there is the issue that, if only the boundary is being used, then arguably much potentially valuable information is being discarded. This may for instance make boundary-only based methods vulnerable to noise.

Another approach is to compute the area of the shape and then use that as a scale factor. While this works successfully for objects of precisely the same shape, if there is some perturbation of the shape, for example an internal hole, the scale will change.

We seek here to develop a technique that avoids the limitations of feature based techniques by finding a linear variate of scale based on the process of phase unwrapping.

2 Fourier Descriptors

We begin our analysis by assuming that we have an image of an object, centred at the origin, but with unknown scale and orientation. In addition, there is a degree of uncertainty as to whether any given point is inside the shape or not. As a result we may not be able to find an accurate boundary. We therefore define a shape function as a probability density

$$p((r, \theta) \in S) \tag{1}$$

defined in the polar coordinate system (r, θ). In order to digitise this shape, we form a polar pixel lattice defined by $p(i, \gamma)$.

2.1 Radial Shape Slices

A radial slice is then defined as the vector

$$\mathbf{v}(\gamma) = [p(0, \gamma), p(1, \gamma), \ldots, p(n, \gamma)]^T \tag{2}$$

The vectors $\mathbf{v}$ may be regarded as feature vectors for the shape, and statistical analysis of this feature space can provide a good characterisation of the overall scale of the shape. However, the unknown scale and orientation parameters modify the feature space in a very non-linear manner.

A change in the orientation of the shape will result in a change in the order of the vectors. We can accommodate this simply by using a statistic which does not depend on the order of the samples.

The effects of scale however, present a more challenging problem. A change in the scale of the shape will result in a change of the indices of the vectors themselves, a highly non-linear process, making analysis of the set of all vectors extremely difficult. We therefore need some method of converting this change in indices into a change in the values at each index.

2.2 Fourier Descriptors of Radial Shape Slices

We commence by applying the Discrete Fourier Transform directly to our radial slices.

$$(x_0, x_1, \ldots, x_n) = \mathrm{DFT}(p(0, \gamma), p(1, \gamma), \ldots, p(n, \gamma)) \tag{3}$$

The quantities $x_0, \ldots, x_n$ are complex numbers that can be represented in terms of a phase and a magnitude component.

This Fourier shape description is completely general, in the sense that the inverse DFT will recover the original data perfectly, for any possible shape. However, realistic shapes will span only a small subset of the possible shape space. For instance, rapid changes in either radial or angular directions are unlikely.

In other words, the amount of information present in a radial slice is typically much less than the size of the feature space would suggest.

Fourier descriptors provide an elegant means of discarding this excess information, in that we can simply discard high-frequency components. In addition, it has been previously shown that much of the information present in a signal is contained in the phase component alone [7], allowing us to discard all the magnitude information, further halving the size of our representation.

As a result, the original radial slice vector can ultimately be replaced by the first few components of the Fourier phase.

$$\mathbf{v}_f(\gamma) = (\phi_0, \phi_1, \ldots, \phi_k)^T \tag{4}$$

By using a reduced representation, we can both reduce the storage space required, and boost the efficiency with which the representation can be manipulated.

3 Scaling and Fourier Descriptors

One important property of the Fourier Transform is that translations in the original data correspond to changes in the phase values of the DFT. Thus, we have converted the effects of scaling from a change in the indices of the values to a change in the values themselves. The change in phase values can be described mathematically by use of the Fourier Shift Theorem:

$$f(t - t_0) = e^{j\omega t_0}\mathrm{F}(\omega) \tag{5}$$

This demonstrates that as the values in the radial slices change index values (the 'shift' in the above equation), then the phase values will be incremented or decremented accordingly. Notice however, that this assumes that all index values in the radial slice will be shifted by a constant amount, something that will not be the case as the shape is resized. Scaling operations, by their very definition, will stretch or contract the waveform represented by the radial slice.

It is easy to obtain a fixed shift for a fixed change in scale by simply sampling the radial slices logarithmically. This is equivalent to sampling the original shape function exponentially. In other words, we re-define eq. 2 as

$$\mathbf{v}(\gamma) = [p(\mathrm{e}^{\beta i}\gamma)]^T \tag{6}$$

The phase representation can then be constructed as before.

3.1 Phase Wrapping and Unwrapping

The problem with adding an offset to each of the phase values is that phase values are constrained to lie between 0 and 2π. Adding values that would take them beyond these limits merely causes them to wrap around. In other words, there are multiple solutions to the phase shift of a particular frequency component. For a measured phase shift of ϕ, the possible solutions are $\phi, \phi + 2\pi, ..., \phi + n\pi$

(where n can also be negative). For this reason, it is not possible to directly infer the shift from the phase change.

The problem of determining the number of times the phase 'wraps' is known, unsurprisingly, as phase unwrapping. Phase unwrapping procedures are perhaps most commonly employed in decoding data supplied by Synthetic Aperture Radar (SAR), or other techniques that represent distances in terms of phases. If the total range of distance to be measured is greater than the wavelength used by the measuring device, then the data will be returned in a wrapped state. Under the assumption that the true data varies smoothly, fringes or jumps in the phase values can be interpreted as wrapping events. However, noise and discontinuities in distance confound this process. There are a number of algorithms for tackling the unwrapping of wrapped-phase-only data [3, 8, 1, 10, 9, 11, 6].

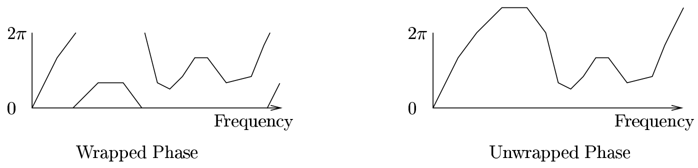


Fig. 1. Phase Unwrapping

In contrast to the problem faced in decoding SAR data however, we have the original data which gives rise to the phase information, making the unwrapping problem more tractable. There are a few techniques that make use of this extra data. Perhaps the most famous was published by Tribolet [12] and functions by re-integrating the derivative of the phase values, on the assumption that the general gradient of the curve formed by the phase values will not change significantly at wrapping points.

Later McGowan and Kuc published an 'exact' method [5], which was subsequently refined by Long [4]. These algorithms purport to calculate an exact solution numerically, but have the drawback that they can be very computationally demanding, both in processing time and storage space.

Most recently of all, Krajník has published a method [2] that claims to be both swift and accurate. It functions in an iterative manner, determining the unwrapped phase for each phase value from the previous phase value, working through the phases in order of their corresponding frequencies. Specifically, he defines the phase increment between adjacent frequencies to be:

$$\varDelta\varPhi_{n+1} = \mathrm{ARG}\left(\frac{X_{n+1}}{X_n}\right) \tag{7}$$

In essence, this complex division results in a third complex number whose phase is the difference in phase between the first two values. For this reason, it is important that the (wrapped) phase does not change by more than π between adjacent phase values, something that can be achieved by padding the original data with zeros, thus interpolating the data in the frequency domain.

One issue that plagues all the aforementioned unwrapping techniques is the case of zeros on the complex plane (i.e.: where the magnitude is zero). At these points, it becomes impossible to tell whether a wrap has occurred or not. Such zeros are most likely to occur with synthetic data however, rather than real-world input. Zeros could potentially be eliminated by adding a small amount of noise to the signal.

4 Shape Scale Representation

4.1 Basic Method

The method we used was to generate 360 radial slices. These slices were sampled from the perimeter inwards (the last sample corresponded to the centre of the shape), making a total of 64 samples per slice. The test shapes were white (value=255) against a black background (value=0). Each of the slices were then extended from 64 to 4096 values by padding with zeros. This was to ensure that the phases of adjacent frequencies would not differ by more than π, as per Krajník's technique.

The Discrete Fourier Transform of each slice was calculated and Krajník's iterative method used to find the first 10 unwrapped phases (this proved to be a good balance between efficiency and accuracy) from the complex output of the DFT. All other phase and magnitude information was discarded.

Finally, all 360 sets of unwrapped phases were averaged together to yield a single set of unwrapped phases whose values represented the scale of the shape in question.

4.2 Measurement of Scale for a Single Object

Our first example consists of a simple image of a car, at 3 scales. The 3 images represent successive doublings in scale.

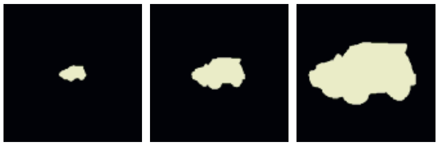
As can be seen from the graph, the doublings in scale are represented by a constant change in the gradient of the curve formed by the unwrapped values. This verifies that we can measure the scale of an object through this technique. Note that we can obtain the scale by using just a single frequency component, eg: the 10th component.

4.3 Shape Matching

We can also make use of this technique to align two shapes in rotation and scale. The technique is as follows:

Given two images to match, we will refer to one of them as the base image, and the other as the test image. We start by generating a set of 360 radial slices for the base image, generating phase information from these slices and unwrapping the phases, as described previously.

We then rotate the test image relative to the base image in a number of increments. At each step of the rotation, we generate the set of unwrapped phases for the test image.



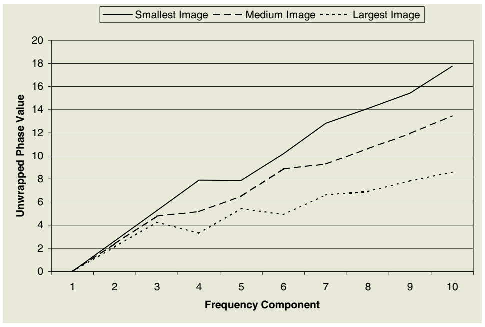


Fig. 2. Unwrapped Phases for Car Images

Finally, we calculate the difference between the unwrapped phase values at a particular frequency (eg: the 10th frequency component) for the corresponding slices of the base and test images. These differences are used to increment an accumulator array, indexed by rotation angle and difference in unwrapped phase.

If the two images are rotationally aligned then we would expect to find the same difference between the unwrapped values for the slices of the test image and the corresponding slices of the base image, for every slice, because the scaling difference is the same for every corresponding pair of slices (this of course assumes isotropic scaling). Hence, in our case, we would expect to see 360 counts of the same difference value, with that difference value representing the overall difference in scale between the two objects. As the two images drift out of rotational alignment, corresponding slices will not represent the same parts of the objects, and instead a broad range of difference values will be obtained.

Ultimately then, we can expect to find a spike in the accumulator array corresponding to the relative rotation and scale factors. This is indeed the case in fig. 4, where we can see a clear spike in the accumulator data indicating that the third step of the test image rotation sequence is that which most closely matches the base image, confirming what we can see from a visual inspection of the data.

The peaks near the bottom of the grid are caused by single-pixel discrepancies due to the pixellated nature of the images, exaggerated by the exponential sampling.

An important point to note is that it is not necessary to actually rotate the test shape relative to the base shape. The same effect can be obtained merely by re-ordering the original radial slices (or, more accurately, the sets of unwrapped phases generated from them) of the initial test shape image. Thus it is only necessary to generate one set of unwrapped phase slices for the test image.

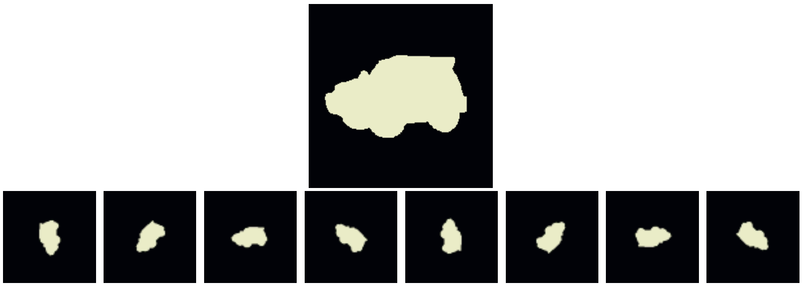


Fig. 3. Base Shape and Series of Rotated Test Shapes

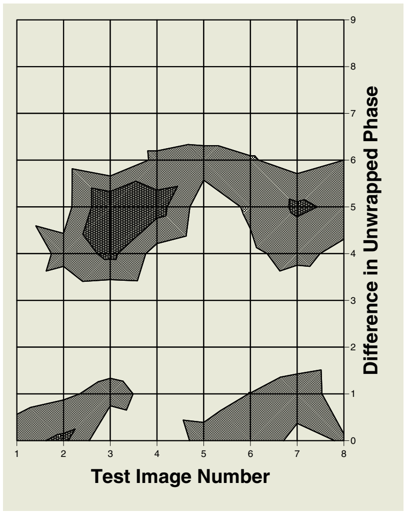


Fig. 4. Accumulator Array

Also, it would be possible to perform the alignment iteratively, starting with 10 coarse radial slices, using these to align the shapes roughly, and then repeating the process at a finer resolution for only part of the images. This would mean that only a few sets of unwrapped data would need to be generated.

4.4 Partial Matches

It is also possible to use this technique to make partial matches. When the shapes shown in fig. 5 were processed, they produced the accumulator graph shown. The two spikes on this graph indicate that there were two possible matching scales. This could potentially be useful when trying to match partially occluded objects.

5 Conclusion

We have presented here a new technique for obtaining a scale factor for unknown shapes that avoids many of the limitations of feature-based scale-finders, while

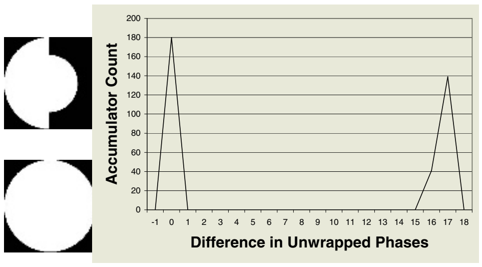


Fig. 5. Shapes That Match at Two Possible Scales

also being more robust and precise than area-based methods. The technique can also be adapted to form the basis of an efficient shape-matching system.

References

1. Kazuyoshi Itoh. Analysis of the phase unwrapping algorithm. *Applied Optics*, 21(14):2470, 1982.
2. Eduard Krajník. A simple and reliable phase unwrapping algorithm. In J. Vandewalle et al., editor, *Signal Processing VI: Theories and Applications*, pages 917–919. Elsevier, Amsterdam, 1992.
3. José M. N. Leitão and Mário A. T. Figueiredo. Absolute phase image reconstruction: A stochastic nonlinear filtering approach. *IEEE Trans. Image Processing*, 7(6):868–881, 1998.
4. David G. Long. Exact computation of the unwrapped phase of a finite-length time series. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 36(11):1787–1790, 1988.
5. Richard McGowan and Roman Kuc. A direct relation between a signal time series and its unwrapped phase. *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-30(5):719–726, 1982.
6. Oleg Michailovich and Dan Adam. Phase unwrapping for 2-D blind deconvolution of ultra-sound images. Submitted to IEEE Trans. Medical Imaging, 2002.
7. A. V. Oppenheim and J. S. Lim. The importance of phase in signals. In *Proc. of the IEEE*, number 69, pages 529–541, 1981.
8. A. V. Oppenheim and R. W. Schafer. *Digital Signal Processing*. Prentice Hall, 1975.
9. Mark D. Pritt and Jerome S. Shipman. Least-squares two-dimensional phase unwrapping using FFT's. *IEEE Trans Geoscience and Remote Sensing*, 32(3):706–708, 1994.
10. Kenneth Steiglitz and Bradley Dickinson. Phase unwrapping by factorization. *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-30(6):984–991, 1982.
11. Torfinn Taxt and Galina V. Frolova. Noise robust one-dimensional blind deconvolution of medical ultrasound images. *IEEE Trans. Ultrasonics, Ferroelectrics, and Frequency Control*, 46(2):291–299, 1999.
12. José M. Tribolet. A new phase unwrapping algorithm. *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-25(2):170–177, 1977.

Lung Segmentation on Postero-anterior Digital Chest Radiographs Using Active Contours

Isaac Iglesias[1], Pablo G. Tahoces[2], Miguel Souto[3], Anxo Martínez de Alegría[3], María J. Lado[4], and Juan J. Vidal[3]

[1] Laboratory for Radiologic Image Research, University of Santiago, 15782 Santiago
isaacigl@usc.es

[2] Department of Electronics and Computer Science. University of Santiago
Campus Sur s/n, 15782 Santiago

[3] Department of Radiology
Complejo Hospitalario Universitario de Santiago (CHUS)
University of Santiago, 15782 Santiago

[4] Department of Computer Science, University of Vigo
Campus Lagoas-Marcosende, 36200 Vigo

Abstract. A computerized pulmonary segmentation based on the detection of oriented edges was performed in postero-anterior (PA) digital radiography (DR) images. To further improve detection of lung contours, a method based on the use of active contours models was developed. First, the technique calculates a set of reference lines to determine the relative position of the lungs in the image. Then, vertical and horizontal rectangular regions of interest (ROIs) are studied to identify the preliminary edge. These points are an approximation to the lung edges that are adjusted using the active contours models. We studied the influence of the different parameters of the active contours on the final result over 30 DR images. Results prove that the active contour models, with selected parameters, can be used to improve the results of a given segmentation scheme.

1 Introduction

Digital radiography (DR) is becoming the standard for pulmonary imaging [1] [2]. DR provides high spatial and contrast resolution for the lung parenchyma. DR also opened the possibilities for computer-aided diagnosis (CAD) applications in chest imaging, such as pulmonary nodules detection [2] [3], characterization of interstitial disease [4], measurement of pulmonary volume [5], and detection of cardiomegaly [6] and pneumotoraces [7]. A survey of the CAD in the chest radiography can be found in [8].

Lung segmentation is a necessary prerequisite for all of these quantitative analysis applications. Several investigators have developed techniques for CAD segmentation of pulmonary DR images. Xu and Doi [9] have determined the rib cage boundary using a set of ROIs, which are selected over the maxima and minima of the gray-level values on horizontal profiles. Duryea et al. [10] searched the lung contour studying the difference between the consecutive maxima and minima of horizontal profiles. Armato et al. employed an iterative global gray-level thresholding and a local gray-level threshold analysis with the resulting image to determine the lung edge in PA [11] and lateral [12] chest radiographs; Carrascal et al. [5] determined a group of reference lines, defined a set of rectangular ROIs and identified the pulmonary border using edge enhancement and thresholding techniques. Vittitoe et al. [13] employed Markov random field modeling for identifying lung regions. Tsujii et al. [14] classified the pixels of the DR using an adaptative-sized hybrid neural network. Van Ginneken et al. [15] developed and compared several segmentation techniques: a matching approach, pixel classifiers based on several combinations of features, a rule-based scheme that detects lung contours using a general framework for the detection of oriented edges and ridges in images, and a hybrid scheme. Later, an active shape model was used by van Ginneken et al. [16] to obtain the segmentation.

Investigators use different algorithms to obtain the segmentation. Of interest is that there is a common step to all of them: it consists on the adjustment of an initial contour in order to correct, complete or smooth it. We believe that the active contour models (snakes) can be very helpful to perform this task. This technique was previously used in chest radiography by Yue et al. [17] to the detection of the rib borders. We propose an automatic lung segmentation scheme that calculates a set of reference lines to determine the relative position of the lungs in the image. Using these lines, a collection of rectangular ROIs to cover the pulmonary border is sequentially defined. To obtain the points of the border, the averaged horizontal or vertical profile in each ROI was analyzed. Finally, the algorithm fits the set of points using an active contour model. The influence of different parameters of the active contour models, such as the relative weights of the three energies (continuity, curvature and image energy) was studied.

2 Materials

A total of 30 PA chest radiographs were employed to develop the method. DR was performed using a Siemens Thorax FD (Siemens AG, Munich, Germany) with a matrix of 3K$\times$3K (pixel size of 143 μm) and 4096 gray level resolution in DICOM format. No distinction was made between normal and pathological cases. The computer programs were written in C++ programming language on Solaris 2.8 operating system. CNT (Mallinckrodt Institute of Radiology, Washington University School of Medicine) libraries have been used for development of DICOM utilities. A Sun Ultra80 Workstation (Sun Microsystems, Inc., Mountain View, CA) was used for all the calculations.

3 Methods

To develop the computerized system to automatically extract the lung contour, we have followed three steps: a) calculation of the reference lines; b) calculation of the points near the lung edge; and c) correction of the rough border using a snake.

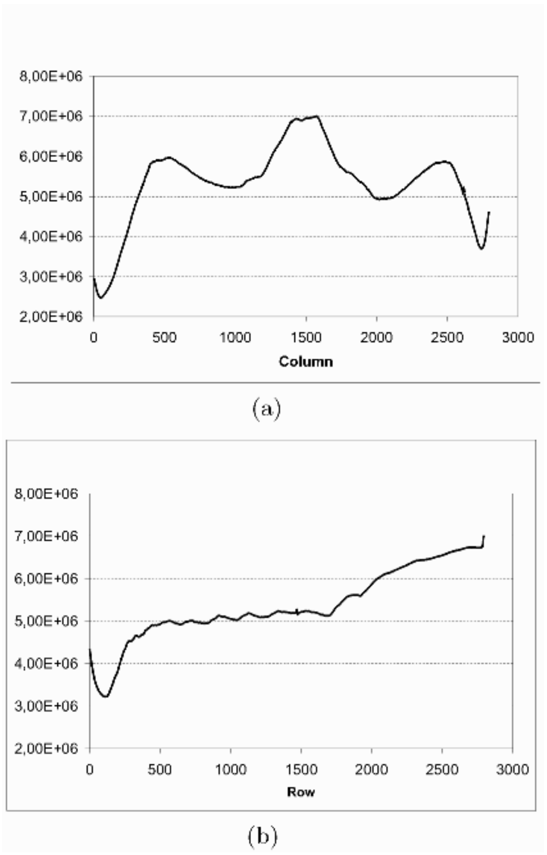


Fig. 1. Horizontal 1(a) and vertical signature 1(b)

3.1 Reference Lines

We used the definition of horizontal and vertical thoracic signature (equations 1 and 2) suggested by Meyers et al. [18]:

$$S_{hor}(j) = \sum_{i=0}^{N_{fil}} I[i,j] \tag{1}$$

$$S_{ver}\left(i\right) = \sum_{j=0}^{N_{col}} I\left[i,j\right] \tag{2}$$

and that are represented in the figures 1(a) and 1(b). The midline C_{mid} (1 in figure 2) of the thoracic cage was determined by searching the central maximum of the horizontal signature of the image (figure 1(a)). The upper and lower limits of the rib cage were obtained from the vertical signature (figure 1(b)), the maximum in the first third of the derivative being the upper limit R_{up} (2), and the lower limit being the maximum of the last half of the derivative. Since the lower limit of the left lung can be very different of the lower limit of the right lung we calculated the vertical signature of each hemithorax and we obtained two lower limits, R_{low}^{right} (3) and R_{low}^{left} (4). The lateral limits of the lungs were calculated as follows: we calculated for each hemithorax an averaged horizontal profile of the $\frac{R_{up}-R_{low}}{3}$ rows over R_{low}. The external lateral limit of the right lung C_{ext}^{rigth} (5) was the first maximum of the profile an the internal limit C_{int}^{right} (7) was the maximum of the derivative close to C_{mid}. For the left lung, C_{ext}^{left} (6) was the last maximum and C_{int}^{left} (8) was the minimum of the derivative close to C_{mid}. In the figure 2 the reference lines are shown.

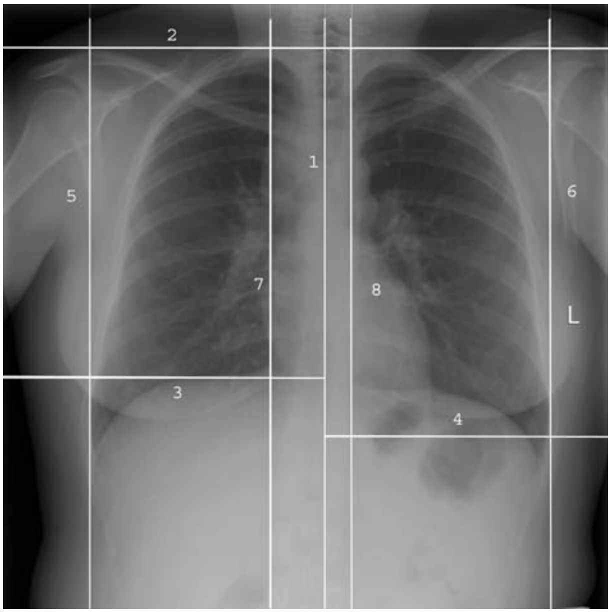


Fig. 2. Reference lines

3.2 Calculation of the Initial Points

Lung contour was divided into four zones: apex, mediastinum, diaphragm and rib cage border, that were calculated separately. The edge points were calculated by using a set of rectangular ROIs oriented either vertically (in the apex and the diaphragm) or horizontally (in the mediastinum and rib border). The position of these ROIs was determined by using the reference lines. In the apex, the ROIs had 30 columns width, and were located under R_{up}. The ROIs of the diaphragm were of the same size being calculated sequentially between C_{int} and C_{ext}, the first was calculated around R_{low} and the position of the edge point then calculated determined the position of the next ROI. The first and the last points of the diaphragm determined the position of the first ROIs of the rib cage border and the mediastinum, the remainder are calculated from the edge points obtained. In order to calculate the edge, the points of each ROI were averaged to obtain an horizontal or vertical average profile. To reduce the number of candidates, every point of the profile was averaged with its 30 neighbors, taking into account that we were searching a large structure such as the lung edge. After this, depending on the type of border that we were searching, the minima or the maxima of the first derivative were calculated. If there were more than one candidate, the difference between the density of gray levels of the 50 neighbors after and before the point was calculated and the one that had the greater difference in absolute value was chosen.

3.3 Correction of the Border Using a Snake

With the method described above we calculated an outline of the lung edge. The points did not join smoothly and any of the points did not correspond with the true lung border, because of this the border needed to be adjust. To do this we used the energy-minimizing curves known as snakes introduced by Kass et al. [19] for detecting lines and curves in an image. The snake is an active contour model which constantly minimizes an energy functional that consists on both the internal and external energy. The internal energy serves to impose continuity and smoothness constraint on the shape of the snake while the external energy tends to pull the snake toward salient image features such as image edges. Representing the snake by a parametric curve $v(s) = [x(s), y(s)]$, the energy functional can be written as:

$$\mathcal{E}_{snake} \equiv \int_0^1 E_{snake}(v(s))ds = \int_0^1 [E_{int}(v(s)) + E_{ext}(v(s))]ds \tag{3}$$

This expression must be discretized for using it in a digital image. Then the curve $v(s) \approx v_i = (x_i, y_i)$, where $0 \leq i < N$, being N the number of points of the snake. In this way, the expression 3 can be written as:

$$\mathcal{E}_{snake} = \sum_{i=0}^{N} (E_{int}(i) + E_{ext}(i)) \tag{4}$$

To minimize this expression Kass et al. [19] used techniques of variational calculus, Amini et al. [20] pointed out some problems with this approach and proposed an algorithm using dynamic programming. Since this method is slow we decided to use the greedy algorithm proposed by Williams et al. [21] that defines the internal and external energies as follows:

$$\mathcal{E}_{snake} = \sum_{i=0}^{N} [\alpha(i)(\bar{d} - |v_i - v_{i-1}|) + \beta(i)|v_{i-1} - 2v_i + v_{i+1}| + \gamma(i)E_{image}(i)] \tag{5}$$

where $\bar{d}$ is the average distance between the points of the snake, E_{image} is a function of the point i of the image and the parameters α, β and γ govern the

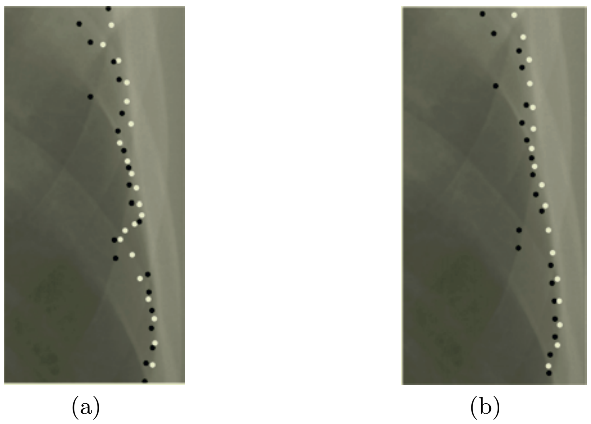


Fig. 3. Fitted rib cage border: using a gradient filter (3(a)) and a Sobel filter (3(b)) as image energy. The black dots are the initial and the white the fitted contour

relative weight of the energies. The evolution of the snake depends on the election of the function $E_{image}(i)$ that allows for determining the type of features of the image that attract the snake and the parameters that allows to determine the shape of the snake. Since each zone of the lung border has different characteristics, a snake with several values to the parameters and $E_{image}(i)$ was created, depending on the position of the point. Then four sets of parameters and four $E_{image}(i)$ (apex, mediastinum, diaphragm and rib cage border) were searched.

4 Results and Discussion

We calculated the initial border in 30 PA chest images, in order to obtain the necessary features of the snake that best fitted the lung contour. The influence of different image energies (the gradient and the Sobel filters to detect horizontal and vertical edges) was studied using standard parameters. The final border was calculated and, for each zone the energy that obtained the best result was selected. In the active contour schemes the use of $E_{image} = -grad[I(x, y)]$ is usual. However, in the thorax DR images this energy did not obtain the best result because there are many edges in the image, some of them become more pronounced than the lung contour. The best example of this are the ribs, when the initial contour was adjusted with $E_{image} = -grad[I(x, y)]$, the snake in the rib border tended to move closer the ribs. This effect is shown in figure 3. To solve this problem, we used the knowledge about the direction of the edges that we were searching. Each zone in the lung have different characteristics, the apex and the diaphragm are mainly horizontal edges and the mediastinum and rib border are vertical. Then we used the Sobel filters ($S_x[I(x, y)]$ and $S_y[I(x, y)]$) as the E_{image}. In the figure 3 is shown how this election changed the final result. With this energies, the influence of the parameters was analysed. In order to reduce the number of possibilities we only used three values $(0,5,\ 1$ and $1,5)$

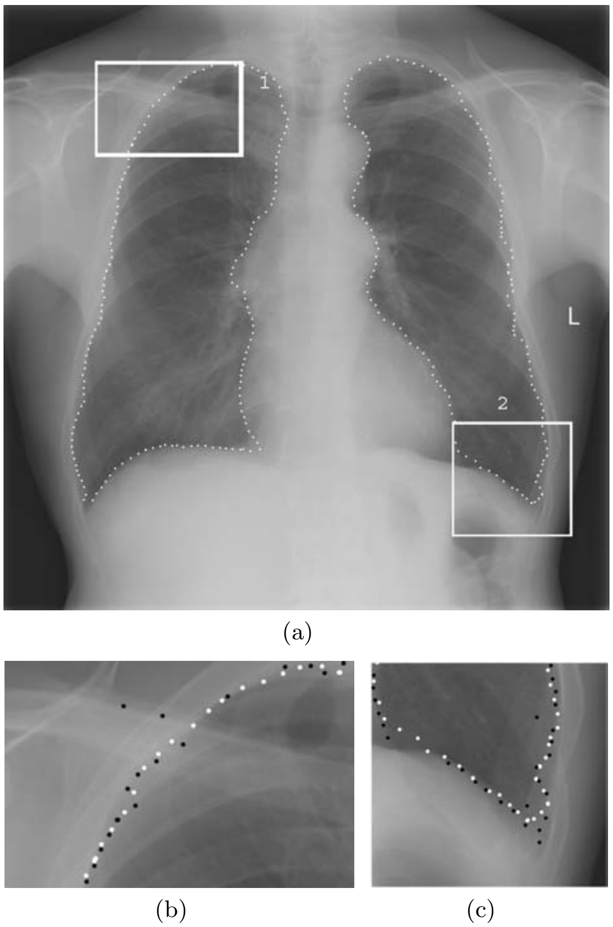


Fig. 4. 4(a) Resulting lung border. Comparison of the initial (black dots) and final contour (white dots): 4(b) in the clavicle region (1); 4(c) in the cardiophrenic and costophrenic angles (2)

for the parameters α, β and γ. Then, there were 27 possible combinations of parameters for each zone. Since the features of the initial border were known, not all of this possibilities were used in the posterior analysis. For example, it was known that the initial contour was very irregular. However, the lung contour must be rounded and as a result the β parameter should be 1 or $1,5$ to eliminate the corners. In the same way γ should not be very small for not loosing the image information. Then the combinations for which γ was less than the other parameters were eliminated.

In the first stage, we obtained the final border after being fitted using the remainder combinations of parameters. An experienced radiologist selected the two or three sets of parameters that best fitted in each region. Once the sets of parameters were selected the fit of the initial border was repeated, but employing different values for the parameters of each region. In the final step, a radiologist evaluated the accuracy of each calculated contour and the parameters that obtained the best result was chosen. In figure 4(a) an example of the final lung contour obtained is shown. The β parameter is always high in order to obtain a smoothness effect, eliminating the irregularities in the contour. The α parameter tends to approximate the points of the snake. It helps in the task of eliminating the irregularities and is fundamental to correct the wrong edge points (example in fig. 4(b). As the first contour is often wrong in the ribs and the clavicle (fig. 3 and 4(b)) it was necessary to increase the value of this parameter in this region. On the other hand, if α and β are high, the snake tends to eliminate the true lung corners, such as the cardiophrenic and costophrenic angles, specially when their edges are fuzzy. To reduce this effect, the parameters α and β for the diaphragmatic region are low compared to the other regions. With a correct choice of the parameters, this effect was reduced but not eliminated as we have shown in the fig. 4(c). To improve the detection, an alternative method to carefully detect these angles, and a snake with an attractive term of energy [19] could be used in these regions.

Finally, the best results were obtained with ($\alpha = 1$, $\beta = 1.5$, $\gamma = 1$) for the rib border, ($\alpha = 0.5$, $\beta = 1.5$, $\gamma = 1.5$) for the apex, ($\alpha = 0.5$, $\beta = 1,5$, $\gamma = 1$) for the mediastinum and ($\alpha = 0.5$, $\beta = 1$, $\gamma = 1.5$) for the diaphragm.

These results demonstrate that a snake with the appropriate features can be used to correct, smooth and complete the results of a given segmentation scheme.

References

1. M. Souto, K. S. Malagari, D. Tucker, P. G. Tahoces, J. Correa, V. S. Benakis, C. Roussos, K. A. Strigaris, J. J. Vidal, G. T. Barnes, and R. G. Fraser. Digital radiography of the chest: state of the art. *Eur. Radiol.*, 4:281–297, 1994.
2. J. Correa, M. Souto, P. G. Tahoces, K. S. Malagari, D. Tucker, J. J. Larkin, J. Kuhlman, G. T. Barnes, E. Zerhouni, R. G. Fraser, and J. J. Vidal. Digital chest radiography: comparision of unprocessed and processed images in the detection of solitary pulmonary nodules. *Radiology*, 195:253–258, 1995.
3. Maryellen L. Giger, Kunio Doi, and Heber MacMahon. Image feature analysis and computer-aided diagnosis in digital radiography. automated detection of nodules in peripheral lung fields. *Medical Physics*, 15:158–166, 1988.
4. S. Katsuragawa, Kunio Doi, and Heber MacMahon. Image feature analysis and computer-aided diagnosis in digital radiography: detection and characterization of interstitial lung disease in digital chest radiographs. *Medical Physics*, 15:311–319, 1988.
5. Francisco M. Carrascal, José M. Carreira, Miguel Souto, Pablo G. Tahoces, Lorenzo Gómez, and Juan J. Vidal. Automatic calculation of total lung capacity from automatically traced lung boundaries in postero-anteior and lateral digital chest radiographs. *Medical Physics*, 25 (7):1118–1131, July 1998.
6. Nobuyuki Nakamori, Kunio Doi, Victoria Sabeti, and Heber MacMahon. Image feature analysis and computer-aided diagnosis in digital radiography: Automated analysis of sizes of heart and lung in chest images. *Medical Physics*, 17 (3):342–350, May/Jun 1990.

7. Shigeru Sanada, Kunio Doi, and Heber MacMahon. Image feature analysis and computer-aided diagnosis in digital radiography: Automated delineation of posterior ribs in chest images. *Medical Physics*, 18 (5):964–971, Sep/Oct 1991.
8. Bram van Ginneken, Bart M. ter Haar Romeny, and Max A. Viergever. Computer-aided diagnosis in chest radiography: A survey. *IEEE Trans. Med. Imag.*, 20 (12):1228–1241, December 2001.
9. Xin-Wei Xu and Kunio Doi. Image feature analisys for computer aided diagnosis: Accurate determination of ribcage boundary in chest radiographs. *Medical Physics*, 22 (5):617 – 626, May 1995.
10. Jeff Duryea and John M.Boone. A fully automated algorithm for the segmentation of lung fields on digital chest radiographics images. *Medical Physics*, 22 (2):183 – 191, February 1995.
11. Samuel G. Armato III, Maryellen L. Giger, and Heber MacMahon. Automated lung segmentation in digitized posteroanterior chest radiographs. *Acad. Radiol.*, 5:245 – 255, 1998.
12. Samuel G. Armato III, Maryellen L. Giger, Kazuto Ashizawa, and Heber MacMahon. Automated lung segmentation in digital lateral chest radiographs. *Medical Physics*, 25 (8):1507 – 1520, August 1998.
13. Neal F. Vittitoe, Rene Vargas-Voracek, and Carey E. Floyd, Jr. Identification of lung regions in chest radiographs using markov random field modeling. *Medical Physics*, 25 (6):976 – 985, June 1998.
14. Osamu Tsujii, Matthew T. Freedman, and Seong K. Mun. Automated segmentation of anatomic regions in chest radiographs using an adaptative-sized hybrid neural network. *Medical Physics*, 25 (6):998 – 1007, June 1998.
15. Bram van Ginneken and Bart M. ter Haar Romeny. Automatic segmentation of lung fields in chest radiographs. *Medical Physics*, 27 (10):2445–2455, October 2000.
16. Bram van Ginneken, Alejandro F. Frangi, Joes J. Staal, Bart M. ter Haar Romeny, and Max A. Viergever. Active shape model segmentation with optimal features. *IEEE Trans. Med. Imag.*, 21 (8):924–933, August 2002.
17. Zhanjun Yue, Ardeshir Goshtasby, and Laurens V. Ackerman. Automatic detection of rib borders in chest radiographs. *IEEE Trans. Med. Imaging*, 14 (3):525 – 536, September 1995.
18. P. H. Meyers, H. C. Nice, C. M.and Becker, N. J. Nettleton, J. W. Sweeney, and G. R. Meckstroht. Automated computer analysis of radiolographic images. *Radiology*, 13:1029–1034, 1964.
19. M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 4:321–331, 1988.
20. A. A. Amini, T. E. Weymouth, and R. C. Jain. Using dynamic programming for solving variational problems in vision. *IEEE Trans. Pattern Anal. Machine Intell.*, 12 (9):855–867, 1990.
21. D. J. Williams and M. Shah. A fast algorithm for active contour and curvature estimation. *CVGIP: Image Understanding*, 55 (1):14 – 26, 1992.

Emotion Recognition from Dance Image Sequences Using Contour Approximation

Hanhoon Park[1], Jong-Il Park[1], Un-Mi Kim[2], and Woontack Woo[3]

[1] Department of ECE, Hanyang Univ., Haengdang-dong 17, Seongdong-gu, Seoul, Korea
hanuni@mr.hanyang.ac.kr, jipark@hanyang.ac.kr

[2] Department of Dance,Hanyang Univ., Haengdang-dong 17, Seongdong-gu, Seoul, Korea
kimunmi@hanyang.ac.kr

[3] U-VR Lab., K-JIST, Oryong-dong, Puk-gu, Kwangju, Korea
wwoo@kjist.ac.kr

Abstract. We present a novel approach that exploits shape context to recognize emotion from monocular dance image sequences. The method makes use of contour information as well as region-based shape information. The procedure of the method is as follows. First, we compute binary silhouette images and its bounding box from dance images. Next, we extract the quantitative features that represent the quality of the motion of a dance. Then, we find meaningful low-dimensional structures, removing redundant information but retaining essential information possessing high discrimination power, of the features using SVD (Singular Value Decomposition). Finally, we classify the low-dimensional features into predefined emotional categories using TDMLP (Time Delayed Multi-Layer Perceptron). Experimental results demonstrate the validity of the proposed method.

1 Introduction

For the last couple of years many researchers have focused on recognizing human emotion to achieve a more efficient and natural human-computer interface. The emotion recognition methods that extract emotional information from speeches or facial expressions have been extensively investigated [12-15]. However, gesture-based methods have been less explored except for a method that uses physiological signals [16]. This may be due to the fact that *gestures* are much too high dimensional, dynamic and ambiguous to analyze and recognize exactly.

Among some interesting exceptions are the following. To analyze the dynamically changing human motion, Kojima et al. defined "rhythm points" that means the time of the start and the end of a motion, so that the whole motion could be represented as the collection of partial motions (a unit of motion) with a period [10]. Wilson et al. also tried to identify temporal aspects of gesture [11]. They proposed a method that detects candidate rest states and gesture phases from gestures spontaneously generated by a person telling a story. Kojima's and Wilson's method are appropriate for analyzing a simple or well-regulated motion such as small baton-like movement. In general, however, human motions (especially in case of dance in this paper) are not so simple so we cannot easily determine the start/end of a unit of motion or rest state. Thus, their promising methods do not seem to be applicable to general human motions. In addition, these methods only aims at analyzing human motion but does not pay attention to emotions.

Recently, some methods that can recognize human emotion from modern dance image sequence have been introduced [3-6]. Dance is the most universal way of expressing human emotion. To represent the high-dimensional and dynamic change of gestures, these methods simplified the dynamic dance to the movement of rectangle surrounding human body and exploited several features related to the motion of rectangle using Laban's effort theory [9]. Then they analyzed the features and tried to find the relationship between the features and human emotion. They showed satisfactory results in most cases. However, they were confronted with difficulties in some cases. For example, when the bounding box associated with a human motion is the same as that associated with another one, they cannot discriminate the differences between the two human motions.

To resolve the problem mentioned above, this paper presents a method of exploiting new shape information for representing human motion. Note that the previous methods focus on only *region-based* shape information – the features such as bounding box or centroid etc. belong to the region-based shape information – but we additionally use the *contour-based* shape information, i.e. the number of dominant points on the boundary of human body. Thus our method can discriminate the subtle difference between the shape of human motion that have the same bounding box information.

The number of the computed dominant points can explain how complex or star-like human motion is and thus be regarded as a contour-based shape descriptor [1]. It cannot explain the details of the shape context of human motion. However, it may suffice to utilize rough contour information because we do not aim to answer the question to "what is the gesture he/she has just made?" but to catch the overall mood i.e. emotion.

The structure of this paper is as follows: In Section 2, we briefly describe the overview of emotion recognition system. The contour approximation algorithm used in our method is presented in Section 3. We then demonstrate the result of a variety of experiments in Section 4. We conclude in Section 5.

2 Overview of the Emotion Recognition System

It is not easy to directly extract high-level (and qualitative) information like emotion from low-level (and quantitative) data like human motion. Recently, however, efforts to extract human emotion from dance have been made. Many researchers thought that it would be easier to recognize emotion from dance than from ordinary human motions because *dance* is what fully expresses human emotion.

Some introduced Laban's movement theory [9] to represent emotional dance quantitatively. Suzuki et al. [3] and Camurri et al. [6] presented a methodology for mapping dance into emotional categories that represent human emotion based on the Laban's theory for the first time. Suzuki et al. and Camurri et al. had shown the possibility of directly recognizing human emotion from dance image sequences. Woo et al. proposed a method that nonlinearly maps dance into emotional categories and emphasized the importance of *flow*[1] by introducing the Time Delayed Multi-Layer Perceptron (TDMLP) [4]. Recently, we extended the work of Woo et al. and proposed a statistical approach that recognizes the human emotion faster and more accurately [5].

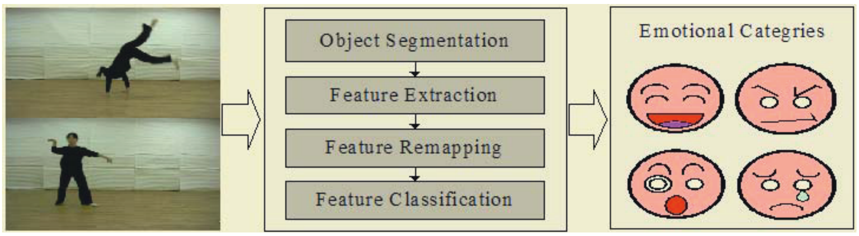


Fig. 1. Overview of emotion recognition system. The system directly maps low-level features extracted from dance images into predefined emotional categories.

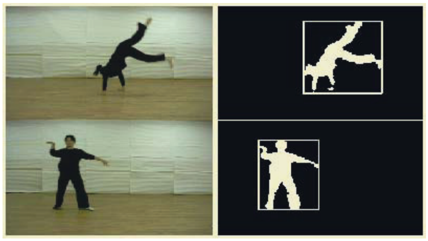


Fig. 2. Object segmentation. This shows the result of applying the background subtraction and the shadow elimination to an image. Refer to [7] for more details.

Table 1. Features extracted from binary images.

The aspect ratio of rectangle	H/W
The coordinate of centroid	(C_x, C_y)
The coordinate of the center of rectangle	(R_x, R_y)
The silhouette area	S_s
The rectangle area	S_r
The velocity of each feature	$f(\cdot)$
The acceleration of each feature	$g(\cdot)$
$f(x_n) = x_n - x_{n-1}$, $g(x_n) = x_n - 2*x_{n-1} + x_{n-2}$	

[1] *Flow* means the temporal variation of human motion, which is one of the factors that Laban adopted to quantitatively represent human motion.

Figure 1 shows the overview of our previous system [5]. First, the background and shadow in dance images are eliminated using the difference keying and normalized difference keying technique separately and then binary silhouette images and their bounding box are computed (Figure 2). Next, the features representing the quality of the motion of a dance are extracted (Table 1). These features are related to the factors i.e. *space*, *time*, *weight*, *flow* that Laban adopted to find out *effort*[2] from human motion. That is, Laban thought that they could quantitatively represent something (it may be emotion here) included in human motion. Next, Singular Value Decomposition (SVD) is applied to the extracted features and then the low-dimensional features associated with large eigen-value are selected. This has an effect of discriminating noisy information from the reliable features. Finally, the low-dimensional features are classified into predefined emotional space using TDMLP.

3 Emotion Recognition Using Contour Approximation

While the features (Table 1) used in our previous system could be easily computed in real-time and showed a good performance, there was difficulty in discriminating between similar but contextually different human motions as shown in Figure 3. Therefore, we need to find new features to resolve this problem. In this paper, we compute the dominant points on the boundary of the silhouette area to represent human motion more exactly. To do this, we introduce the number of the dominant points as a new feature.

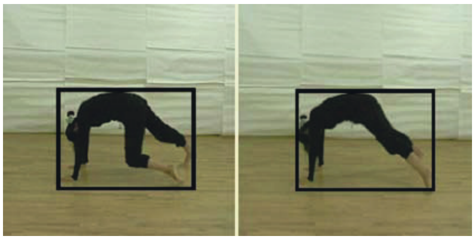


Fig. 3. Similar but contextually different motions. With only a bounding box, we cannot discriminate between two motions.

We use the Teh-Chin's algorithm [2] to detect the dominant points because it has shown reliable results even if the object is dynamically scaled or changed. The Teh-Chin algorithm is as follows: A measure of significance, e.g. curvature must be determined over some region of support. However, there is rarely a sound basis for choosing region of support. The chord length and the perpendicular distance can provide a basis for choosing the appropriate region of support. And the measure of

[2] Laban asserted that human motion include one's temper or propensity. In other words, human beings express themselves and transmit something (Laban called it *effort*) rising form their hearts through performing a motion.

significance of each point is determined by using the neighboring points within the extent of the region of support. The measure of significance and the region of support of each point are then used to guide the selection of points to be removed. The points remaining after the removal process are the dominant points.

Now we can figure out that the dominant points are those points that have a significant change of curvature. This means that an object that has many dominant points is star-like; on the other hand, an object that has few dominant points is circular. The number of dominant points represents the shape complexity of an object. Fig. 4 shows this relationship. The shape of an object has something to do with *space* in Laban's theory.

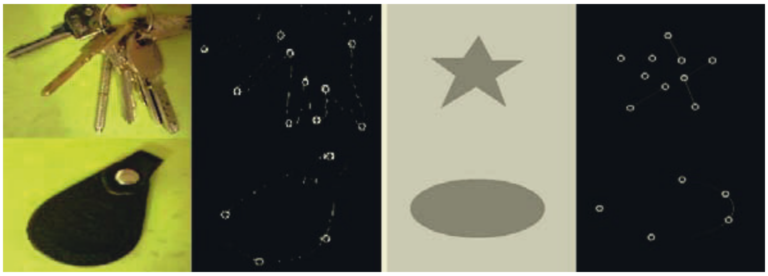


Fig. 4. Dominant point detection on real and synthetic images. The number of dominant points is associated with the complexity of the object.

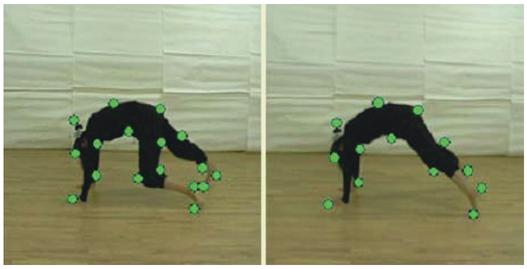


Fig. 5. The difference of the number of dominant points between similar but contextually different motions. In the left, a little motion of her left leg gave rise to more dominant points.

To exactly discriminate the similar but contextually different human motions, we must be able to describe the shape of the motions in detail. To do so, we will have to use the coordinates of respective dominant points. As mentioned earlier, however, we only need to use the number of dominant points because our aim is to catch the overall mood included in the motion of a dance. As shown in Fig. 5, the difference between similar but contextually different human motions can be detected easily even if we only use the number of dominant points. Notice that we could not discriminate the difference between similar but contextually different human motions using the bounding box information in Fig. 3.

4 Experiments and Results

To obtain experimental sequences, we captured the dance motion from 4 professional dancers using a video camera (Cannon MV1). The dancers freely performed the various movements of a dance related to pre-defined emotional categories (*happy*, *surprise*, *angry*, *sad*) within a given period of time. Fig. 6 shows the examples of dance images.

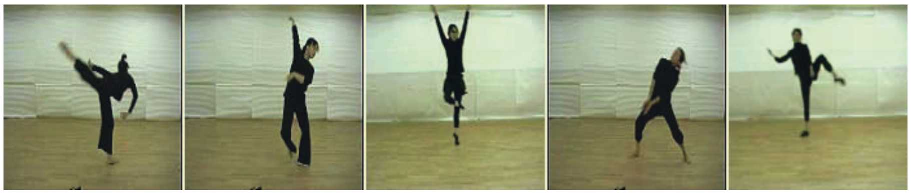


Fig. 6. Dance images. Each dancer performed the movements of a dance freely to express her emotion.

We eliminated the background and shadow of each frame in sequence and extracted binary images using the method previously explained. As shown in Table 2, we extracted 21 (7+7+7) features (of which we used in our previous method [5], we made S_s and S_r into one, i.e. ratio between them, and added N_d as a new one) representing dancing motion and applied SVD to them. We calculated the contribution coefficients (see Appendix) for 12 eigen-vectors having large eigen-values and extracted 12 features weighted with the contribution measure of every frame. Finally, we buffered 24 weighted features with one delay and exploited them as the input of the TDMLP. Then the TDMLP learned to map the weighted features to predefined emotional categories (*happy*, *surprise*, *angry*, *sad*) and was tested (= used) to recognize the emotion that an arbitrary dance image sequences represents. The TDMLP consists of 24 input nodes, 96 hidden nodes, and 4 output nodes. For more details about the experimental environments, refer to [5].

Table 2. Features used in the proposed method.

The aspect ratio of rectangle	H/W
The coordinate of centroid	(C_x, C_y)
The coordinate of the center of rectangle	(R_x, R_y)
The ratio between silhouette area and rectangle area	S_s/S_r
The number of dominant points on boundary	N_d
The velocity of each feature	$f(\cdot)$
The acceleration of each feature	$g(\cdot)$
$f(x_n) = x_n - x_{n-1}$, $g(x_n) = x_n - 2*x_{n-1} + x_{n-2}$	

Table 3. Recognition rate inside the training sequence.

Input \ Output	Happy	Surprised	Angry	Sad
Happy	88%(79%)[3]	2%(11%)	10%(8%)	0%(2%)
Surprised	2%(3%)	80%(78%)	12%(11%)	6%(8%)
Angry	12%(23%)	4%(0%)	84%(70%)	0%(7%)
Sad	0%(0%)	12%(7%)	2%(1%)	86%(92%)

Table 4. Recognition rate outside the training sequence.

Input \ Output	Happy	Surprised	Angry	Sad
Happy	75%(52%)	14%(16%)	11%(24%)	0%(8%)
Surprised	14%(16%)	70%(76%)	11%(0%)	5%(8%)
Angry	14%(14%)	15%(12%)	60%(64%)	11%(10%)
Sad	0%(0%)	11%(4%)	2%(2%)	87%(94%)

Table 3 and 4 shows the cross-recognition rate of four emotions to the dance image sequences inside and outside the training sequence respectively. In Table 3, we trained with sequences from all four dancers and tested with sequences again from all dancers. In Table 4, we tried to train with three dancers and test with the 4th dancer. The proposed method shows improved performance (higher and balanced recognition rate) than our previous system both inside and outside the training sequence. It is clear that this improvement results from using the new contour-based shape information, i.e. N_d because we used the same features excepted for this information in our previous method. Significantly, the proposed system can discriminate between some differences (e.g. the difference between *happy* and *surprise* in Table 3) that the previous method could not.

5 Conclusion

We have presented a new approach for recognizing human emotion from dance image sequence. A key characteristic of our approach is the use of contour-based shape information together with region-based shape information. Thus our method could recognize the difference between similar but contextually different human motions that have the same bounding box information. Consequently, our method noticeably improved the performance of emotion recognition compared with the previous ones that only use the region-based shape information.

It was confirmed that recognizing human emotion using not physical entities but approximated features based on Laban's theory is feasible and shows acceptable performance (above 70% recognition rate in outside the training sequence).

[3] A parenthesized value presents the result of our previous method [5].

Currently, we are continuing this line of research by trying to recognize human emotions from traditional dance performance that are a little bit different from modern ones.

Acknowledgement

This work was supported by the research fund of Hanyang University (HY-2002-T).

References

1. Kim, Y.-S.: Shape Descriptor for Content-Based Image Retrieval. Ph.D. dissertation, Hanyang University. (2000)
2. Teh, C.-H., Chin, R.T.: On the Detection of Dominant Points on Digital Curves. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.11, no.8. (1989) 859-872
3. Suzuki, R., Iwadate, Y., Inoue, M., Woo, W.: MIDAS: MIC Interactive Dance System. IEEE Intl Conf. on Systems, Man and Cybernetics, vol.2. (2000) 751-756
4. Woo, W., Park, J.-I., Iwadate, Y.: Emotion Analysis from Dance Performance Using Time-Delay Neural Netwroks. Proc. of CVPRIP'00, vol.2. (2000) 374-377
5. Park, H., Park, J.-I., Kim, U.-M., Woo, W.: A Statistical Approach for Recognizing Emotion from Dance Sequence. Proc. of ITC-CSCC'02, vol.2. Thailand (2002) 1161-1164
6. Camurri, A., Ricchetti, M., Trocca, R.: Eyeweb-toward gesture and affect recognition in dance/music interactive system. Proc. IEEE Multimedia Systems. (1999)
7. Kim, N., Woo, W., Tadenuma, M.: Photo-realistic Interactive Virtual Environment Generation Using Multiview Cameras. Proc. of SPIE PW-EI-VCIP'01, vol.4310. (2001)
8. Open Source Computer Vision Library. http://www.intel.com
9. Laban, R.: Modern Educational Dance. Trans-Atlantic Publications. (1988)
10. Kojima, K., Otobe, T., Hironaga, M., Nagae, S.: A Human Motion Analysis Using the Rhythm. Proc. of IWRHIC'00, Japan. (2000) 190-193
11. Wilson, A., Bobick, A., Cassell, J.: Temporal Classification of Natural Gesture and Application to Video Coding. Proc. of CVPR'97. (1997) 948-954
12. Fuji, R., Matsumoto, K., Mitsuyoshi, S., Gai, L.: Researches on the emotion measurement system. Proc. of ICSMC'03, vol.2. (2003) 1666-1672
13. Cohen, I., Sebe, N., Cozman, F., Cirelo, M., Huang, T.: Learning Bayesian Network Classifier for Facial Expression Recognition using both Labeled and Unlabeled Data. Proc. of CVPR'03, vol.1. (2003) 595-601
14. Lee, K., Xu, Y.: Real-time Estimation of Facial Expression Intensity. Proc. of ICRA'03, vol.2. (2003) 2567-2572
15. Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., Taylor, J.G.: Emotion Recognition in Human-Computer Interaction. IEEE Signal Processing Magazine. (2001) 32-80
16. Picard, R., Vyzas, E., Healey, J.: Toward Machine Emotional Intelligence: Analysis of Affective Physiological State. IEEE Transaction on Pattern Analysis and Machine Intelligence, vol.23, no.10. (2001) 1175-1191

Appendix: Contribution Coefficient Vector

In case of extracting n features from the dance sequence having m frames, we apply SVD to the measurement matrix ***F*** that has m×n features as its elements:

$$\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T$$

The r eigen-vectors ($\boldsymbol{u_1}$, $\boldsymbol{u_2}$, …, $\boldsymbol{u_r}$) associated with large eigen-values are represented by linear combination of the feature vectors ($\boldsymbol{f_1}$,$\boldsymbol{f_2}$, …,$\boldsymbol{f_n}$). That is:

$$\begin{aligned}
\mathbf{u_1} &= \alpha_{11}\mathbf{f_1} + \alpha_{12}\mathbf{f_2} + \cdots + \alpha_{1n}\mathbf{f_n} \ , \\
\mathbf{u_2} &= \alpha_{21}\mathbf{f_1} + \alpha_{22}\mathbf{f_2} + \cdots + \alpha_{2n}\mathbf{f_n} \ , \\
\vdots & \qquad\qquad \vdots \\
\mathbf{u_r} &= \alpha_{r1}\mathbf{f_1} + \alpha_{r2}\mathbf{f_2} + \cdots + \alpha_{rn}\mathbf{f_n} \ .
\end{aligned}$$

This is rewritten as follows:

$$\mathbf{u_i} = \mathbf{F}\boldsymbol{\alpha}_\mathbf{i} \quad \textit{for } i = 1,\, 2,\, \cdots,\, r,$$

Here, we call α_i contribution coefficient vector and it's solved as follows:

$$\boldsymbol{\alpha}_\mathbf{i} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{u_i}.$$

Matching Concavity Trees

Ossama El Badawy and Mohamed Kamel

Pattern Analysis and Machine Intelligence Laboratory
Department of Systems Design Engineering
University of Waterloo, Waterloo, ON, N2L 3G1, Canada
{obadawy,mkamel}@pami.uwaterloo.ca

Abstract. Concavity trees are structures for 2-D shape representation. In this paper, we present a new recursive method for concavity tree matching that returns the distance between two attributed concavity trees. The matching is based both on the structure of the tree as well as on the attributes stored at each node. Moreover, the method can be implemented on parallel architectures, and it supports occluded and partial matching. To the best of our knowledge, this is the first work to detail a method for concavity tree matching. We test our method on 625 silhouettes in the context of shape-based nearest-neighbour retrieval.

1 Introduction

A new concavity-tree extraction algorithm and a method for shape-based image retrieval have been proposed in [1]. This paper complements that work by detailing a concavity-tree matching method. Together, the two papers constitute, to the best of our knowledge, the first work that reports on the applicability of concavity trees to shape-based image retrieval. The proposed matching method is well suited to labelled concavity trees, but is by no means confined to them; it can be applied to the matching of any labelled trees.

A *concavity tree* is a data structure used for describing non-convex two dimensional shapes. It was first introduced by Sklansky [2] and has since been further researched by others [3–6]. A concavity tree is a rooted tree where the root represents the whole object whose shape is to be analysed/represented. The next level of the tree contains nodes that represent concavities along the boundary of that object. The following level contains nodes, each representing one of the concavities of its parent, *i.e.*, its meta-concavities. If an object or a concavity is itself convex, then the node representing it does not have any children. Typically, each node in a concavity tree stores attributes, or features, describing the underlying object or concavity. Concavity trees are clearly a tool for structural pattern recognition in which a pattern (a 2-D shape in our case) is decomposed into its primitive components. Figure 1 shows an example of a shape (a), its convex hull, concavities, and meta-concavities (b), and its corresponding concavity tree (c). The shape has $five$ concavities as reflected in level *one* of the tree. The four leaf nodes in level *one* correspond to the highlighted triangular concavities shown in (d), whereas the non-leaf node corresponds to the (non-convex)

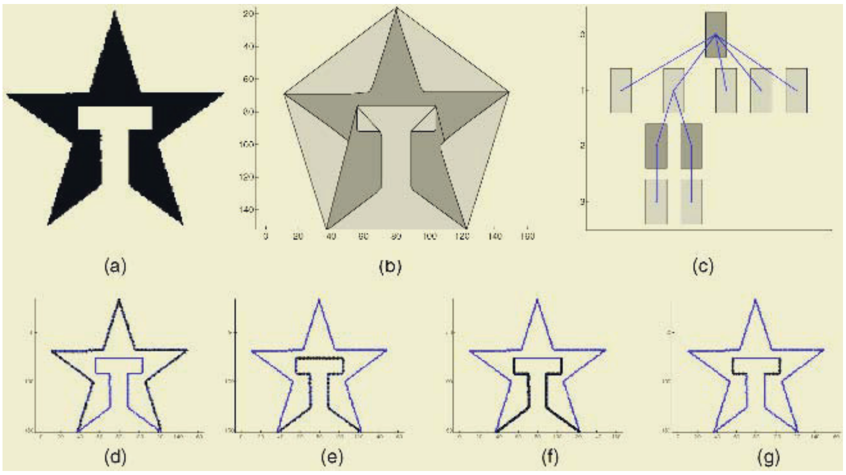


Fig. 1. An object (a), its convex hull and concavities (b), the corresponding concavity tree (c), and contour sections corresponding to concavities (d-g).

concavity shown in (e). Similarly, the nodes in levels *two* and *three* correspond to the meta-concavities highlighted in (f) and (g), respectively. The tree shown in (c) was extracted using the contour-based algorithm presented in [1]. That algorithm is able to deal with the many minor "noisy" concavities seen along the boundary of the shape in (a). In addition to the feature vector labelling the tree nodes, other "helper" features are computed by the tree extraction algorithm and stored at each node (as the tree is being constructed.) They include the level of the node, the height, number of nodes, number of leaves in the subtree rooted at the node, as well as the relative size of the concavity (represented by the node) with respect to its parent.

Structural matching of trees is computationally expensive, especially if the trees are not rooted. Three (structural) tree matching classes exist [7]: perfect matching (isomorphism), partial matching, and similarity matching. The third class is more useful for classification and retrieval purposes. Inexact and heuristic methods for similarity matching of trees exist in the literature in many contexts including shock tree matching [8], remote sensing [9], and VLSI [10].

The decision of whether two objects have similar shapes or not is based on both the shape of the whole object as well as the shapes of the objects' primitives. Moreover, shape can be described using high- and low-level features. The main goal of the proposed matching method is to enhance image similarity retrieval by incorporating high-level structural shape information (trees) as well as low-level shape information (labels) of the whole object and its primitives into the matching process. The analysis and matching of single-object logo images is one viable application of the proposed method.

2 Concavity-Tree Matching

This section details the proposed concavity-tree matching method. Suppose we would like to find a measure of similarity between two general labelled trees T and S. We assume the following:

- Number of nodes in T (S) is n (m), where $n > 0$ ($m > 0$.)
- Number of subtrees rooted in level 1 in T (S) is N (M), where $N \geq 0$ ($M \geq 0$.)
- $M \geq N$. If this is not the case, we just switch the names of the two trees. (Note that $M \geq N$ does not necessarily imply that $m \geq n$.)
- $A = \{T^{(1)}, \cdots, T^{(N)}\}$ and $B = \{S^{(1)}, \cdots, S^{(M)}\}$ are the sets of subtrees rooted in level *one* of T and S, respectively.
- Nodes in T and S are labelled with l-dimensional feature vectors $\mathbf{x}_i = (x_{i_1}, \cdots, x_{i_l})$ and $\mathbf{y}_j = (y_{j_1}, \cdots, y_{j_l})$, where i and j denote the nodes of T and S, respectively, $0 \leq i < n$, and $0 \leq j < m$.
- $\mathbf{x}_0$ and $\mathbf{y}_0$ are the attributes of the root nodes of T and S, respectively.
- Each of the individual features, x_i (y_i) in vector $\mathbf{x}$ ($\mathbf{y}$) is bounded between *zero* and a maximum value max_i, $1 \leq i \leq l$.

The proposed method computes a distance $\mathcal{D}(T, S)$ between the two trees as a measure of dissimilarity. The closer the distance is to *zero*, the more the two shapes are similar and the closer it is to *one*, the more dissimilar they are.

In matching the two trees, we can identify four cases as follows:

1. $N = M = 0$.
2. $N = M, N > 0, M > 0$.
3. $N = 0, M > 0$.
4. $N > 0, M > 0, M > N$ (the most general case.)

2.1 Case 1, $N = M = 0$

This case is the simplest case (see Figure 2a); there is no structural matching and the distance between T and S is

$$\mathcal{D}(T, S) = a\, d(\mathbf{x}_0, \mathbf{y}_0) \tag{1}$$

where a is a constant between *zero* and *one* (discussed later.) Typically, $a = 1/3$. $d(\mathbf{x}, \mathbf{y})$ is defined as follows.

$$d(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{\sum_{i=1}^{l} max_i^2}} \sqrt{\sum_{i=1}^{l} (x_i - y_i)^2} \tag{2}$$

That is, $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$ normalized by the maximum possible distance between $\mathbf{x}$ and $\mathbf{y}$. This will guarantee that $d(\mathbf{x}, \mathbf{y})$ is bounded between *zero* and *one*.

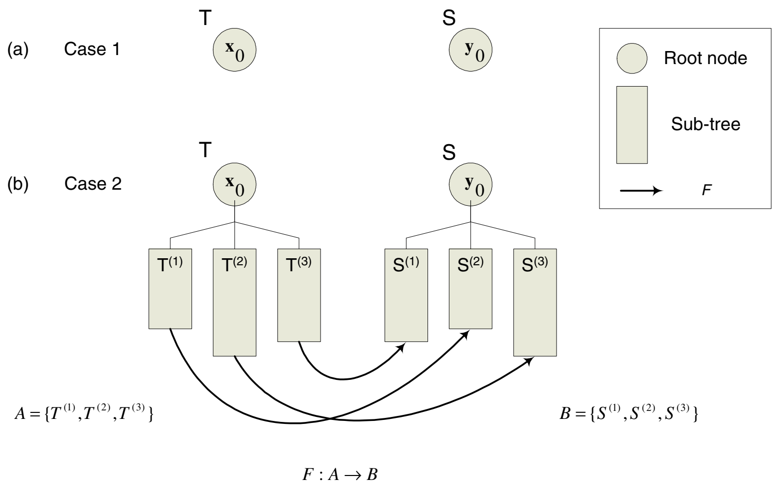


Fig. 2. Concavity-tree matching. Case 1 (a), case 2 (b).

2.2 Case 2, $N = M, N > 0, M > 0$

Here, the first step toward matching the two trees is to find a bijective mapping function $F : A \rightarrow B$ that relates each $T^{(k)}$ in A, $1 \leq k \leq N$, to its "closest" match in B. This mapping is based on the distance between the $\mathbf{x}_0$ and $\mathbf{y}_0$ as well as on other features such as the height and total number of nodes in the subtrees. The goal is to find the best possible match between the components of each pair in the mapping. This case is shown in Figure 2b where, as an example, $M = N = 3$. The distance between T and S is then defined as

$$\mathcal{D}(T, S) = a\, d(\mathbf{x}_0, \mathbf{y}_0) + b \left[\frac{1}{N} \sum_{k=1}^{N} \mathcal{D}(T^{(k)}, F(T^{(k)})) \right] \qquad (3)$$

where a is the same constant as in case 1 and $0 < b < 1 - a$.

2.3 Case 3, $N = 0, M > 0$

In this case, T is a single (root) node and S is not. This is depicted in Figure 3a where S has, as an example, $M = 3$ subtrees. To match the two trees, we need to make the number of subtrees, N and M, equal. We therefore add, at level *one* of T, M dummy single-node subtrees $T^{(k)}$, $1 \leq k \leq M$, and we define

$$A' = \{T^{(1)}, \cdots, T^{(M)}\} \qquad (4)$$

Each node in the added subtrees is labelled with a dummy feature vector $\mathbf{v}$. By definition, $d(\mathbf{v}, \mathbf{y}) = 1$.

A bijective function $G : A' \rightarrow B$ is then defined such that the concavity represented by $G(T^{(i)})$ has an area (relative to its parent) larger than or equal to the area of the one represented by $G(T^{(j)})$, for $i > j$.

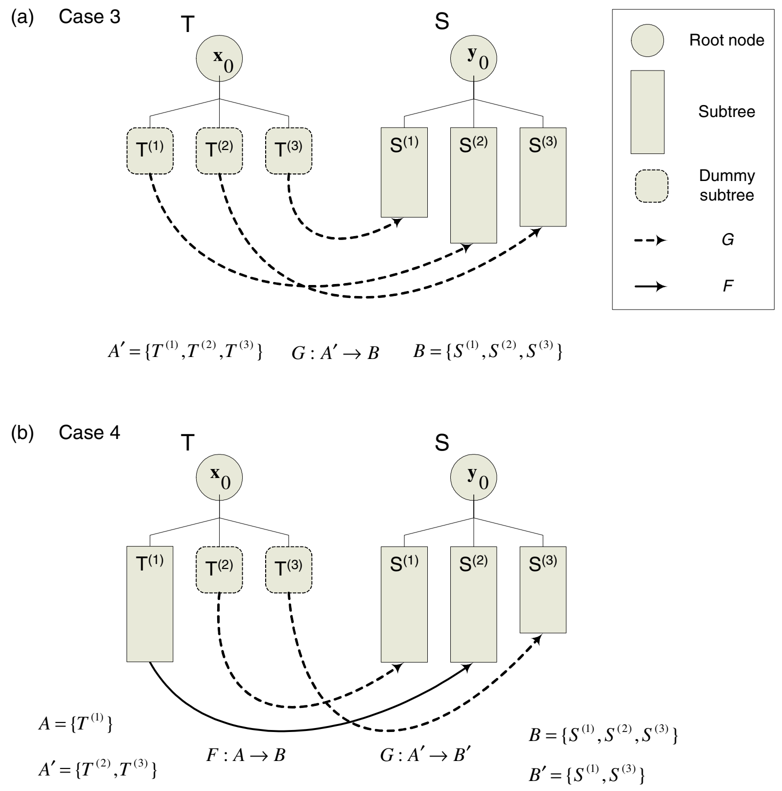


Fig. 3. Concavity-tree matching. Case 3 (a), case 4 (b).

The distance between T and S is given by

$$\mathcal{D}(T,S) = a\, d(\mathbf{x}_0, \mathbf{y}_0) + c\left[h \sum_{k=1}^{M} (1-h)^{k-1}\, \mathcal{D}(T^{(k)}, G(T^{(k)}))\right] \tag{5}$$

where a is the same constant as in cases 1 and 2, $c = 1 - a$, and $0 < h < 1$ (typically 0.5, discussed later.)

2.4 Case 4, $N > 0, M > 0, M > N$

Case 4 (Figure 3b) is the most general one and it combines the previous cases. We first find a function $F : A \to B$ that relates each $T^{(k)}$ in A, $1 \leq k \leq N$, to its "closest" match in B. F in this case is an injective function. That is, $M - N$ elements in B are not in the range of F. Let

$$B' = \{S^{(k)} :\ S^{(k)} \notin \text{range of } F\} \tag{6}$$

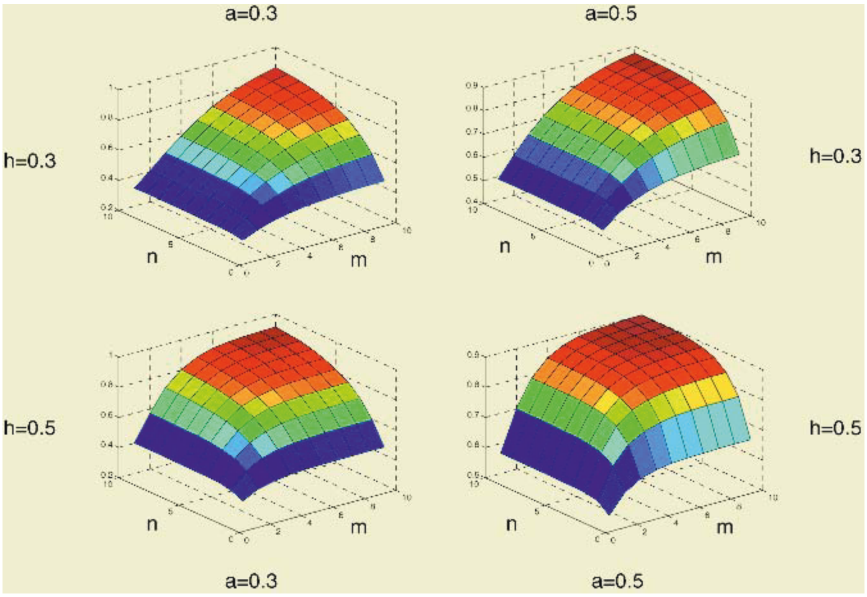


Fig. 4. The distance between a single-node tree and a tree with height n whose nodes (except leaves) have m children.

The next step is to add, at level *one* of T, $M-N$ dummy single-node subtrees $T^{(k)}$, $N < k \leq M$, and to define

$$A' = \{T^{(k)} \,:\, N < k \leq M\} \tag{7}$$

As in case 3, we define a bijective function $G : A' \rightarrow B'$ such that the concavity represented by $G(T^{(i)})$ has an area (relative to its parent) larger than or equal to the area of the one represented by $G(T^{(j)})$, for $i > j$. The distance between T and S is then given by

$$\mathcal{D}(T,S) = a\,d(\mathbf{x}_0,\mathbf{y}_0) + b\left[\frac{1}{N}\sum_{k=1}^{N}\mathcal{D}(T^{(k)},F(T^{(k)}))\right] + c\left[h\sum_{k=N+1}^{M}(1-h)^{k-N-1}\,\mathcal{D}(T^{(k)},G(T^{(k)}))\right] \tag{8}$$

where a is as defined in cases 1, 2, and 3, h is as defined in case 3, and b and c are real positive constants (weights) such that $a+b+c=1$.

3 Discussion and Examples

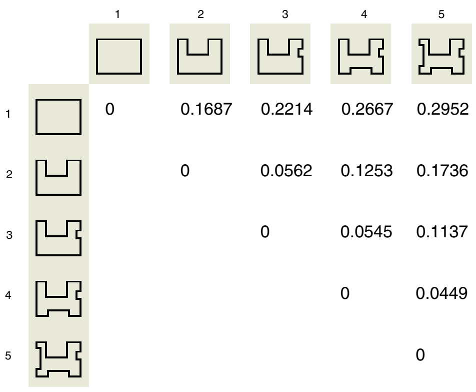
In this section, we examine the role of the method parameters and their effects on the matching process. Four parameters are involved in the tree distance measure, namely, a, b, c, and h. The first three parameters determine the weight given in the matching process to the overall shape attributes (a), the shapes of the underlying primitives (concavities) (b), and the cost of inserting new nodes in the smaller tree (c). Typically, a is first chosen such that it reflects the extent to which the similarity is based on the attributes in the root node, then b and c are chosen such that $a + b = 1$ (case 2), $a + c = 1$ (case 3), or $a + b + c = 1$ (case 4). In case 4, one possible variation is to choose b and c such that their ratio is $N/(N + M)$. If a and c are chosen to be much smaller than b, this will give more weight in the distance to the common part of the objects and hence allow for occluded matching.

Prior to recursively calling itself M times, each call to the matching method requires $1 + MN$ feature vector comparisons, resulting in a complexity of $O(l\,M^2)$, including any sorting operation. Consequently, a worst-case time complexity for the whole matching would be $O(l\,M^{H+1})$, where H is the height of the "higher" tree, and l, as defined before, is the dimensionality of the feature vector labelling the nodes of the trees. M here is the maximum branching factor in the two trees, *i.e.*, the number of children of the node with the highest number of children in any of the two trees. Clearly, this is an upper bound that is rarely reached (in the context of concavity trees) unless every node in both trees has M children. Given that the number of nodes m in a tree is at most $(M^{H+1} - 1)/(M - 1)$, which is $O(M^H)$, then the matching complexity, which was seen to be $O(l\,M^{H+1})$, can also be expressed as $O(l\,m\,M)$. Moreover, because they are independent, recursive calls to $\mathcal{D}(T, S)$ can be done in parallel, which significantly improves the matching speed.

Figure 4 shows the distance between two trees, one is a single-node tree and the other has a height of n and its non-leaf nodes each has m children. The distance between the feature vectors in the root of the two trees is assumed to be 0.8. The plots show that as n and m increase, *i.e.*, as the second tree grows, the distance approaches one. The rate of distance increase as a function of an increase of n and/or m can be controlled by adjusting the values of a and h.

$\mathcal{D}(T, S)$ is bounded by 1. What makes this characteristic suitable for shape matching using concavity trees is the following. If T_1, T_2, and T_3 are three trees such that T_1 has say 10 nodes, T_2 has 100 nodes, and T_3 has 110 nodes, $\mathcal{D}(T_1, T_2)$ is approximately the same as $\mathcal{D}(T_1, T_3)$ (slightly smaller). Which is desirable, since both T_2 and T_3 are much different from T_1. Another important characteristic is that as we move down the tree, each level gets less and less weight in the overall distance. So all the level is treated in a similar way. This is different from other tree matching algorithm where the weight of each node decreases monotonically both horizontally and vertically.

Figure 5 shows the pair-wise distances between 5 shapes. We note that the distance increases in each row from left to right and decreases in each column downward (as would be desired). Moreover, the closest non-identical pairs are in order: (4,5), (3,4), (2,3), and (1,2), which is also desired.



	1	2	3	4	5
1	0	0.1687	0.2214	0.2667	0.2952
2		0	0.0562	0.1253	0.1736
3			0	0.0545	0.1137
4				0	0.0449
5					0

Fig. 5. The progression of the pair-wise distance between similar shapes.

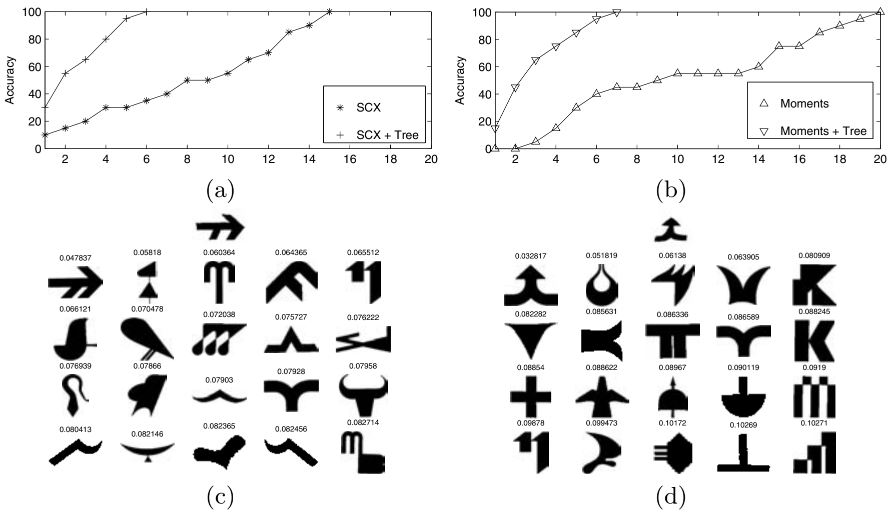


Fig. 6. Performance curves and query example.

4 Experimental Results

We use a database of 625 logo images to assess the new distance measure. We label the trees with two sets of attributes, namely SCX (Solidity, Eccentricity, and eXtent) [11] and moment invariants. We use 50 hand-sketched query images to test the retrieval performance with and without trees. In Figure 6 (a) and (b), the vertical axis is the percentage of the query images that were in the first k (horizontal axis) retrievals. In the plot for example, about 30% of the 50 query images returned the correct database image as the first hit (using SCX labelled trees). From the plots, it is clear that the performance of the SCX feature was improved by 20% when used in conjunction with concavity trees (the accuracy actually tripled). The performance of moment invariants was similarly boosted by 15%. Figure 6 (c) and (d) show two sample queries that returned the target image in first position (using SCX labelled trees.) Using just SCX, the target images were originally in the 13^{th} and 21^{st} positions, respectively. The distance (to the query the image) is shown above each retrieved image.

5 Conclusion

This paper detailed a new recursive method for 2-D shape matching based on matching concavity trees. The method is suitable for concavity trees but can be applied to other attributed trees as well. Experiments using the proposed matching method show an increase of retrieval performance by at least 15%.

References

1. El Badawy, O., Kamel, M.: Shape retrieval using concavity trees. In: Proceedings of the International Conference on Pattern Recognition. (2004)
2. Sklansky, J.: Measuring concavity on a rectangular mosaic. IEEE Transactions on Computers **C-21** (1972) 1355–1364
3. Batchelor, B.: Hierarchical shape description based upon convex hulls of concavities. Journal of Cybernetics **10** (1980) 205–210
4. Borgefors, G., Sanniti di Baja, G.: Methods for hierarchical analysis of concavities. In: Proceedings of the International Conference on Pattern Recognition. Volume 3. (1992) 171–175
5. Borgefors, G., Sanniti di Baja, G.: Analyzing nonconvex 2D and 3D patterns. Computer Vision and Image Understanding **63** (1996) 145–157
6. Xu, J.: Hierarchical representation of 2-D shapes using convex polygons: A morphological approach. Pattern Recognition Letters **18** (1997) 1009–1017
7. Sanfeliu, A.: Matching tree structures. In Bunke, H., Sanfeliu, A., eds.: Syntactic and Structural Pattern Recognition, Theory and Applications. World Scientific (1990) 146–178
8. Torsello, A., Hancock, E.: Computing approximate tree edit distance using relaxation labeling. Pattern Recognition Letters **24** (2003) 1089–1097
9. Peura, M., Saltikoff, E., Syrjasuo, M.: Image analysis by means of attribute trees-remote sensing applications. In: Proceedings of the International Geoscience and Remote Sensing Symposium. (1999) 696–698
10. Sitaraman, K., Ranganathan, N., Ejnioui, A.: A vlsi architecture for object recognition using tree matching. In: Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures and Processors. (2002) 325–334
11. El Badawy, O., Kamel, M.: Shape-based image retrieval applied to trademark images. International Journal of Image and Graphics **2** (2002) 375–393

Combining Classifier for Face Identification at Unknown Views with a Single Model Image

Tae-Kyun Kim[1] and Josef Kittler[2]

[1] HCI Lab., Samsung Advanced Institute of Technology, Yongin, Korea
taekyun@sait.samsung.co.kr
[2] Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, UK
J.Kittler@surrey.ac.uk

Abstract. We investigate a number of approaches to pose invariant face recognition. Basically, the methods involve three sequential functions for capturing nonlinear manifolds of face view changes: representation, view-transformation and discrimination. We compare a design in which the three stages are optimized separately, with two techniques which establish the overall transformation by a single stage optimization process. In addition we also develop an approach exploiting a generic 3D face model. A look-up table of facial feature correspondence between different views is applied to an input image, yielding a virtual view face. We show experimentally that the four methods developed individually outperform the classical method of Principal Component Analysis(PCA)-Linear Discriminant Analysis(LDA). Further performance gains are achieved by combining the outputs of these face recognition methods using different fusion strategies.

1 Introduction

Face recognition has a benefit over other biometric techniques such as fingerprint and iris recognition in that humans can be identified without notice and at distance. However, to realize this potential, it is essential to counteract the degradation in performance exhibited by face recognition systems for views different from the frontal pose. View-changes can be learned from prototype faces and the learned models can be applied to other individuals.

Classically, a generic 3D model of a human face has been used to synthesize face images from different view points[8] and approximate models, such as a cylinder, have also been applied to face recognition. More recently, Vetter and Poggio[1] showed that the 2D image based technique is a viable method for view synthesis and recognition of face classes. In their work, face images are first represented in the view-subspace and the transformation matrix between the different view representations is computed in the sense of Least Square Error(LSE). Blanz[4] utilized a 3D morphable model and Yongmin Li[2] applied Kernel Discriminant Analysis and 3D Point Distribution Model for view-invariant face recognition. In spite of the recent successes, all the above methods have a strong drawback in requiring dense correspondence of facial features for image normalization. The step of feature detection or correspondence solving, which is needed for separating the shape and texture components of face images in these methods, is usually difficult itself. Errors in correspondences seriously degrade the overall face recognition performance of these methods as shown in [4]. Among other relevant works, Graham and Allinson[3] applied a Neural Network for learning the view transfer function of the normalized face images with a fixed eye position. Talukder[6] also proposed the method for the simply normalized images by using fixed eye points, which involves a linear view-transfer matrix obtained by the LSE method[1].

In this paper, we propose robust base classifiers for face identification at unknown views and a combining classifier for accuracy improvement. It is assumed that a single model image is given and face images are registered with reference to the eye positions. The classifiers differ in the way they model face view-changes. They can be categorized into methods based on statistical learning of face images at different poses and methods based on 3D face models. The two piecewise linear methods and the nonlinear kernel method are adopted as the base classifiers, which are based on statistical learning of face images. In addition, a computationally efficient approach, which stores the correspondence information of 3D face models at different views in a look-up table, is also developed for complementing the statistical learning methods. These base classifiers are quite different in their nature owing to different sources of information and architectures used. This motivates us to combine them for further accuracy improvement.

2 Base Classifier Design

There are a number of factors that cause the face data distribution of different poses to be nonlinear; this naturally motivates us to exploit the benefits of non-linear architectures. The existing view-invariant face recognition methods can generally be decomposed into three sequential steps: representation, view-transformation and discrimination function. First an input face image is projected into a view subspace via a function, **S,** which is obtained by linear or nonlinear subspace analysis of face images within a certain range of view-angles, as

$$\mathbf{b}_{v,i} = \mathbf{S}_v(\mathbf{x}_{v,i}, \mathbf{avg}_v) \tag{1}$$

where $\mathbf{x}_{v,i}$ is the *i*-th face image in the set drawn from a certain small range of views, *v*. A linear matrix (LM)[1] or Neural Network[3] can be utilized to learn the transfer function **V** between the different view representations in the sense of LSE,

$$\min_{\mathbf{V}} \sum_{i=1}^{N} \left| \mathbf{b}_{f,i} - \mathbf{V}(\mathbf{b}_{r,i}) \right|^2 \tag{2}$$

where *f* and *r* denote a frontal-view and a rotated-view respectively and *N* is the number of images. The face images transformed to the frontal-view and the original frontal view faces are the input for learning a discriminant function **D**. LDA or Generalized Discriminant Analysis(GDA)[7] can be applied to learn the function mapping the pose corrected face images into discriminative feature vectors, i.e.

$$\mathbf{d}_{f,i} = \mathbf{D}(\mathbf{b}_{f,i}, \mathbf{avg}), \quad \mathbf{d}_{r,i} = \mathbf{D}(\mathbf{V}(\mathbf{b}_{r,i}), \mathbf{avg}) \tag{3}$$

where **avg** is a global mean. The final classification is based on the nearest neighbor matching of the feature vectors $\mathbf{d}$. As the performance of this system depends on the choice of each transformation function, various combinations of linear and nonlinear functions obtained by statistical learning have been compared in [10]. The study showed that the piecewise linear combinatorial method, "PCA(as a **S**)-LM(as a **V**)-LDA(as a **D**)" is one of the most accurate classifiers. As its computational cost is low, PCA-LM-LDA has been adopted as a base classifier in this study. However, it should be noted that this combinatorial method must yield a sub-optimal solution since each step is separately trained.

A novel nonlinear discriminant analysis, called "Locally Linear Discriminant Analysis(LLDA)", has been developed to provide a unified framework for the three stage structure. It concurrently finds the set of locally linear transformations to yield locally linearly transformed face classes that maximize the between-class covariance while minimizing the within-class covariance as shown in Figure 1.

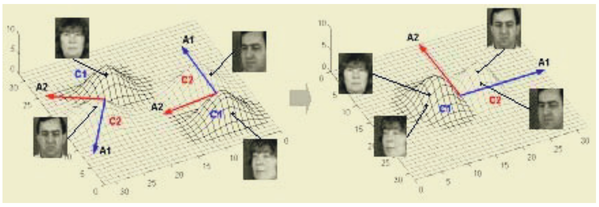


Fig. 1. LLDA for pose-invariant face classification; Left shows the original data distribution and the found components and right shows the transformed data distribution.

The solutions for the frontal and rotated face images found by this method, $\mathbf{U}_f$ and $\mathbf{U}_r$ respectively, correspond to the combined three stage transformation function as discussed above, i.e.

$$\begin{aligned} &\mathbf{d}_{f,i} = \mathbf{U}_f(\mathbf{x}_{f,i}, \mathbf{avg}_f), \quad \mathbf{d}_{r,i} = \mathbf{U}_r(\mathbf{x}_{r,i}, \mathbf{avg}_r), \\ &\rightarrow \ \mathbf{U}_f \equiv \mathbf{D}(\mathbf{S}_f(\cdot)), \quad \mathbf{U}_r \equiv \mathbf{D}(\mathbf{V}(\mathbf{S}_r(\cdot))\cdot \end{aligned} \tag{4}$$

For details of the novel algorithm, LLDA, please refer to the study[5]. LLDA and PCA-LM-LDA will be discussed in more detail in the experimental sections.

Generalised Discriminant Analysis (GDA), which transforms the input space into a high-dimensional feature space by using a kernel function Φ and then linearly separates the data, is also developed. The major difference from the above piece-wise linear methods is that a single nonlinar transformation function $\mathbf{U}^{\Phi}$ such that

$$\mathbf{d}_{f,i} = \mathbf{U}^{\Phi}(\Phi(\mathbf{x}_{f,i}, \mathbf{avg})), \quad \mathbf{d}_{r,i} = \mathbf{U}^{\Phi}(\Phi(\mathbf{x}_{r,i}, \mathbf{avg})) \tag{5}$$

is applied to different view face images while different sets of linear functions are exploited for different view faces in the two piecewise linear classifiers in (3) and (4).

Although the methods based on statistical learning of 2D images are effective for capturing face view changes, a complementary benefit of the more classical method based on 3D face models has also been investigated. We propose to replace the processes of texture mapping, 3D rotation and rendering in graphics with a direct image transformation based on a look-up table(LUT) as shown in Figure 2. By using the average LUT, rotated face images are virtually generated from the frontal face images; Intensity of a pixel of a rotated face image $I_r(x, y)$ is obtained from that of the corresponding pixel of the frontal image I_f as $I_r(x, y) = I_f(\overline{\mathbf{LUT}}(x, y))$, where $\overline{\mathbf{LUT}}$ is a funcntion which yields a stored coodinates. The view-transformation through the LUT is very fast. The rotation direction from the frontal to an arbitrary angle is more beneficial as most of the pixel information is kept. Each pose group has an average correspondence LUT. After transforming frontal faces to a certain view, LDA is applied to the pairs of the transformed and the original images at the same view yielding the output feature vectors **d**. Consequently, the proposed method deploys the view-specific discriminant functions of LDA.

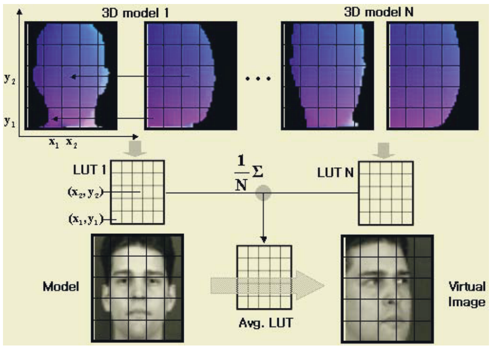


Fig. 2. Virtual View Generation by using the 3D correspondence LUT.

3 Combining Strategies and Experiments

3.1 Combining Techniques

Fusion at the confidence level is considered, where the matching scores reported by the individual classifiers are combined. We have tested the simple fixed combining rules such as the sum, product, maximum, minimum and median rule to access the viability of combining the pose-invariant face classifiers. The use of any trained combiner instead of the fixed rules, provided a suitable evaluation set is available, would be an extension to our work. The confidence value $C_{ij}(\mathbf{x})$ of the base classifier *j* for class *i* is the normalized Euclidean distance of the output vectors **d** produced by the base classifier. The confidence value is scaled by using the small independent evaluation set so that it is in the range of [0,1]. The combining classifier $\mathbf{Q}(\mathbf{x}) = \{\mathbf{Q}_i(\mathbf{x}), i = 1,...,c\}$ is defined as follows:

$$\mathbf{Q}_i = \prod_j \mathbf{C}_{ij}(\mathbf{x}) \quad \mathbf{Q}_i = \sum_j \mathbf{C}_{ij}(\mathbf{x}) \quad \mathbf{Q}_i = \max_j \mathbf{C}_{ij}(\mathbf{x})$$
$$\mathbf{Q}_i = \min_j \mathbf{C}_{ij}(\mathbf{x}) \quad \mathbf{Q}_i = \mathop{median}_j \mathbf{C}_{ij}(\mathbf{x}) \quad \mathbf{Q}_i = \sum_j w_j \mathbf{C}_{ij}(\mathbf{x}) \tag{6}$$

3.2 Experimental Setup

We used the two data sets, XM2VTS[19] as the main set and PIE[12] as the set for further comparison of the base classifiers. XM2VTS data set was annotated with pose labels of the face. The face database consists of 2950 facial images of 295 persons with 5 pose variations (F,R,L,U,D) and 2 different time sessions (S1,S2)(5 months time elapse). This may be the largest public data set which has a sizeable population of subjects taken in different poses. Each pose group has a small view range due to the unexpected error in personal pose. The images were normalized to 46*56 pixel resolution with a fixed eye position. The experimental sets consist of 1250 images of 125 persons, 450 images of 45 persons and 1250 face images of 125 persons for the training, evaluation and test respectively. The training, evaluation and test set have different face identities. The training set was utilized to learn the transformation functions of the base classifiers whereas the evaluation set served to adjust the parameters of the classifier: kernel parameters of GDA, the dimensionality of the output vectors and scaling parameters of the individual classifiers for combining. These were carefully chosen to achieve the best performance of each. The recognition performance is reported as the recognition rate on the test set. The frontal face F-S1 of the test set was selected as a gallery and the 9 rotated face images of the test set were exploited as queries. One more independent protocol was built based on the PIE data set: The selected PIE set consists of 15 images (3 poses x 5 illuminations) of 66 identities as shown in Figure 3. This was equally divided into the training and test set. The frontal face F1 of the test set was selected as a gallery and all the other images of the test set were exploited as queries.

3.3 Performance Comparison of the Base Classifiers

3.3.1 Performance of the Individual Classifiers

All the base classifiers have been tested on the XM2VTS DB. For the method of 3D correspondence LUT, 108 SNU 3D scanned facial models[11] were used. For GDA, an RBF kernel with an adjustable width was deployed. Of the proposed four base classifiers, the two methods, PCA-LM-LDA and 3D LUT, explicitly generate view-rotated images. The characteristics of the two transformation results are quite different as shown in Figure 4. While, the generalization performance of the transformation of the statistical learning based method, PCA-LM-LDA is much degraded for the non-trained individuals, the 3D model based method, 3D LUT, maintains its performance. On the contrary, LLDA and GDA implicitly represent the face images so that the rotated faces have a similar representation to that of the frontal view images. The recognition performance of the base classifiers is shown in Figure 5.

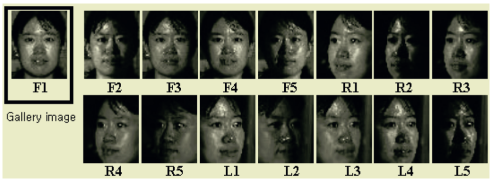


Fig. 3. Sample images of the PIE DB.

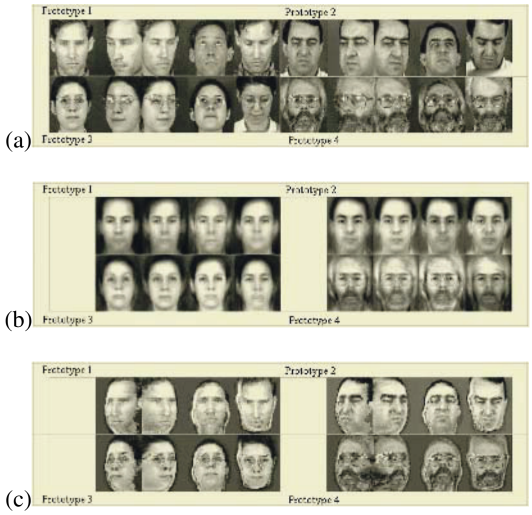


Fig. 4. Examples of the synthesized faces on XM2VTS DB. (a)5 views of the training faces (b)Tranformed faces to a frontal view by PCA-LM-LDA (c)Transformed frontal faces to a rotated view by 3D-LUT.

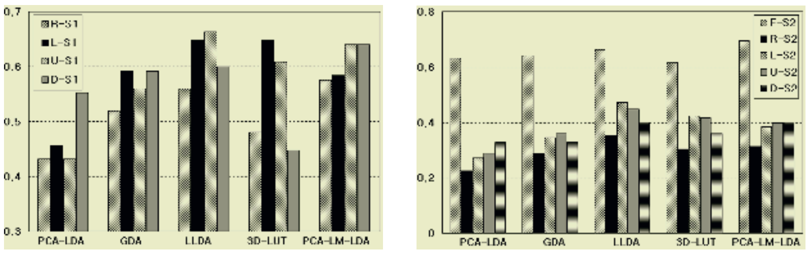


Fig. 5. Recognition rates (in %) of individual experts on XM2VTS DB.

All the four classifiers much outperformed the classical face recognition method, PCA-LDA, where the basis functions of LDA are learned from the eigenfeatures of the training set. The dimensionality of the feature vector at both PCA and LDA stages was carefully controlled to yield its best result. The LLDA method performed best, but the others were also comparable.

3.3.2 PCA-LM-LDA vs. LLDA

We look at the two methods, LLDA and PCA-LM-LDA more closely as the both are trained on the same sources of information and have similar architectures, which are piecewise linear. The PCA-LM-LDA method learns the representation, view-transformation and the discriminant function separately with an indirect objective function for classification. In contrast, in the LLDA, all the procedures are concurrently optimized directly for classification. Their difference is more apparent from the results on a dataset which varies in illumination as well as pose in Table 1. Please refer to the recognition results of the frontal faces by the conventional PCA-LDA for comparison. Illumination changes were relatively well compensated and generalized to novel test faces as compared with pose variations. Some results are ommited here and it is because the results were similar to those of the previous subsection.

Table 1. Recognition rates (in %) on PIE DB.

PCA-LDA		PCA-LM-LDA				LLDA			
		R1	59	L1	44	R1	59	L1	59
F2	85	R2	26	L2	22	R2	41	L2	30
F3	100	R3	56	L3	44	R3	56	L3	63
F4	100	R4	56	L4	37	R4	56	L4	56
F5	67	R5	30	L5	15	R5	30	L5	26
avg	88		45		33		48		47

3.4 Combining Results

Figure 6 shows the combining results of all the 4 base classifiers by the 6 different gating rules. All 6 different gating rules improved the average performance of the best base classifier.

The number of combined experts ranges from 1 up to 4. We first find the best expert, LLDA and then add the next best performing experts, the sequence of which is PCA-LM-LDA, 3D LUT, GDA, yielding the combined results by the sum and product rules. The results are shown in Figure 7. Interestingly, the recognition rate consistently improved as the number of different base classifiers increased. It is also noted that the improvement rate achieved with 2 experts was relatively low in the case of the different session experiment. This might be because the two combined base classifiers, LLDA and PCA-LM-LDA are the most correlated classifiers, due to the similar sources of information used and their piecewise linear structures. In conclusion, the performance improvement achieved by the proposed combining classifier is quite impressive compared with the conventional PCA-LDA method in face recognition: 46.8% $\rightarrow$ 73.2% for the same session and 34.8% $\rightarrow$ 54.4% for the different session respectively.

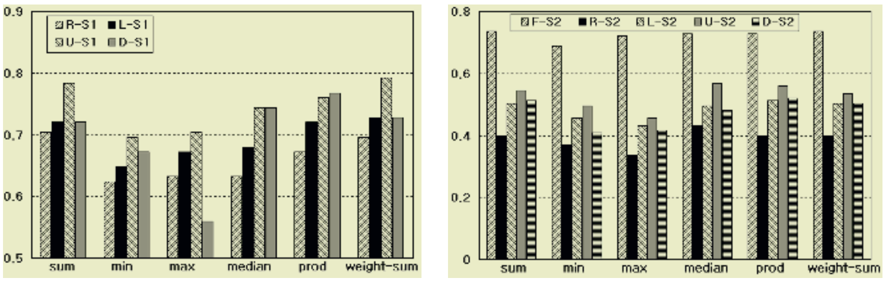


Fig. 6. Recognition rates (in %) of combining classifiers on XM2VTS DB.

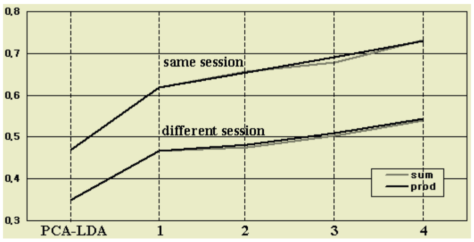


Fig. 7. Average recognition rate for the number of the combined base classifiers.

4 Conclusion

We have proposed a combining classifier based on the modelling of face view-changes. Robust base classifiers are obtained by learning the statistics of 2D images or fitting generic 3D models. The proposed base classifiers outperform the classical method of LDA. The fusion of the different classifiers yields an impressive performance improvement owing to their different characteristics in terms of sources of information exploited and architectures used. We intend to improve the performance of the proposed approach by exploiting dense facial feature correspondences for an image regularization step in the future. The current performance was obtained with the images registered with a fixed eye position and this can be seen as a poor basis of the image normalization for the method.

Acknowledgement

The authors would like to thank Hyun-Chul Kim, Hyung-Soo Lee and Daijin Kim in POSTECH for their helpful discussion. The assistance of Kyung-Ah Sohn, a colleague in SAIT, in implementing the algorithms based on the 3D face models is much appreciated

References

1. T.Vetter and T.Poggio, "Linear object classes and image synthesis from a single example image", IEEE Trans. PAMI, vol. 19, no. 7, pp. 733-742, 1997.
2. Y.Li, S.Gong, and H.Liddell, "Constructing facial identity surfaces in a nonlinear discriminating space", In Proc. of CVPR, 2001.
3. D.B.Graham, N.M.Allinson, "Automatic face representation and classification", In Proc. of British Machine Vision Conference, 1998.
4. V.Blanz,S.Romdhani,T.Vetter, "Face identification across different poses and illuminations with a 3D morphable model", Automatic Face and Gesture Recognition, 2002.
5. T.-K.Kim, J.Kittler, H.-C. Kim and S.-C.Kee, "Discriminant analysis by multiple locally linear transformations", British Machine Vision Conference, pp 123-132, Norwich, UK, 2003.
6. A.Talukder and D. Casasent, "Pose-invariant recognition of face at unknown aspect views", IEEE Joint Conf. on Neural Networks, vol. 5, pp. 3286-3290, 1999.
7. G.Baudat and F.Anouar, "Generalized discriminant analysis using a kernel approach", Neural Computation, vol.12, pp.2385-2404, 2000.
8. A.C.Aitchison and I.Craw, "Synthetic images of faces – an approach to model-based face recognition", British Machine Vision Conference, pp. 226-232, 1991.
9. D.Beymer and T.Poggio, "Face Recognition From One Example View", In Proc. of ICCV, pp. 500-507, 1995.
10. T.-K.Kim, "View-transformation of face images in kernel space-comparative study",Technical Report, Samsung AIT, 2003.
11. B.Choe, H.Lee, and H.-S.Ko, "Performance-driven muscle-based facial animation", The Journal of Visualization and Computer Animation, vol. 12, pp. 67-79, 2001.
12. T.Sim, S.Baker and M.Bsat, "The CMU pose, illumination, and expression (PIE) database", In Proc. of Automatic Face and Gestures Recognition, 2002.
13. K. Okada, C. v. d. Malsburg, "Analysis and synthesis of human faces with pose variations by a parametric piecewise linear subspace method", In Proc. of CVPR, vol. 1 , pp. I -761-8, 2001.
14. A. Pentland, B. Moghaddan, T. Starner, "View-based and modular eigenspaces for face recognition", In Proc. of CVPR, pp. 84-91, 1994.
15. R. Gross, I. Matthews, and S. Baker, "Eigen light-fields and face recognition across pose", In Proc. of Automatic face and Geature Recognition, 2002.
16. A.S.Georghiades, P.N.Belhumeur, and D.J.Kriegman, "From few to many: illumination cone models for face recognition under varialbe lighting and pose", IEEE Trans. on PAMI, vol. 23, no. 6, pp. 643-660, 2001.
17. J. Kittler, M. Hatef, R. P.W. Duin and J. Matas, "On combining classifiers", IEEE Trans. PAMI, vol. 20, no. 3, pp. 226-239, 1998.
18. S. Li, J. Yan, X. Hou, Z. Li and H. Zhang, "Learning low dimensional invariant signature of 3-D object under varying view and illumination from 2-D appearances", In Proc. of ICCV, vol. 1, pp. 635 –640, 2001.
19. K.Messer, J.Matas, J.Kittler, J.Lueettin and G.Maitre, "XM2VTSDB: The extended M2VTS database", In Prof. of Audio and Video-based Biometric Person Authentication, 1999.

A Logodds Criterion for Selection of Diagnostic Tests

Christopher J. Whitaker[1], Ludmila I. Kuncheva[1], and Peter D. Cockcroft[2]

[1] School of Informatics, University of Wales, Bangor, UK
{l.i.kuncheva,c.j.whitaker}@bangor.ac.uk
[2] Department of Clinical Veterinary Medicine, University of Cambridge, UK
pdc24@hermes.cam.ac.uk

Abstract. We propose a criterion for selection of independent binary diagnostic tests (signs). The criterion maximises the difference between the logodds for having the disease and the logodds for not having the disease. A parallel is drawn between the logodds criterion and the standard minimum error criterion. The error criterion is "progression non-monotone" which means that even for independent binary signs, the best set of two signs might not contain the single best sign. The logodds criterion is progression monotone, therefore the selection procedure consists of simply selecting the individually best features. A data set for scrapie in sheep is used as an illustration.

Keywords: feature selection, combining diagnostic tests, independent binary features, logodds criterion, veterinary medicine, diagnosis of scrapie in sheep.

1 Introduction

Diagnosis can be viewed as an example of a classification problem. The features are the diagnostic tests or the clinical signs. The classes are the possible diagnoses. Here we consider two classes, denoted by D^+ (disease present) and D^- (disease absent) and binary features (a sign can only be present or absent, and a test can only be positive or negative). We assume that the only information available to us is in the form of expert estimates of the probabilities $P(T_i^+|D^+)$ and $P(T_i^+|D^-)$, where T_i^+ stands for "test i is positive" and T_i^- stands for "test i is negative". Without loss of generality we can relabel all the signs and tests so that 'present' sign or 'positive' test indicate more strongly D^+ than disease D^-, i.e., $P(T_i^+|D^+) > P(T_i^+|D^-)$.

The common intuition is that the more *independent* signs/tests we have present/positive, the higher is the probability for D^+.

Selection of the best subset of signs or tests is an importrant topic especially in high dimensional problems. While feature selection is a well developed topic within pattern recognition [1,2], selection of classifiers to form an ensemble (diagnostic test selection) is a less developeded topic in the literature. Classifier selection belongs in the field of multiple classifier systems, usually called "overproduce and select" [4]. In our set-up the two selection tasks are equivalent, therefore feature selection techniques can be applied to selecting classifiers. Without loss of generality in the rest of the paper we will talk about test selection only.

The most common selection criterion is the minimum of the classification error. In this paper we derive a different criterion to assess the usefulness of a diagnostic test. The rationale behind this criterion is to maximize the difference between our belief that the individual has the disease (D^+) if all tests are positive and the belief that the individual does not have the disease (D^-) if all the tests are negative. The presumption is that if mixed results are obtained, further tests, probably more expensive or invasive, will be used to clarify the diagnosis. We illustrate the proposed criterion on a set of expert estimates of probabilities $P(T_i^+|D^+)$ and $P(T_i^+|D^-)$ for scrapie in sheep.

2 Criteria for Selection of Diagnostic Tests

2.1 Classification Error

No diagnostic test is perfect so some individuals with a positive test result will not have the disease while others with a negative test result will have the disease. Conventionally the accuracy of a diagnostic test is measured by two values, the sensitivity and the specificity, defined as follows

$$\text{sensitivity} = \frac{\text{number of individuals with the disease and a positive test}}{\text{number of individuals with the disease}}$$

$$\text{specificity} = \frac{\text{number of individuals without the disease and a negative test}}{\text{number of individuals without the disease}}.$$

More formally the notation often seen in the epidemiology literature is shown in the table below where α and β are probabilities

	$T^{\bullet}$	T^-	sum
$D^{\bullet}$	$1-\beta$	β	1
D^-	α	$1-\alpha$	1

In this table $1-\beta$ is the sensitivity of the test and $1-\alpha$ is the specificity of the test. In probabilistic terms

$$\text{sensitivity} = P(T^+|D^+) \qquad \text{and} \qquad \text{specificity} = P(T^-|D^-).$$

The error of a test is

$$P(\text{error}) = P(T^- \ \wedge \ D^+) + P(T^+ \ \wedge \ D^-) \tag{1}$$

For this paper we will assume that both types of error are of equal consequence. Also we will assume that we are equally unsure about whether an individual does or does not have the disease ($P(D^+) = P(D^-) = \frac{1}{2}$). In these circumstances equation (1) leads to

$$\begin{aligned} P(\text{error}) &= P(T^- \ \wedge \ D^+) + P(T^+ \ \wedge \ D^-) \\ &= P(T^-|D^+) \times P(D^+) + P(T^+|D^-) \times P(D^-) \\ &= \frac{1}{2} \times \big(P(T^-|D^+) + P(T^+|D^-)\big) \\ &= 1 - \frac{1}{2} \times ((1 \text{ - } \beta) + (1 \text{ - } \alpha)) \end{aligned}$$

Minimising the error is a sensible criterion. However it is not without its problems. Toussaint (1971) [5] showed that the best test is not necessarily in the best pair of tests. The following hypothetical data shows this. For three tests (labelled A, B and C) we have

test	sensitivity	specificity	P(error)
A	0.90	0.90	0.100
B	0.80	0.95	0.125
C	0.70	0.99	0.155

The best individual test is A, followed by B then C. If we use two tests to make the diagnosis, e.g., A and B, and assume that the tests are independent, then the error can be calculated using the method shown in Table 1.

Table 1. Calculation of the error of the combined test (A and B) for independent A and B

Test results (X)	$P(X\|D^{\bullet})$	$P(X\|D^-)$	minimum
$TA^{\bullet} \ \wedge \ TB^{\bullet}$	$0.9 \times 0.8 = 0.72$	$0.1 \times 0.05 = 0.005$	.005
$TA^{\bullet} \ \wedge \ TB^-$	$0.9 \times 0.2 = 0.18$	$0.1 \times 0.95 = 0.095$	.095
$TA^- \ \wedge \ TB^{\bullet}$	$0.1 \times 0.8 = 0.08$	$0.9 \times 0.05 = 0.045$	.045
$TA^- \ \wedge \ TB^-$	$0.1 \times 0.2 = 0.02$	$0.9 \times 0.95 = 0.855$	.020

When calculating the value for P(error) we have to remember that the probabilities need to be weighted by the probabilities of having or not having the disease. As these are both assumed to be $\frac{1}{2}$ here then we can sum the minimum probabilities and multiply by $\frac{1}{2}$ to get .0825. Similarly we find P(error) is .0695 and .05975 for the pairs A & C and B & C respectively. Thus the best pair is B & C, which does not include the best individual test. P(error) using all three tests is .0495. The errors associated for each possible combination of the three tests are shown in Figure 1.

It is findings like these that make feature selection a difficult problem. We shall refer to this phenomenon by calling the error criterion 'progression non-monotone'. We note that the error criterion is monotone with respect to nested sets, i.e. for any independent tests A and B

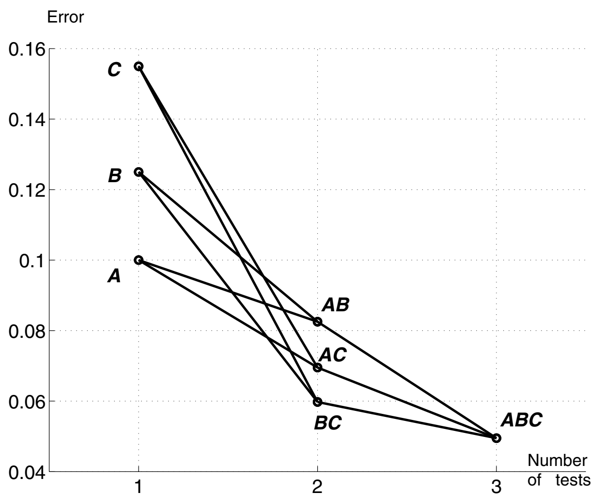


Fig. 1. The error for each combination of tests

$$P(\text{error for A} \wedge \text{B}) \leq P(\text{error for A}) \quad \& \quad P(\text{error for A} \wedge \text{B}) \leq P(\text{error for B})$$

The fact that the error criterion is progression non monotone, even for independent features, means that there is no simple straightforward procedure for selecting the optimal feature subset. Can we find another criterion which is both intuitive for diagnostic purposes, and is progression monotone?

2.2 Logodds Criterion

From a Bayesian viewpoint the probability that an individual has the disease after we have observed a positive test result ($P(D^+|T^+)$) is a measure of our belief that the individual actually has the disease. Probabilities (P) are often more conveniently expressed as odds $\frac{P}{1-P}$, which can then be combined after taking their logarithms. Here the odds of having the disease is

$$\text{ODDS}(D^+ : D^-|T^+) = \frac{P(D^+|T^+)}{1 - P(D^+|T^+)} = \frac{P(D^+|T^+)}{P(D^-|T^+)}$$

Intuitively we would like this value to be high as it means that if we get a positive test result then we increase our belief that the individual has the disease. Similarly we would like our belief in the individual having the disease to be decreased when we get a negative test result. The odds of having the disease given a negative test result is $\text{ODDS}(D^+ : D^-|T^-) = \frac{P(D^+|T^-)}{1-P(D^+|T^-)} = \frac{P(D^+|T^-)}{P(D^-|T^-)}$. We would like this to be as small as possible.

We define the *diagnostic value* of a particular test T to be

$$v(T) = \log(\text{ODDS}(D^+ : D^-|T^+)) - \log(\text{ODDS}(D^+ : D^-|T^-)).$$

This criterion maximises the difference in our belief of being diseased between a positive test result being obtained and a negative test result being obtained. From the definition of Bayesian probability we have

$$P(D^+|T^+) = \frac{P(T^+|D^+) \times P(D^+)}{P(T^+|D^+) \times P(D^+) + P(T^+|D^-) \times P(D^-)} \tag{2}$$

$$= \frac{(1-\beta) \times \delta}{(1-\beta) \times \delta + \alpha \times (1-\delta)} \tag{3}$$

where $\delta = P(D^+)$ and α and β are as defined above.

Similarly

$$P(D^-|T^+) = \frac{\alpha \times (1-\delta)}{\alpha \times (1-\delta) + (1-\beta) \times \delta}$$

This leads to

$$\text{ODDS}(D^+ : D^-|T^+) = \frac{\delta}{1-\delta} \times \frac{1-\beta}{\alpha}$$

Taking the logarithm of this leads to our measure of belief in being diseased when a positive test result is obtained.

$$\log(\text{ODDS}(D^+ : D^-|T^+)) = \log\frac{\delta}{1-\delta} + \log\frac{1-\beta}{\alpha}$$

Similarly our measure of belief in being diseased when a negative test result is obtained is

$$\log(\text{ODDS}(D^+ : D^-|T^-)) = \log\frac{\delta}{1-\delta} + \log\frac{\beta}{1-\alpha}$$

Taking the difference of these two logodds terms leads to our logodds criterion

$$v(T) = \log\frac{1-\beta}{\alpha} - \log\frac{\beta}{1-\alpha} \tag{4}$$

$$= \log\frac{\text{sensitivity}}{1-\text{sensitivity}} + \log\frac{\text{specificity}}{1-\text{specificity}} \tag{5}$$

Notice that δ is not involved in the criterion. This means that the prior probability we have for an individual being diseased plays no part in deciding which test is the best. This has important consequences when we come to look at the situation when we combine the results from more than one test.

Suppose that T is a combined test consisting of $T_1, T_2, \ldots, T_n$. A positive combined test will be equivalent to all individual tests giving positive results, i.e., $T^+ = T_1^+ \cap T_2^+ \cap \ldots \cap T_n^+$. A negative combined test will be equivalent to all individual tests giving positive results, i.e., $T^- = T_1^- \cap T_2^- \cap \ldots \cap T_n^-$. Using the assumption of independence, we can show that

$$v(T) = \sum_{i=1}^{n} v(T_i).$$

for any values of the prior probability δ. Let $\mathcal{S}$ denote the set of all available diagnostic tests. Our proposed criterion is

$$\max_{T\subseteq\mathcal{S}} v(T) = \max_{T\subseteq\mathcal{S}} \sum_{T_i\in T} v(T_i).$$

For the example data shown above the logodds criterion values can be calculated as shown in Table 2. Using this criterion the best test is C, followed by A then B. The real advantage of this criterion can now be seen. After we have got the results of the best test (C here) then our prior probability of the disease has changed. But this plays no part in deciding which test to use in conjunction with it. So the second best test to use in combination with test C is test A. The result of combining all three tests and showing all possible results is given in Figure 2.

Table 2. Calculation of the logodds criterion

test	sensitivity	specificity	$\log\frac{\bullet-\beta}{\alpha}$	$\log\frac{\beta}{\bullet-\alpha}$	logodds
	$1-\beta$	$1-\alpha$			criterion
A	0.90	0.90	2.197	-2.197	4.394
B	0.80	0.95	2.773	-1.558	4.331
C	0.70	0.99	4.248	-1.194	5.442

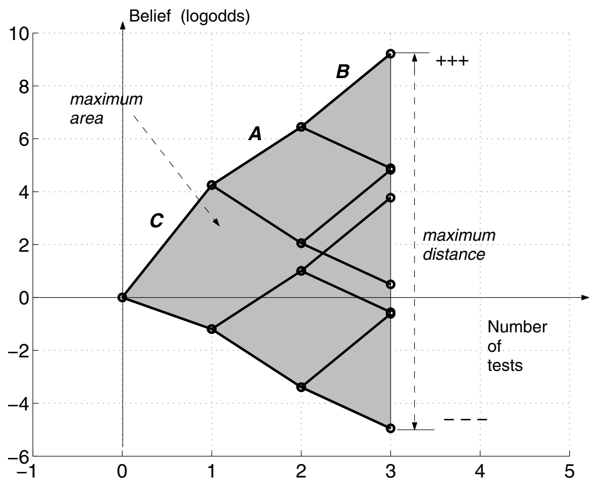


Fig. 2. The logodd criterion for all possible results of 3 tests

Visually the logodds criterion finds the test with the largest difference between the logodds for a positive and a negative test result.

2.3 Differences between the Logodds and the Error Criteria

The following example shows that the logodds criterion is different from the error criterion. Consider again the three tests A, B and C. Let $\alpha_1 = 0.1$ and

$\beta_1 = 0.1$ be the respective errors for test A. Figure 3 depicts the regions for α_2 and β_2 for another test. If the point specified by the pair (α_2, β_2) is inside one of the shaded regions (e.g. Test C), then the second test is worse than A on the error criterion but better than A on the logodds criterion. However if the point is not inside the shaded region (e.g. Test B) then the test is worse than A on both criteria.

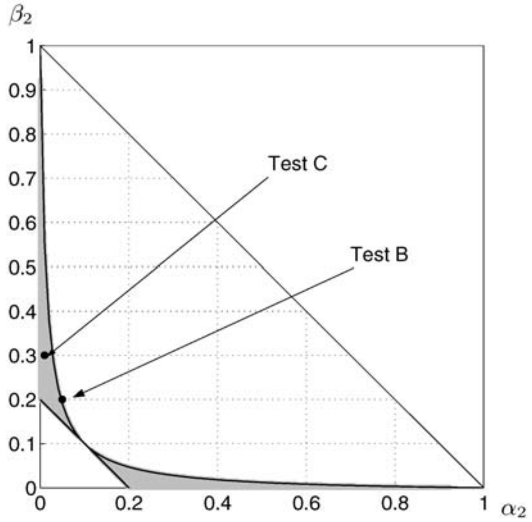


Fig. 3. Regions for $\alpha_{\bullet}$ and $\beta_{\bullet}$ where the (individual) error criterion and logodd criterion disagree

An important difference between the two criteria is the computational effort required for estimating the error for a set of tests $T = \{T_1, \ldots, T_n\} \subseteq \mathcal{S}$. For the error criterion, the error is the sum of 2^n terms, one for each possible combination of test results. These are calculated from the expert estimates of the probabilities. Since the error criterion is progression non monotone, special procedures have to be applied to navigate through the subsets T. On the other hand, the logodds criterion is progression monotone. It is a simple sum whereby we can maximise each component so as to get the a total maximum. Thus the set of the individually best n features is the best set of n features according to the logodds criterion.

3 Results for Scrapie Data

Scrapie is a notifiable disease of sheep. We consider 285 clinical signs that may be observed in either scrapie or in 62 differential diagnoses [6]. The available data is in the form of probabilities $P(T_i^+|D^+)$ and $P(T_i^+|D^-)$, where $i = 1, \ldots, 285$, D^+ is scrapie, and D^- is any of the other diagnoses. It is not practical to observe all 285 clinical signs on sheep suspected for scrapie. Kuncheva et al. (2003) [3] report an analysis which uses sequential feature selection (SFS) to choose 15 signs based on the classification error criterion. Table 3 shows the 15 signs selected by the error criterion through the SFS procedure and the 15 signs selected by the logodds criterion. The numbers in front of the signs show the *individual rank* according to the error criterion. The sign with rank 1, Hyperaesthesia, is the most important discriminatory sign according to the error criterion.

Table 3. The 15 diagnostic signs (tests) selected through the error criterion (via SFS) and the logodds criterion. Given in brackets are $(P(T^{\bullet} \mid D^{\bullet})/P(T^{\bullet} \mid D^{-}))$. 'R' is the *individual* rank of the sign by the error criterion

R	Sign	R	Sign
1	Hyperaesthesia (.87/.06)	2	Weight loss (1/.25)
2	Weight loss (1/.25)	1	Hyperaesthesia (.87/.06)
3	Pruritus (.8/.07)	3	Pruritus (.8/.07)
21	*Increased respiratory rate (0/.19)	4	Abnormal behaviour (.7/.06)
4	Abnormal behaviour (.7/.06)	5	Underweight (.9/.25)
5	Underweight (.9/.25)	21	*Increased respiratory rate (0/.19)
9	Tremor (.63/.09)	22	*Sudden death (0/.18)
22	*Sudden death (0/.18)	18	Abortion or weak newborns (.3/.02)
6	Dysmetria (.67.1)	6	Dysmetria (.67.1)
7	Ataxia (.77/.2)	9	Tremor(.6/.09)
8	Grinding teeth (.67/.11)	10	Trembling (.6/.09)
10	Trembling (.6/.09)	8	Grinding teeth (.7/.11)
11	Alopecia (.57/.08)	23	*Tachycardia (0/0.13)
12	Seizures or syncope (.52/.1)	25	*Reluctant to move (0/.13)
13	Rumen hypomotility (.47/.1)	11	Alopecia (.6/.08)

Notes:
1. The signs marked with a * had to be relabeled so as to ensure that $P(T_i^{\bullet} \mid D^{\bullet}) > P(T_i^{\bullet} \mid D^{-})$.
2. For calculation purposes, all 0's were reassigned to 0.01 meaning "very rare" and all 1's were reassigned to 0.99 meaning "almost sure".

The table shows that the logodds criterion has selected 12 of the 15 features that were chosen by the error criterion. Two of the three different features chosen by the logodds criterion have sensitivities of 0. If the expert estimates are incorrect in the extremes (0 or 1), then the logodds criterion might give undue weight to such features.

4 Conclusions

The difference between the error and logodds criteria can be viewed in the following way. A pharmaceutical company creates a diagnostic test and measures its worth by means of the number of misdiagnoses that ensue. This is a perfectly sensible for the company. However an individual is only interested in the results of the test on themselves. The best test for the individual is the one that leads to the most information about whether the individual actually has the disease. The logodds criterion attempts to measure this. In this sense the criterion is intuitive for diagnostic purposes.

The logodds criterion is progression monotone. This means that we can determine at the outset the ordering of the worth of the tests using this criterion. The advantage is that as the prior belief plays no part in the criterion then this ordering is the one that applies every time we want to include the results of another test. The computational efficiency savings follow from the progression monotonicity.

References

1. D. W. Aha and R. L. Bankert. A comparative evaluation of sequential feature selection algorithms. In *Proc. 5th International Workshop on AI and Statistics*, pages 1–7, Ft Lauderdale, FL, 1995.
2. A. K. Jain and D. Zongker. Feature selection: evaluation, application and small sample performance. *IEEE Transactions on PAMI*, 19(2):153–158, 1997.
3. L. I. Kuncheva, P. D. Cockcroft, C. J. Whitaker, and Z. S. Hoare. Pre-selection of independent binary features in differential diagnosis. submitted.
4. F. Roli and J. Kittler, editors. *Proc. 2nd International Workshop on Multiple Classifier Systems (MCS 2001), Lecture Notes in Computer Science LNCS* **2096**. Springer-Verlag, Cambridge, UK, 2001.
5. G. T. Toussaint. Note on optimal selection of independent binary-valued features for pattern recognition. *IEEE Transactions on Information Theory*, 17:618, 1971.
6. M. E. White. Diagnosis, information management, teaching and record coding using the consultant database. *Canadian Veterinary Journal*, 29:271–273, 1988.

Evaluation of Classical and Novel Ensemble Methods for Handwritten Word Recognition

Simon Günter and Horst Bunke

Department of Computer Science, University of Bern
Neubrückstrasse 10, CH-3012 Bern, Switzerland
sguenter,bunke@iam.unibe.ch

Abstract. Handwritten text recognition is one of the most difficult problems in the field of pattern recognition. Recently, a number of classifier creation and combination methods, known as ensemble methods, have been proposed in the field of machine learning. They have shown improved recognition performance over single classifiers. In this paper a number of ensemble methods for handwritten word recognition are described, experimentally evaluated, and compared to each other. Those methods include classical, general purpose ensemble methods as well as novel ensemble methods specifically developed by the authors for handwritten word recognition. The aim of the paper is to investigate the potential of ensemble methods for improving the performance of handwriting recognition systems. The base recognition systems used in this paper are hidden Markov model classifiers.

Keywords: ensemble methods, handwritten word recognition, hidden Markov model (HMM).

1 Introduction

The field of off-line handwriting recognition has been a topic of intensive research for many years. First only the recognition of isolated handwritten characters was investigated [26], but later whole words [25] were addressed. Most of the systems reported in the literature until today consider constrained recognition problems based on vocabularies from specific domains, e.g. the recognition of handwritten check amounts [13] or postal addresses [15]. Free handwriting recognition, without domain specific constraints and large vocabularies, was addressed only recently in a few papers [16, 22]. The recognition rate of such systems is still low, and there is a need to improve it.

The combination of multiple classifiers was shown to be suitable for improving the recognition performance in difficult classification problems [18, 27]. Also in handwriting recognition, classifier combination has been applied. Examples are given in [20, 28]. Recently new ensemble creation methods have been proposed in the field of machine learning, which generate an ensemble of classifiers from a single classifier [3]. Given a single classifier, the base classifier, a set of classifiers can be generated by changing the training set [2], the input features [12], the input data by injecting randomness [4], or the parameters and the architecture of the classifier [23]. Examples of widely used methods that change the training set are Bagging [2] and AdaBoost [5]. Random subspace method [12] is a well-known approach based on changing the input features. A summary of ensemble methods is provided in [3].

Although the popularity of multiple classifier systems in handwritten recognition has grown significantly, not much work on the use of ensemble methods for handwritten word recognition has been reported in the literature. This issue was recently addressed by the authors in a few papers [6, 7, 9–11]. In [6] ensemble methods using several prototypes were introduced. In [7] new boosting algorithms based on AdaBoost [5] were proposed. Ensemble methods using feature selection algorithms were introduced in [9] while a special combination scheme was described in [10]. Results of classical ensemble methods were reported in [11]. In the present paper we introduce a novel ensemble method and compare the most promising ensemble methods among each other, using a uniform framework for experimental evaluation. In contrast with earlier work reported by the authors, a more sophisticated base classifier is used. Also a new ensemble method is proposed (Subsection 3.2).

The rest of this paper is organized as follows. The base classifiers for handwritten word recognition used in the experiments are presented in Section 2. In Section 3, the ensemble methods evaluated in this paper are described. Then, in Section 4, experimental results of the ensemble methods with optimal parameter values are given. Finally, some conclusions are drawn in Section 5.

2 Handwritten Word Recognizers

The handwritten word recognizers used in this paper are similar to the one described in [22]. We assume that each handwritten word input to the recognizers has been normalized with respect to slant, skew, baseline location and height (for details of the normalization procedures see [22]). A sliding window of one pixel width is moved from left to right over the word and nine geometric features are extracted at each position of the window. Thus an input word is converted into a sequence of feature vectors in a 9-dimensional feature space. The geometric features used in the system include the fraction of black pixels in the window, the center of gravity, and the second order moment. These features characterize the window from the global point of view. The other features give additional information. They represent the position of the upper- and lowermost pixel, the contour direction at the position of the upper- and lowermost pixel, the number of black-to-white transitions in the window, and the fraction of black pixels between the upper- and lowermost black pixel.

For each uppercase and lowercase character, an HMM is build. For all HMMs the linear topology is used, i.e. there are only two transitions per state, one to itself and one to the next state. The character models are concatenated to word models. There is exactly one model for each word from the underlying dictionary. This approach makes it possible to share training data across different words.

Two different versions of the classifier described in the previous two paragraphs, called C_1 and C_2 in the following, are employed in this paper. For classifier C_1 a fixed number of 14 states per HMM were used and the distributions of the features in each state of an HMM were modeled by single Gaussians. Four iterations of the Baum-Welch algorithm [24] were used for the training of the classifier. The number of states per HMM and the number of training iterations are optimal values (as determined by means of a validation set) under the assumption that the same number of states is needed for each individual character.

Classifier C_2 is a significantly refined version of classifier C_1. The number of states of the HMMs were optimized by the Quantile method [30] for each character individually. As a result, each individual character model has a different number of states. In addition the distributions of the features in each state of an HMM were modeled by Gaussian mixtures instead of single Gaussians. The training method of classifier C_2 was optimized on a validation set, using an optimization strategy described in [8].

3 Ensemble Methods

In this section the ensemble methods used in the experiments are described. The ensemble methods are divided in four categories: classical ensemble methods, partitions based ensemble method, novel boosting algorithms, and feature search ensemble method.

3.1 Classical Ensemble Methods

Two classical ensemble methods, AdaBoost and random subspace method were investigated in the experiments.

AdaBoost [5] modifies the original training set for the creation of the ensemble. Each classifier is trained on a different training set of the same size as the original training set. The main idea of AdaBoost is to concentrate the training on “difficult” patterns. The classical AdaBoost algorithm can only be used for two-class problems, but AdaBoost.M1 [5], a simple extension of AdaBoost, can cope with multi-class problems. Consequently, AdaBoost.M1 was applied in the experiments.

In the **random subspace method** [12] the individual classifiers use only a subset of all features for training and testing. The size of the subsets is fixed and the features are randomly chosen from the set of all features. In the experiments of this paper, 6 out of 9 features were selected, i.e. the fixed feature set size is 6.

The classical ensemble Bagging [2] was also tested. As it produced inferior results, no description and no results of Bagging are given in this paper.

3.2 Partitions Based Ensemble Method

The basic algorithm of the partitions based ensemble methods can be described as follows. First, the whole training set is split into several partitions. Then each classifier is trained on all patterns of one of these partitions. This means that if the training set is split in n partitions, n different classifiers are obtained. There are two key parameters of this ensemble method. The first parameter is the algorithm actually applied to perform the partitioning, and the second is the number of partitions

For the experiments the partitioning was based on a clustering of the words according to their writing style. The whole training set was first clustered into groups of words with similar writing style where each group forms a partition. This was done by extracting some features of the handwritten text and applying the k-means clustering algorithm [14]. As we are using the IAM database [21], we always have complete pages of handwritten text, produced by the same writer, at our disposal (compare Section 4). Therefore the features were extracted from a whole page and all words of a page belong to the same cluster. The two features used for the clustering are the following:

- Words per component: The average number of words per connected component is calculated where small components are removed by a filtering procedure. A value of one of this feature corresponds to a complete cursive handwriting, i.e. the case where one connected component always represents exactly one word. By contrast, words consisting of isolated hand-printed characters have significantly lower values.
- Character width: The average width of the characters, in terms of pixels, is calculated. To this aim the lines of the pages are segmented in words using the procedure presented in [29]. The character width is then calculated as the sum of the lengths of the words of a page divided by the number of characters.

Both features were linearly normalized so that the mean of the features was 0 and the standard deviation was 1.

The influence of the second parameter, the number n of partitions, is as follows. The higher the number of n is, the more similar are the writing stiles of the words of the same cluster. On the other hand, the average training set size decreases linearly with the number of clusters. As the performance of a system normally increases with the size of the training set, a rather high number of clusters will lead to classifiers with low performance. Usually, n is a parameter the optimal value of which needs to be experimentally determined.

3.3 Novel Boosting Algorithms

As mentioned before, the original AdaBoost algorithm only works for two-class problems. AdaBoost.M1 [5] is a straightforward extension that basically regards the multi-class problem as a two-class problem with the two classes “correct” and “not correct”. However, by doing this, we loose a great deal of information. In [7] three ensemble methods were introduced which are based on AdaBoost.M1, but which take all classes into account. The simple probabilistic boosting (SPB) defines the selection probability of a training pattern as a linear function of the likelihood of the misclassification of this pattern by the ensemble consisting of all already created classifiers. The effort based boosting (EBB) only trains with training elements for which the inclusion of instances of these elements leads, with a certain likelihood, to their correct recognition. Thus training elements which are always misclassified, even when including many of their instances in the training set, are not used for the training. The simple probabilistic boosting with effort (SPBE) is a combination of SPB and EBB. For more details of the three novel boosting algorithms we refer to [7].

3.4 Feature Search Ensemble Method

The feature search ensemble method described in this subsection was first introduced in [9]. The key idea of the method is not to select feature subsets for the individual classifiers of the ensemble randomly, as it is done in the random subspace methods [12], but to repeatedly apply an algorithm that selects a well performing features subset. In principle any known algorithm for feature selection can be used. In the ensemble method presented in this paper, a feature search starting with the empty set of features and a feature search starting with the full set of features were applied alternatively.

Similarly to most feature selection algorithms, the method applied in this paper tries to maximize the value of an objective function f. For a single classifier system this objective function is simply the classifier's performance on a validation set. However, in the present case we want to maximize the performance of the whole ensemble. Therefore the objective function f measures the performance of the ensemble consisting of all classifiers which were already created and the classifier which is actually considered. The measurement of f is performed on a separate validation set. Ideally f also incorporates an estimation of the potential for improvement when adding more classifiers. In [9] two different objective functions were defined.

4 Experiments

For isolated character and digit recognition, a number of commonly used databases exist. However, for the task considered in this paper, there exists only one suitable database to the knowledge of the authors, holding a sufficiently large number of words produced by different writers [21]. Consequently this database was used in the experiments.

Two sets of experiments were done. The first set of experiments were conducted in order to evaluate the ensemble methods for a rather simple, straightforward base classifier. In these experiments the base classifiers C_1 was used. In the second set of experiments the highly optimized classifier C_2 was used as the base classifier. For the experiments of the second set the optimal number of classifiers was determined in a separate experiment.

To combine the individual classifiers of the ensembles the following combination schemes were applied:

1. Voting scheme (*voting*): Only the top choice of each classifier is considered. The word class that is most often on the first rank is the output of the combined classifier. Ties are broken by means of the maximum rule (*max*) or the median rule (*med*), which are only applied to the competing word classes.
2. Performance weighted voting (*perf weight*): Here we consider again the top class of each classifier. In contrast with regular voting, a weight is assigned to each classifier. The output of the combined classifier is the word class that received the largest sum of weights. The weight of a classifier is the performance of this classifier on the training set.
3. Weighted voting using optimized weights (*ga weight*): This scheme is similar to *perf weight*, but optimal weights are used which are calculated by a genetic algorithm based on the results of the classifiers achieved on the training set [19].
4. Special combination scheme for HMM-based recognizers (*special*): This combination was introduced in [10]. It integrates all HMMs of the different classifiers that correspond to the same character into a single HMM.

In the first set of experiments a data set of 10,927 words with a vocabulary of size 2,296 was used. That is, a classification problem with 2,296 different classes was considered. The total number of writers who contributed to this set is 81. A training set containing 9,861 words and a test set containing 1,066 words were chosen in such a way that none of the writers of the test set were represented in the training set. Thus the experiments are writer independent. For the feature search ensemble method the training set was randomly split in a training set of 8,795 words and a validation set of 1,066 words. For this set of experiments the base classifier C_1, as described in Section 2, was used. The recognition rate of the base classifier C_1 was 66.23 %. The results of the ensemble methods are shown in Table 1. The name of the ensemble method is indicated in the column *ensemble method*. Please note that parameters of some ensemble methods were varied and only the results of the ensemble methods using the best parameter values, i.e. those which lead to the highest recognition rates on the test set, are shown. These parameter values are also given in the column *ensemble method*. (See [7, 9] for details of these parameters). In column *cn* the number of classifiers in the ensemble is given. The column *combination scheme* gives the scheme for which the highest recognition rate is obtained. The obtained recognition rate is finally shown in the last column.

Table 1 shows that the recognition rate of the base classifier was increased by 2.88 % up to 5.81 %. The table furthermore shows that the ensemble methods of all categories are able to significantly improve the performance of the base classifier. We note that the classical ensemble methods were outperformed by all other ensemble methods specially developed by the authors for the domain of handwriting recognition.

In the second set of experiments a larger training set of 18,920 words and a larger test set of 3,264 words were used. The vocabulary of the experiments contains 3,997 words, i.e. a classification problem with 3,997 different classes is

Table 1. Best results of the ensemble methods for base classifier $C_{\bullet}$. The performance of $C_{\bullet}$ is 66.23 %.

ensemble method	cn	combination scheme	rec. rate.
classical:AdaBoost	14	*perf weight*	69.11 %
classical: random subspace method	25	*ga weight*	69.35 %
partitions based	5	*special*	70.83 %
boosting: EBB, $\alpha = 1.5$	10	*ga weight*	69.82 %
feature search, $f_{\bullet}$	10	*vote best med*	72.04 %

Table 2. Best results of the ensemble methods for base classifier $C_{\bullet}$. The performance of $C_{\bullet}$ is 80.36 %.

ensemble method	cn	combination scheme	rec. rate.
classical: AdaBoost	14	*vote max*	82.02 %
classical: random subspace method	25	*vote med*	80.76 %
partitions based	3	*special*	81.07 %
boosting: SPBE, $e_{\bullet}$	14	*vote max*	82.69 %
feature search, $f_{\bullet}$	13	*vote med*	82.69 %

considered. The set of writers of the training set and the set of writers of the test set are disjoint, so the experiments are again writer independent. The total number of writers who contributed to this data set is 153. For the feature search ensemble method the training set was randomly split three times in a training set of 17,920 words and a validation set of 1,000 words. The recognition rate of a system was calculated by averaging over the system's recognition rates obtained for each of the three splits. The base classifier C_2, as described in Section 2, is used. The recognition rate of the base classifier was 80.36 %.

The results of the second set of experiments are shown in Table 2 where the same notation as in Table 1 is used.

The recognition rate of the base classifier was increased by 0.4 % up to 2.33 %. This again shows that the ensemble methods of all categories are able to improve the performance of the base classifier. The random subspace method obtained only moderate improvements. The partitions based ensemble method did not perform as well as the classical method AdaBoost in this case. But both the boosting and the feature search ensemble methods were better than AdaBoost.

5 Conclusions

In this paper classical ensemble methods and novel ensemble methods specially developed for the domain of handwriting recognition were compared to each other. Two series of experiments were conducted. In the first series a rather simple and straightforward base classifier was used, while the base classifier of the second series was highly optimized. The results show that in both cases the performance of the base classifier can be significantly improved through the use of ensemble methods. The ensemble methods developed specifically for the task of handwriting recognition obtained either better results than the classical ones or produced much smaller ensembles. A possible future work is the evaluation of ensemble methods which use several base classifiers with different architectures. By applying an ensemble method on each base classifier and fusing the ensembles the performance may be increased even more. This was already confirmed in some earlier works for the case of two base classifiers. It may be promising to extend this approach to the case of more than two base classifiers.

Acknowledgment

This research was supported by the Swiss National Science Foundation (Nr. 20-52087.97). The authors thank Dr. Urs-Victor Marti for providing the handwritten word recognizer and Matthias Zimmermann for the segmentation of a part of the IAM database. Additional funding was provided by the Swiss National Science Foundation NCCR program "Interactive Multimodal Information Management (IM)2" in the Individual Project "Scene Analysis".

References

1. *Proc of the 7th Int. Conf. on Document Analysis and Recognitio*, Edinbourgh, Scotland, 2003.
2. Leo Breiman. Bagging predictors. *Machine Learning*, (2):123–140, 1996.
3. T. G. Dietterich. Ensemble methods in machine learning. In *[17]*, pages 1–15.
4. T.G. Dietterich and E.B. Kong. Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. Technical report, Departement of Computer Science, Oregon State University, 1995.
5. Yoav Freund and Robert E. Schapire. A decision-theoretic generalisation of on-line learning and an application to boosting. *Journal of Computer and Systems Sciences*, 55(1):119–139, 1997.
6. S. Günter and H. Bunke. Generating classifier ensembles from multiple prototypes and its application to handwriting recognition. In *[18]*, pages 179–188.
7. S. Günter and H. Bunke. New boosting algorithms for classification problems with large number of classes applied to a handwritten word recognition task. In *[27]*, pages 326 – 335.
8. S. Günter and H. Bunke. Optimizing the number of states, training iterations and Gaussians in an HMM-based handwritten word recognizer. In *[1]*, volume 1, pages 472–476.
9. S. Günter and H. Bunke. Creation of classifier ensembles for handwritten word recognition using feature selection algorithms. In *Proc. of the 8th International Conference on Frontiers in Handwriting Recognition*, pages 183–188, 2002.
10. S. Günter and H. Bunke. A new combination scheme for HMM-based classifiers and its application to handwriting recognition. In *16th International Conference on Pattern Recognition*, volume 2, pages 332–337, Quebec, Canada, 2002.
11. S. Günter and H. Bunke. Ensembles of classifiers for handwritten word recognition. *International Journal of Document Analysis and Recognition*, 5(4):224–232, 2003.

12. T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
13. S. Impedovo, P. Wang, and H. Bunke, editors. *Automatic Bankcheck Processing.* World Scientific Publ. Co, Singapore, 1997.
14. A. Jain, M. Murty, and P. Flynn. Data clustering: A survey. *ACM Computing Survey*, 31:264–323, 1999.
15. A. Kaltenmeier, T. Caesar, J.M. Gloger, and E. Mandler. Sophisticated topology of hidden Markov models for cursive script recognition. In *Proc. of the 2nd Int. Conf. on Document Analysis and Recognition, Tsukuba Science City, Japan*, pages 139–142, 1993.
16. G. Kim, V. Govindaraju, and S.N. Srihari. Architecture for handwritten text recognition systems. In S.-W. Lee, editor, *Advances in Handwriting Recognition*, pages 163–172. World Scientific Publ. Co., 1999.
17. J. Kittler and F. Roli, editors. *First International Workshop on Multiple Classifier Systems*, Cagliari, Italy, 2000. Springer.
18. J. Kittler and F. Roli, editors. *Third International Workshop on Multiple Classifier Systems*, Cagliari, Italy, 2002. Springer.
19. L. Lam, Y.-S. Huang, and C. Suen. Combination of multiple classifier decisions for optical character recognition. In H. Bunke and P. Wang, editors, *Handbook of Character Recognition and Document Image Analysis*, pages 79–101. World Scientific, 1997.
20. D. Lee and S. Srihari. Handprinted digit recognition: A comparison of algorithms. In *Third International Workshop on Frontiers in Handwriting Recognition*, pages 153–162, 1993.
21. U. Marti and H. Bunke. The IAM-database: An English sentence database for off-line handwriting recognition. *Int. Journal of Document Analysis and Recognition*, 5:39–46, 2002.
22. U.-V. Marti and H. Bunke. Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system. *Int. Journal of Pattern Recognition and Art. Intelligence*, 15:65–90, 2001.
23. D. Partridge and W. B. Yates. Engineering multiversion neural-net systems. *Neural Computation*, 8(4):869–893, 1996.
24. L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.
25. J.-C. Simon. Off-line cursive word recognition. *Special Issue of Proc. of the IEEE*, 80(7):1150–1161, July 1992.
26. C.Y. Suen, C. Nadal, R. Legault, T.A. Mai, and L. Lam. Computer recognition of unconstrained handwritten numerals. *Special Issue of Proc. of the IEEE*, 80(7):1162–1180, 1992.
27. T. Windeatt and F. Roli, editors. *4th Int. Workshop on Multiple Classifier Systems*, Guildford, United Kingdom, 2003. Springer.
28. L. Xu, A. Krzyzak, and C. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 22(3):418–435, 1992.
29. M. Zimmermann and H. Bunke. Automatic segmentation of the IAM off-line database for handwritten English text. In *Proc. of the 16th Int. Conference on Pattern Recognition*, volume 4, pages 35–39, Quebec, Canada, 2002.
30. M. Zimmermann and H. Bunke. Hidden markov model length optimization for handwriting recognition systems. In *Proc. of the 8th International Workshop on Frontiers in Handwriting Recognition*, pages 369–374, 2002.

Improving Support Vector Classification via the Combination of Multiple Sources of Information

Javier M. Moguerza[1], Alberto Muñoz[2], and Isaac Martín de Diego[2]

[1] University Rey Juan Carlos, c/ Tulipán s/n, 28933 Móstoles, Spain
j.moguerza@escet.urjc.es

[2] University Carlos III de Madrid, c/ Madrid 126, 28903 Getafe, Spain
alberto.munoz@uc3m.es, ismdiego@est-econ.uc3m.es

Abstract. In this paper we describe several new methods to build a kernel matrix from a collection of kernels. This kernel will be used for classification purposes using Support Vector Machines (SVMs). The key idea is to extend the concept of linear combination of kernels to the concept of functional (matrix) combination of kernels. The functions involved in the combination take advantage of class conditional probabilities and nearest neighbour techniques. The proposed methods have been successfully evaluated on a variety of real data sets against a battery of powerful classifiers and other kernel combination techniques.

1 Introduction

Support Vector Machines (SVMs) have proven to be a successful tool for the solution of a wide range of classification problems since their introduction in [3]. The method uses as a primary source of information a kernel function $K(i, j)$, where K is Mercer's kernel and i, j represent data points in the sample: By the Representer Theorem (see for instance [16]), SVM classifiers always take the form $f(x) = \sum_i \alpha_i K(x, i)$. The approximation and generalization capacity of the SVM is determined by the choice of the kernel K [4]. A common way to obtain SVM kernels is to consider a linear differential operator D, and choose K as Green's function for the operator D^*D, where D^* is the adjoint operator of D [15]. It is easy to show that $\|f\|^2 = \|Df\|^2_{L_2}$. Thus we are imposing smoothing conditions on the solution f. However, it is hard to know in advance which particular smoothing conditions to impose for a given data set. Fortunately, kernels are straightforwardly related to similarity (or equivalently distance) measures, and this information is actually available in many data analysis problems.

Nevertheless, using a single kernel may be not enough to solve accurately the problem under consideration. This happens, for instance, when dealing with text mining problems, where analysis results may vary depending on the document similarity measure chosen [9]. Thus, the information provided by a single similarity measure (kernel) may be not enough for classification purposes, and the combination of kernels appears as an interesting alternative to the choice of the 'best' kernel.

The specific literature on the combination of kernels is rather in its beginnings. A natural approach is to consider linear combinations of kernels. This is the approach followed in [10], and it is based on the solution of a semi-definite programming (SDP) problem to calculate the coefficients of the linear combination. The solution of this kind of optimization problems is computationally very expensive [19]. Another problem regarding this method is the overfitting due to lack of capacity control. A different approach is proposed in [2]. The method, called MARK, builds a classifier (not the specific kernel matrix) by a boosting type algorithm.

In this paper we describe several methods to build a kernel matrix from a collection of kernels for classification purposes. The key idea is to extend the concept of linear combination of kernels to the concept of functional (matrix) combination of kernels.

The paper is organized as follows. Section 2 describes the proposed methods for combining kernels. The experimental setup and results on real data sets are described in section 3. Section 4 concludes.

2 Methods

Let $K_1, K_2, ...K_M$ be a set of M input kernels defined on a data set X, and denote by K^* the desired output combination. Let y denote the label vector, where for simplicity $y_i \in \{-1, +1\}$ (the extension to the multilabel case is straighforward).

To motivate the discussion, consider the following (functional) weighted sum of the kernels:

$$K^* = \sum_{m=1}^{M} W_m \cdot K_m \,, \tag{1}$$

where '$\cdot$' denotes the element by element product between matrices, and $W_m = [w_m(i,j)]$ is a matrix whose elements are nonlinear functions $w_m(i,j)$, with i and j data points in the sample. Notice that if $w_m(i,j) = \mu_m$, where $\mu_1, \ldots, \mu_M$ are constants, then the method reduces to a simple linear combination of matrices:

$$K^* = \sum_{m=1}^{M} \mu_m K_m \,. \tag{2}$$

As mentioned in Section 1, in [10] a method is suggested to learn the coefficients μ_m of the linear combination by solving a semi-definite programming problem. We will refer to this method as **SDP**.

Taking $\mu_m = \frac{1}{M}$, the average of the kernels is obtained. This method will be refered in the following as **AKM** (Average Kernel Method).

Regarding our proposals, consider the (i,j) element of the matrix K^* in (1):

$$K^*(i,j) = \sum_{m=1}^{M} w_m(i,j) K_m(i,j) \,. \tag{3}$$

Next we will show how to calculate the weighting functions $w_m(i,j)$. To this aim, we will make use of conditional class probabilities. Consider the pair (i, y_i) and an unlabeled observation j. Given the observed value j, define $P(y_i|j)$ as the probability of j being in class y_i. If i and j belong to the same class this probability should be high. Unfortunately this probability is unknown and it has to be estimated. In our proposals we will estimate it by $P(y_i|j) = \frac{n_{ij}}{n}$, where n_{ij} is the number of the n-nearest neighbours of j belonging to class y_i. Notice that each kernel induces a different type of neighborhood. Hence, it is advisable to estimate this probability for each kernel representation, that is, for the kernel K_m we will estimate the conditional probabilities $P_m(y_i|j)$.

2.1 The Probability Weighting Scheme ('ProbWS')

The first proposed method builds K^* by defining $w_m(i,j)$ in (3) as:

$$w_m(i,j) = \tau \left(P_m(y_i|j) + P_m(y_j|i) \right)^q , \tag{4}$$

where τ is introduced to assure that $\sum_m w_m(i,j) = 1$, and q is a positive constant to control the value of the weights. Within this setting, the weights quantify the relative importance of each kernel: If i and j belong to the same class (say y_i), the proportion of the nearest neighbours of j belonging to y_i should be high. So, the method favours the kernel whose induced neighbourhood shows the highest agreement with the data label information.

Given that K^* is not necessarily a linear combination of kernels, positive definiteness of K^* is not guaranteed. Several solutions have been proposed to face this problem [14]: A first possibility is to replace K^* by $K^* + \lambda I$, for $\lambda > 0$ large enough to make all the eigenvalues of the kernel matrix positive. Another direct approach is to use Multidimensional Scaling to represent the data set in an Euclidean space. Finally, it is also possible to define a new kernel matrix as $K^{*T} K^*$ [17].

2.2 The Exponencial and Polynomial Weighting Scheme Methods

The next two methods are influenced by the ideas in [12, 5] where the variables are weighted according to their relative discrimination power. We make use of similar ideas to raise the weight of kernels with expected good classification performance and, analogously, to diminish the influence of less informative kernels.

Let

$$\bar{P}(y_i|j) = \frac{1}{M} \sum_{m=1}^{M} P_m(y_i|j) , \tag{5}$$

and

$$r_m(i,j) = \frac{\left(\frac{\bar{P}(y_i|j) + \bar{P}(y_j|i)}{2} - \frac{P_m(y_i|j) + P_m(y_j|i)}{2} \right)^2}{\frac{P_m(y_i|j) + P_m(y_j|i)}{2}} , \tag{6}$$

where $r_m(i,j)$ is designed to measure the ability of kernel m to predict $\bar{P}(y_i|j)$ and $\bar{P}(y_j|i)$. The value of $r_m(i,j)$ will be inversely related to the discrimination power of K_m with respect to the whole set of kernels: The numerator in (6) approaches zero when the information conveyed by K_m tends to be similar to the information collected by the entire set of kernels.

Now, we construct $w_m(i,j)$ as a function of $r_m(i,j)$. The relative relevance of kernel K_m can be evaluated by:

$$w_m(i,j) = \tau \, exp \left(q \, \frac{1}{r_m(i,j)} \right) . \tag{7}$$

We call this method 'exponential weighting scheme' (**ExpWS**). The parameter q is used to control the influence of $r_m(i,j)$ on $w_m(i,j)$. If $q = 0$, this influence is ignored, and the method reduces to the AKM method. On the other hand, for large values of q, changes in r_m will be exponentially reflected in w_m. A different choice to quantify the relative importante of K_m is given by:

$$w_m(i,j) = \tau \left(\frac{1}{r_m(i,j)} \right)^q , \tag{8}$$

where τ and q play the same role as before. Using $q = 1, 2$ we have linear and quadratic weighting schemes, respectively. We will label this method as 'polynomical weighting scheme' (**PolyWS**).

2.3 The Percentile Methods

Unlike the previous methods, the techniques introduced in this section do not build a functional (matrix) combination of kernels. Consider the ordered sequence of kernels:

$$\min_{1 \le m \le M} K_m(i,j) = K_{[1]}(i,j) < K_{[2]}(i,j) < \ldots < K_{[M]}(i,j) = \max_{1 \le m \le M} K_m(i,j) .$$

The two new methods we propose build each element of K^* using, respectively, the following formulae:

$$K^*(i,j) = K_{\left[\left(\frac{\bar{P}(y_i|j) + \bar{P}(y_j|i)}{2} \right) M \right]} , \tag{9}$$

$$K^*(i,j) = \frac{K_{\left[\bar{P}(y_i|j) M \right]} + K_{\left[\bar{P}(y_j|i) M \right]}}{2} . \tag{10}$$

We will denote these methods by **'percentil-in method'** and **'percentil-out method'**, respectively.

If the class probabilities $\bar{P}(y_i|j)$ and $\bar{P}(y_i|j)$ are high, we can expect a high similarity between i and j and both methods will guarantee a high $K^*(i,j)$. If the class probabilities $\bar{P}(y_i|j)$ and $\bar{P}(y_i|j)$ are both low, $K^*(i,j)$ will be also low.

2.4 The 'MaxMin' Method

The next method can be considered as a mixture of the previous combination techniques. The method produces a functional combination of two kernels, namely, the maximun and the minimun of the ordered sequence of kernels:

$$\begin{aligned} K^*(i,j) = K_{[M]}(i,j)\frac{\left(\bar{P}(y_i|j) + \bar{P}(y_j|i)\right)}{2} &+ \\ K_{[1]}(i,j)\frac{\left(\bar{P}(-y_i|j) + \bar{P}(-y_j|i)\right)}{2} \end{aligned} \tag{11}$$

If i and j belong to the same class then the conditional class probabilities $\bar{P}(y_i|j)$ and $\bar{P}(y_j|i)$ will be high and the method guarantees that $K^*(i,j)$ will be large. On the other hand, if i and j belong to diferent classes the conditional class probabilities $\bar{P}(y_i|j)$ and $\bar{P}(y_j|i)$ will be low and the method will produce a value close to the minimun of the kernels.

2.5 The 'Pick-out' Method

This is the limiting case of the 'MaxMin' method. We take $\bar{P}(y_i|j) = \bar{P}(y_j|i) = 1$ if i and j belong to the same class and $\bar{P}(y_i|j) = \bar{P}(y_j|i) = 0$ otherwise.

$$K^*(i,j) = \begin{cases} \max\limits_{1\le m\le M} K_m(i,j), & \text{if } i \text{ and } j \text{ belong to the same class} \\ \min\limits_{1\le m\le M} K_m(i,j), & \text{if } i \text{ and } j \text{ belong to different classes} \end{cases} \tag{12}$$

In this way, if i and j are in the same class, it is guaranteed that $K^*(i,j)$ will be the largest possible according to the available information. If i and j belong to different classes, we can expect a low similarity between them, and this is achieved by the choice of the minimum kernel value. This method was first introduced in [13], in the context of classification problems with asymmetric similarity measures.

3 Experiments

To test the performance of the proposed methods, a SVM has been trained on several real data sets using the corresponding kernel matrix K^*. For the ProbWS, ExpWS and PolyWS methods the value of the parameter q has been assigned via cross-validation.

Given a non labelled data point x, $K(x,i)$ has to be evaluated. We can calculate two different values for $K(x,i)$, the first one assumming x belongs to class $+1$ and the second assumming x belongs to class -1. For each assumption, all we have to do is to compute the distance between x and the SVM hyperplane and to assign x to the class corresponding to the largest distance from the hyperplane.

In the following, for all the data sets, we will use 80% of the data for training and 20% for testing.

We have compared the proposed methods with the following classifiers: Multivariate Additive Regression Splines (MARS) [6], Logistic Regression (LR), Linear Discriminant Analysis (LDA), k-Nearest Neighbour classification (kNN) and SVMs using a RBF kernel $K_c(x_i, x_j) = e^{-\|x_i - x_j\|^2/c}$, with $c = 0.5d$, where d is the data dimension (see [18] for details).

3.1 Cancer Data Set

In this section we have dealt with a database from the UCI Machine Learning Repository: the Breast Cancer data set [11]. The data set consists of 683 observations with 9 features each. For this data set we have combined three kernels: a polynomial kernel $K_1(x_i, x_j) = (1 + x_i^T x_j)^2$, a RBF kernel $K_2(x_i, x_j) = exp(-||x_i - x_j||^2)$ and a linear kernel $K_3(x_i, x_j) = x_i^T x_j$. We have normalized the kernels in order to compare them: $\bar{K}_{ij} = \frac{K_{ij} - \min(K_{ls})}{\max(K_{ls}) - \min(K_{ls})}$. The results, averaged over 10 runs, are shown in Table 1.

Table 1. Classification errors for the cancer data.

Method	Train error	Test error	Support vectors
$K.$:**Polynomial**	0.1 %	8.6 %	8.0 %
$K.$:**RBF**	0.0 %	10.2 %	65.7 %
$K.$:**Linear**	2.6 %	3.6 %	7.1 %
AKM	1.3 %	3.8 %	31.4 %
ProbWS	1.8 %	3.2 %	39.6 %
ExpWS	0.0 %	3.2 %	90.6 %
PolyWS	2.4 %	2.9 %	33.4 %
Percentil-in	1.9 %	3.4 %	59.9 %
Percentil-out	1.5 %	3.5 %	33.1 %
MaxMin	0.7 %	2.9 %	24.5 %
Pick-out	1.7 %	2.9 %	11.4 %
kNN	2.79%	3.6 %	
MARK-L	0.0 %	11.7 %	18.3 %
MARS	2.7 %	3.2 %	
LDA	3.8 %	4.4 %	
LR	13.1 %	13.1 %	
SDP	0.0 %	6.6 %	61.4 %
SVM	0.0 %	4.4 %	49.5 %

The Pick-out, MaxMin and PolyWS methods show the best overall performance. All our combination methods provide better results than the SVM with a single kernel, using usually significantly less support vectors. The standard deviation of the test error (over 10 runs) was below 1% for all the studied methods.

It is well known that the choice of kernel parameters is often critical for the good performance of SVMs. Combining kernels provides a solution that minimizes the effect of a bad parameter choice. Next we illustrate this situation using a battery of RBF kernels on the cancer data set. Let $\{K_1, \ldots, K_{12}\}$ be a set of RBF kernels with parameters $c = 0.1, 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100$ respectively. Table 2 shows the performance of the proposed methods when combining all these kernels. Again, the results have been averaged over 10 runs.

Table 2. Classification errors for the cancer data using a battery of RBF kernels.

Method	Train error	Test error	Support vectors
Best RBF	2.3 %	2.6 %	13.6 %
Worst RBF	0.0 %	24.8 %	73.5 %
AKM	1.6 %	2.9 %	21.6 %
ProbWS	1.4 %	2.9 %	19.2 %
ExpWS	1.6 %	2.8 %	21.4 %
PolyWS	0.1 %	2.9 %	31.6 %
Percentil-in	1.9 %	2.6 %	7.7 %
Percentil-out	1.7 %	2.5 %	9.5 %
MaxMin	1.9 %	2.6 %	9.0 %
Pick-out	2.7 %	3.2 %	7.7 %
MARK-L	0.0 %	3.5 %	18.3 %
SDP	0.0 %	3.1 %	39.1 %

The Percentil-out method improves the best RBF kernel under consideration while the MaxMin and Percentil-in methods show a similar performance to that of the best RBF. In particular, the results provided by the combination methods are not degraded by the inclusion of kernels with a bad generalization performance.

3.2 A Handwritten Digit Recognition Problem

The experiment in this section is a binary classification problem: the recognition of digits '7' and '9' from the Alpaydin and Kaynak database [1]. The data set is made up by 1128 records, represented by 32×32 binary images. We have employed three different methods to specify features in order to describe the images. The first one is the 4×4 method: features are defined as the number of ones in each of the 64 squares of dimension 4×4 . The second method was introduced by Frey and Slate [7]: 16 attributes are derived from the image, related to the horizontal/vertical position, width, height, etc. The last method under consideration was designed by Fukushima and Imagawa [8]: features are defined as a collection of 12 different representations in a 4×4 square. This is a typical example with several different sources of information and probably complementary. We have used these representations to calculate three kernels from the Euclidean distance. Classifier performance for all the methods is tabulated in Table 3. We have taken the 4×4 representation to train kNN, MARS, LR and LDA methods. The Percentil-in and Percentil-out methods achieve the best results on classification. Furthermore, the MaxMin, Pick-out and ExpWS combinations improve the results obtained using the rest of the techniques except kNN. The excellent performance of kNN may be due to the fact that the result in the table has been obtained using only the best digit codification (4×4). When the other codification methods are used, the error of kNN increases up to 1.7%.

Table 3. Classification errors for the handwritten digit data set.

Method	Train error	Test error	Support vectors
4×4	0.0 %	3.6 %	3.6 %
Frey-Slate	5.5 %	11.1 %	9.8 %
Fukushima	0.0 %	4.5 %	7.0 %
AKM	0.0 %	4.5 %	13.1 %
ProbWS	0.0 %	4.5 %	13.1 %
ExpWS	0.5 %	3.3 %	15.1 %
PolyWS	0.5 %	3.6 %	4.6 %
Percentil-in	0.0 %	1.1 %	10.9 %
Percentil-out	0.0 %	1.1 %	32.8 %
MaxMin	0.0 %	1.9 %	34.2 %
Pick-out	0.0 %	3.1 %	16.9 %
kNN	0.0 %	0.6 %	
MARK-L	0.0 %	4.2 %	13.0 %
MARS	0.1 %	3.9 %	
LDA	0.4 %	5.0 %	
LR	0.0 %	3.6 %	
SDP	0.0 %	3.6 %	6.2 %

4 Conclusions

In this work we have proposed several techniques for the combination of kernels within the context of SVM classifiers. The suggested methods compare favorably to other well established classification techniques and also to other kernel combining techniques in a variety of real data sets. Within the group of the combining techniques proposed in this paper, there is not an overall better method. Further research will focus on the theoretical properties of the methods and extensions.

Acknowledgments

This work was partially supported by Spanish grants BEC2000-0167 (DGICYT), TIC2003-05982-C05-05 (MCyT) and PPR-2003-42 (URJC).

References

1. E. Alpaydin and C. Kaynak. *Cascading Classifiers.* Kybernetika 34(4):369-374, 1998.
2. K. Bennett, M. Momma, and J. Embrechts. *MARK: A Boosting Algorithm for Heterogeneous Kernel Models.* Proceedings of SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002.

3. C. Cortes and V. Vapnik. *Support Vector Networks.* Machine Learning, 20:273-297, 1995.
4. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines.* Cambridge University Press, 2000.
5. C. Domeniconi, J. Peng and D. Gunopulos *Adaptive Metric Nearest Neighbor Classification.* Proc. of IEEE Conf. on CVPR, 517-522, 2000.
6. J. Friedman. *Multivariate adaptative regression splines (with discussion).* Annals of Statistics, 19 (1):1-141, 1991.
7. P.W. Frey and D.J. Slate. *Letter Recognition Using Holland-Style Adaptive Classifiers.* Machine Learning, 6 (2):161-182, 1991.
8. K. Fukushima and T. Imagawa. *Recognition and segmentation of connected characters with selective attention.* Neural Networks, 6:33-41, 1993.
9. T. Joachims. *Learning to Classify Text using Support Vector Machines.* Kluwer, 2002.
10. G.R.G. Lanckriet, N. Cristianini, P. Barlett, L. El Ghaoui and M.I. Jordan. *Learning the kernel matrix with semi-definite programming.* Journal of Machine Learning Research, 5:27-72, 2004.
11. O.L. Mangasarian and W.H. Wolberg. *Cancer diagnosis via linear programming.* SIAM News, 23 (5):1-18, 1990.
12. A. Muñoz and T. Villagarcía. *Unsupervised neural networks for variable selection with mixed covariates.* Analyse Multidimensionelle des Donnes, CSIA Ceresta, 217-227, 1998.
13. A. Muñoz, I. Martín de Diego and J.M. Moguerza. *Support Vector Machine Classifiers for Assymetric Proximities.* Proc. ICANN (2003), LNCS, Springer, 217-224.
14. E. Pekalska, P. Paclík and R.P.W. Duin. *A Generalize Kernel Approach to Dissimilarity-based Classification.* Journal of Machine Learning Research, Special Issue on Kernel Methods 2 (2):175-211, 2002.
15. T. Poggio and F. Girosi. *Networks for Approximation and Learning.* Proceedings of the IEEE, 78(10):1481-1497, 1990.
16. B. Schölkopf, R. Herbrich, A. Smola and R. Williamson. *A Generalized Representer Theorem.* NeuroCOLT2 TR Series, NC2-TR2000-81, 2000.
17. B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K. Müller, G. Rätsch and A. Smola. *Input Space versus Feature Space in Kernel-based Methods.* IEEE Transactions on Neural Networks 10 (5):1000-1017, 1999.
18. B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola and R.C. Williamson. *Estimating the Support of a High Dimensional Distribution.* Neural Computation, 13(7):1443-1471 , 2001.
19. L. Vandenberghe and S. Boyd. *Semidefinite programming.* SIAM Review, 38(1):49-95, 1996.

Fusing Segmentation and Classification from Multiple Features

Roberto Manduchi

University of California, Santa Cruz
Santa Cruz, CA 95064

Abstract. This paper presents a strategy for combining the results of image classification and image segmentation. The visual features used for classification and segmentation may be different in general. Fusion is performed in a Maximum Likelihood framework using the Expectation Maximization algorithm. Preliminary results show that segmentation may effectively contribute to increase the quality of classification.

1 Introduction

In several computer vision problems, the analyst has access to different types of observables (let's call them "features") for the same image. These features often correspond to very different physical causes. For example, the color of a pixel depends on the combination of the reflectance of the surface, the spectrum of the illuminant, and the illumination geometry. The texture around a pixel depends on local albedo and/or geometrical variations. The optical flow at one point depends on the 3D motion of the imaged surface relative to the camera. All of these different observables should be combined to infer information about the scene.

Two fundamental low–level tasks of image analysis are segmentation/grouping and classification, as summarized below.

1. *Segmentation/grouping*: The aim is to identify perceptually homogeneous regions in the image. Such regions don't have pre-defined *labels*: they do not necessarily correspond to semantic categories known in advance. Segmentation is typically performed by first establishing a suitable similarity metric, sometimes derived by generative models. The segmentation problem can then be recast as an optimization task (energy or cost minimization [17], Maximum Likelihood or Maximum a posteriori [16]). This approach is inherently global, in that the decision at any pixel requires examination of all other points in the image.
2. *Classification (labeling)*: In this case, a number of classes is known in advance, together with their statistical description or at least with a number of labeled training samples. The goal is to assign each image point or region to just one class. Classification may use image features defined at the pixel level (such as

color, motion field or texture[1]), or at a higher abstraction level (e.g., shape or spatial distribution). Note that even in the case of pixel–based classification, global reasoning is often invoked to enforce spatial coherence priors.

Whereas these two families of algorithms are traditionally used in different contexts, we show in this paper that segmentation, an unsupervised process that is blind to the semantic categories defined by the user, can actually be used to improve the classification process. This is particularly useful in two cases of practical importance. The first case is when the visual features used to segment the image cannot be used for classification. For example, gradient–based techniques such as snakes can effectively isolate an image region, but classification based on the boundary contour alone may be difficult. Another example is the use of optical flow to identify areas corresponding to different motion models. Once these regions have been segmented out, further reasoning, possibly using different features such as color, texture and shape, should be used for region labeling.

The second case of interest is when there is shortage of training data. Due to the "curse of dimensionality", high–dimensional features (e.g., texture) require much larger training data sets than low–dimensional features (e.g., color). Whereas learning a classifier may be unpractical in such cases, clustering only requires a notion of distance in feature space, which can be defined without reliance on training data.

Intuitively, segmentation should provide some sort of prior information to the classifier. If two image points belong to the same segment, it is reasonable to expect (although by no means necessary) that they also belong to the same class. A naive application of this intuition could lead to a procedure that assigns all points in the same segment to the class that is best represented in the segment. This "hard" fusion policy, however, would be unsatisfactory in general; a softer strategy that takes into consideration the degree of confidence in both classification and segmentation would be much more desirable. Indeed, our algorithm requires that the result of these two operation is expressed either in terms of a class– (or segment–) posterior distribution or, equivalently, in terms of class– (or segment–) conditional likelihoods. These type of information is normally available from the classifier. For what concerns the segmenter, some algorithms do produce soft cluster assignment (e.g., Expectation Maximization), while others produce hard (binary) assignments (e.g. k–means, graph cutting, snakes). It is always possible, though, to artificially "soften up" the result of hard segmenter, by creating at each point a distribution over the set of segments, peaking at the segment assigned to that point.

The intuition behind our approach is simple. The segmenter identifies areas that are homogeneous, according to one considered feature. We hypothesize that there is a correlation between these segments and the semantic classes that we actually interested in. The result of the classifier (based on a different feature)

[1] Strictly speaking, texture is an attribute of a region, not of a single pixel. However, one may define a texture field, by assigning to each pixel the texture of a small region centered in that pixel.

will be biased toward using any evidence of such correlation. The correlation between clusters and classes, however, is unknown, and must be estimated from the image being analyzed. This "chicken and egg" problem is solved using the elegant formalism of the Expectation Maximization algorithm.

The main hypothesis used by our approach is the conditional independence of the two features for each class/cluster combination. This is a generalization of class–conditional feature independence, often assumed in decision fusion. When this property is satisfied, it is well known that the posterior class distribution given the two observed features factorizes into the product of the two marginal posterior class distributions (divided by the class prior). Classifiers that compute the product of these two posterior distributions are often called "naive Bayes". Since the conditional assumption is at the core of our algorithm, we discuss its relevance and shortcomings in Section 2. Our iterative solution to the class/segment fusion is introduced in Section 3, where we also show an example of application. Section 4 has the conclusions.

2 Conditional Independence and Bayes Fusion

We introduce here the notation that will be used throughout this article. Let f_1 and f_2 be two different local feature vectors. For example, f_1 could be the (r,g,b) color at a pixel, and f_2 the texture descriptor at the same pixel. Consider a set of N classes $\{c_j\}$. The class–conditional likelihood of feature f_i given class c_j is represented by $p_i(f_i|c_j)$. P_j represents the prior probability of class c_j, while $P_i(c_j|f_i)$ is represents the class–posterior probability distribution for a given feature f_j. Bayes' rule can thus be expressed as $P_i(c_j|f_i) = p_i(f_i|c_j)P(c_j)/p(f_i)$, where $p_i(f_i)$ is the total likelihood of feature f_i. The mode of the posterior probability yields the Bayes classification at the chosen pixel.

The fusion problem arises when we have independent information about class assignment from the two features f_1 and f_2. We would like to infer $P_{1,2}(c_j|f_1, f_2)$ from $P_1(c_j, f_1)$ and $P_2(c_j, f_2)$ (or, equivalently, $p_{1,2}(f_1, f_2|c_j)$ from $p_1(f_1|c_j)$ and $p_2(f_2|c_j)$). Unfortunately, it is impossible, in general,to infer the joint density $p_{1,2}(f_1, f_2|c_j)$ from its marginals. A popular simplifying assumption is the class–conditional independence of the two features, that is:

$$p_{1,2}(f_1, f_2|c_j) = p_1(f_1|c_j)p_2(f_2|c_j) \tag{1}$$

for each choice of class c_j. This assumption is easily transformed into an equivalent condition on the posterior distributions:

$$P_{1,2}(c_j|f_1, f_2) = P_1(c_j|f_1)P_2(c_j|f_2)/P(c_j) \tag{2}$$

Equation (2) determines a *Bayes fusion* classifier, more commonly known as a *naive Bayes* system [11, 3]. These two terms will be liberally interchanged in this work.

How acceptable is the conditional independence assumption? Although it is a weaker condition than total independence, conditional independence can be grossly violated in practice. The real question, however, is how much this (generally wrong) assumption affects the classification performances. For example, it would be interesting to compare the misclassification rate of the Bayes fusion classifier with the (necessarily lower) Bayes rate (that is, the misclassification rate of the Bayes classifier which uses the real joint posterior $P_{1,2}(c_j|f_1, f_2)$ [16]). An analytical expression for such quantities cannot be found in general, although Shi and Manduchi [18] computed an upper bound for the difference of these two misclassification rates as a function of the correlation between the two features in a simple equivariant Gaussian case.

It is well known that, in practical applications, Bayes fusion (or naive) classifiers perform rather well, despite the possible inaccuracy of the approximation in (1) [11, 3]. Experimental studies include [8, 9]. Friedman [4] justifies the sometimes surprisingly good results achieved by naive Bayes classifiers in light of the *bias/variance dilemma.* The bias/variance theory, first introduced by Geman for the regression problem [5], links the expected quadratic estimation error to the randomness in the choice of the training data set and the complexity of the algorithm. More precisely, the *bias* represents the difference between the estimates, averaged over all possible choices of training samples, and the optimal (in L_2 sense) estimate (i.e., the conditional expectation). The *variance* is the actual variance of estimation, again computed using the distribution over the training samples. In general, complex regression algorithms have low bias but high variance (i.e. they may overfit the data), while this behavior is reversed for simpler algorithms. Since the squared bias and the variance contribute as additive terms to the overall estimation error, it is seen that lower complexity algorithms may outperform more complex algorithms when only a limited amount of training data is available (see also [14, 15, 13]).

A similar situation occurs in the case of classification, although the definition of bias and variance is somewhat different here. Friedman [4] first showed that even in this case, variance with respect to the choice of training sample has an important role in the quality of the result (that is, the misclassification rate). Further work in the field includes [10, 1, 2, 19, 20]. In spite of their obvious bias (consequent to the approximation in (1)), naive Bayes systems are described by a "simple" posterior distribution, and it is reasonable to assume that they are less sensitive to the choice of the training data [4]. Shi and Manduchi [18] confirmed this hypothesis, by showing experimentally that the difference between the misclassification rate of a Bayes fusion classifier and the Bayes rate decreases as fewer and fewer data are used for training.

3 Bayes Fusion of Segmentation and Classification

In this section we tackle the main objective of this contribution, namely the fusion of a classifier with a segmenter. As we mentioned in the Introduction, we will assume that segmentation is expressed by either a posterior distribution $P_k(s_k|f_i)$ over the set of segments $\{s_k\}$, or a conditional likelihood $p_k(f_i|s_k)$. Let's assume that f_1 is the feature used for classification, and f_2 is the feature used for segmentation. We would like to be able to use the segmentation using f_2 to assist the classification over the set of classes $\{c_j\}$. Formally, our problem an be formulated as follows:

Given $P_1(c_j|f_1)$ and $P_2(s_k|f_2)$, estimate $P_{1,2}(c_j|f_1, f_2)$.

We could also consider a parallel problem, but defined using the conditional likelihoods:

Given $p_1(f_1|c_j)$ and $p_2(f_2|s_k)$, estimate $P_{1,2}(c_j|f_1, f_2)$.

This formulation makes our fusion problem similar to the case of Section 2, with one important difference: now the two marginal posterior distributions are defined over different sets, $\{c_j\}$ and $\{s_k\}$, that are semantically different (and have different cardinality in general). In order to attempt a solution to this problem, we first extend the notion of conditional independence to the case of conditional likelihoods defined over the cartesian product of the features and the cartesian product of the class/segment sets:

$$p_{1,2}(f_1, f_2|c_j, s_k) = p_1(f_1|c_j)p_2(f_2|s_k) \tag{3}$$

The same cautionary disclaimer about the validity and consequences of the conditional independence approximation, discussed in Section 2, applies to this case as well. Given this assumption, we can use Bayes' rule to write the joint posterior distribution given the two features as follows:

$$P_{1,2}(c_j, s_k|f_1, f_2) = \frac{p_1(f_1|c_j)p_2(f_2|s_k)P_{1,2}(c_j, s_k)}{\sum_{\bar{j},\bar{k}} p_1(f_1|c_{\bar{j}})p_2(f_2|s_{\bar{k}})P_{1,2}(c_{\bar{j}}, s_{\bar{k}})} \tag{4}$$

The posterior distribution $P_{1,2}(c_j|f_1, f_2)$ can then be obtained by marginalizing $P_{1,2}(c_j, s_k|f_1, f_2)$ in (4):

$$P_{1,2}(c_j|f_1, f_2) = \sum_k P_{1,2}(c_j, s_k|f_1, f_2)$$

The only unknown quantity in (4) is the joint prior distribution $P_{1,2}(c_j, s_k)$. In fact, this distribution is the key to understanding our fusion strategy. One easily proves that if the priors are separable, that is, if $P_{1,2}(c_j, s_k) = P_1(c_j)P_2(s_k)$, then segmentation does not contribute to the fusion process. Indeed, in this case $P_{1,2}(c_j, s_k|f_1, f_2)$ factorizes into $P_1(c_j|f_1)P_2(s_k|f_2)$, and marginalization over s_j simply yields $P_1(c_j, s_k|f_1, f_2) = P_1(c_j|f_1)$.

The more interesting cases are when $P_{1,2}(c_j, s_k)$ is not separable, that is, when knowledge about which segment the point belongs to gives us some prior information about the class. Since we don't know $P_{1,2}(c_j, s_k)$, we should try to extract it from the data. We will first consider the case the conditional likelihoods $p_1(f_1|c_j)$ and $p_2(f_2|s_k)$ (and therefore $p_{1,2}(f_1, f_2|c_j, s_k)$ from (3)) are known, or that a reasonable assumption about their values can be made. We can then use a Maximum Likelihood criterion, and search for the joint priors that maximize the likelihood of the data $p_{1,2}(f_1, f_2)$ according to our model, where

$$p_{1,2}(f_1, f_2) = \sum_{j,k} p_{1,2}(f_1, f_2|c_j, s_k) P_{1,2}(c_j, s_k)$$

A classic solution is given by the Expectation Maximization algorithm, based on the following iterations:

1. For each pixel x in the image, use the current values for $P_{1,2}(c_j, s_k)$ to estimate an updated posterior distribution $P_{1,2}(c_j, s_k|f_1(x), f_2(x))$ as by (4);
2. For each class c_j and segment s_k, average the posterior probabilities $P_{1,2}(c_j, s_k|f_1(x), f_2(x))$ over the image to obtain the updated prior distribution $P_{1,2}(c_j, s_k)$.

At each step, the total likelihood $p_{1,2}(f_1, f_2)$ increases or stays the same, and therefore this procedure is guaranteed to converge to a (possibly local) maximum.

If the conditional likelihood of one feature (or both) is not known, but the class– or segment–conditional probability is known, then some modifications are in order. This could be the case when a hard segmenter is artificially transformed into a soft segmenter by creating a simple posterior distribution at each pixel, as mentioned earlier in the Introduction. For example, for a pixel x with feature $f_2(x)$ that was assigned to segment s_k, we could hypothesize a posterior distribution[2]

$$P_2(s_r|f_2) = \begin{matrix} 1 - \epsilon \ , r = k \\ \epsilon/(N_s - 1) \ , r \neq k \end{matrix}$$

where N_s is the number of segments, and ϵ is a (small) positive constant. By averaging the values of $P_2(s_k|f_2)$ over the image, one can estimate the prior $P_2(s_k)$. One could then set an artificial total likelihood $p_2(f_2)$ that is constant over all features in the image. At this point, the conditional likelihoods can be computed using Bayes' rule.

As an application example, consider the image of Figure 1. In this case, color was used for classification, while texture was used for segmentation. A poorly trained color classifier produced the unsatisfactory labeling of Figure 1 (b). The three classes are: obsidian (the blue-ish rock, c_1); basalt (the red rock, c_2); and sand (c_3.) Texture–based unsupervised segmentation into two regions s_1 and s_2 (using Gabor features) yielded the results shown in Figure 1 (c). Texture was unable to separate the two rocks, but did a good job at separating both rocks from the sand. The fused classification (into the original three classes) is shown in Figure 1 (d). It is seen that the quality of classification has improved through fusion, although a small region surrounding the blue rock has been misclassified. Table 1 shoes the joint prior distribution $P_{1,2}(c_j, s_k)$, which cannot be factorized into the product of the two marginal priors.

• This distribution may be inconsistent if two points with exactly the same feature $f_\bullet$ belong to different segments. This has not proven to be a problem in practice.

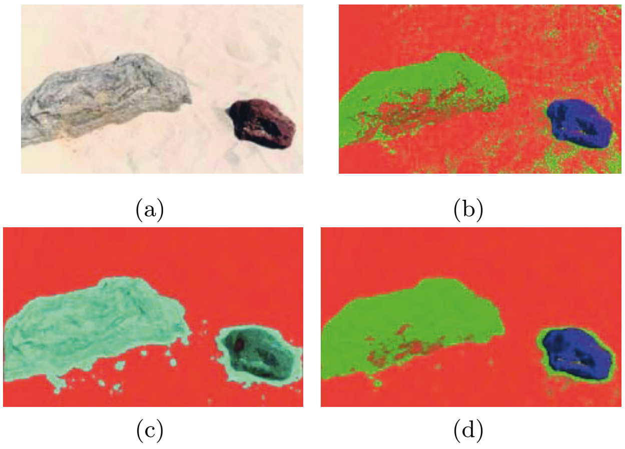


Fig. 1. (a): Original image. (b): Supervised color–based classification (green: obsidian; blue: basalt; red: sand.) (c) Unsupervised texture–based segmentation. (d) Fusion of segmentation and classification.

Table 1. The prior distribution $P_{\bullet,\bullet}(c_j, s_k)$ for the example of Figure 1. Note that $P_{\bullet,\bullet}(c_j, s_k)$ is not separable.

	$s_\bullet$	$s_\bullet$
$c_\bullet$	0.120	0.262
$c_\bullet$	0.001	0.045
$c_\bullet$	0.564	0.008

4 Conclusions

Our fusion technique merges information from classification and segmentation (with each point of the image characterized by an assignment distribution.) In some sense, this corresponds to looking at the segmentation as a kind of classification itself, with classes that don't have a logical correspondence with those used by the classifier.

A more intriguing form of hybrid fusion would also consider cases where only partial segmentation is available, such as an edge segment partially separating two regions. This edge segment may provide useful information to the classifier (the regions at the two sides of the edge are likely to contain points from two different classes). Unless one enforces contour closure, however, this information cannot be directly exploit using the framework discussed in this paper, and more research is needed for this type of problems.

References

1. L. Breiman, "Bias, variance and arcing classifiers", Tech. Report 460, Statistics Department, UC Berkeley, 1996.
2. P. Domingos, "A unified Bias-Variance decomposition for zero-one and squared loss", *AAAI*/IAAI, 564–569, 2000.

3. P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero–one loss", *Machine Learning*, 29(2/3):103–130, November 1997.
4. J.H. Friedman, "On bias, variance, 0/1–loss, and the curse-of-dimensionality", *Data Mining and Knowledge Discovery*, 1, 55-77 (1997).
5. S.L. Geman, E. Bienenstock and R. Doursat, "Neural networks and bias/variance dilemma", *Neural Computation* 4: 1-58, 1992.
6. T. Heskes, "Bias/Variance decomposition for likelihood–based estimators",*Neural Computation* 10, 1425–1433 (1998).
7. N.R. Howe and D.P Huttenlocher, "Integrating Color, Texture, and Geometry for Image Retrieval", *Proc. Computer Vision and Patter Recognition*, 2000, 239–247.
8. J. Kittler, M. Hatef. R. Duin and J. Matas, "On combining classifiers", *IEEE Trans. PAMI* 20(3), March 1998.
9. J. Kittler and A.A. Hojjatoleslami, "A weighted combination of classifiers employing shared and distinct representations", *Proc. CVPR*, 924–929 (1998).
10. E.B. Kong and T.G. Dietterich, "Error–correcting output coding corrects bias and variance", *Proc. Intl. Conf. Machine Learning*, 313–21, 1995.
11. Pat Langley, "Induction of selective Bayesian classifiers", *Proc. of the 10th Conference on Uncertainty in Artificial Intelligence*, Seattle, WA (1994).
12. R. Manduchi, "Bayesian Fusion of Color and Texture Segmentations", *7th IEEE International Conference on Computer Vision*, Kerkyra, September 1999, 956–962.
13. J.K. Martin and D.S. Hirschberg, "Small sample statistics for classification error rates I: Error rate measurements", Dept. of Inf. and Comp. Sci., UC Irvine, Tech. Report 96–21 (1996).
14. S. Raudys and A.K. Jain, "Small sample size effects in statistical pattern recognition: Recommendations for practitioners", *IEEE Trans. PAMI*, 13(3):252–64, 1991.
15. S. Raudys, "On dimensionality, sample size, and classification error of nonparametric linear classification aglgorithms", *IEEE Trans. PAMI*, 19(6):337–71, 1997
16. B.D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, 1996.
17. J. Shi, J. Malik, "Normalized Cuts and Image Segmentation", *Proc. Computer Vision and Patter Recognition*, 1997, 731–737.
18. X. Shi and R. Manduchi, "A Study on Bayes Feature Fusion for Image Classification", *IEEE Workshop on Statistical Algorithms for COmputer Vision*, 2003.
19. R. Tibshirani, "Bias, variance and predicition error for classification rules", Tech. Report, Dept. pf Prev. Medicine and Biosatistics, Univ, of Toronto, 1996.
20. D. Wolpert, "On bias plus variance", *Neural Computation* 9, 1211–1243 (1997).

A Classifier Combination Tree Algorithm

Ross A. McDonald[1], Idris A. Eckley[2], and David J. Hand[1]

[1] Imperial College London
[2] Shell Research Ltd.

Abstract. In recent years a number of authors have suggested that combining classifiers within local regions of the measurement space might yield superior classification performance to rigid global weighting schemes. In this paper we describe a modified version of the CART algorithm, called ARPACC, that performs local classifier combination. One obstacle to such combination is the fact that the 'optimal' covariance combination results originally assumed only two classes and classifier unbiasedness. In this paper we adopt an approach based on minimizing the Brier score and introduce a generalized matrix inverse solution for use in cases where the error matrix is singular. We also report some preliminary experimental results on simulated data.

Keywords: Local Combination, Brier Score, CART

1 Introduction

The notion that the combination of classifiers based on local accuracy estimates may prove superior to an ensemble based on globally defined weights is one that has emerged several times in recent years ([10],[9],[13],[2]), though as yet there would seem to exist little understanding of why such methods should work in practice beyond the purely intuitive, and no algorithmic method for defining a useful partition of the feature space for local combination. In this paper we propose using a modified version of the CART tree algorithm [3] for this purpose. We combine the classifiers in our ensemble locally, by assigning separate weighting schemes to each leaf node of our 'combination tree'. Note the distinction between this idea and the model trees proposed by other authors [6], where local models are fitted within nodes of a tree; we assume that our ensemble classifiers are induced using the full design set, and given this our aim is to combine their output in the most efficient way possible.

Our algorithm, called ARPACC (Automatic Recursive Partitioning Algorithm for Classifier Combination), relies on the use of the 'optimal' results for the weighted sum combination of classifiers through the minimisation of combined error covariance. We have extended these results in a number of ways. We drop the assumption of classifier unbiasedness with respect to the training data, and propose a natural extension to the case of more than two classes. We motivate this through the use of the *Brier score* measure of classifier fit. We also introduce a solution for the optimal weights in cases where the error matrix is singular, together with a necessary and sufficient condition for this to be true (singular error matrices become more common as we consider smaller and smaller subsamples of the design data).

Our intention is to increase the flexibility of our combination scheme and thereby exploit a corresponding decrease in the bias of the combined classifier. The computation of the optimal weighting scheme across a node depends, however, upon our ability to estimate the mean error of a classifier across the corresponding subsample of the design data. As we recursively partition our training space the variance of these estimates is bound to increase, due to the diminishing size of the sample over which it is calculated. This is the familiar bias-variance tradeoff that affects most recursive classification methods. We have observed, however, that ARPACC trees tend to be robust to overfitting especially when combining classifiers that tend naturally to underfit. This is most likely to occur in feature spaces of high dimensionality, or when the number of class labels is large.

In Section 2 of this paper we describe the optimal combination scheme argument that we use to calculate weighting schemes for terminal nodes. Section 3 details the ARPACC algorithm and the results of some simulated trials, while Section 5 provides a visual illustration of local classifier combination.

2 Optimal Weighted-Sum Combination

Suppose that we are presented with a supervised classification problem in some measurement space $\mathbf{x}$, where for a fixed location the posterior class probability of observing class c is generated by some underlying target function $f(c|\mathbf{x})$ taking real values in $[0, 1]$. Suppose also that we have an ensemble of classifiers $\hat{f}_k(c|\mathbf{x})$ that are approximations to this underlying function. Our goal is to combine these classifiers in such a way as to ensure optimal classifier performance, measured in terms of test error rate on future observations. Kittler [11] has demonstrated that one of the most robust combination strategies in this circumstance is the weighted sum rule, or

$$\hat{f}(c|\mathbf{x}) = \sum_{k=1}^{K} w_k \hat{f}_k(c|\mathbf{x}).$$

where the w_k are positive classifier weights to be determined. It is natural to treat this as a regression problem and adopt the strategy of minimizing the total mean-squared distance across the training data between each observed value and our approximation,

$$\frac{1}{N} \sum_{c} \sum_{n=1}^{N} \left(\delta(c|\mathbf{x}_n) - \hat{f}(c|\mathbf{x}_n) \right)^2,$$

where $\delta(c|\mathbf{x}_n) = 1$ if training example n has class c, and 0 otherwise. The above is the *Brier* or *quadratic* score, a common measure of classifier performance. The Brier score is an estimate of the *Brier Inaccuracy* over the full space χ:

$$E_{\chi} \sum_{c} E_{\delta|c,\mathbf{x}} \left(\delta(c|\mathbf{x}) - \hat{f}(c|\mathbf{x}) \right)^2 .$$

It can be shown (see [7]) that minimizing the Brier inaccuracy is equivalent to minimising

$$E_{\mathcal{X}} \sum_{c} (f(c|\mathbf{x}) - \hat{f}(c|\mathbf{x}))^2,$$

which is the mean squared error of our $\hat{f}$ relative to f. This is important because it means that in the absence of any knowledge about $f(c|\mathbf{x})$, we can use our observed design set class labels as a proxy in solving for the optimal weights. Of course, any solution derived in this way is only 'optimal' with respect to the training data: as always we rely on a large sample that is in some sense a good reflection of the target function.

If we denote the 'error' of a particular classifier within our ensemble relative to a specific training object as $e_k(\mathbf{x}_n) = \delta(c|\mathbf{x}_n) - \hat{f}_k(\mathbf{x}_n)$ for class c, then minimising the Brier score of our ensemble is the same as minimising

$$\frac{1}{N} \sum_{n=1}^{N} \sum_{c} \left[\sum_{k=1}^{K} w_k e_k(\mathbf{x}_n) \right]^2,$$

which may be written in matrix form as $w'\mathrm{E}w$, where w is a vector of weights and the ijth entry of the square 'error' matrix E, $e_{ij} = \frac{1}{N} \sum_n [e_i e_j]$. Prior applications of this method set out to minimize either the total error variance of the combined classifier, or the total mean square error of a two class combined classifier. By adopting the multi-class Brier score, we are able to extend the solution to more than two classes in a natural way. We assume that each class label has its own separate associated weighting scheme, so that by minimizing the error of the combined classifier for each $\delta(c|\mathbf{x})$ we are minimizing the total Brier score.

The solution for the weights vector was first derived by Bates and Granger in 1969 [1] for ensembles of size two, and extended to ensembles of more than two classifiers by Dickinson [4]. Dickinson began by assuming that the errors were normally distributed with mean zero, ie. that $E_{\mathcal{X}} e_k = 0 \; \forall \; k$, and this assumption leads to a solution that minimizes the error *variance* of the combined classifier. In fact the assumption is unnecessary, and for our purposes it would clearly be restrictive. Across the global space it may be fair to assume that the expected error of a classifier is zero with respect to the target function, but within a small local region of that space this is very unlikely to be true.

Dropping the assumption leads us to the following revised solution for classifier combination based on mean square error, derived by Hashem and Schmeiser [8] in the context of regression neural networks. They impose the restriction that the w_k should sum to one by introducing the Lagrange multiplier λ, and minimising the function

$$\sigma = \sum_{i,j}^{K} w_i w_j e_{ij} + \lambda \left(\sum_{i=1}^{k} w_i - 1 \right).$$

We differentiate with respect to each of the parameters $(w_1, ..., w_k, \lambda)$ in turn, so that

$$\frac{\delta\sigma}{\delta w_i} = 2\sum_{j=1}^{K} w_j e_{ij} + \lambda \qquad (i = 1, ..., K)$$

and

$$\frac{\delta\sigma}{\delta\lambda} = \sum_{i=1}^{K} w_i - 1.$$

Setting these equal to 0, we get

$$\begin{pmatrix} 1'_K & (0) \\ E & 1_K \end{pmatrix} \begin{pmatrix} w_{\min} \\ \lambda/2 \end{pmatrix} = \begin{pmatrix} (1) \\ 0_K \end{pmatrix}$$

where 1_K and 0_K are the column vectors of K ones and K zeros respectively, and the notation () denotes a single integer. Thus

$$\begin{pmatrix} w_{\min} \\ \lambda/2 \end{pmatrix} = \begin{pmatrix} 1'_K & (0) \\ E & 1_K \end{pmatrix}^{-1} \begin{pmatrix} (1) \\ 0_K \end{pmatrix}.$$

The solution to the above is given by the first entry in the inverted matrix, so:

$$w_{\min} = \frac{E^{-1}1_K}{1'_K E^{-1} 1_K}.$$

Negative weights may arise for ensembles of more than two classifiers. In extreme cases, this can lead to combined predictions outside the range $[0, 1]$. As a stop-gap measure, we can deal with this by setting negative output predictions to 0, and output predictions greater than 1 to 1.

In the course of our research we also extended the solution to cases where the matrix E proves to be singular. This can frequently happen by accident, especially where component classifiers make similar predictions, or where we perform a large number of calculations on training sets of diminishing size, as with the algorithm in the next section. As we base our combination weights on smaller and smaller subsets of the training data, the likelihood of our encountering a singular error matrix becomes very high. Our more general solution is as follows:
If E is non-singular

$$w_{min} = \frac{E^{-1}1_K}{1'_K E^{-1} 1_K}.$$

If E is singular form the matrix $Q = E + 1_K 1'_K$. If Q is non-singular

$$w_{min} = Q^{-1} 1_K.$$

If Q is singular

$$w_{min} = \frac{Q^g 1_K}{1'_K Q^g 1_K}$$

where g denotes the generalized matrix inverse. We derived the following as a necessary and sufficient condition for the matrix E to be singular. Let $\hat{Y}_i$ denote the vector of predictions of ensemble classifier i on the training data and Y denote the vector of true class labels. If the matrix E is singular, then for some subset(s) of classifiers $i \in S$, say, there exists a linear combination $\sum_{i \in S} s_i \hat{Y_i}$ such that

$$(\sum_{i \in S} s_i) Y = \sum_{i \in S} s_i \hat{Y_i}.$$

The converse is also true.

The above condition will be met if one or more classifiers match the training data predictions exactly (but note that this does not necessarily imply that the classifiers are identical). It will also be met if two or more classifiers, or linear combinations of subsets of classifiers, make identical predictions (this corresponds to $\sum_{i \in S} s_i = 0$). This is most likely to happen when the number of training samples is small, as for a fine partition of the training data.

It is also possible to show that the total squared error of the combined ensemble is less than or equal to that of any component classifier. The proof is similar to the proof in [5] relating to combined variance.

3 The ARPACC Algorithm

In this section we outline the results of some preliminary experiments on simulated data produced using ARPACC, a modification of the CART algorithm [3] that uses the optimal weighting results of the previous section to build classifier combination trees.

To modify the CART algorithm to perform local classifier combination we need to change the the splitting criterion and the calculation of node predictions. Rather than associating each terminal node with a prediction based on the assigned training data, each node is treated as a partition and is assigned the optimal local weighting scheme for the given training sample at that node, calculated via the expressions at the end of the previous section. The splitting criterion is calculated by applying these weights to the predictions of the classifiers under consideration, and summing the training error of the combined classifier.

We deal with problems of more than two classes by associating a separate set of weights with each class (so that we might simultaneously decide that we trust ensemble classifier A's class 1 predictions, but assign a low weight to its class 3 predictions, for example). During splitting the total current error of the tree is replaced by the expected Brier score across all nodes and training examples.

The chief difference between this and the standard tree classifier is that our algorithm *does not make any predictions in its own right.* The model output by ARPACC is only a framework in which global classifiers are combined locally. This is underlined by the fact that any or all of these classifiers could themselves be trees.

Our first simulated experiment consists of a series of trials, each comprising five random repetitions of classifier fitting and combination. In each trial we first fix the value of three parameters: s, the sample size for each class, d, the dimensionality of the problem, and c, the total number of classes.

For each class and each dimension, we generate two integer values between 1 and 100 uniformly and at random, and then generate s random normal variates using these values as the mean and variance. These values are rounded down to the nearest integer (to speed up computation). The resulting clusters are rotated by forty-five degrees in consecutive pairs of dimensions. Two classifiers are fitted to the training sample: a CART tree computed using S-Plus with the default parameters, and Venables and Ripley's multinomial method, computed with S-Plus using the function *multinom* [12], which uses neural networks to fit log-linear models. The CART tree stopping criterion is a node size of 5, and the ARPACC trees are grown to a fixed size of 50 terminal nodes.

Five trials are run for each choice of parameters. The results in Table 1 below summarize the sample sizes, and the mean improvement of the global and local combination methods over the best single-classifier test error rate, as a percentage of points misclassified.

Table 1. Results of simulated experiments with a local combination algorithm.

Classes	Dimensions	Train Size	Test Size	Global Test Error Improvement	ARPACC Test Error Improvement
2	2	2,000	2,000	0.00% s.d 0.14	0.16% s.d 0.13
3	3	3,000	3,000	-0.03% s.d 0.10	0.52% s.d. 0.42
5	5	5,000	5,000	1.00% s.d. 1.12	2.90% s.d. 1.55
6	4	2,000	2,000	0.42% s.d. 0.34	1.70% s.d. 0.39
10	8	2,000	2,000	1.72% s.d. 2.89	3.90% s.d. 2.00

The above results would appear to suggest that the performance of the ARPACC tree improves as the number of dimensions and classes is increased. We believe that this is due to the propensity of our standard classifiers to underfit problems of high dimension, with the combination tree providing us with a measured means of correcting for this underperformance.

4 Combining a Tree and a Neural Network Using ARPACC

The aim of this section is to provide a visual representation of what an ensemble classifier created using the ARPACC algorithm might look like. We begin by generating 500 training and 2,000 test observations distributed uniformly in two dimensions, and use the target probability function

$$f(1|\mathbf{x}) = \frac{cos\left(2\pi\sqrt{(x_1^2 + x_2^2)}\right) + 1}{2}$$

to randomly assign each observation to one of two categorical classes. Panel 1 in Figure 1 shows how the design sample appears in two dimensional space. Panel 2 is a contour plot of the generating function f across the same range.

To our design sample we fit a CART tree, using the *tree* function in S-Plus, and a neural net created using Venables and Ripley's neural net library for S-Plus [12]. This is a 2-3-1 neural network with 13 weights, and the fitting algorithm is allowed to run for 100 iterations.

Contour plots of the predictions made by the tree model and the neural network model are shown in panels 3 and 4 of figure 5. Here the greyscale level represents the 'confidence' level of the prediction, ranging from black (high confidence in the first class) to white (high confidence in the second class). The test error rates for the tree and the neural net, ie. the number of misclassified test examples divided by the total number of test examples, are 0.221 and 0.2225 respectively. Although these test error rates are close, a visual inspection tells us that the forms of the fitted models are in fact very different, and that both underfit. Empirically we have found that the pairing of a tree and a neural net model is a good one for combination, since they tend to complement each other well.

We now combine the two models globally using two sets of weights generated using the optimal MSE solution. The weights are 0.91 for the tree and 0.09 for the neural network. This means that the combined model heavily favours the tree over the neural network. The resultant predictions are plotted in panel 5. The test error rate of this globally combined model is 0.215, lower than that of either of the component models. Visually, the combined classifier appears a slightly better fit to the true f.

Our final model is that produced when the ARPACC algorithm uses the training data to fit a fifty node combination tree. The test error rate for this model is 0.2, a further improvement on the global combination. A visual examination of the plot in panel 6 tells us that at least some of this improvement probably results from an improved model fit in the region around the centre of the plot.

The test error results for all our models, as a fraction of points misclassified, are summarized in Table 2.

Table 2. Absolute training and test errors for the models used in the simulated example.

Model	**Training error**	**Test Error**
CART Tree	0.1280	0.2210
Neural Net	0.1780	0.2225
Global Combined Model	0.1260	0.2150
50-Node ARPACC	0.1100	0.2000

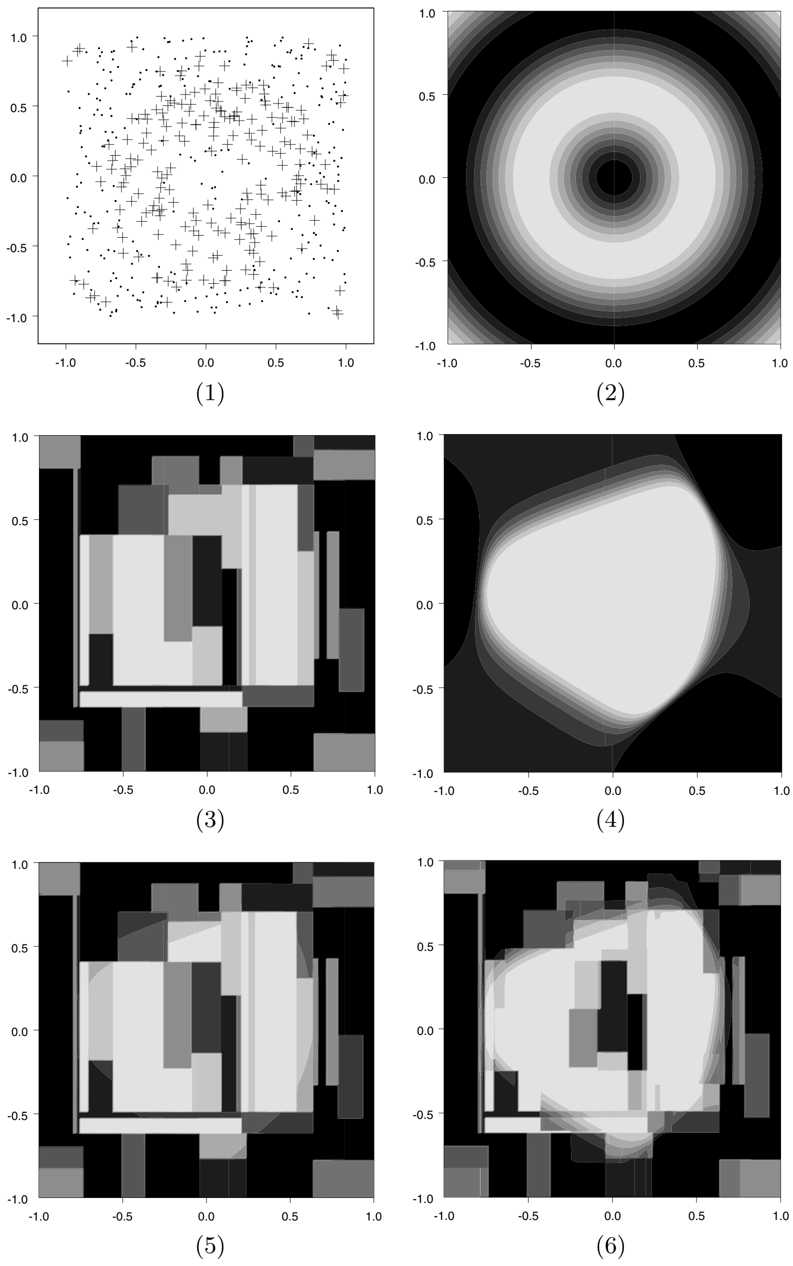


Fig. 1. An example of local classifier combination. (1) 500 data points in 2 dimensions randomly mapped to 2 classes with probabilities generated via a smooth cosine function. (2) The underlying function f. (3) A CART tree fitted to the training data. (4) A 2-3-1 neural network fitted to the training data. (5) The optimal global combined classifier. (6) A local combination as fitted by a 50-node ARPACC tree.

5 Conclusions and Future Work

In this paper we have proposed the use of optimal weights within local regions of the measurement space, have introduced solutions for more than two classes and singular error matrices, and have experimented with the ARPACC algorithm as one means of defining a suitable feature space partition.

We have also experimented with a pruning method for ARPACC based on the minimal cost-complexity pruning methods used for CART. At present our Matlab implementation of this proves computationally expensive, and we hope to improve it in the future. We also plan to experiment with ensembles of more than two classifiers, and to produce test error results for real classification problems.

Acknowledgements

RAM was supported in this work by Shell Research Ltd. and research grant number 0130322X from the Engineering and Physical Sciences Research Council.

References

1. J. M. Bates and W. J. Granger. The combination of forecasts. *Operational Research Quarterly*, 20:451 – 468, 1969.
2. H. Blockeel and J. Struyf. Frankenstein classifiers: Some experiments on the sisyphus data set. *Proceedings of IDDM 2001*, pages 1 – 12, 2001.
3. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, U.S., 1984.
4. J. P. Dickinson. The combination of short term forecasts. *Proc. Univ. of Lancaster Forecasting Conference*, 1972.
5. J. P. Dickinson. Some comments on the combination of forecasts. *Operational Research Quarterly*, 26:205 – 210, 1975.
6. E. Frank, Y. Wang, S. Inglis, G. Holmes, and I. Witten. Using model trees for classification. *Machine Learning*, 32:63–76, 1998.
7. D. J. Hand. *Construction and Assessment of Classification Rules*. John Wiley & Sons, Chichester, 1997.
8. S. Hashem and B. Schmeiser. Improving model accuracy using optimal linear combinations of trained neural networks. *IEEE Transactions on Neural Networks*, 6:792 – 794, 1995.
9. T. M. Joergensen and C. Linneberg. Feature weighted ensemble classifiers - a modified decision scheme. In *Multiple Classifier Systems (MCS) 2001*, pages 218 – 227. Springer-Verlag, 2001.
10. M. S. Kamel and N. M. Wanas. Data dependence in combining classifiers. In *Multiple Classifier Systems (MCS) 2003*, pages 1 – 14. Springer-Verlag, 2003.
11. J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3), 1998.
12. W. Venables and B. Ripley. *Modern Applied Statistics with S-Plus*. Springer-Verlag, 1994.
13. K. Woods, W. P. Kegelmeyer, and K. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machie Intelligence*, 19:405 – 410, 1997.

Bounds for the Average Generalization Error of the Mixture of Experts Neural Network

Luís A. Alexandre[1,⋆], Aurélio Campilho[2], and Mohamed Kamel[3]

[1] Networks and Multimedia Group, IT, Covilhã, Portugal
lfbaa@di.ubi.pt
[2] INEB - Instituto de Engenharia Biomédica, Portugal
[3] Dept. Systems Design Engineering, Univ. Waterloo, Canada

Abstract. In this paper we derive an upper bound for the average-case generalization error of the mixture of experts modular neural network, based on an average-case generalization error bound for an isolated neural network. By doing this we also generalize a previous bound for this architecture that was restricted to special problems.
We also present a correction factor for the original average generalization error, that was empirically obtained, that yields more accurate error bounds for the 6 data sets used in the experiments. These experiments illustrate the validity of the derived error bound for the mixture of experts modular neural network and show how it can be used in practice.

Keywords: modular neural networks, mixture of experts, generalization error bounds

1 Introduction

This paper addresses generalization error bounds for supervised learning. In [1], an average-case generalization error bound was introduced that is not as pessimistic as the error bounds derived using Vapnik-Chervonenkis theory [2] since the later are based on a worst case analysis. In this paper we derive an upper bound for the average-case generalization error of the mixture of experts (ME) [3] modular neural network (MNN), based on the average-case generalization error bound introduced in [1] for a single multi-layer perceptron (MLP). By doing this we also generalize a previous bound for this architecture [4] that was restricted to special problems (problems that could be completely separated into two subproblems without overlapping classes, by a hyperplane in the input space) and was derived assuming that the gate does not introduce errors and does not use adjustable parameters.

The bound proposed here assumes that the generalization error of the experts is higher than their training set error. In fact, this assumption was already made in the derivation of the bound for the isolated MLP in [1]. It also assumes independency between the errors of the gate and of the experts.

⋆ We acknowledge the support of project number POSI/35590/SRI/2000 approved by the Portuguese ‘Fundação para a Ciência e Tecnologia’(FCT) and POSI and partially financed by FEDER. Corresponding author: Luís Alexandre.

We tested the error bound on 6 publically available data sets corresponding to real classification problems, with distinct characteristics (number of features, number of patterns and number of classes). These experiments showed that the derived bound was not useful because it was quite loose. This was due to the original expression for the average generalization error (AvGE) in [1] and not to the derivation of the bound for this particular architecture. Thus, we found empirically a modified expression for the AvGE of the isolated MLP, that when used to derive the AvGE of the ME architecture, produces much tighter bounds.

The paper is organized as follows: the next section describes the ME MNN, section 3 introduces the average error bound for a single NN. In section 4, the AvGE error bound for the ME is derived. Three experiments are presented in section 5, with a discussion of their results. The final section presents the conclusions.

2 The Mixture of Experts MNN

The idea behind the MNN is the divide-and-conquer paradigm: the problem should be divided into smaller subproblems that are solved by experts and their partial solutions should be integrated to produce a final solution.

To use an MNN, three stages have to be considered: first, the task decomposition where the problem is divided into smaller problems, each one to be given to one of the modules or expert networks. Then each individual expert is trained until it learns to solve its particular subproblem. Finally, a decision integration strategy is used to combine the decisions of the experts to produce a final network output.

The decision integration can be obtained through different approaches: using a gating network [5], making the modules vote [6] or through hierarchical integration (which can also use voting and/or gating networks) [7, 8].

In this paper we consider the use of a gating network. The gate is trained to learn which region of the input space should be classified by which expert.

During the test phase, when the gate receives an input pattern it decides which is the expert that should classify the pattern and selects one of the experts to produce the final classification. This type of MNN is called a Mixture of Experts.

3 Average Error Bound

3.1 Problem Definition

The bounds discussed in this paper apply to the generalization error of a learning machine, and in particular, to an MLP and its generalization, an MNN.

We now discuss the general learning problem and define explicitly the generalization error and the empirical error (the one measured in the section 4).

Consider a learning problem in which a learning machine is given a set of data $\{x_1, \dots, x_m\}$ and it is expected that, by adjusting a set of parameters, w, it can learn to associate the respective targets $\{y_1, \dots, y_m\}$ to each input.

Typically, for a classification problem, which is the type of problem we are addressing in this paper, $x_i \in X$ and $X \subseteq \mathbb{R}^n$ and $y_i \in Y$ and $Y = \{1, 2, \dots, L\}$, where L is the number of classes in the problem.

Let $Z = X \times Y$. Each $z_i = (x_i, y_i)$ is called a training sample. The joint distribution $P(x, y)$ is represented by $P(z)$.

For a particular choice of the parameters w, the learning machine produces a hypothesis. Each hypothesis is represented by $f(x, w)$. When the prediction $f(x_i, w)$ is different from the respective target y_i, a loss occurs. This loss is measured by the loss function $l(y, f(x, w))$.

The expected value of the loss is called generalization error and is given by

$$R(w) = \int_Z l(y, f(x, w)) dP(z) \tag{1}$$

For the majority of real problems, $R(w)$ is not zero. Usually it is not possible to find $R(w)$ since the distribution $P(z)$ is not known. Instead, the empirical error is found,

$$E_m(w) = \frac{1}{m} \sum_{i=1}^{m} l(y_i, f(x_i, w)) \tag{2}$$

This is an estimate of $R(w)$.

Usually the data set is divided into two disjoint sets, one is used for training, or the adjustment of the weights w, and is called the training set. The other is used to estimate $R(w)$ and is called a test set. The empirical error measured in the training set is usually called the training set error.

In this paper we are concerned with an average error bound for $R(w)$. We call it the Average Generalization Error (AvGE).

3.2 AvGE for the MLP

The AvGE for an MLP was introduced in [1]. It has the form of

$$AvGE_{MLP} \leq \alpha + \frac{1}{2}\sqrt{\frac{d}{m}} \tag{3}$$

where α is the training set error, d represents the number of weights and m is the number of training samples. Note that it is an upper bound for the generalization error based on the training set error and a penalty for the number of weights that is reduced by the number of training samples.

3.3 AvGE for the ME MNN

To produce an AvGE for the ME MNN we first introduce the complement of the AvGE, the AvGC, which we define as

$$AvGC = 1 - AvGE \tag{4}$$

It is the 'Average Generalization Correctness'.

Using this definition it is possible to write, assuming independency between the errors made by the gate and by the experts,

$$AvGC_{ME} = AvGC_g \sum_{i=1}^{N} P(mod_i) AvGC_i \tag{5}$$

where $AvGC_g$ is the $AvGC$ of the gate, $P(mod_i)$ is the probability that the module i is chosen by the gate and $AvGC_i$ is the $AvGC$ of module i.

The assumption of the independency of the errors will not be true only in points on the frontiers between distinct regions of expertize of different experts.

To obtain the AvGE for the ME, we replace in expression (5) the relation in expression (4) for the gate and the individual modules, we get

$$AvGE_{ME} = 1 - AvGC_{ME} = 1 - \left[(1 - AvGE_g) \sum_{i=1}^{N} P(mod_i)(1 - AvGE_i) \right] \tag{6}$$

After some simple manipulation, we get

$$AvGE_{ME} = \beta + AvGE_g(1 - \beta) \tag{7}$$

with

$$\beta = \sum_{i=1}^{N} P(mod_i) AvGE_i \tag{8}$$

To obtain the upper bound for the $AvGE_{ME}$ we have to find a lower bound for β. A trivial lower bound is zero, but we can assume that the generalization error will be higher than the training set error, and thus β has the following upper and lower bounds

$$\sum_{i=1}^{N} P(mod_i)\alpha_i \le \beta \le \sum_{i=1}^{N} P(mod_i) \left(\alpha_i + \frac{1}{2}\sqrt{\frac{d_i}{m_i}} \right) \tag{9}$$

The upper bound is obtained directly from expression (8) by simply replacing $AvGE_i$ by expression (3), with each variable using a subscript i that corresponds to the expert i.

Now it is possible to write the final expression for the $AvGE_{ME}$

$$\begin{aligned} AvGE_{ME} \le & \sum_{i=1}^{N} P(mod_i) \left(\alpha_i + \frac{1}{2}\sqrt{\frac{d_i}{m_i}} \right) \\ & + \left(\alpha_g + \frac{1}{2}\sqrt{\frac{d_g}{m_g}} \right) \left(1 - \sum_{i=1}^{N} P(mod_i)\alpha_i \right) \end{aligned} \tag{10}$$

We will now use this bound to predict the generalization error of the ME MNN. Note that the parameters d_i, d_g, m_i and m_g are set a priori. The $P(mod_i)$ can be estimated from the proportion of points in the training set that belong to the region assigned for expert i. The α_i (the training set errors) are estimated and the number of modules used, N, is defined by the task decomposition algorithm or set a priori.

4 Experiments

4.1 Introduction

In these experiments we intend to show how the expression (10) can be used to bound the generalization error.

We use six data sets that illustrate different conditions: the number of classes ranges from 2 to 10; the number of data points ranges from 106 to 2126; the number of features ranges from 2 to 60.

Table 1 contains information about these data sets. They are all publically available, and the table shows references to their sources. It also shows the name, number of data points, number of features and the number of classes.

Table 1. The data sets used in the experiments.

Data set	Name	N. points	N. features	N. classes	Source
1	Breast cancer	106	9	6	[9]
2	Cleveland	296	13	5	[10]
3	CTG	2126	22	10	[9]
4	Diabetes	768	8	2	[10]
5	Speech recognition	608	2	4	[5]
6	Sonar	208	60	2	[10]

4.2 Classification

As we made the experiments, we noticed that the original error bound in expression (3) was quite loose. This behavior was also observed by Gu and Takahashi in [11] when they applied another average error bound to MLPs.

We empirically found that if instead of using $d = (number\ of\ weights)$ we used $d = (number\ of\ weights)/18$, the error bound becomes much tighter. A justification for this replacement is that in fact, although d was considered the number of weights for an MLP in [1], it is more recently considered to be an upper bound on the number of adjustable weights of the MLP (Lemma 1 in [11]). Hence, it can be replaced by a smaller value. Of course, a theoretical guideline for the value of d to use is desirable.

In what follows we present measures of the two versions of the error bound for the ME: $AvGE_{ME}$ and $AvGE_{ME2}$. They both correspond to expression (10), but the former uses $d = (number\ of\ weights)$ and the later uses $d = (number\ of\ weights)/18$.

The experiments were made with the holdout method: half the data set was used for training and the other half for testing. Then the data sets were used with inverted roles (the original training set became the test set and the original test set became the training set) and the error and error bounds were averaged between the two repetitions.

The task decomposition was made with the fuzzy c-means algorithm [12]. The experts and the gate were one hidden-layer multi-layer perceptrons (MLPs) trained with resilient backpropagation [13], for 100 epochs. The number of hidden layer neurons used for each expert and for the gate are listed in tables 2 and 3. These numbers were obtained empirically and gave acceptable classification errors for all data sets. We were

Table 2. The topology of the experts and gate along with the error and error bounds with standard deviations when using two experts.

Data set	Topology	$AvGE_{ME}$ (std)	$AvGE_{ME^{\bullet}}$ (std)	Error (std)
1	10,10,4	163.0 (2.3)	40.6 (3.6)	39.6 (10.7)
2	5,5,2	89.6 (4.7)	29.6 (5.9)	47.6 (0.5)
3	21,21,4	76.1 (0.2)	21.4 (0.1)	19.3 (1.1)
4	1,1,6	50.2 (0.2)	30.2 (0.2)	27.9(0.4)
5	2,2,2	29.2 (0.6)	11.3 (0.7)	9.6 (1.9)
6	2,2,6	174.0 (0)	41.0 (0)	20.2 (1.6)

Table 3. The topology of the experts and gate along with the error and error bounds with standard deviations when using three experts.

Data set	Topology	$AvGE_{ME}$ (std)	$AvGE_{ME^{\bullet}}$ (std)	Error (std)
1	10,10,10,4	186.3 (3.4)	48.2 (1.2)	32.1 (8.0)
2	5,5,5,2	108.9 (2.3)	33.2 (0.2)	45.3 (3.8)
3	21,21,21,4	88.1 (0.9)	24.1 (1.1)	20.0 (1.0)
4	1,1,1,6	51.1 (0.6)	25.2 (0.7)	23.1 (2.8)
5	2,2,2,2	32.9 (0.2)	11.9 (0.2)	7.9 (0.9)
6	2,2,2,6	191.1 (1.0)	45.0 (0.2)	17.3 (6.8)

not concerned in finding the optimal value of the number of neurons, we intended only to illustrate the behavior of the error bound derived in section 3.2.

Table 2 shows the values obtained in the experiments using two experts in the mixture of experts. The second column (topology) gives the number of hidden layer neurons used with each expert and gate, respectively. For instance, 18,18,2 means that both the first expert and the second had 18 neurons in the hidden layer and that the gate used 2 neurons in the hidden-layer.

Table 3 shows the values obtained in the experiments using three experts in the mixture of experts.

4.3 Discussion

The Errors. The true errors are smaller when the experiment is done with three experts than when it is done with only two. This is not surprising since in the first case the number of weights used in the MNN is larger and it can adjust itself better to the data. There is one exception, that is data set 3. This indicates that for this particular problem a division in three sub-problems does not improve its learnability when compared to a division in only two subproblems.

The Bounds. In all cases, the $AvGE_{ME2}$ is tighter than $AvGE_{ME}$, indicating that d should not be set to the number of weights but to a smaller value. This is due to the fact that the $AvGE_{ME2}$ gives a smaller penalty for the increase in the number of adjustable parameters in the model (the network weights) than $AvGE_{ME}$.

In all cases, both the $AvGE_{ME}$ and the $AvGE_{ME2}$ increase their values when going from an MNN with two experts to one with three experts. This is due to an increase in the total number of weights for the MNNs with 3 experts when compared to the MNN used for the same problem but with only 2 experts.

The bound $AvGE_{ME}$ is always larger than the measured error. But for the bound $AvGE_{ME2}$ this does not happen with data set 2. Both with the 2 expert MNN and with the 3 expert MNN, the bound $AvGE_{ME2}$ is considerably smaller than the measured error. The bounds have an important component that is the measured training error. We checked the training and test set errors for an isolated MLP for this data set and there is a big difference between their values. Training set error is about 15% while the test set error is over 40%. We think this explains the poor performance of the bound for this particular case. The reason for the difference between the training and test errors might be due to a training overfitting.

The predictions of the $AvGE_{ME2}$ bound are very close to the measured errors. The exceptions are data set 3, which was already mentioned, and data set 6. This data set represents a problem in a high dimensional space (60-dimensional). This makes the number of weights of an MLP vary a great deal with the introduction of a single neuron in the hidden layer (each new hidden layer neuron adds 61+(number of classes) weights to the MLP). So it is easy to go from one situation where the number of weights is insufficient to learn the problem to one where the network overfits the problem. We argue that the difference between the bound and the errors measured for this data set is due to the possibly larger than necessary number of weights used in the network.

5 Conclusions

In this paper we developed an average error bound for the mixture of experts MNN. It is an extension of the bound presented in [1] for an isolated NN. It also generalizes the bound in [4] since there are no restrictions on the data sets or on the type of gate used. This way it can be applied to any classification problem.

We also propose an empirical correction factor to the original expression in [1] that produces much tighter bounds. A theoretical justification for the particular value of this adjustment is still lacking.

These bounds can be used to estimate the generalization error of the ME MNN, as illustrated in the experiments section and they behave well for most of the tests made.

References

1. Gu, H., Takahashi, H.: Towards more practical average bounds on supervised learning. IEEE Trans. Neural Networks **7** (1996) 953–968
2. Vapnik, V.: The Nature of Statistical Learning Theory. Springer (1999)
3. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edition. Prentice Hall (1999)
4. Alexandre, L., Campilho, A., Kamel, M.: Average error bound for the mixture of experts MNN architecture. In: Proceedings of the 12th Portuguese Conference on Pattern Recognition, Aveiro, Portugal (2002)

5. Jacobs, R., Jordan, M., Nowlan, S., Hinton, G.: Adaptive mixtures of local experts. Neural Computation (1991) 79–87
6. Auda, G., Kamel, M.: Modular neural network classifiers: A comparative study. J. Intel. Robotic Systems (1998) 117–129
7. Jordan, M., Jacobs, R.: Hierarchical mixture of experts and the EM algorithm. Neural Computation (1994) 181–214
8. Jacobs, R., Peng, F., Tanner, M.: A bayesian approach to model selection in hierarchical mixtures-of-experts architectures. Neural Networks **10** (1997) 231–241
9. Marques de Sá, J.: Pattern Recognition: Concepts, Methods and Applications. Springer (2001)
10. Blake, C., Keogh, E., Merz, C.: UCI repository of machine learning databases (1998) http://www.ics.uci.edu/˜mlearn/MLRepository.html.
11. Gu, H., Takahashi, H.: Estimating learning curves of concept learning. Neural Networks **10** (1997) 1089–1102
12. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
13. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: IEEE International Conference on Neural Networks, San Francisco (1993) 586–591

An Effective EM Algorithm for PCA Mixture Model

Zhong Jin[1,3], Franck Davoine[2], and Zhen Lou[3]

[1] Centre de Visio per Computador
Universitat Autonoma de Barcelona
08193 Barcelona, Spain
zhong.jin@cvc.uab.es

[2] Heudiasyc - CNRS Mixed Research Unit
Compiegne University of Technology
60205 Compiegne cedex, France
franck.davoine@hds.utc.fr

[3] Department of Computer Science
Nanjing University of Science and Technology
Nanjing, People's Republic of China

Abstract. This paper studied PCA mixture model in high dimensional space. A novel EM learning approach by using perturbation was proposed for the PCA mixture model. Experiments showed the novel perturbation EM algorithm is more effective in learning PCA mixture model than an existing constrained EM algorithm.

1 Introduction

In recent years, there has been increasing interest in PCA mixture model. Mixture model provides a simple framework for modelling data complexity by a weighted combination of component distributions [1, 2]. It has been widely used in machine learning, image processing, and data mining due to its great flexibility and power. However, since the component distributions in mixture model are usually formalized to be probability density functions, there are limited applications to practical problems in high dimensional space.

As a variation of mixture model, PCA mixture model is proposed to use principal component analysis (PCA) to express component distributions. The idea of PCA mixture model is motivated by a mixture-of-experts technique that models a non-linear distribution by a combination of local linear sub-models, each with a relatively simple distribution [3, 4]. Hinton et al. [5] proposed a PCA mixture model based on the reconstruction error. Tipping and Bishop defined a mixture model for probabilistic principal component analyzers (PPCA), whose parameters can be determined using an EM algorithm [6]. Recently, mixtures of probabilistic principal component analyzers was used to model data that lies on or near a low dimensional manifold in a high dimensional observation space [7]. Kim et al. discussed the problem to select model order for PCA mixture model [8].

Expectation-maximization (EM) algorithm is a powerful algorithms for maximum likelihood (ML) or maximum a posterior (MAP) estimation in problems with incomplete data, e.g., fitting a mixture model to observed data [9, 10]. The EM algorithm provides iterative formulae for the estimation of the unknown parameters of the mixture. The drawbacks of EM algorithm for a problem in high dimensional space is that the mixing components are assumed to have probability density functions in the data space. In practical problems, the mixing components may only have probability density functions in low dimensional manifolds.

In this paper, the EM algorithm for PCA Gaussian mixture model is studied. It is organized as follows. Section 2 gives an introduction to the Gaussian mixture model. Section 3 proposes a new EM algorithm for PCA mixture model. Experiments are performed in Section 4. Finally, conclusions are drawn in Section 5.

2 The EM Algorithm for Gaussian Mixtures

A Gaussian mixture is defined as a combination of Gaussian densities. A Gaussian density in a d-dimensional space, parameterized by its mean $\mathbf{m} \in \Re^{\mathbf{d}}$ and $d \times d$ covariance matrix C, is defined by the density:

$$\phi(x;\theta) = \frac{1}{(2\pi)^{\frac{d}{2}}|C|^{\frac{1}{2}}} exp\{-\frac{(x-m)^t C^{-1}(x-m)}{2}\}, \tag{1}$$

where $\theta = (m, C)$.

A mixture of k Gaussian densities is then defined as:

$$f_k(x) = \sum_{j=1}^{k} \pi_j \phi(x;\theta_j), \tag{2}$$

with

$$\sum_{j=1}^{k} \pi_j = 1, \tag{3}$$

where $\theta_j = (m_j, C_j)$ and $\pi_j \geq 0$, for $j = 1, 2, \cdots, k$. The π_j are called the mixing weights and $\phi(x;\theta_j)$ the components of the mixture.

A training set $X_n = \{x_1, x_2, \ldots, x_n\}$ of independent and identically distributed points $x_i \in \Re^d$ is assumed to be sampled from Eq. (2). The task is to estimate the parameters of the mixture that maximize the log-likelihood

$$\mathbf{L} = \frac{1}{n}\sum_{i=1}^{n} \log f_k(x_i). \tag{4}$$

The Expectation-Maximization (EM) algorithm is a well-known statistical tool for maximum likelihood problems involving hidden or unobserved variables [9]. In the case of mixtures, the unobservable variable can be regarded as the component from which each input point has been sampled. The EM algorithm enables us to update the parameters of a given k-component mixture with respect to X_n such that the log-likelihood of X_n is never smaller under the new mixture.

Let

$$r_{ji} = P(j|x_i) \tag{5}$$

be the posterior probability that the point x_i is sampled from the ^{j}th mixing component. The EM iteration consists of one expectation step (E-step) and one maximization step (M-step) as follows.

- **E-step**
  By the Bayes rule, the expectation values of r_{ji} can be given from the mixture model of Eq. (2) from the previous iteration:

$$r_{ji} = \frac{\pi_j \phi(x_i; \theta_j)}{f_k(x_i)}, \tag{6}$$

  where $\theta_j = (m_j, C_j)$.
- **M-step**
  The component parameters can be estimated from samples and the expectation values of r_{ji} of Eq. (6):

$$\pi_j = \frac{1}{n} \sum_{i=1}^{n} r_{ji}, \tag{7}$$

$$m_j = \frac{1}{n\pi_j} \sum_{i=1}^{n} r_{ji} x_i, \tag{8}$$

$$C_j = \frac{1}{n\pi_j} \sum_{i=1}^{n} r_{ji}(x_i - m_j)(x_i - m_j)^t. \tag{9}$$

3 EM Algorithm for PCA Mixture Model

In this section, we will discuss the limitation of EM in high dimensional space firstly and then propose a novel EM algorithm for PCA mixture model.

3.1 Limitation of EM

In a high dimensional space, some mixing components may lies on or near a low dimensional manifold. In other words, for some mixing components, the covariance matrices C may be singular in high dimensional space so that there are not Gaussian densities of Eq. (1). In this case, Eq. (1), then E-step of Eq. (6), can not be conducted since there are not inverse matrices for such mixing components which are degenerate in a low dimensional manifold.

3.2 PCA Mixture Model

One solution to the above limitation of EM is to describe Gaussian density of Eq. (1) by using principal component analysis (PCA).

Let $\Psi = (\psi_1, \cdots, \psi_d)$ be the matrix whose columns are the unit-norm eigenvectors of the covariance matrix C of Eq. (1). Let $\Lambda = diag(\lambda_1, \cdots, \lambda_d)$ be the diagonal matrix of the eigenvalues of C, where λ_i are the eigenvalues corresponding to the eigenvectors ψ_i $(i = 1, \cdots, d)$. We have

$$\Psi^t C \Psi = \Lambda. \tag{10}$$

Let

$$y = \Psi^t (x - m), \tag{11}$$

where $y = [y(1), y(2), \cdots, y(d)]'$ and $y(i)$ is the ^{i}th principal component of the d-dimensional vector y.

Assume that λ_i $(i = 1, \cdots, d)$ are ranked in order from larger to smaller as follows:

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0. \tag{12}$$

The covariance matrix C can be any covariance matrix of the k mixing components and the mean m can be any mean of the k mixing components. Let C be C_j, the covariance matrix of the ^{j}th mixing component. If C_j is non-singular, we will have $\lambda_d > 0$. Then, the Gaussian density of Eq. (1) in x space can be expressed in y space as follows:

$$\phi(y; \theta_j) = \prod_{i=1}^{d} \frac{1}{(2\pi\lambda_i)^{\frac{1}{2}}} exp\{-\frac{[y(i)]^2}{2\lambda_i}\}. \tag{13}$$

Note that y not only depends on x, but also depends on $\theta_j = (m_j, C_j)$.

If C_j is singular, some last λ's will have a value of zero. This means that the ^{j}th mixing component is distributed in a low dimensional manifold in original x space. So, there doesn't exist a Gaussian density of Eq. (13) in x space for the mixing component.

Assume that

$$\lambda_1 \geq \cdots \geq \lambda_{d_j} > \lambda_{d_{j+1}} = \cdots = \lambda_d = 0, \tag{14}$$

where d_j is the number of non-zero eigen-values of the covariance matrix C_j of the ^{j}th mixing component. For the ^{j}th mixing component, there exists a Gaussian density in a low dimensional $[y(1), y(2), \cdots, y(d_j)]$ space as follows:

$$\phi(y; \theta_j) \approx \prod_{i=1}^{d_j} \frac{1}{(2\pi\lambda_i)^{\frac{1}{2}}} exp\{-\frac{[y(i)]^2}{2\lambda_i}\}. \tag{15}$$

The $d_j (j = 1, \cdots, k)$ in Eq. (15) can be called as an order of principal components for the ^{j}th mixing component. If d_j is equal to d, i.e., $d_j = d$, Eq. (15) is the same as Eq. (13). In other word, Eq. (13) is a special case of Eq. (15).

What is a PCA mixture model? Now, we can give a definition. We define the mixture model of Eqs. (2), (3) and (15) as a PCA mixture model, where $d_j (j = 1, \cdots, k)$, the orders of principal components for the mixing components, can be determined by Eq. (14) or by any other assumptions. This definition can be regarded as a generalization of the PCA mixture model used in [8], where $d_j (j = 1, \cdots, k)$ is assumed to be the same value.

Note that the $f_k(x)$ in the PCA mixture model is not a probability density function now. It is just a structure description of the data set in high dimensional space.

3.3 A Perturbation Approach

Recently, Kim et al. discussed the problem to select model order for PCA mixture model [8], and proposed a constraint with $d_j (j = 1, \cdots, k)$ as follows:

$$d_1 = d_2 = \cdots = d_k. \tag{16}$$

Under the constraint of Eq. (16), $d_j (j = 1, \cdots, k)$ have to be the smallest of all the orders of principal components for the mixing components. This assumption increases the difficulty in learning the PCA mixture model.

In general, when $d_j < d$, no matter whether $d_j (j = 1, \cdots, k)$ are equal to each other, it is difficult to estimate r_{ji} in classical E-step by using Eq. (15) directly. The reason is that Eq. (15) may be a probability density in a different space for the different $j (j = 1, \ldots, k)$, even under the constraint of Eq. (16).

Our idea is to assign a very small positive value ε to the last eigenvalues $\lambda_{d_{j+1}}, \cdots, \lambda_d$ in Eq. (14). So, we have

$$\lambda_1 \geq \cdots \geq \lambda_{d_j} > \lambda_{d_{j+1}} = \cdots = \lambda_d = \varepsilon > 0, \tag{17}$$

where ε can be called a perturbation factor.

By introducing a perturbation factor in Eq. (17), we make the covariance matrix C_j of the ^{j}th mixing component be non-singular. Therefore, Eq. (13) can be directly used to estimate the raw values of r_{ji} of Eq. (5) as follows:

$$r_{ji} = \pi_j \phi(y; \theta_j), \tag{18}$$

where according to Eq. (11), $y = y_i$ is obtained from the point x_i instead of x, and $\theta_j = (m_j, C_j)$ instead of $\theta = (m, C)$.

Note that all the raw values $r_{ji} (j = 1, \cdots, k)$ will further be normalized by

$$\sum_{j=1}^{k} r_{ji} = 1. \tag{19}$$

So, a novel E-step by using perturbation can be introduced as follows:

- **Perturbation E-Step**

  ◇ For $C = C_j$, the covariance matrix of the ^{j}th mixing component, make a PCA decomposition of Eq. (10) , and obtain all the d eigenvalues λ_i $(i = 1, \cdots, d)$ and corresponding eigenvectors ψ_i $(i = 1, \cdots, d)$.
  ◇ According to Eq. (17), determine d_j and assign ε to the last $d - d_j$ eigenvalues $\lambda_{d_{j+1}}, \cdots, \lambda_d$.
  ◇ For each point $x_i (i = 1, \cdots, n)$, compute the projection of x_i on all the d principal components according to Eq. (11).
  ◇ Update the expectation values of r_{ji} of Eq. (5) according to Eqs. (13,17, 18,19).

Our novel EM algorithm by using perturbation for PCA mixture model includes the novel perturbation E-step and the classical M-step of Eqs. (7,8,9).

4 Experiments

In this section, some experiments are performed to see if the novel EM algorithm by using perturbation is effective to learn both non-degenerate mixing component and degenerate mixing component.

4.1 Synthetic Data

Although PCA mixture model is motivated by difficulties in the estimating of distributions in high dimensional space, it is of some significance to have an initial evaluation on the learning algorithms with problems in low dimensional space.

For simplicity, a problem of only two mixing components in two dimensional space is considered: one with a degenerate density in one dimensional subspace, and the other with a non-degenerate density in two dimensional space. We have considered four examples as follows.

- **Example (a)** One mixing component has a degenerated density in the direction of $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$. The other has a covariance of $\begin{bmatrix} 1 & 0.6 \\ 0.6 & 1 \end{bmatrix}$. The first principal directions of these two mixing components are orthogonal to each other.
- **Example (b)** One mixing component has a degenerated density in the direction of $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$. The other has a covariance of $\begin{bmatrix} 1 & 0.6 \\ 0.6 & 3 \end{bmatrix}$. The first principal directions of these two mixing components are approximately orthogonal to each other.
- **Example (c)** One mixing component has a degenerated density in the direction of $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$. The other has a covariance of $\begin{bmatrix} 1 & -0.6 \\ -0.6 & 3 \end{bmatrix}$. The first principal directions of these two mixing components are approximately parallel to each other.

- **Example (d)** One mixing component has a degenerated density in the direction of $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$. The other has a covariance of $\begin{bmatrix} 1 & -0.6 \\ -0.6 & 1 \end{bmatrix}$. The first principal directions of these two mixing components are parallel to each other.

We draw 100 samples respectively for each mixing component. These four examples are displayed in Fig. 1 respectively.

4.2 Experimental Results and Analysis

For each example, we have performed experiments to learn PCA mixture model by our perturbation EM algorithm and the constrained EM algorithm used in [8] with a constraint of Eq. (16).

The experimental results with four examples by our perturbation EM algorithm are displayed in Fig. 1 (a1), (b1), (c1) and (d1) respectively. The experimental results with four examples by the constrained EM algorithm are displayed in Fig. 1 (a2), (b2), (c2) and (d2) respectively. In Fig. 1, the distribution centers are marked in green color for all the learned mixing components, and the probability ellipses in blue color are drawn to enclose about 90% samples for all the learned mixing components if without degeneration.

From Fig. 1, we can have the following facts and discussion:

- Our perturbation EM algorithm is effective to learn PCA mixture model. For each example, the novel perturbation EM algorithm is effective to learn both non-degenerate mixing component and degenerate mixing component.
- The constrained EM algorithm [8] is not as effective as the perturbation EM algorithm to learn PCA mixture model. It becomes more difficult for the constrained EM algorithm to learn the mixing components when the first principal directions of two mixing components become more parallel to each other.

5 Conclusion and Future Work

In this paper, we have studied PCA mixture model in high dimensional space and have given an analysis to the limitation of EM algorithms for PCA mixture model. A novel EM learning approach by using perturbation is proposed for PCA mixture model. Experiments with synthetic data show the novel perturbation EM algorithm is more effective to learn both non-degenerate mixing component and degenerate mixing component in PCA mixture model than an existing constrained EM algorithm.

In our future work, we are focusing on investigating some learning algorithms for PCA mixture model for practical problems in high dimensional space, such as face recognition and facial expression analysis. In these problems, only a small number of samples are available, and a singular decomposition theorem is used to obtain some eigen-vectors for non-zero eigenvalues. In other words, for practical application problems, the dimension d may be too high to obtain all the d eigenvectors in Eq. (10) and the transformation from x-space to y-space in Eq. (11) is not available.

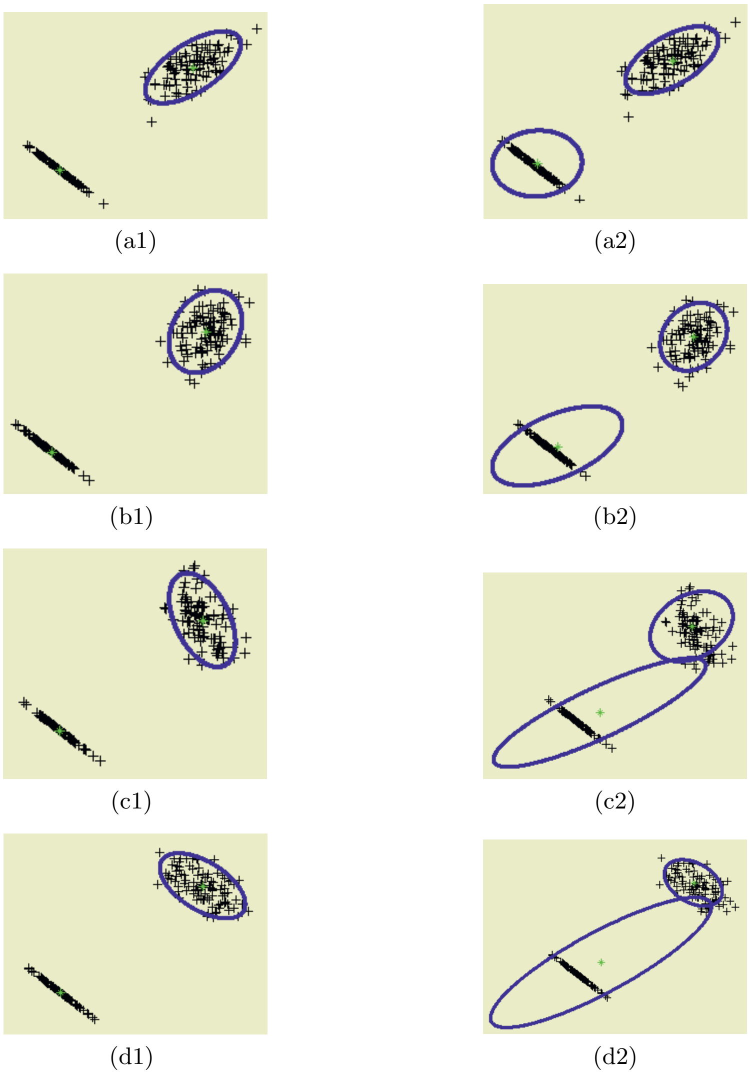


Fig. 1. Experiment results with four examples: (a1,b1,c1,d1) by our perturbation EM algorithm, and (a2,b2,c2,d2) by the constrained EM algorithm [8].

References

1. D. M. Titterington, A. F. M. Smith, and U. E. Makov. *Statistical analysis of finite mixture distribution*. Wiley, New York, 1985.
2. G. McLachlan and D. Peel. *Fixed mixture models*. Wiley, New York, 2000.
3. R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

4. M. Jordan and R. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6(5):181–214, 1994.
5. G. Hinton, P. Dayan, and M. Revow. Modeling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Network*, 8(1):65–74, 1997.
6. M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.
7. J.J. Verbeek, N. Vlassis, and B. Kröse. Coordinating mixtures of probabilistic principal component analyzers. Technical report, Computer Science Institute, University of Amsterdam, The Netherlands, Feb 2002. IAS-UVA-02-01.
8. Hyun-Chul Kim, Daijin kim, and Sung Yang Bang. An efficient model order selection for PCA mixture model. *Pattern Recognition Letters*, 24(9-10):1385–1393, 2003.
9. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1–38, 1977.
10. R. J. A. Little and D. B. Rubin. *Statistical analysis with missing data.* Wiley, New York, 1987.

EM Initialisation for Bernoulli Mixture Learning*

Alfons Juan, José García-Hernández, and Enrique Vidal

DSIC, Universitat Politècnica de València, 46071 València, Spain
{ajuan,jogarcia,evidal}@dsic.upv.es

Abstract. Mixture modelling is a hot area in pattern recognition. This paper focuses on the use of Bernoulli mixtures for binary data and, in particular, for binary images. More specifically, six EM initialisation techniques are described and empirically compared on a classification task of handwritten Indian digits. Somehow surprisingly, we have found that a relatively good initialisation for Bernoulli prototypes is to use slightly perturbed versions of the hypercube centre.

Keywords: Mixture Models, EM Algorithm, Multivariate Bernoulli Distribution, Initialisation Techniques, Binary Data, Indian Digits

1 Introduction

Mixture modelling is a popular approach for density estimation in both supervised and unsupervised pattern classification [7]. On the one hand, mixtures are flexible enough for finding an appropriate tradeoff between model complexity and the amount of training data available. Usually, model complexity is controlled by varying the number of mixture components while keeping the same (often simple) parametric form for all components. On the other hand, maximum likelihood estimation of mixture parameters can be reliably accomplished by the well-known *Expectation-Maximisation (EM)* algorithm.

Although most research in mixture modelling has focused on mixtures for continuous data, there are many pattern recognition tasks for which binary or discrete mixture models are better suited. This paper focuses on the use of *(multivariate) Bernoulli mixtures* for binary data and, in particular, for *binary images.* EM-based maximum likelihood estimation of Bernoulli mixtures is known even before the general statement of the EM algorithm in 1977 [3]. In fact, the basic formulae appear in a proposed problem of the classic 1973 book by Duda and Hart [4, pp. 256 and 257], who attribute to Wolfe their derivation in 1970 [4, p. 249]. In spite of being known for more than three decades, Bernoulli mixtures as such have seldom been assessed in practice. In [6], for instance, a more complex yet closely-related model is successfully tested on a conventional OCR task, but no comparative results are provided for the simpler, pure Bernoulli mixture

* Work supported by the Spanish "Ministerio de Ciencia y Tecnología" under grant DPI2001-0880-CO2-02.

model. It seems that this pure model has been only applied to non-conventional tasks such as unsupervised modelling of *electropalatographic data* [2].

During the past few years, we have found that Bernoulli mixtures are really effective in certain supervised text classification tasks [5, 9]. Moreover, since these tasks can be considered somewhat non-conventional, we have recently tried out Bernoulli mixtures on a more conventional pattern recognition task involving binary images [8]. As in the case of text classification, the results obtained are encouraging.

In view of their potential, we think that Bernoulli mixtures deserve more attention. In particular, as with any kind of mixtures, it is important to take care of *EM initialisation* in order to fine-tune Bernoulli mixture learning. This paper compares six initialisation techniques, which are described in section 4, after a review of the model and the basic theory on its EM-based maximum likelihood estimation (sections 2 and 3). Then, experimental results are reported on a classification task of handwritten Indian digits.

2 Bernoulli Mixtures

A (finite) mixture model consists of a number of *mixture components,* I. It generates a D-dimensional *sample* $\boldsymbol{x} = (x_1, \ldots, x_D)^t$ by first selecting the ith component with *prior probability* $p(i)$, and then generating $\boldsymbol{x}$ in accordance with the ith *component-conditional probability (density) function* $p(\boldsymbol{x} \mid i)$. The priors must satisfy the constraints:

$$\sum_{i=1}^{I} p(i) = 1 \quad \text{and} \qquad p(i) \geq 0 \quad (i = 1, \ldots, I). \tag{1}$$

The *posterior probability* of $\boldsymbol{x}$ being actually generated by the ith component can be calculated via the *Bayes' rule* as

$$p(i \mid \boldsymbol{x}) = \frac{p(i)\, p(\boldsymbol{x} \mid i)}{p(\boldsymbol{x})} \tag{2}$$

where

$$p(\boldsymbol{x}) = \sum_{i=1}^{I} p(i)\, p(\boldsymbol{x} \mid i) \tag{3}$$

is the *(unconditional) mixture probability (density) function.*

A Bernoulli mixture model is a particular case of (3) in which each component i has a D-dimensional Bernoulli probability function governed by its own vector of parameters or *prototype* $\boldsymbol{p}_i = (p_{i1}, \ldots, p_{iD})^t \in [0, 1]^D$,

$$p(\boldsymbol{x} \mid i) = \prod_{d=1}^{D} p_{id}^{x_d}\, (1 - p_{id})^{1 - x_d} \tag{4}$$

Consider an arbitrary component $p(\boldsymbol{x} \mid i)$. It identifies a certain *subclass* of binary vectors "resembling" its parameter vector or *prototype* $\boldsymbol{p}_i$. In fact, each p_{id} is the probability of bit x_d to be one, whereas $1 - p_{id}$ is the opposite.

Equation (4) is just the product of independent, unidimensional Bernoulli probability functions. Therefore, a single multivariate Bernoulli component can not capture any kind of dependencies or correlations between individual bits. As with other types of mixtures, this is implicitly done by mixing several components in the right proportions.

Also as with other types of mixtures, Bernoulli mixtures can be used as class-conditional models in supervised classification tasks. Let C denote the number of supervised classes. Assume that, for each supervised class c, we know its prior $p(c)$ and its class-conditional probability function $p(\boldsymbol{x} \mid c)$, which is a mixture of I_c Bernoulli components,

$$p(\boldsymbol{x} \mid c) = \sum_{i=1}^{I_c} p(i \mid c)\, p(\boldsymbol{x} \mid c, i) \tag{5}$$

Then, the optimal Bayes decision rule is to assign each pattern vector $\boldsymbol{x}$ to a class $c^*(\boldsymbol{x})$ giving maximum a posteriori probability:

$$\begin{aligned} c^*(\boldsymbol{x}) &= \arg\max_c p(c \mid \boldsymbol{x}) & (6)\\ &= \arg\max_c \left(p(c)\, p(\boldsymbol{x} \mid c) \right) & (7)\\ &= \arg\max_c \left(\log p(c) + \log p(\boldsymbol{x} \mid c) \right) & (8)\\ &= \arg\max_c \left(\log p(c) + \log \sum_{i=1}^{I_c} p(i \mid c) p(\boldsymbol{x} \mid c, i) \right) & (9) \end{aligned}$$

3 Maximum Likelihood Estimation

Let $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ be a set of samples available to learn a Bernoulli mixture model. This is a statistical parameter estimation problem since the mixture is a probability function of known functional form, and all that is unknown is a parameter vector including the priors and component prototypes:

$$\boldsymbol{\Theta} = (p(1), \ldots, p(I), \boldsymbol{p}_1, \ldots, \boldsymbol{p}_I)^t. \tag{10}$$

Here we are excluding the number of components from the estimation problem, as it is a crucial parameter for controlling model complexity and receives special attention in section 5.

Following the maximum likelihood principle, the best parameter values maximise the log-likelihood function of $\boldsymbol{\Theta}$,

$$\mathcal{L}(\boldsymbol{\Theta} \mid X) = \sum_{n=1}^{N} \log \left(\sum_{i=1}^{I} p(i)\, p(\boldsymbol{x}_n \mid i) \right). \tag{11}$$

In order to find these optimal values, it is useful to think of each sample $\boldsymbol{x}_n$ as an *incomplete* component-labelled sample, which can be completed by an indicator vector $\boldsymbol{z}_n = (z_{n1}, \dots, z_{nI})^t$ with 1 in the position corresponding to the component generating $\boldsymbol{x}_n$ and zeros elsewhere. In doing so, a complete version of the log-likelihood function (11) can be stated as

$$\mathcal{L}_C(\boldsymbol{\Theta}|X, Z) = \sum_{n=1}^{N} \sum_{i=1}^{I} z_{ni} \left(\log p(i) + \log p(\boldsymbol{x}_n|i) \right), \tag{12}$$

where $Z = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_N\}$ is the so-called *missing* data.

The form of the log-likelihood function given in (12) is generally preferred because it makes available the well-known *EM* optimisation algorithm (for finite mixtures) [3]. This algorithm proceeds iteratively in two steps. The E(xpectation) step computes the expected value of the missing data given the incomplete data and the current parameters. The M(aximisation) step finds the parameter values which maximise (12), on the basis of the missing data estimated in the E step. In our case, the E step replaces each z_{ni} by the posterior probability of $\boldsymbol{x}_n$ being actually generated by the ith component,

$$z_{ni} = \frac{p(i)\, p(\boldsymbol{x}_n \mid i)}{\sum_{i'=1}^{I} p(i')\, p(\boldsymbol{x}_n \mid i')} \qquad \begin{pmatrix} n = 1, \dots, N \\ i = 1, \dots, I \end{pmatrix}, \tag{13}$$

while the M step finds the maximum likelihood estimates for the priors,

$$p(i) = \frac{1}{N} \sum_{n=1}^{N} z_{ni} \qquad (i = 1, \dots, I), \tag{14}$$

and the component prototypes,

$$\boldsymbol{p}_i = \frac{1}{\sum_{n=1}^{N} z_{ni}} \sum_{n=1}^{N} z_{ni} \boldsymbol{x}_n \qquad (i = 1, \dots, I). \tag{15}$$

To start the EM algorithm, initial values for the parameters are required. To do this, it is recommended to avoid "pathological" points in the parameter space such as those touching parameter boundaries and those in which the same prototype is used for all components [2]. Provided that a non-pathological starting point is used, each iteration is guaranteed not to decrease the log-likelihood function and the algorithm is guaranteed to converge to a proper stationary point (local maximum). Also, for the sake of robustness, it is important to introduce some sort of model smoothing.

4 Initialisation Techniques

As said above, the only condition for proper EM initialisation is to avoid "pathological" points in the parameter space. Unfortunately, this does not say too much about the actual technique we should use. So, to clarify ideas, let us first distinguish between mixture proportions and Bernoulli prototypes. Clearly, a natural choice for the initialisation of mixture proportions is to be as impartial as possible, that is, to set them all to the same value:

$$p(i) = \frac{1}{I} \qquad (i = 1, \ldots, I) \tag{16}$$

The tricky problem is to devise an adequate initialisation technique for Bernoulli prototypes. In this case, the simplest yet natural option is to randomly draw each prototype from the open unit hypercube:

$$\boldsymbol{p}_i^{\text{rand}} = rand\{\boldsymbol{x} \in [\epsilon, 1-\epsilon]^D\} \qquad (i = 1, \ldots, I) \tag{17}$$

where ϵ $(0 < \epsilon \leq 0.5)$ is a positive constant intended to exclude extreme probability values. This *random* initialisation was used in [2]. Generally speaking, each possible non-pathological prototype has the same chance of being chosen.

A possible drawback of initialisation (17) comes from the fact that almost all potential prototypes have nothing in common with the training data. Therefore, it is almost sure that it will pick a poor starting point in terms of likelihood. A possible remedy to this drawback is to restrict the set of potential prototypes to the training data,

$$\boldsymbol{p}_i^{\text{proto}} = \boldsymbol{x}_{rand\{1,\ldots,N\}} \qquad (i = 1, \ldots, I) \tag{18}$$

but these prototypes are completely pathological (made up of zeros and ones). A straightforward solution to this pathological nature is to linearly combine (17) and (18):

$$\boldsymbol{p}_i^{\text{rproto}} = \alpha\, \boldsymbol{p}_i^{\text{rand}} + (1-\alpha)\, \boldsymbol{p}_i^{\text{proto}} \qquad (i = 1, \ldots, I) \tag{19}$$

where α $(0 \leq \alpha < 1)$ measures the "global randomness" of $\boldsymbol{p}_i^{\text{rproto}}$, as opposed to $1-\alpha$, which measures its "closeness" to the training data. We call this technique *random prototypes.* We used it in [5] $(\alpha \approx 0)$ and [8] $(\alpha = 0.75)$.

Although (19) will usually provide acceptable initialisations in terms of likelihood, it may be improved in some cases, especially when initialising a mixture of many prototypes. In this case, it is hardly likely that the prototypes chosen will uniformly cover all regions in which training bit vectors appear. On the contrary, it is more likely that some of these regions will become "overpopulated" by prototypes, while other regions will not be covered enough. This eventual failure can be easily prevented by considering prototypes as "facilities" to be located in a "maximally dispersed" way. More specifically, the following algorithm does the job:

$$\boldsymbol{p}_i^{\text{maxmin}} = \begin{cases} \boldsymbol{x}_{rand\{1,\ldots,N\}} & \text{if } i = 1 \\ \displaystyle\arg\max_{\boldsymbol{x} \in X} \min_{i'=1,\ldots,i-1} d(\boldsymbol{x}, \boldsymbol{p}_{i'}^{\text{maxmin}}) & \text{if } i > 1 \end{cases} \tag{20}$$

where $d(\cdot,\cdot)$ is an appropriate distance function for bit vectors (e.g. the Hamming distance). The basic idea behind (20) is simple: the ith prototype chosen is the training bit vector which is farthest away from its closest prototype among those

($i-1$ prototypes) previously chosen. As with (18), some sort of randomisation or smoothing is also required to avoid exact zeros and ones. We used this algorithm in [9], where it was called *maxmin* initialisation.

The three initialisation techniques discussed so far (random, random prototypes and maxmin) have been used in previous works but they have not been yet compared on the same basis. Since this work is a good opportunity to consider any reasonable initialisation heuristic, we have also considered three additional techniques that can be interpreted as minor variations on the same idea: to use the same vector for all mixture prototypes. The rationale behind this idea is that we have to be as neutral as possible during EM initialisation and then rely on the EM itself for the purpose of specialising each prototype in a different data subclass. Of course, each prototype has to be (randomly) perturbed since the use of the very same vector in all components leads to a well-known pathological starting point [2]. The three minor variations we are talking about are:

1. *Hypercube centre:* all prototypes are slightly perturbed versions of $\mathbf{0.5}$.
2. *Data mean:* all prototypes are slightly perturbed versions of the data mean.
3. *Class mean:* all prototypes of the mixture for class c are perturbed versions of the class c data mean.

5 Experiments

The experiments reported in this section correspond to an OCR task consisting in the recognition of handwritten Indian digits. They were designed to compare, on the same basis, the six initialisation techniques described in the previous section. Also, they can be considered as a continuation of the experiments reported in [8].

The dataset used here comprises the 10425 digit samples included in the non-touching part of the *Indian digits database* recently provided by CENPARMI [1]. Original digit samples are given as binary images of different sizes (minimal bounding boxes). To obtain properly normalised images, both in size and position, two simple preprocessing steps were applied. First, each digit image was pasted onto a square background whose centre was aligned with the digit centre of mass. This square background was a white image large enough (64×64) to accommodate most samples though, in some cases, larger background images were required. Second, given a size S, each digit image was subsampled into $S \times S$ pixels, from which its corresponding binary vector of dimension $D = S^2$ was built. Figure 1 shows one preprocessed example of each Indian digit ($S = 30$).

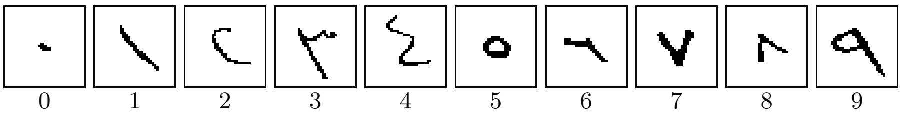


Fig. 1. 30×30 examples of each Indian digit.

The standard experimental procedure for classification error rate estimation in the CENPARMI Indian digits task is a simple partition with 7390 samples for training and 3035 for testing (excluding the extra classes *delimiter* and *comma*). Using this procedure, we obtained the results shown in Figure 2. This Figure includes six graphs arranged in a matrix of two columns and three rows: the graphs in the left column correspond to the random, random prototypes ((19) with $\alpha = 0.75$) and maxmin initialisation techniques; while those in the right column refer to the three minor variations of the "same-vector" idea proposed in the preceding section. For each initialisation technique and each $I \in \{1, 2, 5, 10, 15, 20, 25\}$, the standard experimental procedure was run 50 times, each one entailing an I-component Bernoulli mixture classifier trained from a different random seed. Each graph includes four curves computed from these runs: the (normalised) average log-likelihood of the classifier parameters for both the training and test sets, and the average classification error rate, also for both sets (error bars show standard error). Taking into account the results reported in [8] for this task, here we have only considered a resolution of $S = 20$ pixels. Also, in order to allow direct comparison, only classifiers with class-conditional mixtures of identical number of components, $I_c = I$, have been considered.

From the results shown in Figure 2, it can be said that all techniques give similar results, except random initialisation, which does not seem to be as good as the others. An immediate consequence of this result is that maxmin initialisation becomes less attractive since, in comparison with its alternatives, it is more computationally demanding and difficult to implement. Let us compare random prototypes with hypercube centre, which somehow surprisingly appears to be a good choice among the three variations of the "same-vector" idea. For ease of comparison, the error rate curves (for test data) of these techniques are plotted together in Figure 3. Although standard error bars overlap, we would say that hypercube centre is a bit superior to random prototypes.

6 Conclusions

The results presented in this paper can be considered as a continuation of previous work on the use of Bernoulli mixtures for binary data and, in particular, for binary images. Six EM initialisation techniques have been described and compared on an OCR task consisting in the recognition of handwritten Indian digits. Three of these techniques have been already used in our previous works, though here we have tried to provide a better description of them and, more importantly, a common basis for empirical comparison. The other three techniques, which are proposed here, can be interpreted as minor variations on a very simple idea (to use the same vector for all prototypes). From the empirical results obtained in the Indian digits recognition task, we can conclude that "random prototypes" (linear combination of random parameters and randomly chosen training bit vectors) and "hypercube centre" (all prototypes are slightly perturbed versions of $\mathbf{0.5}$) are relatively good initialisation techniques.

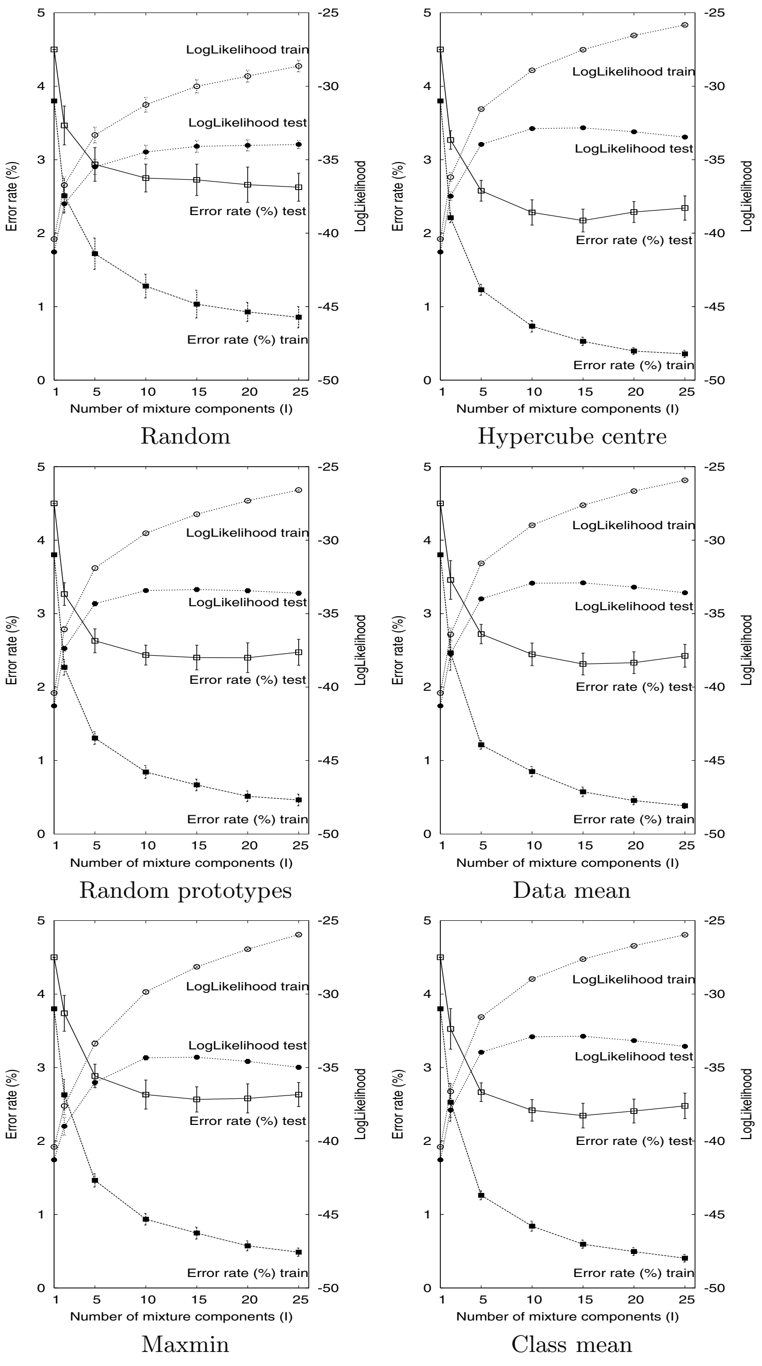


Fig. 2. Comparison of six initialisation techniques: log-likelihood and error rate (for training and test data) of the I-component Bernoulli mixture classifier ($I = 1, \ldots, 25$).

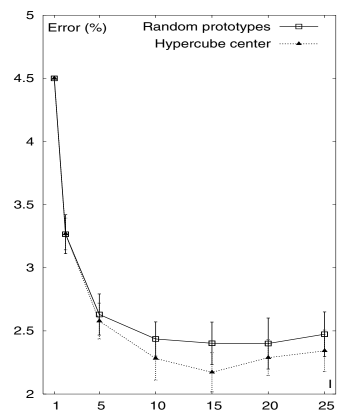


Fig. 3. Comparison of "random prototypes" and "hypercube centre" initialisations: error rate (for test data) of the I-component Bernoulli mixture classifier ($I = 1, \ldots, 25$).

References

1. Y. Al-Ohali, M. Cheriet, and C. Suen. Databases for recognition of handwritten Arabic cheques. *Pattern Recognition*, 36:111–121, 2003.
2. M. A. Carreira-Perpiñán and S. Renals. Practical identifiability of finite mixtures of multivariate Bernoulli distributions. *Neural Computation*, 12(1):141–152, 2000.
3. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38, 1977.
4. R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley, 1973.
5. J. González, A. Juan, P. Dupont, E. Vidal, and F. Casacuberta. A Bernoulli mixture model for word categorisation. In *Proc. of the IX Spanish Symposium on Pattern Recognition and Image Analysis*, volume I, pages 165–170, Benicàssim (Spain), May 2001.
6. J. Grim, P. Pudil, and P. Somol. Multivariate Structural Bernoulli Mixtures for Recognition of Handwritten Numerals. In *Proc. of the ICPR 2000*, volume 2, pages 585–589, Barcelona (Spain), September 2000.
7. A. K. Jain, R. P. W. Duin, and J. Mao. Statistical Pattern Recognition: A Review. *IEEE Trans. on PAMI*, 22(1):4–37, 2000.
8. A. Juan and E. Vidal. Bernoulli mixture models for binary images. In *Proc. of the ICPR 2004.* Submitted.
9. A. Juan and E. Vidal. On the use of Bernoulli mixture models for text classification. *Pattern Recognition*, 35(12):2705–2710, December 2002.

A New Family of Weak Estimators for Training in Non-stationary Distributions

B. John Oommen[1,⋆] and Luis Rueda[2,⋆⋆]

[1] School of Computer Science, Carleton University
1125 Colonel By Dr., Ottawa, ON, K1S 5B6, Canada
oommen@scs.carleton.ca
[2] School of Computer Science, University of Windsor
401 Sunset Avenue, Windsor, ON, N9B 3P4, Canada
lrueda@uwindsor.ca

Abstract. In this paper, we formally present a novel estimation method, referred to as the Stochastic Learning Weak Estimator (SLWE), which is used to estimate the parameters of a binomial distribution, where the convergence of the estimate is weak, i.e. *in law*. The estimation is based on the principles of stochastic learning. Even though our new method includes a learning coefficient, λ, it turns out that the mean of the final estimate is independent of λ, the variance of the final distribution decreases with λ, and the speed decreases with λ. Similar results are true for the multinomial case. An empirical analysis on synthetic data shows the advantages of the scheme for non-stationary distributions. Conclusive results demonstrate the advantage of SLWE for a pattern-recognition problem which has direct implications in data compression. In this case, the underlying distribution in the data file to be compressed is non-stationary, and it is estimated and learnt using the principles highlighted here. By classifying *its* variation and using it in the compression, the superiority of the scheme is documented.

1 Introduction

The theory of estimation has been studied for hundreds of years [1, 15]. It is also easy to see that the learning (training) phase of a statistical pattern recognition system is, indeed, based on estimation theory [2, 17]. Estimation methods generally fall into various categories, including the Maximum Likelihood Estimates (MLE) and the Bayesian family of estimates.

Although the MLEs and Bayesian estimates have good computational and statistical properties, the fundamental premise for establishing the quality of estimates is based on the assumption that the parameter being estimated does not change with time. In other words, the distribution is assumed to be *stationary*. Thus, it is generally assumed that there is an underlying parameter θ, *which does not change with time*, and as the number

⋆ *Fellow of the IEEE.* Partially supported by NSERC, the Natural Science and Engineering Research Council of Canada.

⋆⋆ Member, IEEE. Partially supported by NSERC, the Natural Science and Engineering Research Council of Canada.

of samples increases, we would like the estimate $\hat{\theta}$ to converge to θ with probability one, or in a mean square sense.

There are numerous problems which we have recently encountered, where these strong estimators pose a real-life concern. One scenario occurs in pattern classification involving moving scenes. We also encounter the same situation when we want to perform adaptive compression of files which are interspersed with text, formulae, images and tables. Similarly, if we are dealing with adaptive data structures, the structure changes with the estimate of the underlying data distribution, and if the estimator used is "strong" (i.e., w. p. 1), it is hard for the learned data structure to emerge from a structure that it has converged to. Indeed, we can conclusively demonstrate that it is sub-optimal to work with strong estimators in such application domains, i.e., when the data is truly *non-stationary*.

In this paper, we shall present one such "weak" estimator, referred to as the Stochastic Learning Weak Estimator (SLWE), and which is developed by using the principles of stochastic learning. In essence, the estimate is updated at each instant based on the value of the current sample. However, this updating is not achieved using an *additive* updating rule, but rather by a *multiplicative* rule, akin to the family of linear action-probability updating schemes [6, 7]. The formal results that we have obtained for the binomial distribution are quite fascinating. Similar results have been achieved for the multinomial case, except that the variance (covariance) has not been currently derived.

The entire field of obtaining weak estimators is novel. In that sense, we believe that this paper is of a pioneering sort – we are not aware of any estimators which are computed as explained here. We also hope that this is the start of a whole new body of research, namely those involving weak estimators for various distributions, and their applications in various domains. In this regard, we are currently in the process of deriving weak estimators for various distributions. To demonstrate the power of the SLWE in real-life problems, we conclude the paper with conclusive results applicable to data compression problems, in which the underlying distribution is non-stationary.

The more detailed application of these results in adaptive data compression, statistical pattern recognition, adaptive data structures, and scene analysis is currently underway. Finally, the application of non-linear updating schemes to obtain weak estimators is also open, although some initial results are currently available.

To devise a new estimation method, we have utilized the principles of learning as achieved by the families of Variable Structure Stochastic Automata (VSSA) [3, 4, 6]. In particular, the learning we have achieved is obtained as a consequence of invoking algorithms related to families of linear schemes, such as the Linear Reward-Inaction (L_{RI}) scheme. The analysis is also akin to the analysis used for *these* learning automata. This involves first determining the updating equations, and taking the conditional expectation of the quantity analyzed. The condition disappears when the expectation operator is invoked a second time, leading to a difference equation for the specified quantity, which equation is later explicitly solved. We have opted to use these families of VSSA in the design of our SLWE, because it turns out that the analysis is considerably simplified and (in our opinion) fairly elegant.

The question of deriving weak estimators based on the concepts of *discretized* LA [8, 10] and *estimator-based* LA [5, 9, 11, 13, 14, 16] remains open. We believe, however, that even if such estimator-based SLWE are designed, the analysis of their properties will not be trivial.

2 Weak Estimators of Binomial Distributions

Through out this paper we assume that we are estimating the parameters of a binomial/multinomial distribution. The binomial distribution is characterized by two parameters, namely, the *number* of Bernoulli trials, and the parameter characterizing *each* Bernoulli trial. In this regard, we assume that the number of observations is the number of trials. Thus, all we have to do is to estimate the *Bernoulli* parameter for each trial. This is what we endeavor to do using stochastic learning methods.

Let X be a binomially distributed random variable, which takes on the value of either '1' or '2'[1]. We assume that X obeys the distribution S, where $S = [s_1, s_2]^T$. In other words,

$X =$ '1' with probability s_1
$=$ '2' with probability s_2 ,

where, $s_1 + s_2 = 1$.

Let $x(n)$ be a concrete realization of X at time 'n'. The intention of the exercise is to estimate S, i.e., s_i for $i = 1, 2$. We achieve this by maintaining a running estimate $P(n) = [p_1(n), p_2(n)]^T$ of S, where $p_i(n)$ is the estimate of s_i at time 'n', for $i = 1, 2$. Then, the value of $p_i(n)$ is updated as per the following simple rule:

$$p_1(n+1) \leftarrow \lambda p_1(n) \qquad \text{if } x(n) = 2 \tag{1}$$
$$\leftarrow 1 - \lambda p_2(n) \qquad \text{if } x(n) = 1\,. \tag{2}$$

where λ is a user-defined parameter, $0 < \lambda < 1$.

In order to simplify the notation, the vector $P(n) = [p_1(n), p_2(n)]^T$ refers to the estimates of the probabilities of '1' and '2' occurring at time 'n', namely $p_1(n)$ and $p_2(n)$ respectively. In the interest of simplicity, we omit the index n, whenever there is no confusion, and thus, P implies $P(n)$.

The first theorem, whose proof can be found in [12], concerns the distribution of the vector P which estimates S as per Equations (1) and (2). We shall state that P converges in distribution. The mean of P is shown to converge exactly to the mean of S. The proof, which is a little involved, follows the types of proofs used in the area of stochastic learning [12].

Theorem 1. *Let X be a binomially distributed random variable, and $P(n)$ be the estimate of S at time 'n'. Then, $E[P(\infty)] = S$.* □

The next results that we shall prove indicates that the distribution of $\mathrm{E}[P(n+1)]$ follows $\mathrm{E}[P(n)]$ in a Markovian manner. We derive the explicit Markovian dependence, and allude to the resultant properties by virtue of the ergodic nature of the Markov matrix. This leads us to two results, namely that of the limiting distribution of the chain,

[1] We depart from the traditional notation of the random variable taking values of '0' and '1', so that the notation is consistent when we consider the multinomial case.

and that which concerns the rate of convergence of the chain. It turns out that while the former is independent of the learning parameter, λ, the latter is determined *only* by λ.

The reader will observe that the results we have derived are asymptotic results. In other words, the mean of $P(n)$ is shown to converge exactly to the mean of S. The implications of the "asymptotic" nature of the results will be clarified presently. The proofs of Theorems 2, 3 and 4 can be found in [12].

Theorem 2. *Let X be a binomially distributed random variable governed by the distribution S. If $P(n+1)$ is the estimate of S at time '$n+1$', $E[P(n)]$ obeys an ergodic Markov chain whose steady state distribution converges to S.* □

Theorem 3. *Let X be a binomially distributed random variable governed by the distribution S, and $P(n)$ be the estimate of S at time 'n' obtained by (1) and (2). Then, the rate of convergence of P to S is fully determined by λ.* □

We now derive the explicit expression for the asymptotic variance of the SLWE. A small value of λ leads to fast convergence and a large variance and vice versa.

Theorem 4. *Let X be a binomially distributed random variable governed by the distribution S, and $P(n)$ be the estimate of S at time 'n' obtained by (1) and (2). Then, the algebraic expression for the variance of $P(\infty)$ is fully determined by λ.* □

When $\lambda \to 1$, it implies that the variance tends to zero, implying mean square convergence. The plot of the variance of $p_1(\infty)$ against λ, is depicted in Fig. 1, where $s_1 = 0.6$ and $s_2 = 0.4$. Observe that the *maximum* value of the variance, Var$[p_1(\infty)]$ $= s_1 s_2 = 0.24$, is attained when $\lambda = 0$, and the *minimum* value of the variance, Var$[p_1(\infty)] = 0$, is achieved when $\lambda = 1$.

Although the result derived is of an asymptotic sort, if the value of λ is even as "small" as 0.9, the SLWE will be able to track the change, even if the environment switches its Bernoulli parameter after 50 steps. Observe too that we do not need to introduce or consider the use of a "sliding window".

3 Weak Estimators of Multinomial Distributions

In this section, we shall consider the problem of estimating the parameters of a multinomial distribution. The multinomial distribution is characterized by two parameters, namely, the *number* of trials, and a probability vector which determines the probability of a specific event (from a pre-specified set of events) occurring. In this regard, the number of observations is the number of trials. Thus, we are to estimate the latter probability *vector* associated with the set of possible outcomes or trials. Specifically, let X be a multinomially distributed random variable, which takes on the values from the set $\{`1', \ldots, `r'\}$. We assume that X is governed by the distribution $S = [s_1, \ldots, s_r]^T$ as:
$X = `i'$ with probability s_i, where $\sum_{i=1}^{r} s_i = 1$.

Also, let $x(n)$ be a concrete realization of X at time 'n'. The intention of the exercise is to estimate S, i.e., s_i for $i = 1, \ldots, r$. We achieve this by maintaining a running estimate $P(n) = [p_1(n), \ldots, p_r(n)]^T$ of S, where $p_i(n)$ is the estimate of s_i at time

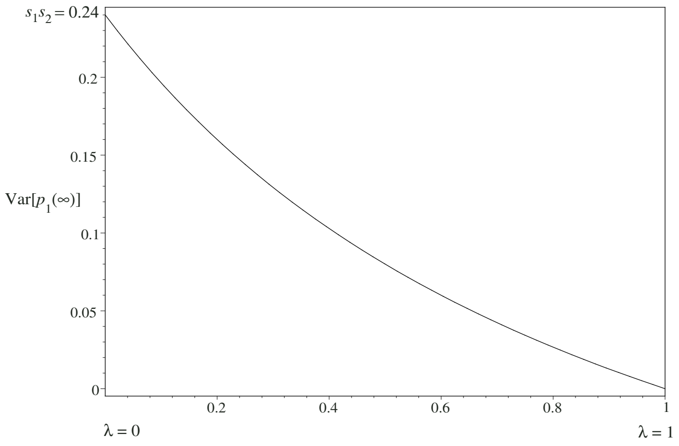


Fig. 1. Plot of the function Var$[p_\bullet(\infty)]$ for different values of λ in the interval $[0, 1]$, where $s_\bullet = 0.6$ and $s_\bullet = 0.4$.

'n', for $i = 1, \ldots, r$. Then, the value of $p_1(n)$ is updated as per the following simple rule (the rules for other values of $p_j(n)$ are similar):

$$p_1(n+1) \leftarrow p_1 - (1-\lambda) \sum_{j \neq 1} p_j \quad \text{when } x(n) = 1 \tag{3}$$

$$\leftarrow \lambda p_1 \quad \text{when } x(n) \neq 1 \tag{4}$$

As in the binomial case, the vector $P(n) = [p_1(n), p_2(n), \ldots, p_r(n)]^T$ refers to the estimate of $S = [s_1, s_2, \ldots, s_r]^T$ at time 'n', and we will omit the reference to time 'n' in $P(n)$ whenever there is no confusion.

The results that we present now are as in the binomial case, i.e. the distribution of $\mathrm{E}[P(n+1)]$ follows $\mathrm{E}[P(n)]$ in a Markovian manner. This leads us to two results, namely to that of the limiting distribution of the chain, and that which concerns the rate of convergence of the chain. It turns out that both of these, in the very worst case, could only be dependent on the learning parameter λ. However, to our advantage, while the former is *independent* of the learning parameter, λ, the latter is *only* determined by it (and not a function of it). The complete proofs of Theorems 5, 6 and 7 are found in [12].

Theorem 5. *Let X be a multinomially distributed random variable governed by the distribution S, and $P(n)$ be the estimate of S at time 'n' obtained by (3) and (4). Then, $E[P(\infty)] = S$.* □

We shall now state that the convergence of P to S occurs in a Markovian manner and derive the explicit form of the underlying Markovian matrix, say, M. Thus, we analyze the properties of M, and in particular, its eigenvalue properties.

Theorem 6. *Let X be a multinomially distributed random variable governed by the distribution S, and $P(n)$ be the estimate of S at time 'n' obtained by (3) and (4). Then,* *the expected estimated probability vector follows a Markovian behavior in which every off-diagonal term of the underlying Markov matrix has the same multiplicative factor,* $(1 - \lambda)$*. Furthermore, the final expected solution is independent of* λ*.* □

The convergence and eigenvalue properties of $\mathbf{M}$ follow.

Theorem 7. *Let X be a multinomially distributed random variable governed by the distribution S, and $P(n)$ be the estimate of S at time 'n' obtained by (3) and (4). Then, all the non-unity eigenvalues of $\mathbf{M}$ are exactly λ, and thus the rate of converge of P is fully determined by λ.* □

A small value of λ leads to fast convergence and a large variance, and vice versa. Again, although the results we have derived are asymptotic, if λ is even as "small" as 0.9, after 50 iterations, the variation from the asymptotic value will be of the order of 10^{-50}. In other words, after 50 steps, the SLWE will be able to track this change. Our experimental results demonstrate this fast convergence.

4 Experimental Results

To assess the efficiency of the SLWE introduced here, we have estimated the parameters for binomial and multinomial random variables using a value of $\lambda = 0.99$. We have also estimated the parameters of these random variables by following the traditional MLE.

4.1 Binomial Random Variables

The estimation of the parameters for binomial random variables has been extensively tested for numerous distributions. In the interest of brevity, we merely cite one specific example. Also, to make the comparison meaningful, we have followed the MLE computation using the identical data stream. In each case, the estimation algorithms were presented with random occurrences of the variables for $n = 4,000$ time instances. In the case of the SLWE, the true underlying value of s_1 was initially set to be 0.8, and randomly modified after every 500 steps. The new values in the next periods were uniformly drawn from the interval $[0, 1]$, being 0.42, 0.92, etc.

We report two kinds of results. In the first case, the results obtained are from a single run. The plot of this behavior is shown in Fig. 2. Observe that the MLE starts with an initial value of 0.5 and attains a value quite close to 0.8 in the first time period. As opposed to this, the SLWE starts close to the value 0.5, but quickly (in less than 100 iterations) converges to a value close to 0.8. Thereafter, the MLE is not capable of tracking the "switching" of the Bernoulli parameter. Thus, at the end of the second period, during which the value of s_1 is 0.42, the MLE attains a value of 0.638794, which is significantly different from the true one. The SLWE, however, quickly learns that the parameter has changed, and its estimate, p_1, of s_1 is much closer to the true value. In the interest of playing on a level field, we have assumed that neither algorithm is aware of the switching nature of the environment. In other words, the MLE does not use a "sliding window" to estimate the running mean. The behavior of the MLE would probably be superior if this information were provided, but the question of how large

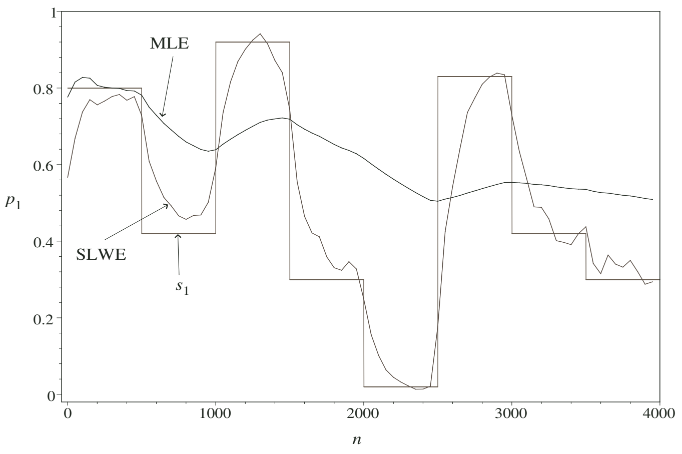


Fig. 2. Plot of the expected value of $p_{\bullet}(n)$, at time 'n', which was estimated by using the SLWE and the MLE. The actual values of $s_{\bullet}$, which change every 500 time steps, are also plotted.

this window should be, is a consideration that is not easily determined. This would be specially crucial in fast-changing environments, where the SLWE would not require any additional information.

In order to demonstrate the superiority of the SLWE over the MLE on an ensemble of experiments, we also report the respective ensemble averages. The same experiment was repeated $1,000$ times, and the ensemble average at every time step was recorded. Clearly, the variations of the estimates would be much smoother. This behavior is shown in Fig. 3. Observe that the MLE follows s_1 *exactly* in the first window, but is thereafter severely handicapped in tracking the variations. In contrast, the SLWE quickly adjusts to the changes, as expected, in a geometric manner. In our opinion, the results are quite impressive. The plot of the "empirical" variance (shown in [12]) is identical.

4.2 Multinomial Random Variables

We have also performed simulations for multinomial random variables, where the parameters were estimated by following the SLWE and the MLE. We considered a multinomial random variable, X, which can take any of four different values, namely '1', '2', '3' or '4'. As in the binomial case, we ran the estimators for $4,000$ steps, and repeated this $1,000$ times. We then took the ensemble average. For each experiment, we computed $||P - S||$, the *Euclidean distance* between P and S. This was a measure of how good our estimate, P, was of S.

The plot of the latter distance obtained from our experiments is depicted in Fig. 4. While the distance of the SLWE quickly drops towards zero for every "switch" of S, the MLE is not able to capture these "switches" properly. Experimental results on real-life problems, involving adaptive data compression show a similar pattern [12].

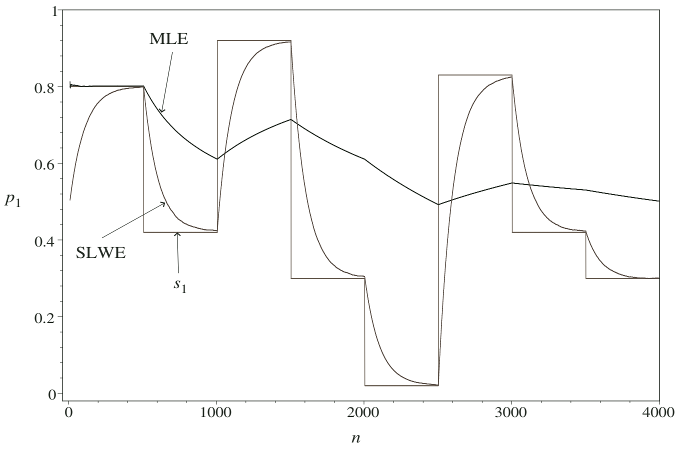


Fig. 3. Plot of the estimated probability of $s_{\bullet}$, $p_{\bullet}(n)$, at time 'n' by utilizing two approaches: the SLWE and the MLE. The actual values of $s_{\bullet}$, which change every 500 time steps, are also plotted.

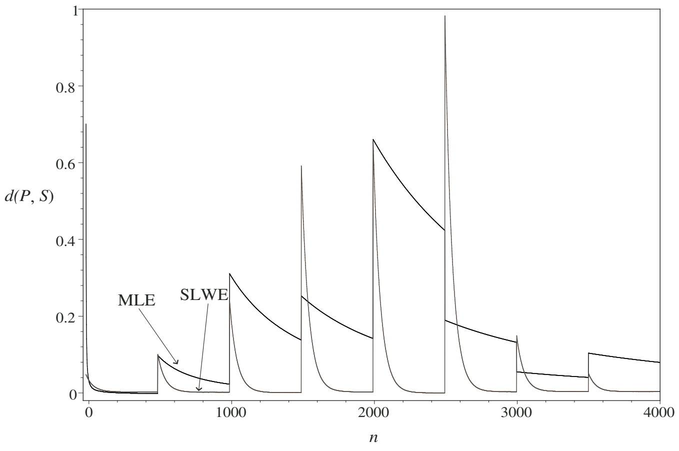


Fig. 4. Plot of the *Euclidean norm* of $P - S$ (or *Euclidean distance* between P and S), where P was estimated by using both the SLWE and the MLE. The values for the distances are computed for groups of 500 steps.

5 Conclusions and Future Work

In this paper, we have considered the problem of estimating the parameters of a distribution from its observations. Unlike traditional estimates that possess *strong* convergence properties, we have argued that there is a need for estimates that do not possess such strong convergence properties. Motivated by real-life applications, we formally presented an estimation method based on the principles of stochastic learning, which yields the binomial and multinomial estimates that converge *weakly*, i.e. in law. In the case of the binomial distribution, even though our new method includes a learning coefficient, λ, the mean of the final estimate is independent of λ, but the asymptotic variance decreases with λ, as does the convergence. Similar results are true for the multinomial case. Experimental results for both binomial and multinomial random variables, and for compressing data files drawn from non-stationary sources demonstrate the superiority of the SLWE over the MLE in these application domains.

References

1. G. Casella and R. Berger. *Statistical Inference*. Brooks/Cole Pub. Co., second edition, 2001.
2. R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley and Sons, Inc., New York, NY, 2nd edition, 2000.
3. S. Lakshmivarahan. *Learning Algorithms Theory and Applications*. Springer-Verlag, New York, 1981.
4. S. Lakshmivarahan and M. A. L. Thathachar. Absolutely Expedient Algorithms for Stochastic Automata. *IEEE Trans. on System, Man and Cybernetics*, SMC-3:281–286, 1973.
5. J. K. Lanctôt and B. J. Oommen. Discretized Estimator Learning Automata. *IEEE Trans. on Systems, Man and Cybernetics*, 22(6):1473–1483, 1992.
6. K. Narendra and M. Thathachar. *Learning Automata. An Introduction*. Prentice Hall, 1989.
7. J. Norris. *Markov Chains*. Springer Verlag, 1999.
8. B.J. Oommen. Absorbing and Ergodic Discretized Two-Action Learning Automata. *IEEE Trans. on System, Man and Cybernetics*, SMC-16:282–296, 1986.
9. B.J. Oommen and M. Agache. Continuous and Discretized Pursuit Learning Schemes: Various Algorithms and Their Comparison. *IEEE Trans. on Systems, Man and Cybernetics*, SMC-31(B):277–287, June 2001.
10. B.J. Oommen and J. R. P. Christensen. Epsilon-Optimal Discretized Reward-Penalty Learning Automata. *IEEE Trans. on System, Man and Cybernetics*, SMC-18:451–458, 1988.
11. B.J. Oommen and J. K. Lanctôt. Discretized Pursuit Learning Automata. *IEEE Trans. on System, Man and Cybernetics*, 20(4):931–938, 1990.
12. B.J. Oommen and L. Rueda. Stochastic Learning-based Weak Estimation of Multinomial Random Variables and Its Applications to Non-stationary Environments. (Submitted for Publication).
13. G. I. Papadimitriou. A New Approach to the Design of Reinforcement Schemes for Learning Automata: Stochastic Estimator Learning Algorithms. *IEEE Trans. on Knowledge and Data Engineering*, 6:649–654, 1994.
14. K. Rajaraman and P. S. Sastry. Finite Time Analysis of the Pursuit Algorithm for Learning Automata. *IEEE Trans. on Systems, Man and Cybernetics*, 26(4):590–598, 1996.
15. S. Ross. *Introduction to Probability Models*. Academic Press, second edition, 2002.
16. M. A. L. Thathachar and P.S. Sastry. A Class of Rapidly Converging Algorithms for Learning Automata. *IEEE Trans. on Systems, Man and Cybernetics*, SMC-15:168–175, 1985.
17. A. Webb. *Statistical Pattern Recognition*. John Wiley & Sons, N.York, second edition, 2002.

Path Relinking Particle Filter for Human Body Pose Estimation

Juan José Pantrigo[1], Ángel Sánchez[1], Kostas Gianikellis[2], and Abraham Duarte[1]

[1] Universidad Rey Juan Carlos, c/ Tulipán s/n
28933 Móstoles, Spain
{j.j.pantrigo,an.sanchez,a.duarte}@escet.urjc.es
[2] Universidad de Extremadura, Avda. Universidad s/n
10071 Cáceres, Spain
kgiannik@unex.es

Abstract. This paper introduces the Path Relinking Particle Filter (PRPF) algorithm for improving estimation problems in human motion capture. PRPF hybridizes both Particle Filter and Path Relinking frameworks. The proposed algorithm increases the performance of general Particle Filter by improving the quality of the estimate, by adapting computational load to problem constraints and by reducing the number of required evaluations of the weighting function. We have applied the PRPF algorithm to 2D human pose estimation. Experimental results show that PRPF drastically reduces the MSE value to obtain the set of markers with respect to Condensation and Sampling Importance Resampling (SIR) algorithms.

1 Introduction

Automatic visual analysis of human motion is an active research topic in Computer Vision and its interest has been growing during last decade [1][2][3]. Biomechanics of Human Movement is an interdisciplinary area, supported by Biomedical Sciences, Mechanics and other different technologies, that studies the human body behavior subject to mechanical loads [4]. This area has evolved into three main fields [4]: medical, sports and occupational.

A typical biomechanical study involves four phases: (i) defining a suitable theoretical model, (ii) obtaining relevant point (marker) coordinates, (iii) performing the kinematic analysis, and (iv) determining parameters of interest. Manual digitizing is generally used to obtain the marker coordinates. These procedures are slow and require the supervision of specialists in human anatomy. Automated marker-based systems [5] permit to automatically obtain the set of marker coordinates, although they are intrusive and force the use of expensive specialised hardware. Therefore, the goal of research in human motion capture is the development of an automatic full-body tracking system, which runs under conventional hardware, and oriented to processing realistic applications.

Most of human motion analysis studies are based on articulated models that properly describe the human motion [1][6][7][8]. Recent research in human motion analysis makes use of the particle filter framework. The Particle Filter (PF) algorithm, (also termed as Condensation algorithm) enables the modelling of a stochastic process with an arbitrary probability density function (pdf), by approximating it numerically with a set of points called particles in a state-space process [9]. Deutscher [5] developed an algorithm termed Annealed Particle Filter (APF) for tracking people using articulated models.

The principal contribution of this paper is the development of the Path Relinking Particle Filter (PRPF) algorithm. The algorithm is inspired by the Path Relinking metaheuristic proposed by Glover [10][11] as a way to integrate intensification and diversification strategies in the context of combinatorial optimization problems. PRPF hybridizes both Particle Filter (PF) and Path Relinking (PR) frameworks in two different stages. In the PF stage, a particle set is propagated and updated to obtain a new particle set. In PR stage, a selected elite set from the particle set is selected, and new solutions are constructed by exploring trajectories that connect each of the particles in the elite set. PRPF algorithm significantly improves the performance of both general and other optimized particle filters.

2 Particle Filter

General particle filters (PF) are sequential Monte Carlo estimators based on particle representations of probability densities which can be applied to any state-space model [12]. The state-space model consists of two processes: (i) an observation process $p(\mathbf{Z}_t|\mathbf{X}_t)$, where $\mathbf{X}$ denotes the system state vector and $\mathbf{Z}$ is the observation vector, and (ii) a transition process $p(\mathbf{X}_t|\mathbf{X}_{t-1})$. Assuming that observations $\{\mathbf{Z}_0, \mathbf{Z}_1, \ldots, \mathbf{Z}_t\}$ are sequentially measured in time, the goal is to estimate the new system state $\{\chi_0, \chi_1, \ldots, \chi_t\}$ at each time. In the framework of Sequential Bayesian Modelling, posterior pdf is estimated in two stages:

(i) Prediction: the posterior pdf $p(\mathbf{X}_{t-1}|\mathbf{Z}_{t-1})$ is propagated at the time *t* using the Chapman-Kolmogorov equation:

$$p(\mathbf{X}_t \mid \mathbf{Z}_{t-1}) = \int p(\mathbf{X}_t \mid \mathbf{X}_{t-1}) p(\mathbf{X}_{t-1} \mid \mathbf{Z}_{t-1}) d\mathbf{X}_{t-1} \tag{1}$$

A predefined object motion model is used to obtain an updated particle set.

(ii) Evaluation: the posterior pdf is computed using the observation vector $\mathbf{Z}_t$:

$$p(\mathbf{X}_t \mid \mathbf{Z}_t) = \frac{p(\mathbf{Z}_t \mid \mathbf{X}_t) p(\mathbf{X}_t \mid \mathbf{Z}_{t-1})}{p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})} \tag{2}$$

The aim of the PF algorithm is to recursively estimate the posterior pdf $p(\mathbf{X}_t|\mathbf{Z}_t)$, that constitutes the complete solution to the sequential estimation problem. This pdf is represented by a set of weighted particles $\{(\mathbf{x}_t^0, \pi_t^0)\ldots(\mathbf{x}_t^N, \pi_t^N)\}$, where the weights $\pi_t^n \propto p(\mathbf{Z}_t| \mathbf{X}_t = \mathbf{x}_t^n)$ are normalised. The state χ_t can be estimated by the equation:

$$\chi_t = \sum_{n=1}^{N} \pi_t^n \mathbf{x}_t^n \tag{3}$$

A pseudocode of a general PF is shown in Figure 1. PF starts by setting up an initial population $\mathbf{X}_0$ of *N* particles using a known pdf. The measurement vector $\mathbf{Z}_t$ at time *t*, is obtained from the image. Particle weights $\mathbf{\Pi_t}$ are computed using weighting function. Weights are normalized and a new particle set $\mathbf{X}^*_t$ is selected. As particles with larger weight values can be chosen several times, a diffusion stage are applied to avoid the loss of diversity in $\mathbf{X}^*_t$. Finally, particle set at time *t+1*, $\mathbf{X}_{t+1}$, is predicted by using the motion model.

```
algorithm PARTICLE_FILTER ((IN)M: video_sequence;
(IN)N:number_of_particles; (OUT)$\chi_t$: estimates_set)
var  t: time;
     $\Pi_t$: weight_set;
     $X_t$, $X^*_t$, $X_{t+1}$: particle_set;
     $Z_t$: measurement_vector;
begin
 t := 0;
 $X_t$ := Initialize(N);
 repeat
    $Z_t$ := ObtainMeasures(S, t);
    $\Pi_t$ := Evaluate($X_t$, $Z_t$);
    [$X_t$, $\Pi_t$] := Normalize($X_t$, $\Pi_t$);
    $\chi_t$ := Estimate($X_t$, $\Pi_t$);
    $X^*_t$ := Select($X_t$, $\Pi_t$);
    $X^*_t$ := Diffuse($X^*_t$);
    $X_{t+1}$ := Predict($X^*_t$);
    t := t+1;
  until (termination_condition)
end.
```

Fig. 1. General Particle Filter Algorithm.

Several optimized algorithms from the general PF approach, which use different strategies to improve its performance have been proposed [5][10][12]. For example, the Sampling Importance Resampling (SIR) algorithm [12] reduces the effects of the degeneracy phenomenon. The goal of SIR algorithm is to eliminate those insignificant particles and to consider the contribution of those ones with larger weight values. Reference [5] presents an Annealed Particle Filter (APF) to track human motion using a proposal articulated body model. APF is applied to searching in high dimensionality spaces. This filter works well for articulated models with 29 DOFs. Partitioned Sampling (PS) [12] is a statistical approach to tackle hierarchical search problems. PS consists of dividing the state-space into two or more partitions, and sequentially applying the stated dynamic model for each partition followed by a weighted resampling stage. The advantages of this technique are that the number of required weighting function evaluations is reduced.

3 Path Relinking

Path Relinking (PR) [10][11] is an evolutionary metaheuristic in the context of the combinatorial optimization problems. PR constructs new high quality solutions by combining others obtained solutions by exploring paths that connect these solutions. To yield better solutions than the original ones, PR starts from a given set of elite solutions obtained during a search process, called *RefSet* (short for "Reference Set"). These solutions are ordered according to their quality, and new solutions are then generated, by exploring trajectories that connect solutions in the *RefSet*. The metaheuristic starts with two of these solutions *x'* and *x''*, and it generates a path: $x' = x(1), x(2), \ldots, x(r) = x''$, in the neighbourhood space that leads towards the new solution sequence. In order to produce better quality solutions, it is convenient to add a local search optimization phase. Figure 2 sketches a pseudocode of the PR metaheuristic.

```
algorithm PATH_RELINKING ((IN)Solutions: Solution_set; (IN)$\Pi_S$: weight_set; (IN)b: RefSet_size; (IN)NumImp: Integer; (IN)$Z_t$: measurement_vector; (OUT)RefSet: Solution_set; (OUT)$\Pi_R$: weight_set)
var  $\Pi_P$, $\Pi_{NR}$: weight_set;
      RefSetNew, Path: Solution_set;
      NewSubSets: Pairs_of_Solutions;
      NewSolutions: Boolean;
begin
 [RefSet, $\Pi_R$] := MakeRefSet(Solutions, $\Pi_S$, b);
 [RefSet, $\Pi_R$] := Order(RefSet);
 NewSolutions := TRUE;
 while (NewSolutions) do
    NewSubsets := MakeNewSubsets(b);
    NewSolutions := FALSE;
    while (NewSubsets = ∅) do
       Path := MakePath(NewSubSets(1));
       $\Pi_P$ := Evaluate(Path, $Z_t$);
       [Path, $\Pi_P$] := Improve(Path, $\Pi_P$, NumImp);
       [RefSetNew, $\Pi_{RN}$] := Update(RefSet, $\Pi_R$, Path, $\Pi_P$);
       if (RefSetNew <> RefSet) then
          NewSolutions := TRUE;
       end if
       NewSubsets := Delete(NewSubsets, 1);
    end while
    RefSet := RefSetNew;
    $\Pi_R$ := $\Pi_{RN}$
 end while
end.
```

Fig. 2. Template of the Path Relinking metaheuristic.

4 Path Relinking Particle Filter

The Path Relinking Particle Filter (PRPF) algorithm is introduced in this paper to be applied to estimation problems in sequential processes that can be expressed using the state-space model abstraction. As pointed in section 1, PRPF integrates PF and PR frameworks in two different stages. The PRPF algorithm is centered on a delimited region of the state-space in which it is highly probable to find new better solutions than the initial computed ones. PRPF increases the performance of general PF by improving the quality of the estimate, by adapting computational load to constraints and by reducing the number of required evaluations of the particle weighting function. Figure 3 shows a graphical template of the PRPF method. Dashed lines separate the two main components in the PRPF scheme: PF and PR optimization, respectively.

PRPF starts with an initial population of *N* particles drawn from a known pdf. Each particle represents a possible solution of the problem. Particle weights are computed using a weighting function and a measurement vector. PR stage is later applied to

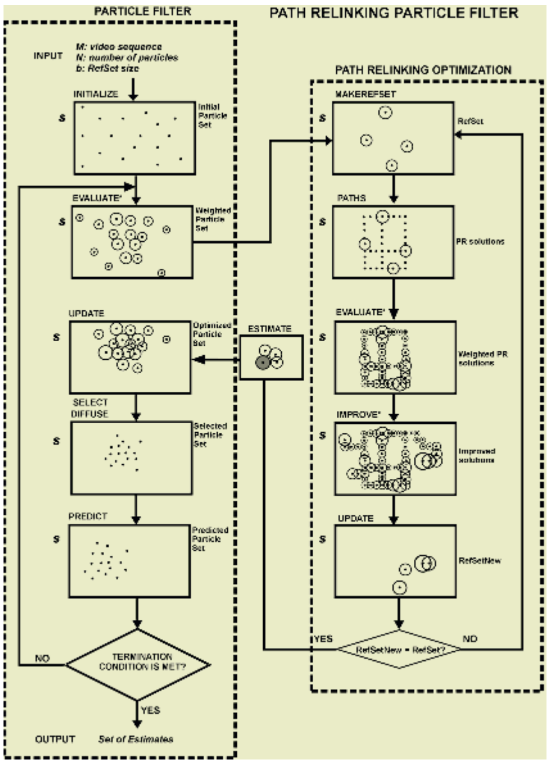


Fig. 3. Path Relinking Particle Filter scheme. Actual frame measures are required during. EVALUATE and UPDATE stages (*).

improve the best obtained solutions of the particle filter stage. A Reference Set (*RefSet*) is created selecting the *b* (*b*<<*N*) best particles in particle set. New solutions are generated and evaluated, by exploring predefined trajectories that connect all possible pairs of particles in the *RefSet*. Paths are generated following those directions according to the state-space axes. In order to improve the solution fitness, a local search from some of the generated solutions within the PR procedure is performed. Worst solutions in the *RefSet* are replaced when there are better performance new ones. PR stage ends when new generated solutions *RefSetNew* do not improve the quality of the *RefSet*. Once the PR stage is finished, the "worst" particles are replaced with the *RefSetNew* solutions. Then, a new population of particles is created by selecting the individuals from particle set with probabilities according to their weights. In order to avoid the loss of diversity, a diffusion stage is applied to the particles in new set. Finally, particles are projected into the next time step by making use of the update rule. The proposed pseudocode of PRPF algorithm for visual tracking is shown in Figure 4.

PRPF system estimator quality is improved and the required number of evaluations for the weighting function is also reduced. Therefore, PRPF search in state-space is not performed randomly like in a general particle filter. PRPF is time-adaptive since the number of evaluations of the weighting function changes in each time step. If the initial solutions in the *RefSet* are far away one from each other, then paths connecting solutions are long, and the number of explored solutions increases. It is not possible to have any estimate of the previous state of the system at the beginning of the visual tracking, thus the particle filter is usually randomly initialized. The number of individuals in the particle filter does not change during the algorithm execution. PRPF algorithm reduces the total required number of evaluations of the weighting function when increasing the number of total time steps.

```
algorithm PRPF((IN)M: video_sequence; (IN)N: number_of_particles;
(IN)b: RefSet_size; (IN)NumImp: Integer; (OUT)$\chi_t$: estimate_set)
var  t: integer;
     $\Pi_t$, $\Pi_P$, $\Pi_R$, $\Pi_{NR}$: weight sets;
     $X_t$, $X^*_t$, RefSet, RefSetNew, Path: particle sets;
     NewSubSets: Pairs of Particles;
     NewSolutions: Boolean;
     $Z_t$: measurement vector;
begin
 t := 0;
 $X_t$ := Initialize(N);
 repeat
    $Z_t$ := ObtainMeasures(M, t);
    $\Pi_t$ := Evaluate($X_t$, $Z_t$);
    [RefSet, $\Pi_R$]:=PATH_RELINKING($X_t$, $\Pi_t$, b, NumImp, $Z_t$); {See fig. 2}
    $\chi_t$ := Estimate(RefSet, $\Pi_R$);
    [$X_t$, $\Pi_t$] := Update($X_t$, $\Pi_t$, RefSet, $\Pi_R$);
    [$X_t$, $\Pi_t$] := Normalize($X_t$, $\Pi_t$);
    $X^*_t$ := Select($X_t$, $\Pi_t$);
    $X^*_t$ := Diffuse($X^*_t$);
    $X_{t+1}$ := Predict($X^*_t$);
    t := t+1;
 until (termination_condition)
end.
```

Fig. 4. Path Relinking Particle Filter Algorithm.

5 Considered Upper-Body Model for Pose Estimation

The automatic computation of marker coordinates is the aim of human pose estimation system. Our proposed PRPF system is applied to determine the position and orientation of body segments in the global frame. The set of particles describe complete solutions for the tracking problem. The particle structure in an eight-limbs model is:

$$\left[x_1, y_1, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \dot{x}_1, \dot{y}_1, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5, \dot{\theta}_6, \dot{\theta}_7, \dot{\theta}_8\right] \tag{4}$$

where x and y are the spatial positions, θ is the angle and $\dot{x}$ represents the first derivative of magnitude (velocity).

A geometrical model is used to represent the human upper-body in a 2D space as a hierarchical set of articulated limbs. It stores time-independent parameters describing the body components. On the other hand, each particle stores time-dependent values relating to limbs position and orientation. Therefore, it is possible to build blobs and edges pixel maps combining the particle state prediction and the geometrical model. Figure 5(a) shows the proposed blobs and edge models for upper body tracking.

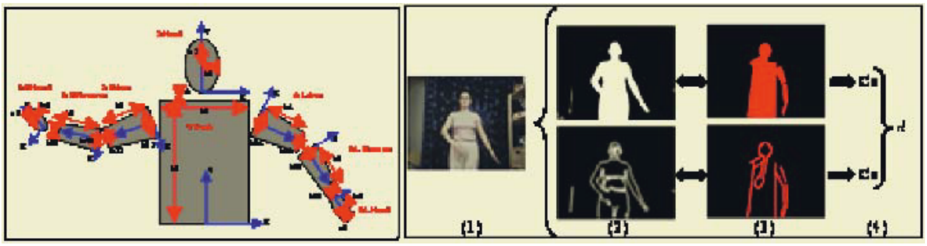


Fig. 5. (a) Proposed human upper-body model. (b) Measurement process: (1) initial image, (2) feature extraction, (3) particle prediction and (4) particle weight computation.

Figure 5(b) represents the measurement process to obtain the particle weights. To construct the weighting function it is necessary to use adequate image features. Continuous edges extracted from a human image usually provide a good measure of visible body limbs. Canny edge detection method can be used to extract edges. Edges outside from human silhouette are removed using background subtraction. The resulting edges are then smoothed using a convolution operation. This produces a pixel map E^M which assigns each pixel a value related to its proximity to an edge. Another pixel map E^P is built using edges produced by the geometrical model of the configuration predicted by the *ith* particle, for each pixel (*j*) in the pixel map. Differences between these two pixel maps are computed by:

$$C_E^i = \sum_j \left| E_j^M - E_j^P \right| \qquad (5)$$

Similarly, background subtraction was used to obtain human silhouette. Two pixel maps B^M and B^P are built and compared to compute the corresponding values of C^i_B. Finally, edges and blobs coefficients are combined to obtain *ith* particle weight using the following weighting function:

$$\pi^i = e^{-\alpha(C_E^i + C_B^i)} \qquad (6)$$

where α is an experimental parameter.

6 Results

To demonstrate the advantages of proposed PRPF algorithm, three different particle filter algorithms (Condensation, SIR and PRPF) were implemented using MATLAB 6.1. We tested the developed algorithms for tracking people who performed planar movements in different scenes. A Pentium 4 1.7 GHz. computer and a webcam were used for the experiments. Performance of Condensation using 4000 particles, SIR using 2000 particles (at each sampling stage) and PRPF using 200 particles in the particle set and four solutions in the *RefSet* were evaluated. Manual digitizing was performed to take a reference in order to evaluate the qualitative performance of the different algorithms. Different estimations of the *x* coordinate for the right wrist marker over a 50 frames video sequence using PRPF, SIR, Condensation and manual digitizing are shown in figure 6. The required number of evaluations of the weighting function related to each frame in the video sequence of figure 7 was 4000 for SIR particle filter algorithm and 2838 for the PRPF algorithm.

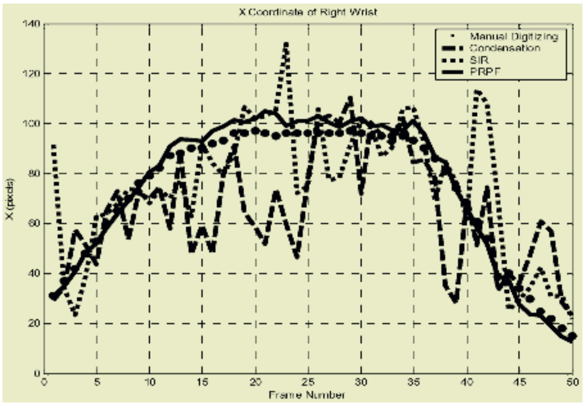


Fig. 6. (a) Estimation of the *x* coordinate of the right wrist using a manual procedure, the Condensation, SIR and PRPF algorithms.

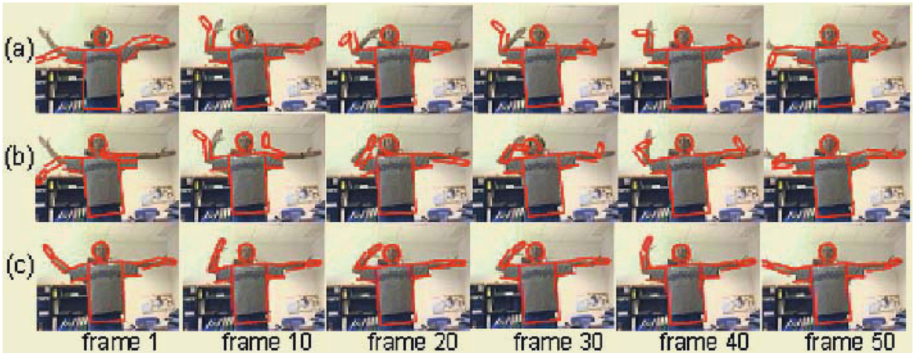


Fig. 7. Estimates in different frames using: (a) Condensation (b) SIR and (c) PRPF.

Table 1. MSE respect to manual digitizing of Condensation, SIR and PRPF for two sequences.

Marker	Image Sequence 1			Image Sequence 2		
	Condens	**SIR**	**PRPF**	**Condens**	**SIR**	**PRPF**
X Vertex	3.19 e+001	4.92 e+001	1.26e+001	1.17e+002	1.26e+002	7.65e+001
Y Vertex	6.49 e+001	9.52 e+001	1.20e+001	6.07e+001	6.47e+001	3.04e+001
X Neck	3.26 e+001	2.93 e+001	2.83e+001	2.23e+001	2.68e+001	1.16e+001
Y Neck	3.00 e+001	2.48 e+001	1.04e+001	2.78e+001	2.66e+001	1.19e+001
X R Shoulder	2.59 e+001	2.10 e+001	1.31e+001	2.15e+001	2.11e+001	7.10e+000
X L Wrist	3.12 e+002	1.69 e+001	5.71e+000	1.37e+003	1.19e+003	9.37e+001
Y L Wrist	8.29 e+002	4.54 e+002	8.69e+000	2.18e+003	9.64e+002	1.76e+001
Mean	152.15	107.52	15.31	425.69	341.62	37.38

Figure 7 shows the achieved experimental model configuration results using the Condensation, SIR and PRPF algorithms for the same video sequence. Table 1 shows the deviations from manual digitizing of the estimates using the considered algorithms. The Mean Square Error (MSE) deviation for the set of all markers averaged by the two sequences was 26.34 using PRPF, 224.57 using SIR and 288.92 using Condensation algorithm respectively. Note that MSE has decreased to 11.73% of the

SIR error and to 9.12% of the Condensation error. This is due to a drastic improvement of the *RefSet* quality after Path Relinking optimization.

7 Conclusion

The main contribution of this work is the Path Relinking Particle Filter (PRPF) algorithm, developed for estimation problems in sequential processes that are represented by the state-space model abstraction. Experimental results have shown that PRPF appreciably increases the performance of general and SIR particle filters. We have applied the proposed PRPF algorithm to the 2D human pose estimation problem. PRPF increases the accuracy for automatically obtaining the marker coordinates with a more reduced particle set in 2D biomechanical analysis. As future work the PRPF will be applied to perform the tracking of 3D images for human pose estimation in biomechanics applications. In addition, a study of PRPF properties oriented to reduce execution time of the proposed algorithm is necessary.

References

1. Wang, L., Weiming, H., Tieniu, T.: Recent developments in human motion analysis. Pattern Recognition 36 (2003) 585–601
2. Moeslund, B., Granum, E.: A Survey on Computer Vision-Based Human Motion Capture. Computer Vision and Image Understanding 81 (2001) 231–168
3. Gavrila, D: The visual analysis of human movement: a review. CVIU 73 1: (1999)
4. IBV: Biomechanics. http://www.ibv.org/ingles/ibv/index2.html (1992)
5. Deutscher, J., Blake, A., Reid, I.: Articulated body motion capture by annealed particle filtering. IEEE Conf. Computer Vision and Pattern Recognition, Vol. 2 (2000) 126–133
6. Aggarwal, J.K., Cai, Q., Liao, W., Sabata, B.: Articulated and elastic non-rigid motion: a review. IEEE Workshop on Motion of Non-Rigid and Articulated Objects (1994) 2–14
7. Ju, S., Black, M, Yaccob, Y.,: Cardboard people: a parameterized model of articulated image motion. IEEE Int. Conf. on Automatic Face and Gesture Recognition (1996) 38–44
8. Wachter, S., Nagel, H.-H.: Tracking persons in monocular image sequences. Computer Vision Image Understanding 74 Vol 3 (1999) 174–192
9. Zotkin, D., Duraiswami, R., Davis, L.: Joint Audio-Visual Tracking Using Particle Filters. EURASIP Journal on Applied Signal Processing, Vol, 11 (2002) 1154–1164
10. Glover, F., Laguna, M., Martí, R.: Scatter Search and Path Relinking: Foundations and Advances Designs. To appear in New Optimization techniques in Engineering (2003)
11. Glover, F.: A Template for Scatter Search and Path Relinking. LNCS, 1363, (1997) 1-53
12. Arulampalam, M., et al.: A Tutorial on Particle Filter for Online Nonlinear/Non-Gaussian Bayesian Tracking. IEEE Trans. On Signal Processing, V 50 (2): 174–188 (2002)

Clustering with Soft and Group Constraints*

Martin H.C. Law, Alexander Topchy, and Anil K. Jain

Department of Computer Science and Engineering
Michigan State University
East Lansing, MI, 48824, USA
{lawhiu,topchyal,jain}@cse.msu.edu

Abstract. Several clustering algorithms equipped with pairwise hard constraints between data points are known to improve the accuracy of clustering solutions. We develop a new clustering algorithm that extends mixture clustering in the presence of (i) soft constraints, and (ii) group-level constraints. Soft constraints can reflect the uncertainty associated with a priori knowledge about pairs of points that should or should not belong to the same cluster, while group-level constraints can capture larger building blocks of the target partition when afforded by the side information. Assuming that the data points are generated by a mixture of Gaussians, we derive the EM algorithm to estimate the parameters of different clusters. Empirical study demonstrates that the use of soft constraints results in superior data partitions normally unattainable without constraints. Further, the solutions are more robust when the hard constraints may be incorrect.

1 Introduction

Modern cluster analysis [1] is largely driven by the quest for scalable and more robust clustering algorithms capable of detecting clusters with diverse shapes and densities. Data clustering is an ill-posed problem when the associated objective function is not well defined, leading to fundamental limitations of generic clustering algorithms. Multiple clustering solutions may seem to be equally plausible due to an inherent arbitrariness in the notion of a cluster. Any side (auxiliary) information must be used in order to reduce this degeneracy of possible solutions and improve the quality of clustering.

Unlike supervised classification, only recently some attention has been given to the role of prior information in data clustering. Prior information can be available in several forms: labelled data, known data groupings or associations, additional inter-pattern similarity estimates, feature relevance, object ranks, etc. We are primarily interested in various inter-point constraints that can complement already known pattern or proximity matrix. For example, pairwise constraints on the data points tell us which pairs of points must be placed in the same cluster (positive constraint) or different clusters (negative constraint). Ideally,

* This work was supported by the U.S. ONR grant no. N000140410183.

the target partition must satisfy all the given data constraints. Hence, a clustering algorithm should be driven by attribute (feature) values as well as the constraint information, such that the clustering solution is biased in favor of the constraints.

Constraints are naturally available in many clustering applications. For instance, in image segmentation one can have partial grouping cues for several regions to assist in the overall clustering [2]. Clustering of customers in market-basket database can have multiple records pertaining to the same person. In video retrieval tasks different users may provide alternative annotations of images in small subsets of a large database [3]. Such groupings may be used for semi-supervised clustering of the entire database.

The prior knowledge was provided at the instance level in the form of positive (must-link) and negative (cannot-link) pairwise constraints in [4, 5]. A constrained k-means algorithm is proposed in [4]: must-link data points are replaced by their centroid, and a data point is assigned to the closest cluster center that does not violate any constraints. "Soft" constraints were introduced in the dissertation of Wagstaff [6], where a heuristic is employed to assign a point to a cluster that gets the lowest penalty for constraint violations. Similarly, the constrained version of COBWEB algorithm is considered in [5]. Constrained modification of the complete-link algorithm was proposed in [7]. Spectral clustering is modified in [8] to work with constraints. Again, a heuristic procedure augments the affinity matrix derived from feature space by the constraints. The EM algorithm for mixture model clustering with hard data constraints was developed in [9] and was shown to be superior to the constrained k-means algorithm [4]. Constraints were also incorporated into image segmentation algorithms using graph-based clustering in [10, 2]. Recently, Xing et al. [11] proposed a way to perform clustering by metric learning using side-information: metric can be learned from the constraints and then applied globally in the feature space to obtain the final clustering. Correlation clustering [12] uses only the positive and negative constraints to partition the vertices in a graph. It has been extended to cope with soft constraints [13, 14].

The main contribution of this paper is to adopt soft constraints in mixture (model-based) clustering. Each constraint becomes a real valued variable in between 0 and 1. The value of the constraint reflects the certainty of the prior knowledge that a pair of objects comes from the same cluster. Variable strength of the constraint allows for better control of clustering bias introduced by the constraints. Our main clustering algorithm is based on a generative model, where constraint variables are explicitly identified with the nodes in the corresponding Bayesian network. In this sense, we extend the work by Shental et al. [9] whose method included only the hard constraints into the mixture model clustering. To account for soft constraints, we use a more sophisticated graphical model, yet preserve linear complexity of the inference process (clustering) in the model. Coupled with mixture clustering, soft constraints are not strictly enforced but rather serve as prior values that can be changed by the observed data. Mutually conflicting "soft" constraints are allowed. Moreover, our model can operate with group constraints, namely we can specify the certainty of each of several points belonging to the same cluster (not in a pairwise manner).

2 Clustering with Constraints

Consider a data set $\mathcal{D} = \{\mathbf{y}_1, \dots, \mathbf{y}_N\}$ of size N. Let $z_i \in \{1, \dots, K\}$ be the (hidden) cluster label of point $\mathbf{y}_i$, where K is the number of clusters. Suppose we want to incorporate the constraint that the first three points $\mathbf{y}_1$, $\mathbf{y}_2$ and $\mathbf{y}_3$ are in the same group and should belong to the same cluster. This can be done by setting z_1, z_2 and z_3 to a common value w_1, $z_1 = w_1$, $z_2 = w_1$ and $z_3 = w_1$. Here, w_1 is an auxiliary random variable that serves as a "group-label"[1] – cluster label of the group $\{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3\}$. The likelihood function for the observed data $\mathcal{D}$ can be derived based on the group membership assumptions, and the EM algorithm is used for parameter estimation [9].

Note that the equalities $z_1 = w_1$ and $z_2 = w_1$ mean that this pairwise constraint is "hard", namely, the points $\mathbf{y}_1$ and $\mathbf{y}_2$ are certain to have the same cluster label. In general, this may not be true. Alternatively, we require $z_i = w_l$ to be true only with a probability γ_{il}. The value of $\gamma_{il} \in [0, 1]$ can be interpreted as the strength of the constraint that $\mathbf{y}_i$ belongs to the l-th group. To have a logically consistent framework, any $\mathbf{y}_i$ without any associated constraint information should be equivalent to $\gamma_{il} = 0$, $\forall l$. This ensures uniform treatment for data points with and without constraints. If $\gamma_{il} = 0$, z_i is chosen independently of the other group and cluster labels.

Formally, let α_j be the prior probability for the j-th mixture component $q_j(\mathbf{y}; \theta_j)$, which is parameterized by θ_j. For simplicity, we write $q_j(\mathbf{y}) = q_j(\mathbf{y}; \theta_j)$. Let w_l $(l = 1, \dots, L)$ be the set of (hidden) group-labels. Each group label can take a value from 1 to K. Let v_i be a discrete random variable that takes value in $\{0, \dots, L\}$ and determines how z_i is generated. If $v_i = l$, the point $\mathbf{y}_i$ participates in the l-th group and thus $z_i = w_l$ and $P(v_i = l) = \gamma_{il}$. When $v_i = 0$, label z_i is generated independently according to the prior probabilities $\{\alpha_j\}$. Hence, the model for $\mathbf{y}_i$ is specified as follows:

$$P(w_l = j) = \alpha_j, \quad l \in \{1, \dots, L\}, j \in \{1, \dots, K\} \tag{1}$$

$$P(v_i = l) = \gamma_{il}, \quad i \in \{1, \dots, N\}, l \in \{1, \dots, L\} \tag{2}$$

$$P(z_i = j | v_i, w_1, \dots, w_L) = \begin{cases} \alpha_j & \text{if } v_i = 0 \\ \delta_{w_l, j} & \text{if } v_i = l \end{cases} \tag{3}$$

$$p(\mathbf{y} | z = j) = q_j(\mathbf{y}), \tag{4}$$

where δ_{ij} is the Kronecker delta. An example of such model with seven data points and three group labels is shown in Figure 1. Typically, when a user specifies γ_{il}, many of them are set to zero. It means that the label z_i is only tied to a small number of group-labels. The case when z_i has more than one group-label corresponds to the existence of possibly incompatible constraint, because z_i can

[1] It corresponds to the term "chunklet" defined in [9].

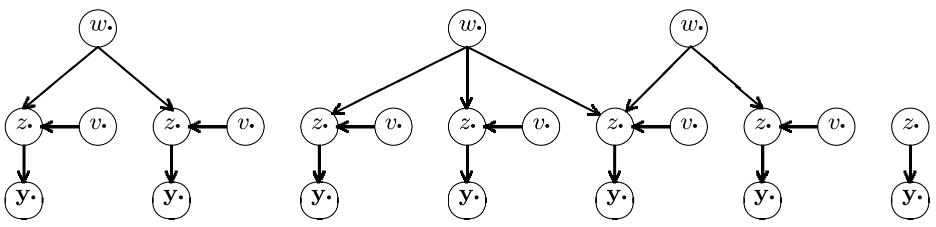


Fig. 1. An example of the graphical model for the proposed soft constraint. There are seven data points $\{\mathbf{y}_\cdot, \mathbf{y}_\cdot, \ldots, \mathbf{y}_\cdot\}$ with three group-labels $\{w_\cdot, w_\cdot, w_\cdot\}$. There are competing constraints for $z_\cdot$. Note that each connected component in the graph is a polytree and hence the belief propagation algorithm can be used. The number of clusters, K, determine the possible values that w_l and z_i can assume.

belong to more than one group. This probabilistic model can be given generative interpretation (Figure 1). First, the L group-labels $\{w_l\}$ are generated according to the component prior probabilities $\{\alpha_j\}$. For each $i \in \{1, \ldots, N\}$, we generate v_i with the probabilities $\{\gamma_{il}\}$. The outcome determines how z_i gets its value: if v_i is between 1 and L, z_i is set to w_l; otherwise, z_i is generated independently according to $\{\alpha_j\}$. Based on the value of z_i, the point $\mathbf{y}_i$ is generated from $q_{z_i}(\mathbf{y})$.

2.1 Constraints Specification

One important advantage of adopting soft constraints is its robustness. It is usually difficult to obtain definitive statements on the properties of patterns in real world applications. Thus a practical clustering algorithm using constraint information should tolerate noisy constraints. However, a single erroneous "cannot-link" constraint can break down the constrained k-means algorithm in [4]. The dissimilarity values between multiple items can be drastically altered by a single bad constraint. In our proposed approach, $z_i = w_l$ is required to be true only with a certain probability. This flexibility protects us from disastrous clustering solution when some constraints may be wrong. Although the algorithm in [9] also does not break down in view of erroneous constraints, later in section 3 we shall demonstrate that the use of soft constraints in the proposed algorithm can lead to superior results when the constraints are noisy.

Different constraints are specified by assigning different values to γ_{il}, which in turn specifies the topology of the graphical model by the sparsity of the matrix $\{\gamma_{il}\}$. Note that we have made the abstraction that the group-label may not correspond to the label of any particular data point. However, it is easy to enforce the group-label w_l to be the same as the cluster label z_i by setting $\gamma_{il} = 1$. Equivalence constraint information between $\mathbf{y}_i$ and $\mathbf{y}_j$ can then be incorporated by setting γ_{jl} to be the confidence that they are in the same cluster. The abstraction of group-label is useful in the distributed learning scenario described in [9]. Different teachers are asked to assign group labels to different subsets of the data. Further, suppose the teachers also provide confidence values in their assignment. Let w_l correspond to a group labelled by a certain teacher. The confidence that $\mathbf{y}_i$ belongs to the l-th group is represented by γ_{il}.

2.2 Parameter Estimation

The model parameters $\{\alpha_j\}$ and $\{\theta_j\}$ can be estimated by maximizing the data log-likelihood function. Note that γ_{il} values are provided by the user and do not need to be estimated. Since $\{z_i\}$, $\{v_i\}$ and $\{w_l\}$ are hidden variables, this is a missing data problem and the EM algorithm can be used. We refer the readers to texts like [15] for more details on the EM algorithm. The complete data log-likelihood can be written as

$$\begin{aligned}\mathcal{L} &= \log p(\{\mathbf{y}_i\},\{z_i\},\{w_l\},\{v_i\}) \\ &= \begin{cases} -\infty & \exists v_i \neq 0 : z_i \neq w_{v_i} \\ \sum_{i=1}^{N}\left(\log q_{z_i}(\mathbf{y}_i) + \log \gamma_{v_i,i} + \delta_{v_i,0}\log\alpha_{z_i}\right) + \sum_{l=1}^{L}\log\alpha_{w_l} & \text{otherwise}\end{cases}\end{aligned} \tag{5}$$

The data is said to be inconsistent (have zero probability) if there exists $v_i \neq 0$ such that $z_i \neq w_{v_i}$. Let $\boldsymbol{\theta}$ denote the current parameter estimate. Taking expectation of $\mathcal{L}$ with respect to the missing data, given $\boldsymbol{\theta}$ and $\mathcal{D}$, we obtain

$$\begin{aligned}&E[\log p(\{\mathbf{y}_i\},\{z_i\},\{w_l\},\{v_i\})] \\ &= \sum_{i=1}^{N}\sum_{j=1}^{K} P(z_i = j|\{\mathbf{y}_i\})\log q_j(\mathbf{y}_i) + \sum_{l=1}^{L}\sum_{j=1}^{K} P(w_l = j|\{\mathbf{y}_i\})\log\alpha_j + \\ &\quad \sum_{i=i}^{N}\sum_{l=0}^{L} P(v_i = l|\{\mathbf{y}_i\})\log\gamma_{li} + \sum_{i=1}^{N}\sum_{j=1}^{K} P(v_i = 0, z_i = j|\{\mathbf{y}_i\})\log\alpha_j\end{aligned} \tag{6}$$

Note that different $\mathbf{y}_i$'s may not be independent because they can be related indirectly by a common w_l. Also, the inconsistency of hidden data does not depend on the parameter values. The expected value of $\mathcal{L}$ is computed over only the set of consistent values of hidden variables and hence no infinite values are encountered. The expected complete data log-likelihood can be maximized with respect to the parameters $\{\alpha_j, \theta_j\}$ by

$$\hat{\alpha_j} = \frac{\sum_{l=1}^{L} P(w_l = j|\{\mathbf{y}_i\}) + \sum_{i=1}^{N} P(v_i = 0, z_i = j|\{\mathbf{y}_i\})}{\sum_{j=1}^{K}\left(\sum_{l=1}^{L} P(w_l = j|\{\mathbf{y}_i\}) + \sum_{i=1}^{N} P(v_i = 0, z_i = j|\{\mathbf{y}_i\})\right)} \tag{7}$$

$$\hat{\boldsymbol{\mu}}_j = \frac{\sum_{i=1}^{N} P(z_i = j|\{\mathbf{y}_i\})\mathbf{y}_i}{\sum_{i=1}^{N} P(z_i = j|\{\mathbf{y}_i\})} \tag{8}$$

$$\hat{\mathbf{C}}_j = \frac{\sum_{i=1}^{N} P(z_i = j|\{\mathbf{y}_i\})(\mathbf{y}_i - \hat{\boldsymbol{\mu}}_j)(\mathbf{y}_i - \hat{\boldsymbol{\mu}}_j)^T}{\sum_{i=1}^{N} P(z_i = j|\{\mathbf{y}_i\})}, \tag{9}$$

where the j-th component is assumed to be a Gaussian with mean $\boldsymbol{\mu}_j$ and covariance $\mathbf{C}_j$. The parameter update in Equations (7) to (9) corresponds to the M-step of the EM algorithm. The E-step consists of the computation of the probabilities $P(w_l{=}j|\{\mathbf{y}_i\})$, $P(z_i{=}j|\{\mathbf{y}_i\})$ and $P(v_i{=}0, z_i{=}j|\{\mathbf{y}_i\})$. Unlike the standard Gaussian mixture, it is not easy to express these probabilities by simple equations because of the interdependence of $\{\mathbf{y}_i\}$ via w_l. Instead, these probabilities can be computed by standard Bayesian network inference algorithms like belief propagation or junction tree. The two-variable query $P(v_i{=}0, z_i{=}j|\{\mathbf{y}_i\})$ can be easily handled since the node v_i is a parent of z_i. Because of the simplicity of the structure of the graphical model, inference can be carried out efficiently. In particular, the complexity is virtually the same as the standard EM algorithm when there are no competing constraints for all the data points. This is the most usual scenario in constraint clustering.

3 Experiments

3.1 Synthetic Data

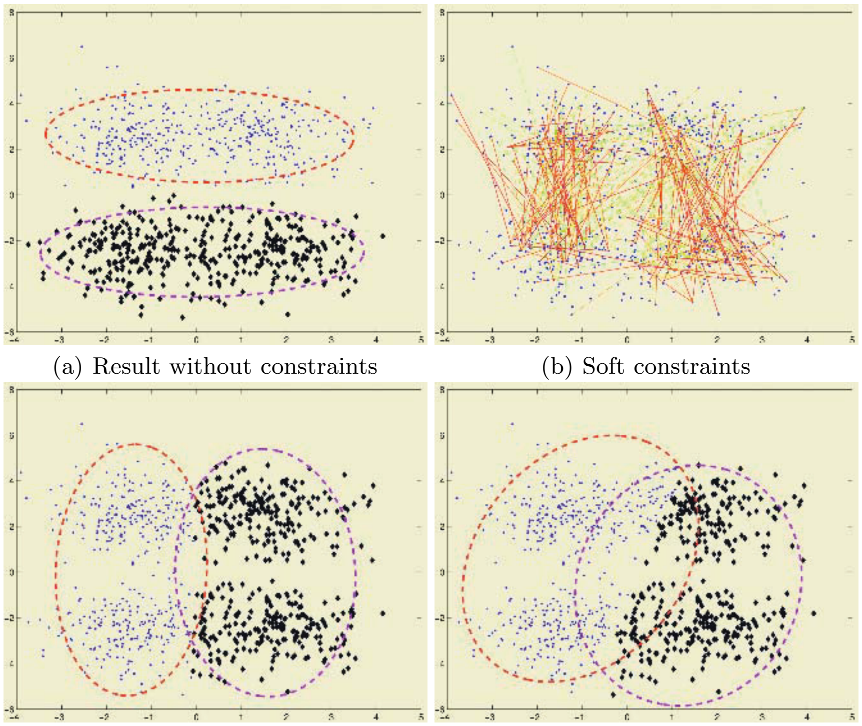
In the first experiment, we investigate how constraint information can be used to bias the search for the appropriate clusters. Four 2D Gaussian distributions with mean vectors $\left[\begin{smallmatrix}1.5\\2.5\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}-1.5\\2.5\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}-1.5\\-2.5\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1.5\\-2.5\end{smallmatrix}\right]$, and identity covariance matrix are considered (Figure 2). 200 data points are generated from each of the four Gaussians. The number of target clusters (K) is two. The two natural clusters are recovered by the EM algorithm without any constraint (Figure 2(a)). Ten multiple random restarts are used to avoid poor local minima.

Now suppose that prior information favors two vertical clusters instead of the more natural horizontal clusters. This prior information can be incorporated by constraining a data point in the leftmost (rightmost) top cluster to belong to the same cluster as a data point on the leftmost (rightmost) bottom cluster. We select 50 points randomly ($L = 50$) and link them to seven different points. To create more realistic constraints, a link can be absent with a probability of 0.05. The strength of the constraint is randomly drawn from the interval [0.6,1]. To demonstrate the importance of soft constraints, the constraints are corrupted with some noise: a data point is connected to a randomly chosen point with probability one minus the constraint strength. An example of the constraints is shown in Figure 2(b).

The proposed algorithm is run using the specified soft constraints and the obtained clustering solution is shown in Figure 2(c). The constraint information indeed helps to detect the preferred cluster structure, instead of natural clusters in Figure 2(a) when the constraints are absent. The soft constraints can be converted to hard constraints by changing all nonzero γ_{li} to 1. In this case, the proposed algorithm becomes equivalent to the algorithm in [9] for positive constraints. The result of using hard constraints is shown in Figure 2(d). While the estimated cluster structure is close to what we seek, the noise in the constraints notably distorts the detected clusters. This confirms that the use of soft constraints can significantly improve the robustness of mixture model clustering with hard constraints.

3.2 Real World Data Set

In the second experiment, we investigate how constraints can assist in obtaining superior cluster boundaries. Two data sets from the UCI machine learning



(a) Result without constraints (b) Soft constraints

(c) Result of soft constraints (d) Result of hard constraints

Fig. 2. The results of soft constraint clustering on a synthetic data set of 800 points. The ellipses represent the estimated Gaussian components. The solid lines in (b) correspond to the "strong" constraints, while the dotted lines correspond to the "weak" constraints.

repository are considered. The Iris data set (`iris`) has 150 points in 4D from 3 classes. The wine recognition data set[2] (`wine`) has 178 points with 13 features from 3 classes. For each data set, half of the points are used for training (learning the clusters), and the rest for testing (comparing the clusters obtained with the ground truth). A Gaussian is fit to each of the classes and the ambiguous data points (5% of the total number of data) are identified by examining the class posterior probabilities and the true class labels. The ambiguous data points are then constrained to be in the same cluster as the points near the center of the class. Examples of ambiguous data points are shown in Figure 3. For each data set, we randomly split them into two parts. As in the previous experiment, the strength of the constraint is drawn randomly from [0.6,1], and the constraint is additionally corrupted by noise. The clusters obtained (with soft constraints) are used to "classify" the other half of the data points. The experiment is re-

[2] The variables of `wine` are standardized to have means equal to zero and variances equal to one.

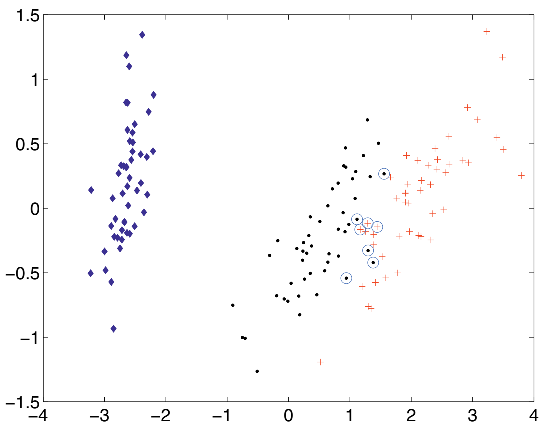


Fig. 3. The iris data set projected to the first two principal components. The eight ambiguous data points are circled.

peated 20 times. The error rates of “no constraint’, “soft constraint” and “hard constraint” are 6.7%, 2.7% and 8% for `iris`, and 5.6%, 3.4% and 3.4% for `wine`. For both the data sets, soft constraints yield clusters that are at least as good as clusters obtained by hard constraints 19 times out of 20, with 8 ties for `iris` and 6 ties for `wine`. Soft constraints also give better clusters than no constraints (18 out of 20 for `iris` and 19 out of 20 for `wine`), with 6 ties for `iris` and 7 ties for `wine`. Soft constraint information indeed helps to identify the target clusters more accurately and tolerates potentially erroneous constraints.

4 Conclusion and Future Work

We have proposed a new EM algorithm for clustering in the presence of soft constraints. Experimental results demonstrate that the proposed approach is promising and can be superior to hard constraints in the presence of noise. One notable property of the proposed approach is its efficiency. Despite the apparent increase in the complexity of the model, no additional parameters need to be estimated when compared with a standard mixture of Gaussians. Also, the inference procedure is of similar complexity as the standard EM algorithm when each data point is associated with few group-labels. One limitation of the proposed algorithm is that it does not deal with the negative constraints. In principle, the graphical model can be extended in a manner similar to [9] to include the negative constraints. We choose not to do so, however, for two reasons. First, the addition of negative constraints results in only a slight improvement as reported in [9]. Secondly, the presence of negative constraints can increase the complexity of the graphical model and hence increase the inference complexity.

There are several directions for future work. The total strength of constraint information is currently determined by the number of constrained data points.

This can be undesirable as a large amount of data can dilute the constraint information. This relates to the fundamental issue of how to appropriately weight the information contained in the data and the constraints. One possibility is to include an additional penalty term in the likelihood function that balances the posterior probabilities of cluster labels with the constraints. The number of cluster, K, is assumed to be given. Since we are using a mixture model, the idea of minimum message length described in [16] can be adopted to the current algorithm to estimate K.

References

1. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall (1988)
2. Yu, S.X., Shi, J.: Segmentation given partial grouping constraints. IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004) 173–183
3. Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning via equivalence constraints, with applications to the enhancement of image and video retrieval. In: Proc. IEEE Confernce on Computer Vision and Pattern Recognition. (2003)
4. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: Proc. International Conference on Machine Learning. (2001) 577–584
5. Wagstaff, K., Cardie, C.: Clustering with instance-level constraints. In: Proc. International Conference on Machine Learning. (2000) 1103–1110
6. Wagstaff, K.: Intelligent Clustering with Instance-Level Constraints. PhD thesis, Department of Computer Science, Cornell University (2002)
7. Klein, D., Kamvar, S.D., Manning, C.D.: From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In: Proc. International Conference on Machine Learning. (2002) 307–314
8. Kamvar, S., Klein, D., Manning, C.D.: Spectral learning. In: Proc. of the Eighteenth International Joint Conference on Artificial Intelligence, MIT Press (2003)
9. Shental, N., Bar-Hillel, A., Hertz, T., Weinshall, D.: Computing gaussian mixture models with EM using equivalence constraints. In: Advances in Neural Information Processing Systems 16, MIT Press (2004)
10. Yu, S.X., Shi, J.: Grouping with bias. In: Advances in Neural Information Processing Systems 13, MIT Press (2001)
11. Xing, E.P., Ng, A.Y., Jordan, M.I., Russell, S.: Distance metric learning, with application to clustering with side-information. In: Advances in Neural Information Processing Systems 15, Cambridge, MA, MIT Press (2003)
12. Bansal, N., Blum, A., Chawla, S.: Correlation clustering. In: Proc. of the 43d Annual IEEE Symp. on Foundations of Computer Science. (2002)
13. Charikar, M., Guruswami, V., Wirth, A.: Clustering with qualitative information. In: Proc. of the 44th Annual IEEE Symposium on Foundations of Computer Science. (2003)
14. Demaine, E.D., Immorlica, N.: Correlation clustering with partial information. In: Proc. of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, Princeton, New Jersey (2003)
15. McLachlan, G., Peel, D.: Finite Mixture Models. John Wiley & Sons, New York (2000)
16. Figueiredo, M.A.T., Jain, A.K.: Unsupervised learning of finite mixture models. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 381–396

Learning from General Label Constraints

Tijl De Bie, Johan Suykens, and Bart De Moor

Katholieke Universiteit Leuven, ESAT-SCD
Kasteelpark Arenberg 10, 3001 Leuven, Belgium
{tijl.debie,johan.suykens,bart.demoor}@esat.kuleuven.ac.be
www.esat.kuleuven.ac.be/sista-cosic-docarch

Abstract. Most machine learning algorithms are designed either for supervised or for unsupervised learning, notably classification and clustering. Practical problems in bioinformatics and in vision however show that this setting often is an oversimplification of reality. While label information is of course invaluable in most cases, it would be a huge waste to ignore the information on the cluster structure that is present in an (often much larger) unlabeled sample set. Several recent contributions deal with this topic: given partially labeled data, exploit all information available. In this paper, we present an elegant and efficient algorithm that allows to deal with very general types of label constraints in class learning problems. The approach is based on spectral clustering, and leads to an efficient algorithm based on the simple eigenvalue problem.

1 Introduction

We address the clustering problem where general information on the class labels y_i of some of the samples $\mathbf{x}_i$ ($i = 1, \ldots n$) is given. This problem of taking general label information into account has received increasing attention in recent literature, and is know under different names as side information learning, semi-supervised learning, transductive learning (in a more restrictive setting), learning from (in)equivalence constraints, and more.

Roughly two ways of addressing the problem can be distinguished. Some of the methods try to learn a metric that is in accordance with the side information given, after which a standard clustering method can be applied, such as in [7, 11, 2]. Other methods propose actual adaptations of algorithms that were originally designed for clustering or for classification, such as in [9, 12, 3, 4, 1, 6, 10, 13]. These adaptations make it possible to take general types of label information into account. In this paper, we describe a novel fast, principled and highly general method that belongs to the second category of algorithms.

The label information to be dealt with can be of two general forms: in the first setting subsets of samples are given for which is specified that they belong to the same class; in the second setting, similarly subsets of samples with the same label are given, but now additionally, for some pairs of such subsets, it is given that they contain samples that do not belong to the same class. Note that the standard transduction setting, where part of the samples is labeled, is in fact a special case of this type of label information.

In this paper, we present an elegant way to handle the first type of label information in both the two class and the multi class learning settings. Furthermore, we show how the second type of label information can be dealt with in full generality for the two class case.

In a first section we will review spectral clustering as a relaxation of a combinatorial problem. In the second section, we will show how to enforce the constraints to the spectral clustering method, first for the two class case, and subsequently for the multi class case. Then, without going into detail, we will point out how the constraints can be imposed in a soft way as well. Finally, empirical results are reported and compared to a recently proposed approach [4] that is able to deal with similar settings and has comparable computational cost.

General Notation: $\mathbf{1}$ is a column vector containing all ones, sometimes its size n is indicated as a subscript: $\mathbf{1}_n$. The identity matrix is denoted by $\mathbf{I}$. A transpose will be denoted by a prime $'$. The matrix containing all zeros is denoted by $\mathbf{0}$. Matrices are denoted by upper case bold face, vectors by lower case bold face, and scalars by standard lower case symbols.

2 Spectral Clustering

Spectral clustering methods can best be seen as relaxations of graph cut problems, as clearly presented in [8]. Below a detailed derivation will be discussed only for the two class setting.

2.1 Two Class Clustering

Consider a weighted graph over the nodes each representing a sample $\mathbf{x}_i$. The edge weights correspond to some similarity measure to be defined in an appropriate way. These similarities can be arranged in a symmetric *affinity* matrix $\mathbf{K}$: its entry at row i and column j, denoted by K_{ij}, represents the similarity between sample $\mathbf{x}_i$ and $\mathbf{x}_j$ [1].

A graph cut algorithm searches for a partition of the nodes in two sets (corresponding clusters of the samples $\mathbf{x}_i$) such that a certain cost function is minimized. Several cost functions are proposed in literature, among which the *average cut cost* and the *normalized cut cost* are best known and most widely used.

For the normalized cut cost, the discrete optimization problem can be written in the form (see e.g. [8]):

$$\min_{\mathbf{y}} \frac{\mathbf{y}'(\mathbf{D}-\mathbf{K})\mathbf{y}}{\mathbf{y}'\mathbf{D}\mathbf{y}} \quad (1)$$

$$\text{s.t.} \quad \mathbf{1}'\mathbf{D}\mathbf{y} = 0 \quad (2)$$

$$y_i \in \{y_+, y_-\} \quad (3)$$

[1] In many practical cases this similarity will be given by a kernel function $k(\mathbf{x}_i, \mathbf{x}_j) = K_{ij}$ in which case $\mathbf{K}$ is semi positive definite, often named the kernel matrix.

where y_+ and y_- are the two possible values the y_i take depending on the class $\mathbf{x}_i$ is assigned to, and $\mathbf{D} = \text{diag}(\mathbf{K1})$ is a diagonal matrix containing all row sums d_i of $\mathbf{K}$ as its diagonal entries. The matrix $\mathbf{D} - \mathbf{K}$ is generally known as the Laplacian of the graph associated with $\mathbf{K}$. Note that the Laplacian is always semi positive definite.

It is constraint (3) that causes this problem to be combinatorial. However, the relaxed problem obtained by dropping this constraint can be solved very easily as we will show now. Furthermore, using the resulting vector $\mathbf{y}$ as an approximation has been observed to be very effective in practical problems. Note that since the scale of $\mathbf{y}$ in fact does not matter, we can as well solve

$$\begin{aligned} \min_{\mathbf{y}} \ & \mathbf{y}'(\mathbf{D}-\mathbf{K})\mathbf{y} \\ \text{s.t.} \ & \mathbf{y}'\mathbf{D}\mathbf{y} = 1 \\ & \mathbf{1}'\mathbf{D}\mathbf{y} = 0 \end{aligned} \tag{4}$$

If we would drop constraint (4), the minimization becomes equivalent to solving for the minimal eigenvalue of

$$(\mathbf{D}-\mathbf{K})\mathbf{y} = \lambda\mathbf{D}\mathbf{y} \tag{5}$$

or, after left multiplication with $\mathbf{D}^{-1/2}$

$$\mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{K})\mathbf{D}^{-1/2}\mathbf{v} = \lambda\mathbf{v} \tag{6}$$
$$\text{with} \qquad \mathbf{v} = \mathbf{D}^{1/2}\mathbf{y}.$$

Now note that this is an ordinary symmetric eigenvalue problem, of which the eigenvectors are orthogonal. Since the Laplacian and thus also $\mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{K})\mathbf{D}^{-1/2}$ is always semi positive definite, none of its eigenvalues can be smaller than 0. We can see immediately that a 0 eigenvalue is achieved by the eigenvector $\mathbf{v}_0 = \mathbf{D}^{1/2}\mathbf{1}$. This means that all other eigenvectors $\mathbf{v}_i$ of $\mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{K})\mathbf{D}^{-1/2}$ are orthogonal to $\mathbf{v}_0$, such that for all other eigenvectors $\mathbf{v}_i$ and thus for $\mathbf{y}_i = \mathbf{D}^{-1/2}\mathbf{v}_i$, we have that $\mathbf{v}_0'\mathbf{v}_i = \mathbf{1}'\mathbf{D}\mathbf{y}_i = 0$. It thus follows that constraint (4) is automatically taken into account by simply solving for the *second* smallest eigenvalue of (6) or (5). This is the final version of the spectral clustering method as a relaxation of the normalized cut problem[2].

2.2 Multi-class Clustering

In the k-class case, one usually extracts the eigenvectors $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{k-1}$ corresponding to the smallest $k-1$ eigenvalues (excluding the 0 eigenvalue). Then, these vectors are put next to each other in a matrix $\mathbf{Y} = \begin{pmatrix}\mathbf{y}_1 \ \mathbf{y}_2 \cdots \mathbf{y}_{k-1}\end{pmatrix}$, and subsequently any clustering algorithm can be applied to the rows of this matrix[3]. Every sample $\mathbf{x}_i$ is then assigned a label corresponding to which cluster row i of $\mathbf{Y}$ is assigned to.

[2] We can follow a similar derivation for the average cut cost function, ultimately leading to solving for the second smallest eigenvalue of $(\mathbf{D}-\mathbf{K})\mathbf{y} = \lambda\mathbf{y}$. All results presented in this paper can immediately be transferred to the average cut variant of spectral clustering.

[3] In [5] it is suggested to first normalize the rows of $\mathbf{Y}$ before performing the clustering.

3 Constrained Spectral Clustering

The results in this paper derive from the observations that

- constraining the labels according to the information as specified in the introduction can be seen as constraining the label vector $\mathbf{y}$ to some subspace;
- it is easy, in principle and computationally, to constrain the vector $\mathbf{y}$ to this subspace, while optimizing the Rayleigh quotient (1) subject to (2).

We will first tackle the two class learning problem subject to general label equality and inequality constraints. Afterwards, we show how equality constraints can be handled in the multi class setting.

3.1 Two Class Learning

Consider again the unrelaxed graph cut problem (1),(2),(3). We would now like to solve it with respect to the label information as additional constraints. For this we introduce the label constraint matrix $\mathbf{L} \in \{0,1\}^{n\times m}$ (with $n \geq m$) associated with the label equality and inequality constraints:

$$\mathbf{L} = \begin{pmatrix} \mathbf{1}_{s_1} & \mathbf{1}_{s_1} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_2} & -\mathbf{1}_{s_2} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_3} & \mathbf{0} & \mathbf{1}_{s_3} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_4} & \mathbf{0} & -\mathbf{1}_{s_4} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{1}_{s_{2p-1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_{2p-1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p}} & \mathbf{0} & \mathbf{0} & \cdots & -\mathbf{1}_{s_{2p}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p+1}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{1}_{s_{2p+1}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{1}_{s_{2p+2}} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{1}_{s_{2p+2}} & \cdots & \mathbf{0} \\ \cdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \mathbf{1}_{s_c} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_c} \end{pmatrix}.$$

Hereby, every row i of $\mathbf{L}$ corresponds to sample $\mathbf{x}_i$, in such a way that samples corresponding to one block row of size s_k are given to belong to the same class (i.e., for ease of notation the samples are sorted accordingly; of course also the rows of $\mathbf{K}$ need to be sorted in the same way). On the other hand, inequality constraints are encoded by the first $2p$ block rows: for all $k \leq p$, samples from block row k are given to belong to a different class as samples from block row $k+1$. For the last $c - 2p$ blocks no inequality constraints are given. Note that in most practical cases, many block row heights s_k will be equal to 1, indicating that no constraint for the corresponding sample is given.

Using the label constraint matrix $\mathbf{L}$, it is possible to impose the label constraints explicitly, by introducing an auxiliary vector $\mathbf{z}$ and equating

$$\mathbf{y} = \mathbf{L}\mathbf{z}.$$

Then again constraint (3) is dropped, leading to

$$\min_{\mathbf{z}} \frac{\mathbf{z}'\mathbf{L}'(\mathbf{D}-\mathbf{K})\mathbf{L}\mathbf{z}}{\mathbf{z}'\mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z}} \quad \text{s.t.} \quad \mathbf{1}'\mathbf{D}\mathbf{L}\mathbf{z} = 0$$

or equivalently

$$\begin{aligned} \min\nolimits_{\mathbf{z}} \ & \mathbf{z}'\mathbf{L}'(\mathbf{D}-\mathbf{K})\mathbf{L}\mathbf{z} \\ \text{s.t.} \ & \mathbf{z}'\mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z} \\ & \mathbf{1}'\mathbf{D}\mathbf{L}\mathbf{z} = 0 \end{aligned} \tag{7}$$

Note that (similarly as in the derivation on standard spectral clustering above) after dropping the constraint (7), we would only have to solve the following eigenvalue problem

$$\mathbf{L}'(\mathbf{D}-\mathbf{K})\mathbf{L}\mathbf{z} = \lambda\mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z} \tag{8}$$

or, by left multiplication with $(\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}$ and an appropriate substitution:

$$(\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}[\mathbf{L}'(\mathbf{D}-\mathbf{K})\mathbf{L}](\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}\mathbf{v} = \lambda\mathbf{v} \tag{9}$$
$$\text{with} \qquad \mathbf{v} = (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2}\mathbf{z}$$

Again, one can see that the extra constraint is taken into account automatically by picking the *second* smallest eigenvalue and associated eigenvector of this eigenvalue problem. To this end, note that $(\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}[\mathbf{L}'(\mathbf{D}-\mathbf{K})\mathbf{L}](\mathbf{L}'\mathbf{D}\mathbf{L})^{-1/2}$ is semi positive definite, such that its smallest eigenvalue is larger than or equal to 0. Now, note that the 0 eigenvalue is actually achieved for[4]

$$\mathbf{v}_0 = (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2} \cdot \begin{pmatrix} 1 \ 0 \cdots 0 \end{pmatrix}'.$$

Thus, since (9) is an ordinary symmetric eigenvalue problem, all other eigenvectors $\mathbf{v}_i$ have to be orthogonal: $\mathbf{v}_0'\mathbf{v}_i = 0$. This means that

$$\begin{pmatrix} 1 \ 0 \cdots 0 \end{pmatrix} (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2} \cdot (\mathbf{L}'\mathbf{D}\mathbf{L})^{1/2}\mathbf{z} = 0$$

and thus $\mathbf{1}'\mathbf{D}\mathbf{L}\mathbf{z} = 0$.

As a result, it suffices to solve (9) or equivalently (8) for its second smallest eigenvalue, and constraint (7) is taken into account automatically.

In summary, the procedure is as follows

– Compute the affinity matrix $\mathbf{K}$ and the matrix $\mathbf{D} = \mathbf{K}\mathbf{1}$.
– Compute the label constraint matrix $\mathbf{L}$.
– Compute the eigenvector $\mathbf{z}$ corresponding to the second smallest eigenvalue of the eigenvalue problem $\mathbf{L}'(\mathbf{D}-\mathbf{K})\mathbf{L}\mathbf{z} = \lambda\mathbf{L}'\mathbf{D}\mathbf{L}\mathbf{z}$.
– Compute the resulting relaxed label vector as $\mathbf{y} = \mathbf{L}\mathbf{z}$.

[4] To see this note that $\mathbf{L} \cdot \begin{pmatrix} 1 \ 0 \cdots 0 \end{pmatrix}' = \mathbf{1}$.

All operations can be carried out very efficiently, thanks to the sparsity of $\mathbf{L}$. The most expensive step is the eigenvalue problem, which is even smaller than in the unconstrained spectral clustering algorithm: the size of the matrices is only $m \times m$ instead of $n \times n$ (where m is the number of columns of $\mathbf{L}$).

As a last remark in this section, note that we are actually not interested in the component of $\mathbf{y}$ along $\mathbf{1}$. Thus, we could choose to take $\widetilde{\mathbf{y}} = \mathbf{L} \cdot \begin{pmatrix} 0 \; z_2 \cdots z_m \end{pmatrix}'$ as an estimate for the labels, instead of $\mathbf{y} = \mathbf{Lz}$. This results in the fact that estimates for labels $\widetilde{y}_i$ that were specified to be different are actually *opposite in sign*. Therefore thresholding the vector $\widetilde{\mathbf{y}}$ around 0 in fact makes more sense than thresholding $\mathbf{y}$ around 0.

3.2 Multi-class Learning

In the multi class setting, it is not possible anymore to include label inequality constraints in the same straightforward elegant way. The reason is that the true values of the labels can not be made equal to 1 and -1 anymore.

We can still take the equality constraints into account however. This means we would use a label constraint matrix of the form

$$\mathbf{L} = \begin{pmatrix} \mathbf{1}_{s_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{1}_{s_2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}_{s_c} \end{pmatrix},$$

constructed in a similar way. Note that this time we don't need a column containing all ones, as such vector $\mathbf{1}$ is included in its column space already.

As in the unconstrained spectral clustering algorithm, often $k-1$ eigenvectors will be calculated when k clusters are expected, leading to $\mathbf{Y} = \begin{pmatrix} \mathbf{y}_1 \; \mathbf{y}_2 \cdots \mathbf{y}_{k-1} \end{pmatrix}$. Finally, the clustering of $\mathbf{x}_i$ is obtained by clustering the rows of $\mathbf{Y}$.

4 Softly Constrained Spectral Clustering

In both the two class case and the multi class case, the constraints could be imposed in a soft way as well. This can be done by adding a cost term to the cost function that penalizes the distance between the weight vector and the column space of $\mathbf{L}$, in the following way (we give it without derivation or empirical results due to space restrictions):

$$\begin{array}{l} \min_{\mathbf{y}} \; \gamma \mathbf{y}'(\mathbf{D} - \mathbf{K})\mathbf{y} + (1-\gamma)\mathbf{y}'(\mathbf{D} - \mathbf{DL}(\mathbf{L}'\mathbf{DL})^{-1}\mathbf{L}'\mathbf{D})\mathbf{y} \\ \text{ s.t. } \;\; \mathbf{y}'\mathbf{Dy} = 1 \;\text{ and }\; \mathbf{1}'\mathbf{Dy} = 0 \end{array}$$

where γ is called the regularization parameter. Again the same reasoning can be applied, leading to the conclusion that one needs to solve for the second smallest eigenvalue (i.e. the smallest eigenvalue different from 0) of the eigenvalue problem:

$$\left[\gamma(\mathbf{D} - \mathbf{K}) + (1-\gamma)(\mathbf{D} - \mathbf{DL}(\mathbf{L}'\mathbf{DL})^{-1}\mathbf{L}'\mathbf{D})\right] \mathbf{y} = \lambda \mathbf{Dy} \qquad (10)$$

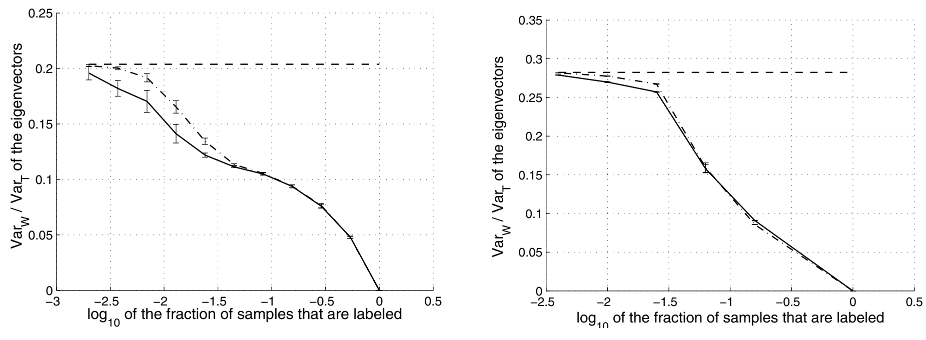


Fig. 1. The cost (within class variance divided by total variance of the eigenvectors) for spectral learning [4] in dash-dotted line, as compared with our method in full line as a function of the fraction of labeled samples (on a log scale), on the left for the three newsgroup dataset, on the right for the 10-class USPS dataset. For reference, the unconstrained cost is plotted as a horizontal dashed line.

For γ close to 0, the side-information is enforced very strongly, and $\mathbf{y}$ will satisfy the constraints nearly exactly. In the limit for $\gamma \to 0$, the soft constraints become hard constraints.

5 Empirical Results

We report experiments for the transduction setting, and this for the three newsgroup dataset (2995 samples) as also used in [4], and for the training subset of the USPS dataset (7291 samples). Comparisons are shown with the method proposed in [4]. This method works in similar settings and has a similar computational cost. We construct the affinity matrix in the same way as in that paper, namely by equating the entries $\mathbf{K}_{ij}$ equal to 1 if j is among the 20 samples lying closest (in euclidian distance) to i or vice versa.

Since spectral clustering methods provide eigenvectors on which subsequently a clustering of the rows has to be performed, and since we are only interested in evaluating the spectral clustering part, we used a cost function defined on the eigenvectors themselves (without doing the actual clustering step). Specifically, the within cluster variance divided by the total variance in the eigenvectors is used as a quality measure, attaining values in between 0 and 1. All experiments are averaged over 10 randomizations of the labeled part of the training set; each time the standard deviation on the estimated average is shown on the figures.

Figures 1 show that the performance of both methods effectively increases for increasing fractions of labeled samples on the three newsgroup dataset as well as on the USPS dataset. Moreover, for small fractions of labeled samples (which is when side information methods are most useful in practice), the newly proposed performs slightly but significantly better.

Subsequently, we solve a binary classification problem derived from the USPS dataset, where one class contains the samples representing numbers from 0 up

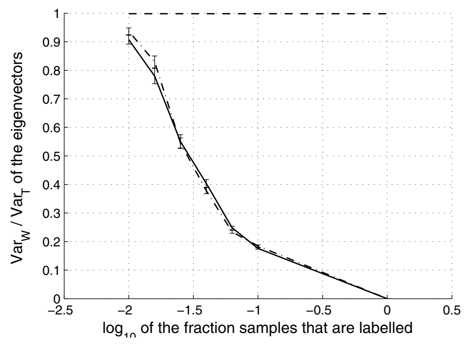


Fig. 2. Similar experiment as in figures 1, but now in the binary classification setting. One class contains all handwritten digits from 0 to 4, the other class contains the digits from 5 to 9. As can be expected the score for no labeled data at all is really bad.

to 4, and the other class from 5 up to 9. In figure 2 we see that the performance of both methods is indistinguishable in this case. Note however that relatively few information is already sufficient to provide a significant improvement over the clustering with no label information at all.

6 Conclusions and Further Work

We have presented an efficient, performant and natural method to incorporate general constraints on the labels in class learning problems. The performance of the method compares well with a recently proposed approach that has a similar computational cost and that is designed to deal with a similar generality of learning settings. However further empirical investigation would be useful.

As compared to other related approaches in literature, the constrained spectral clustering method compares favorably in two respects. First, computationally the method is very attractive since basically it only requires the computation of a few dominant eigenvectors of a matrix with less than n rows and columns (n being the number of samples). Second, the method not only deals with the transductive learning setting, but addresses more general side information learning in the same natural way, both for two class and multi class problems.

Note that the softly constrained version can be seen as the application of the spectral clustering method to a sum of two affinity matrices, where one of both is derived from the label constraints. In principle, one may be able to construct a label affinity matrix for very general label information, also for the the multi class case. This will be subject of a later paper.

Acknowledgements

This work was supported by grants and projects for the Research Council K.U.L (GOA-Mefisto 666, IDO, PhD/Postdocs & fellow grants), the Flemish Government (FWO: PhD/Postdocs grants, projects G.0240.99, G.0407.02, G.0197.02, G.0141.03, G.0491.03, G.0120.03, ICCoS, ANMMM; AWI;IWT:PhD grants, Soft4s), the Belgian Federal Government (DWTC: IUAP IV-02, IUAP V-22; PODO-II CP/40), the EU(CAGE, ERNSI, Eureka 2063-Impact;Eureka 2419-FLiTE) and Contracts Research/Agreements (Data4s, Electrabel, Elia, LMS, IPCOS, VIB). T. De Bie is a Research Assistant with the Fund for Scientific Research – Flanders (F.W.O–Vlaanderen). J. Suykens and B. De Moor are an assistant professor and a full professor at the K.U.Leuven, Belgium, respectively. The scientific responsibility is assumed by its authors.

References

1. T. De Bie and N. Cristianini. Convex methods for transduction. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.
2. T. De Bie, M. Momma, and N. Cristianini. Efficiently learning the metric with side-information. In *Proceedings of the International Conference on Algorithmic Learning Theory (ALT)*. Nara, Japan, April, 2003.
3. T. Joachims. Transductive learning via spectral graph partitioning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2003.
4. S. D. Kamvar, D. Klein, and C. D. Manning. Spectral learning. In *IJCAI*, 2003.
5. A. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.
6. N. Shental, A. Bar-Hillel, T. Hertz, and D. Weinshall. Computing gaussian mixture models with em using equivalence constraints. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.
7. N. Shental, T. Hertz, D. Weinshall, and M. Pavel. Adjustment learning and relevant component analysis. In *Proceedings of the 7th European Conference of Computer Vision*, May, 2002.
8. J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
9. V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 2nd edition, 1999.
10. J. Weston, C. Leslie, D. Zhou, A. Elisseeff, and W. Noble. Semi-supervised protein classification using cluster kernels. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.
11. E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.
12. Stella X. Yu and Jianbo Shi. Grouping with bias. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.
13. D. Zhou, O. Bousquet, T. Navin Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

Solving the One-Class Problem Using Neighbourhood Measures

Javier M. Moguerza[1] and Alberto Muñoz[2]

[1] University Rey Juan Carlos, c/ Tulipán s/n, 28933 Móstoles, Spain
j.moguerza@escet.urjc.es
[2] University Carlos III, c/ Madrid 126, 28903 Getafe, Spain
alberto.munoz@uc3m.es

Abstract. The problem of estimating high density regions from univariate or multivariate data samples is studied. To be more precise, we estimate minimum volume sets, whose probability is specified in advance. This problem arises in outlier detection and cluster analysis, and is strongly related to One-Class Support Vector Machines (SVM). In this paper we propose a new method to solve this problem, the Support Neighbour Machine (SNM). We show its properties and introduce a new class of kernels. Finally, numerical results illustrating the advantage of the new method are shown.

1 Introduction

The task of estimating high density regions from data samples arises explicitly in a number of works involving interesting problems such as outlier detection or cluster analysis (see for instance [7, 4] and references herein). One-Class Support Vector Machines (SVM) [7, 9] are designed to solve this problem with tractable computational complexity. We refer to [7] and references therein for a complete description of the problem and its ramifications.

In this work, a new algorithm to estimate high density regions from data samples is presented. The algorithm relaxes the density estimation problem in the following sense: instead of trying to estimate the density function at each data point, an easier to calculate data-based measure is introduced in order to establish a density ranking among the sample points.

The concrete problem to solve is the estimation of minimum volume sets of the form $S_\alpha(f) = \{x|f(x) \geq \alpha\}$, such that $P(S_\alpha(f)) = \nu$, where f is the density function and $0 < \nu < 1$. Throughout the paper, sufficient regularity conditions on f are assumed. For space reasons, proofs of propositions and theorems are omitted.

The rest of the paper is organized as follows. Section 2 introduces the Support Neighbour Machine and its properties. In Section 3, a kernel formulation of Support Neighbour Machines is shown. In Section 4, the performance of One-Class SVM and Support Neighbour Machines is compared on a variety of both artificial and real data sets. Section 5 concludes.

2 The Support Neighbour Machine

There are data analysis problems where the knowledge of an accurate estimator of the density function $f(x)$ is sufficient to solve them, for instance, mode estimation [1], or the present task of estimating $S_\alpha(f)$. However, density estimation is far from trivial [8, 7]. The next definition is introduced to relax the density estimation problem: the task of estimating the density function at each data point is replaced by a simpler measure that asymptotically preserves the order induced by f.

Definition 1 (Neighbourhood Measures). *Consider a random variable X with density function $f(x)$ defined on $\mathbb{R}^d$. Let S_n denote the set of random independent identically distributed (iid) samples of size n (drawn from f). The elements of S_n take the form $s_n = (x_1, \cdots, x_n)$, where $x_i \in \mathbb{R}^d$. Let $M : \mathbb{R}^d \times S_n \longrightarrow \mathbb{R}$ be a real-valued function defined for all $n \in \mathbb{N}$. (a) If $f(x) < f(y)$ implies $\lim_{n\to\infty} P(M(x, s_n) > M(y, s_n)) = 1$, then M is a **sparsity measure**. (b) If $f(x) < f(y)$ implies $\lim_{n\to\infty} P(M(x, s_n) < M(y, s_n)) = 1$, then M is a **concentration measure**.*

Example 1. $M(x, s_n) \propto 1/\hat{f}(x, s_n)$, where $\hat{f}$ can be any consistent non-parametric density estimator, is a sparsity measure; while $M(x, s_n) \propto \hat{f}(x, s_n)$ is a concentration measure. A commonly used estimator is the kernel density one $\hat{f}(x, s_n) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{\|x - x_i\|}{h})$.

Example 2. Consider the distance from a point x to its k^{th}-nearest neighbour in s_n, $x^{(k)}$: $M(x, s_n) = d_k(x, s_n) = d(x, x^{(k)})$: it is a sparsity measure. Note that d_k is neither a density estimator nor is it one-to-one related to a density estimator. Thus, the definition of 'sparsity measure' is not trivial. Another valid choice is given by the average distance over all the k nearest neighbours: $M(x, s_n) = \bar{d}_k = \frac{1}{k} \sum_{j=1}^{k} d_j = \frac{1}{k} \sum_{j=1}^{k} d(x, x^{(j)})$. Extensions to other centrality measures, such as trimmed-means are straightforward.

Our goal is to obtain some decision function $h(x)$ which solves the problem stated in the introduction, that is, $h(x) = +1$ if $x \in S_\alpha(f)$ and $h(x) = -1$ otherwise. We will show how to use sparsity measures to build $h(x)$. To this aim a new algorithm, the Support Neighbour Machine, is introduced. Consider a sample $s_n = \{x_1, \ldots, x_n\}$. The SNM method works by solving the following optimization problem:

$$\begin{aligned} &\max_{\rho,\xi} \ \nu n \rho - \sum_{i=1}^{n} \xi_i \\ &\text{s.t.} \quad g(x_i) \geq \rho - \xi_i \,, \\ &\qquad\ \ \xi_i \geq 0, \qquad\qquad i = 1, \ldots, n \,, \end{aligned} \tag{1}$$

where $g(x_i) = M(x_i, s_n)$ is a sparsity measure, $x_i \in s_n$, ξ_i are slack variables, $\nu \in [0, 1]$ is a predefined constant and ρ is a variable whose role will become clear below. Note that the calculus of $g(x)$ is not involved in the optimization process; it has to be determined in advance.

The SNM problem formulation is very similar to the linear One-Class SVM formulation (see [5]), but the solution to problem (1) is simpler and its continuity and differentiability is straightforward (while the use of the L_1 norm in the linear One-Class SVM problem implies non derivability, and using any other L_p norm would imply non linearity).

The motivation to adopt the name 'Support Neighbour Machines' is simple: 'sparsity' and 'concentration' are both neighbourhood measures.

The next proposition shows that the decision function $h(x) = sign(\rho^* - g(x))$ will be non-negative for at least a proportion equal to ν of the training s_n sample, where ρ^* is the value of ρ at the solution of problem (1). Following [7], this result is called ν-property.

Proposition 1 (ν-property). *At the solution of problem (1) the following two statements hold:*

1. $\frac{1}{n}\sum_{i=1}^{n} I(g(x_i) < \rho) \leq \nu \leq \frac{1}{n}\sum_{i=1}^{n} I(g(x_i) \leq \rho)$*, where I stands for the indicator function and $x_i \in s_n$.*
2. *With probability 1, asymptotically, the preceding inequalities become equalities.*

Remark 1. If $g(x)$ is chosen to be a concentration measure, then the decision function has to be defined as $h(x) = sign(g(x) - \rho^*)$.

Notice that ν in problem (1) represents the fraction of points inside the support of the distribution if $g(x)$ is a sparsity measure. If a concentration measure is used, ν represents the fraction of outlying points. The role of ρ becomes now clear: it represents the decision value which, induced by the sparsity measure, determines if a given point belongs to the support of the distribution.

As the next theorem states an asymptotical result, we will denote every quantity depending on the sample s_n with the subscript n. The theorem goes one step further from the ν-property, showing that, asymptotically, the SNM algorithm finds the desired α-level sets.

In order to formulate the theorem, we need a measure to estimate the difference between two sets. We will use the d_μ-distance. Given two sets A and B

$$d_\mu(A, B) = \mu(A \Delta B)\,, \tag{2}$$

where μ is a measure on $\mathbb{R}^d$, Δ is the symmetric difference $A\Delta B = (A \cap B^c) \cup (B \cap A^c)$, and A^c denotes the complementary set of A.

Theorem 1. *Consider a measure μ absolutely continuous with respect to the Lebesgue measure. The set $R_n = \{x : h_n(x) = sign(\rho_n^* - g(x)) \geq 0\}$ d_μ-converges to a region of the form $S_\alpha(f) = \{x | f(x) \geq \alpha\}$, such that $P(S_\alpha(f)) = \nu$. Therefore, the Support Neighbour Machine estimates a density contour cluster $S_\alpha(f)$ (which, in probability, includes the mode).*

We provide an estimate of a region $S_\alpha(f)$ with the property $P(S_\alpha(f)) = \nu$. Among regions S with the property $P(S) = \nu$, the region $S_\alpha(f)$ will have minimum volume as it has the form $S_\alpha(f) = \{x|f(x) \geq \alpha\}$. Therefore we provide an estimate that asymptotically, in probability, has minimum volume.

Finally, it is important to remark that the quality of the estimation procedure heavily depends on using a sparsity or a concentration measure (the particular choice is not – asymptotically – relevant). If the measure used is neither a concentration nor a sparsity measure, there is no reason why the method should work.

3 Kernel Formulation of SNM

In this section we will show the relation between SNM and One-Class SVM. In order to do so we have to define a class of neighbourhood measures.

Definition 2 (Positive and Negative Neighbourhood Measures). *$MP(x, s_n)$ is said to be a* ***positive sparsity (concentration) measure*** *if $MP(x, s_n)$ is a sparsity (concentration) measure and $MP(x, s_n) \geq 0$. $MN(x, s_n)$ is said to be a* ***negative sparsity (concentration) measure*** *if $-MN(x, s_n)$ is a positive concentration (sparsity) measure.*

Given that negative neighbourhood measures are in one-to-one correspondence to positive neighbourhood measures, only positive neighbourhood measures need to be considered. The following classes of kernels can be defined using positive neighbourhood measures.

Definition 3 (Neighbourhood Kernels). *Consider the mapping $\phi : \mathbb{R}^d \rightarrow \mathbb{R}^+$ defined by $\phi(x) = MP(x, s_n)$, where $MP(x, s_n)$ is a positive neighbourhood measure. The function $K(x, y) = \phi(x)\phi(y)$ is called a* ***neighbourhood kernel****. If $MP(x, s_n)$ is a positive sparsity (concentration) measure, $K(x, y)$ is a* ***sparsity (concentration) kernel****.*

Note that the set $\{\phi(x_i)\}$ is trivially separable in the sense of [7], since each $\phi(x_i) \in \mathbb{R}^+$. Separability is guaranteed by Definition 2.

The strategy of One-Class support vector methods is to map the data points into a feature space determined by a kernel function, and to separate them from the origin with maximum margin (see [7] for details). In order to build a separating hyperplane between the origin and the points $\{\phi(x_i)\}$, the quadratic One-Class SVM method solves the following problem:

$$\begin{aligned} \min_{w,\rho,\xi} \quad & \frac{1}{2}\|w\|^2 - \nu n \rho + \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & \langle w, \phi(x_i) \rangle \geq \rho - \xi_i \,, \\ & \xi_i \geq 0, \qquad\qquad i = 1, \ldots, n \,, \end{aligned} \tag{3}$$

where ϕ is the mapping defining the kernel function, ρ and ξ_i are variables whose meaning is the same as that in problem (1), and $\nu \in [0, 1]$ is an a priori fixed constant. In the following we will refer to 'quadratic One-Class SVM' simply as 'One-Class SVM'.

The next theorem illustrates the relation between SNM and One-Class SVM when neighbourhood kernels are used.

Theorem 2. *Define the mapping $\phi(x) = MP(x, s_n)$. The decision function $h(x) = sign(\rho_V^* - w^* \phi(x))$ obtained from the solution ρ_V^* and w^* to the One-Class SVM problem (3) using the sparsity kernel $K(x,y) = \phi(x)\phi(y)$ coincides with the solution obtained by solving the SNM problem (1) using a positive sparsity measure.*

It remains open to show if the decision function obtained from One-Class SVM algorithms within the framework in [7, 5] can be stated in terms of positive sparsity or concentration measures. The next remark provides the answer.

Remark 2. The exponential kernel $K_c(x,y) = e^{-\|x-y\|^2/c}$ is neither a sparsity kernel nor a concentration kernel. For instance, consider a univariate bimodal density f with finite modes m_1 and m_2 such that $f(m_1) = f(m_2)$. Consider any positive sparsity measure $MP(x, s_n)$ and the induced mapping $\phi(x) = MP(x, s_n)$. As $n \to \infty$, the sparsity kernel $K(x,y) = \phi(x)\phi(y)$ would attain its minimum at (m_1, m_2) (or at two points in the sample s_n near to the modes). On the other hand, as the exponential kernel $K_c(x,y)$ depends exclusively on the distance between x and y, any pair of points (a,b) whose distance is larger than $\|m_1 - m_2\|$ will provide a value $K_c(a,b) < K_c(m_1, m_2)$, which asymptotically can not happen for kernels induced by positive sparsity measures. In this case, the neighbourhood kernel has four minima while the exponential kernel has the whole diagonal as minima. The reasoning for concentration kernels is analogous. A similar argument applies for polinomial kernels with even degrees (odd degrees induce mapped data sets that are non separable from the origin, which discards them).

Note that, while SNM work with every neighbourhood measure, the separability condition of the mapped data is necessary when One-Class SVM are being used, restricting the use of neighbourhood measures to positive or negative ones. This restriction and the fact that SNM provide a simpler linear approach make the use of SNM advisable when neighbourhood measures are being used.

4 Experiments

In this section we compare the performance of One-Class SVM and SNM for a variety of artificial and real data sets. Systematic comparisons of the two methods as data dimension increases are carried out. First of all we describe the implementation details concerning both algorithms.

With regards to One-Class SVM we adopt the proposal in [7], that is, the exponential kernel $K_c(x_i, x_j) = e^{-\|x_i - x_j\|^2/c}$ is used (the only kernel tested for experimentation in that work). To perform the experiments, a range of values for c has been chosen, following the widely used rule $c = hd$ (see [6, 7]), where $h \in \{0.1, 0.2, 0.5, 0.8, 1.0\}$ and d is the data dimension.

Concerning SNM, two different sparsity measures have been considered:

- $M_1(x, s_n) = d_k = d(x, x^{(k)})$, the distance from a point x to its k^{th}-nearest neighbour $x^{(k)}$ in the sample s_n. The only parameter in M_1 is k, which takes a finite number of values (in the set $\{1, \cdots, n\}$). We have chosen k to cover a representative range of values, namely, k will equal the 10%, 20%, 30%, 40% and 50% sample proportions. Therefore we choose k as the closest integer to hn, where n is the sample size and $h \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$.
- $M_2(x, s_n) = \frac{1}{\sum_{i=1}^{n} \exp\left(-\frac{\|x-x_i\|^2}{2\sigma}\right)}$, where $\sigma \in \mathbb{R}^+$. The only parameter in M_2 is σ. We want σ to be related to the sample variability and, at the same time, to scale well with respect to the data sample distances. We choose $\sigma = hs$, where $s = \max d_{ij}^2/\varepsilon$, $h \in \{0.1, 0.2, 0.5, 0.8, 1.0\}$, $d_{ij}^2 = \|x_i - x_j\|^2$ and ε is a small value which preserves scalability in M_2. For all the experiments we have chosen $\varepsilon = 10^{-8}$.

Measure M_1 has been described in Example 2 in Section 2. Measure M_2 is of the type described in Example 1. M_2 uses as density estimator the Parzen window [8]. Note that Theorem 1 guarantees that asymptotically every sparsity measure (and in particular the two chosen here) will lead to sets containing the true mode.

4.1 Artificial Data Sets

In the first experiment we have generated 2000 points from a gamma $\Gamma(\alpha, \beta)$ distribution, with $\alpha = 1.5$ and $\beta = 3$. Figure 1 shows the histogram, the gamma density curve, the true mode $(\alpha - 1)/\beta$ as a bold vertical line, the SNM estimations with sparsity measure M_1 (five upper lines) and the One-Class SVM (five lower lines) estimations of the 50% highest density region. The parameters have been chosen as described at the beginning of Section 4, and lines are drawn for each method in increasing order in the h parameter, starting from the bottom. Being our goal to detect the shortest region of the form $S_\alpha(f) = \{x : f(x) > \alpha\}$ (that must contain the mode), it is apparent that the SNM regions improve upon the One-Class SVM regions. All the SNM regions contain the true mode and are connected. All the One-Class SVM regions are wider and show a strong bias towards less dense zones. Furthermore, only in two cases the true mode is included in the estimated SVM regions, but in these cases the intervals obtained are not simply connected. SNM using measure M_2 provide similar intervals to those obtained using measure M_1, and are not shown for space reasons.

In the second experiment a mixture of a normal $N(0, 1)$ and a uniform $U(6, 9)$ distribution is considered. Figure 2 shows the results. All the One-Class SVM (five lower lines) estimations spread part of the points in the uniform zone. However, all points in this zone have lower density than those found by the SNM procedure.

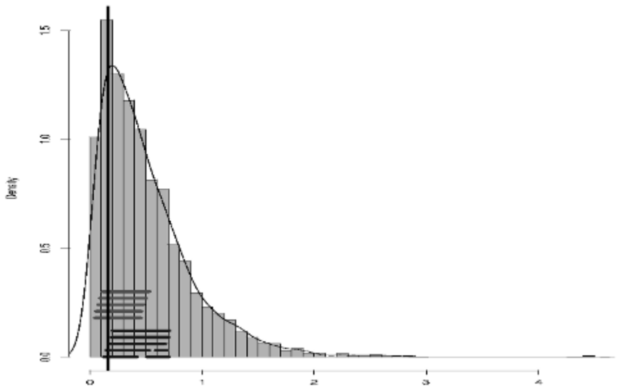


Fig. 1. Gamma sample with 2000 points. The figure shows the histogram, the density curve, a vertical line at the true mode, the SNM estimations with sparsity measure $M_\bullet$ (five upper lines) and One-Class SVM (five lower lines) estimations of the 50% highest density region.

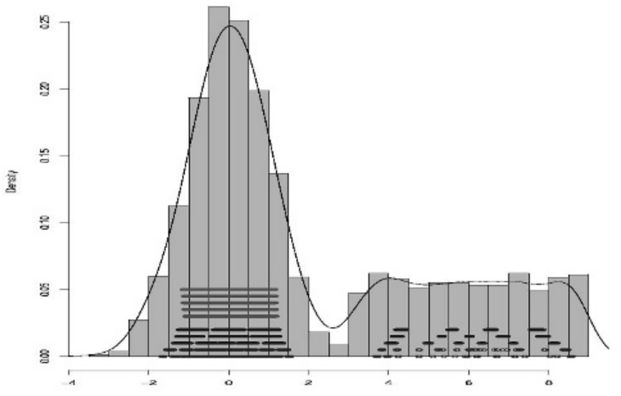


Fig. 2. Mixture sample with 3000 points. The figure shows the histogram, the estimated density curve, the SNM estimations with sparsity measure $M_\bullet$ (five upper lines) and One-Class SVM (five lower lines) estimations of the 50% highest density region.

4.2 Increasing the Dimension of the Data Space

In this experiment we want to evaluate whether the performance of the SNM and SVM algorithms degrades as the data dimension increases. To this end, we have generated 20 data sets with increasing dimension from 2 to 200. Each data set contains 2000 points from a multivariate normal distribution $N(0, I_d)$, where I_d is the identity matrix in $\mathbb{R}^d$. Detailed results are not shown for space reasons. We will only show the conclusions. Since the data distribution is known, we can retrieve the true outliers, that is, the true points outside the support corresponding to any percentage specified in advance. For each dimension and

Table 1. Percentage of true outliers detected using the One-Class SVM, $d = 9$.

Results for One-Class SVM – Cancer Data					
$c = hd$	$0.1d$	$0.2d$	$0.5d$	$0.8d$	$1.0d$
Success %	53.7 %	61.3 %	72.4 %	78.6 %	79.7 %

Table 2. Percentage of true outliers detected using the SNM with $M_{\bullet}$, $n = 683$.

Results for SNM with $M_{\bullet}$ **– Cancer Data**					
$k \simeq hn$	$0.1n$	$0.2n$	$0.3n$	$0.4n$	$0.5n$
Success %	94.1 %	95.0 %	95.0 %	95.0 %	95.8 %

each method, we have determined, from the points retrieved as outliers, the proportion of true ones.

As the data dimension increases, the performance of One-Class SVM degrades: it tends to retrieve as outliers an increasing number of points. The best results for One-Class SVM are obtained for the largest magnitudes of the parameter c. Regarding the SNM procedure, robustness with regard to the parameter choice is observed. Dimension barely affects the performance of the SNM method, and results are consistently better than those obtained with One-Class SVM. For instance, for a percentage of outliers equal to 1%, the best result for One-Class SVM is 15%, against 100% using the SNM method (for all the sparsity measures considered). For a percentage of outliers equal to 5%, the best result for One-Class SVM is 68%, against 99% using the SNM method.

4.3 Cancer Data Set

This data set is given by a 699×9 matrix of clinical measures taken on breast cancer patients [2]. After removing cases with missing values, the matrix dimensions are 683×9 and the real class distribution becomes 65% benign and 35% malignant. The cancer data set is interesting for various reasons: relatively high dimension, overlap, unbalanced classes and different density distributions for each group. This data set has traditionally been approached from a supervised point of view, using classification schemes. Here we focus on a different point of view: we hypothesize that there is only one multivariate (approximate normal) distribution made up of bening cases (65% of the sample) and that the malignant cases are (asymmetrically distributed) outliers. This hypothesis is graphically supported by two-dimensional projections (see for instance [3]). Therefore, we run SNM and One-Class SVM on the cancer data set to detect the 65%-level set, and the percentage of outliers (malignant cases) detected by each of the algorithms is checked. Results are shown in Tables 1 and 2. The best One-Class SVM model detects 79.7% of the outliers, while the worst SNM model detects 94.1% of the outliers, with the best SNM result being equal to 95.8%. Once more the choice of the sparsitiy measure does not affect the results.

5 Conclusions

In this paper the Support Neighbour Machine, a new method to estimate minimum volume sets of the form $S_\alpha(f) = \{x|f(x) \geq \alpha\}$, has been proposed. The new algorithm introduces the use of neighbourhood measures. These measures asymptotically preserve the order induced by the density function f. In this way we avoid the complexity of solving a pure density estimation problem.

Regarding computational results, SNM performs consistently better than One-Class SVM in all the tested problems. The advantage that the SNM has over the One-Class SVM is due to Theorem 1 which guarantees that the SNM algorithm tends to (asymptotically) find the desired α-level sets. The suboptimal performance of One-Class SVM may arise from the fact that its decision function is not based on sparsity or concentration measures.

Acknowledgments

This work was partially supported by MCyT grant TIC2003-05982-C05-05 and by URJC grant PPR-2003-42 (Spain).

References

1. L. Devroye. *Recursive estimation of the mode of a multivariate density.* The Canadian Journal of Statistics, 7(2):159-167, 1979.
2. O.L. Mangasarian and W.H. Wolberg. *Cancer diagnosis via linear programming* SIAM News,Volume 23, Number 5, 1990, 1-18.
3. J.M. Moguerza, A. Muñoz and M. Martin-Merino. *Detecting the Number of Clusters Using a Support Vector Machine Approach.* Proc. ICANN 2002, LNCS 2415:763-768, Springer, 2002.
4. A. Muñoz and J. Muruzabal. *Self-Organizing Maps for Outlier Detection.* Neurocomputing, 18:33-60, 1998.
5. G. Rätsch, S. Mika, B. Schölkopf and K.R. Müller. *Constructing Boosting Algorithms from SVMs: an Application to One-Class Classification.* IEEE Trans. on Pattern Analysis and Machine Intelligence, 24(9):1184-1199, 2002.
6. B. Schölkopf, C. Burges and V. Vapnik. *Extracting Support Data for a given Task.* Proc. of the First International Conference on Knowledge Discovery and Data Mining, AAAI Press, 1995.
7. B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola and R.C. Williamson. *Estimating the Support of a High Dimensional Distribution.* Neural Computation, 13(7):1443-1471 , 2001.
8. B.W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, 1990.
9. D.M.J. Tax and R.P.W. Duin. *Support Vector Domain Description.* Pattern Recognition Letters, 20:1991-1999, 1999.

Detection of a Single Texture in Colour Images

Linjiang Yu and Georgy Gimel'farb

CITR, Department of Computer Science
Tamaki Campus, The University of Auckland
Auckland, New Zealand
lyu011@ec.auckland.ac.nz, g.gimelfarb@auckland.ac.nz

Abstract. Detection of regions similar to a single given texture in arbitrary colour images is difficult for conventional supervised or unsupervised segmentation techniques. We introduce a novel partially supervised algorithm that solves this problem using similarity between local statistics on different levels of pyramidal representations of the texture and the image. Most characteristic statistics for the texture are estimated in accord with a generic Gibbs random field model with spatially homogeneous pairwise pixel interactions. Empirical distributions of the self-similarity values for the texture itself are used to separate the desired texture from an arbitrary background. Experiments with different images, including aerial images of the Earth's surface, show this algorithm effectively detects regions with spatially homogeneous and weakly homogeneous textures.

1 Introduction

Both unsupervised and supervised texture segmentation has been intensively studied for long time. In both the cases the goal is to separate an image into regions containing each an individual spatially homogeneous texture. But most typical practical segmentation problems belong to partially supervised segmentation such that the training information is available only for few regions of interest, while the information about all other non-target regions is absent. The partially supervised segmentation is necessary for identifying regions of interest in remote sensing of the Earth's surface, medical diagnostics, and industrial vision. One more typical application is the content-based image retrieval (CBIR) (see, e.g., comprehensive surveys in [1, 9, 10]). The partially supervised segmentation allows for retrieving images from a large image database that contain regions similar to a given small query image [13].

Up to now, partially supervised segmentation has not been under intensive studies although it cannot be performed with existing supervised or unsupervised techniques. Nonetheless, several schemes for one-class classification were proposed for some applications [2, 6, 11].

The algorithm in [14] separates homogeneous texture from an arbitrary background using local and global distributions of colours and colour cooccurrences. In line with other statistical approaches, it is suitable for translation invariant textures. But many of real visually homogeneous textures are weakly variant with respect to translation, and their spatial homogeneity is recovered better using a pyramidal image representation. This paper attempts to detect a weakly inhomogeneous texture in an arbitrary colour image as well as compare a few different measures of similarity between two probability distributions.

2 Basic Steps of Detection Algorithm

The basic steps are as follows given a value of $\bar{\zeta}$:

1. Create a codebook with a fixed number of codevectors for the training sample using colour space vector quantization (CSVQ).
2. Convert the training sample into the index image with respect to the codebook.
3. Find the maximum deviation of each codevector with respect to the training sample.
4. Create the index image for the original image with the codevectors and their deviations.
5. The above two index images become the base levels of pyramids. Find the number K of pyramid levels based on the sizes of the training area and a fixed moving window.
6. Find a characteristic subset of colour cooccurrence histograms (CCHs) for the base training level image of the index training sample and collect the corresponding global normalized CCHs (nCCHs). Set $k = 0$.
7. Find a cumulative empirical distribution of distances between the local nCCHs over the moving window around each pixel and the like global nCCHs over the whole training image at level k.
8. Calculate the distances between the local nCCHs for the candidate regions and the global nCCHs for the training image at level k. Get the cumulative distance map.
9. Calculate the rejection rate ρ if $k = 0$, then select a distance threshold ζ_k using the distribution at step 7.
10. If $\zeta_k \leq \bar{\zeta}$ or $k \geq K$, go to next step, otherwise go to step 13.
11. Set the next level: $k = k + 1$.
12. Build up the index image of the candidate regions and training image at level k. Go to step 7.
13. Detect the regions by thresholding the cumulative distance map at step 8 with ζ_k.

Details of the CSVQ (step 1) has been already discussed in paper [14]. The codebook $\mathbf{B} = [\mathbf{b}_k : k = 1, \ldots, N]$ was obtained in [14] by approximating an empirical colour distribution for the training sample $\mathbf{S}^{tr}$ with a mixture of Gaussians (GM). But such an approximation is not effective in the case of a small number of sparsely or uniformly distributed codevectors. To avoid the above difficulties, the colour thresholding in this paper uses the codebook $\mathbf{B}$ and its deviation array $\Delta = [\delta_k : k = 1, \ldots, N]$ with respect to the training sample. Each element δ_k is the maximum deviation of the codevector $\mathbf{b}_k$ within its corresponding partition in the training sample.

In the index image, each initial colour is replaced by the index of the closest codebook colour (in terms of the Cartesian distance) if the distance is less than a predefined threshold, δ, or is considered as background otherwise.

Let $\mathbf{G}(0) = [g_i(0) : i \in \mathbf{R}_0; g_i(0) \in \mathbf{N}]$ and $\mathbf{G}^{tr}(0) = [g_j^{tr}(0) : j \in \mathbf{R}_0^{tr}; g_j^{tr}(0) \in \mathbf{N}]$ be the base index image for the original colour image to be segmented and the base training image, respectively, on the supporting finite arithmetic lattices $\mathbf{R}_0 = \{(x, y) : x = 0, \ldots, X_0 - 1; y = 0, \ldots, Y_0 - 1\}$ and $\mathbf{R}_0^{tr} = \{(x, y) : x = 0, \ldots, X_0^{tr} - 1; y = 0, \ldots, Y_0^{tr} - 1\}$. At level $k+1$ of the pyramid, the image $\mathbf{G}(k+1) = [g_{x',y'}(k+1) : (x', y') \in \mathbf{R}_{k+1}; g_{x',y'}(k+1) \in \mathbf{N}]$ can be obtained from the preceding image $\mathbf{G}(k) = [g_{x,y}(k) : (x, y) \in \mathbf{R}_k; g_{x,y}(k) \in \mathbf{N}]$ at level k by downsampling: $X_{k+1} = \lfloor X_k/2 \rfloor$, $Y_{k+1} = \lfloor Y_k/2 \rfloor$ and $g_{x',y'}(k+1) = g_{\lfloor x/2 \rfloor, \lfloor y/2 \rfloor}(k)$ for all (x', y').

3 Texture Similarity Measure

Let $\mathbf{A} = \{(\xi_a, \eta_a) : a = 1, \ldots, A\}$ be the most "energetic" translation invariant families of interacting pixel pairs (see [14] in detail). Let $\mathbf{F}_a(\mathbf{G}) = [F_a(q, s|\mathbf{G}) : q, s \in \mathbf{N}]$ and $\mathbf{F}_{a,i}(\mathbf{G}) = [F_{a,i}(q, s|\mathbf{G}) : q, s \in \mathbf{N}]$ denote the global normalized colour cooccurrence histogram (nCCH) for the family $\mathbf{C}_a$ and the local nCCH over the rectangular moving window $\widetilde{\mathbf{W}} = W_x \times W_y$ around a position i in the image $\mathbf{G}$, respectively. The interaction energy $\varepsilon_a(\mathbf{G})$ for the family $\mathbf{C}_a$ (see [3] in detail) is defined as:

$$\varepsilon_a(\mathbf{G}) = \sum\nolimits_{(q,s)\in\mathbf{N}^2} F_a(q, s \mid \mathbf{G}) \left(F_a(q, s \mid \mathbf{G}) - \frac{1}{|\mathbf{N}|^2} \right)$$

The number K of pyramid levels depends on the radius r of the base training area and the width of the moving window $\widetilde{\mathbf{W}}$ so that the training pixels at the highest pyramidal level are still sufficient for collecting the CCHs to represent the texture at this level:

$$K = \frac{\ln(2r) - \ln(\min(W_x, W_y))}{\ln 2} \tag{1}$$

Let the characteristic subset of pixel neighbours $\mathbf{A}(0) = \{(\xi_a^0, \eta_a^0) : a = 1, \ldots, A\}$ be the clique families with the top-rank interaction energies $\varepsilon_a(\mathbf{G}^{tr}(0))$ over the index training image $\mathbf{G}^{tr}(0)$ and the maximum energy $\varepsilon^{max} = \max\{\varepsilon_a(\mathbf{G}^{tr}(0)) : a = 1, \ldots, A\}$. The corresponding subset $\mathbf{A}(k)$ at level k is denoted as $\mathbf{A}(k) = \{(\xi_a^k, \eta_a^k) : a = 1, \ldots, A\}$ so that

$\xi_a^k = \xi_a^0$, if $\lfloor \xi_a^0/2^k \rfloor < 2$; $\xi_a^k = \lfloor \xi_a^0/2^k \rfloor$, otherwise.
$\eta_a^k = \eta_a^0$, if $\lfloor \eta_a^0/2^k \rfloor < 2$; $\eta_a^k = \lfloor \eta_a^0/2^k \rfloor$, otherwise.

The paper [12] has investigated the changes of pairwise pixelwise interaction structures along the Gaussian pyramid. The structure $\mathbf{A}(k)$ at level k reflects more long-range interaction than $\mathbf{A}(k-1)$ at level $k-1$. Pyramidal similarity may be justified only if textures become more homogeneous when adding more up levels so that the assumption of translation invariant pixel interactions can be satisfied. For weakly homogeneous or inhomogeneous textures, the most characteristic interactions belong to the long-range ones which cannot be captured in the base level given a small moving window but can be reduced and captured at the up levels, so the detection performance may be improved. However, for homogeneous textures, their short-range ones are already captured at the base level, adding more up levels may not improve the performance and may decrease the discriminability due to the lack of their characteristic short-range interactions at up levels. Therefore, this algorithm will start from the base level of pyramid and go up level by level until a criterion is satisfied. Let the pixelwise distance measure be the cumulative distance along the pyramid levels $k = 0, \ldots, L$:

$$D_i^L(\mathbf{F}_i(\mathbf{G}), \mathbf{F}(\mathbf{G}^{tr})) = \sum_{k=0}^{L} \frac{1}{|\mathbf{A}(k)|} \sum_{a \in \mathbf{A}(k)} \frac{\varepsilon^{max}}{\varepsilon_a(\mathbf{G}^{tr}(k))} \bullet D_{a,\frac{i}{2^k}}\left(\mathbf{F}_{a,\frac{i}{2^k}}(\mathbf{G}(k)), \mathbf{F}_a(\mathbf{G}^{tr}(k))\right) \tag{2}$$

where $D_{a,\frac{i}{2^k}}\left(\mathbf{F}_{a,\frac{i}{2^k}}(\mathbf{G}(k)), \mathbf{F}_a(\mathbf{G}^{tr}(k))\right)$ is the distance between the local nCCH at the position $\frac{i}{2^k}$ over the image $\mathbf{G}(k)$ at level k and the global nCCH over the training image $\mathbf{G}^{tr}(k)$ for the clique family $\mathbf{C}_a$, $i \in \mathbf{R}_0$, and it is scaled with the ratio $\frac{\varepsilon^{max}}{\varepsilon_a(\mathbf{G}^{tr}(k))}$ to be comparable each other.

Let $\mathbf{f}_a = \{f_{a,i} : i = 1, \ldots, n; \sum_{i=1}^{n} f_{a,i} = 1\}$ and $\mathbf{f}_b = \{f_{b,i} : i = 1, \ldots, n; \sum_{i=1}^{n} f_{b,i} = 1\}$ be two discrete density functions. The dissimilarity measure $d(\mathbf{f}_a, \mathbf{f}_b)$ between the two functions can be done in many ways. The experiments in [7] had shown that the fidelity measure tends to give sharper results if two density functions to be compared are quite close each other, comparing to χ^2 [8] and Jensen-Shannon divergence (JS) [5] measures. The quadratic form distance (QF) [4] can be the best solution because its similarity matrix may consider the closeness between the training colours.

(1) The symmetric χ^2-distance:

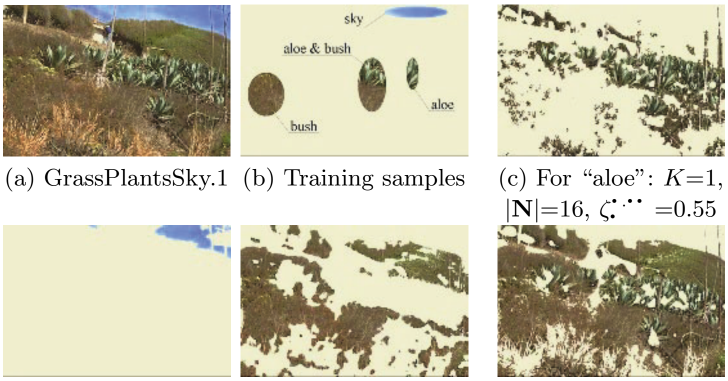
$$d(\mathbf{f}_a, \mathbf{f}_b) = \sum_{i=1}^{n} \frac{(f_{a,i} - f_{b,i})^2}{f_{a,i} + f_{b,i}}, \ 0 \le d(\mathbf{f}_a, \mathbf{f}_b) \le 2 \tag{3}$$

(2) The fidelity measure:

$$d(\mathbf{f}_a, \mathbf{f}_b) = \sqrt{1 - \left(\sum_{i=1}^{n} \sqrt{f_{a,i}}\sqrt{f_{b,i}}\right)^2}, \ 0 \le d(\mathbf{f}_a, \mathbf{f}_b) \le 1 \tag{4}$$

(3) The quadratic form distance (QF):

$$d(\mathbf{f}_a, \mathbf{f}_b) = \sqrt{(\mathbf{f}_a - \mathbf{f}_b)^T \mathbf{A} (\mathbf{f}_a - \mathbf{f}_b)}, \ 0 \le d(\mathbf{f}_a, \mathbf{f}_b) \le \sqrt{2} \tag{5}$$



(a) GrassPlantsSky.1 (b) Training samples (c) For “aloe”: K=1, $|\mathbf{N}|$=16, $\zeta^{\bullet\bullet\bullet}_{\bullet}$ =0.55

(d) For “sky”: K=1, $|\mathbf{N}|$=14, $\zeta^{\bullet\bullet\bullet}_{\bullet}$ =1.13 (e) For “bush”: K=3, $|\mathbf{N}|$=16, $\zeta^{\bullet\bullet\bullet}_{\bullet}$ =0.47 (f) For “aloe & bush”: K=2,$|\mathbf{N}|$=16,$\zeta^{\bullet\bullet\bullet}_{\bullet}$ =0.50

Fig. 1. Results of detecting for different training samples using $\chi^{\bullet}$-distance measure from the image GrassPlantsSky.1.

where $\mathbf{A} = [a_{ij}]$ is the similarity matrix, and a_{ij} denote the cross-bin similarity between bins i and j. Let d_{ij} be the Euclidean distance between colour i (or a pair of colours i) and colour j (or a pair of colours j), $d_{max} = \max_{i,j}(d_{ij})$, then $a_{ij} = 1 - \frac{d_{ij}}{d_{max}}$.

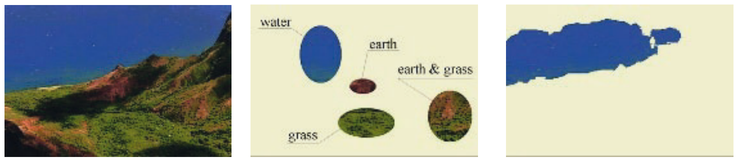
The resulting regions can be obtained from the distance map $\{D_i^L(\mathbf{F}_i(\mathbf{G}), \mathbf{F}(\mathbf{G}^{tr})) : i \in \mathbf{R}_0\}$ by a distance threshold ζ_L^ρ. All the pixels where their distances are less than ζ_L^ρ will be remained. Let ζ_L^ρ be chosen from the empirical distribution of distances $\{D_j^L(\mathbf{F}_j(\mathbf{G}^{tr}), \mathbf{F}(\mathbf{G}^{tr})), j \in \mathbf{R}_0^{tr}\}$ over the training texture pyramid $\{\mathbf{G}^{tr}(0), \ldots, \mathbf{G}^{tr}(L)\}$ by rejecting a certain small percentage ρ of the topmost training distances. The rejection rate ρ depends on the homogeneity of the training sample. Let $D_{max}^0 = \max\{D_j^0(\mathbf{F}_j(\mathbf{G}^{tr}), \mathbf{F}(\mathbf{G}^{tr})) : j \in \mathbf{R}_0^{tr}\}$ be the maximum distance over the base level of the training sample. Assume that a fixed value $\bar{\zeta}$ is the threshold of homogeneity depending on specific distance measure and $\hat{D}$ is the upbound of a distance measure, and $\hat{\rho}$ is the pre-defined maximum rejection rate. Then the rejection rate ρ is defined as:

1. $\rho = 0\%$, if $D_{max}^0 \le \bar{\zeta}$ (homogeneous training sample);
2. $\rho = \frac{\hat{\rho}(D_{max}^0 - \bar{\zeta})}{\hat{D} - \bar{\zeta}} \times 100\%$, otherwise (inhomogeneous training sample, e.g. $\hat{\rho} = 20\%$).

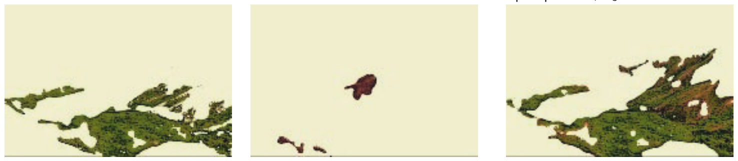
4 Experimental Results and Conclusions

The experimental analysis and comparisons are based on visual observation due to the lack of the “ground truth” for the specific detection problems. Furthermore, the detecting requirements usually depend on specific applications.

The first examples use two colour images (768×512) GrassPlantsSky.1 (a) in Fig. 1 and ValleyWater.2 (a) in Fig. 2 from the MIT Media Lab VisTex database, and (b) shows the different training samples cut from the original ones. The



(a) ValleyWater.2 (b) Training samples (c) For "water": K=3, $|\mathbf{N}|$=11, $\zeta_{\bullet}^{\cdot\bullet\bullet}$ =0.92



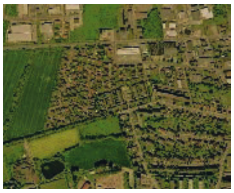
(d) For "grass": K=3, $|\mathbf{N}|$=16, $\zeta_{\bullet}^{\cdot\bullet\bullet}$ =0.49 (e) For "earth": K=2, $|\mathbf{N}|$=16, $\zeta_{\bullet}^{\cdot\bullet\bullet}$ =0.54 (f) For "earth & grass": K=3,$|\mathbf{N}|$=16,$\zeta_{\bullet}^{\bullet\bullet}$ =0.54

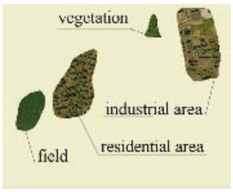
Fig. 2. Results of detecting for different training samples using $\chi^{\bullet}$-distance measure from the image ValleyWater.2.

parameters are chosen as follows: the moving window $W_x \times W_y = 17 \times 17$, only one ($|\mathbf{A}| = 1$) most energetic family of pixel pairs per each training sample $\mathbf{G}^{tr}$ selected within all the relative shifts ($\xi_a \le W_x, \eta_a \le W_y$), the maximum rejection rate $\hat{\rho} = 20\%$, the χ^2-distance measure in Eq. (3) and $\bar{\zeta} = 0.5$. The detecting results with the different training samples are illustrated in (c)-(f) in Fig. 1 and 2 respectively, the white areas are the rejected regions. Note that K is the number of pyramid levels in Eq. (1), the subscript L and superscript ρ of ζ_L^ρ represents the stop level and the rejection rate, respectively. Clearly, the training areas can be always detected. The similar regions to the training area are almost found out. Meanwhile, most of these training regions are inhomogeneous, the results can be refined by using more levels of pyramids.

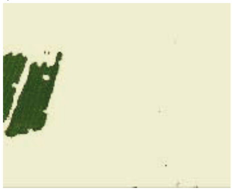
A special colour aerial image in (a) of Fig. 3 provided by the Institute of Communication Theory and Signal Processing (TNT), University of Hannover, is used below as a test example to search for each of the four training regions in (b) of Fig. 3 : field (homogeneous), vegetation (homogeneous), residential area (weakly homogeneous), and industrial area (inhomogeneous). The curves in (a) of Fig. 4 illustrate the empirical distributions of the quantized distances $D_j^0(\mathbf{F}_j(\mathbf{G}^{tr}), \mathbf{F}(\mathbf{G}^{tr}))$ for their base level $L = 0$ only (see Eq. (2)) over the training samples, using χ^2-distance measure with D_{max}^0 as the distance threshold (i.e. $\rho = 0\%$). The narrower the range of distances,the more homogeneous the texture. It is reasonable to set $\bar{\zeta} = 0.5$ being the threshold of homogeneity for χ^2-distance measure. In the same way, $\bar{\zeta} = 0.5$ for fidelity measure and $\bar{\zeta} = 0.25$ for quadratic form (QF) from the curves in (b) and (c) of Fig. 4, respectively.

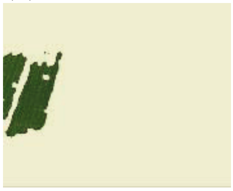
Meanwhile the ratio between $\bar{\zeta}$ and a distance upbound $\hat{D}$ is 0.25, 0.5, and 0.176 for the χ^2-distance, fidelity measure, and QF distance respectively, we can expect that the QF has the best performance and the χ^2-distance can perform better than fidelity measure. This can be verified by the comparison using dif-

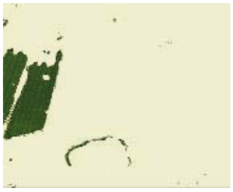

(a) Original aerial image

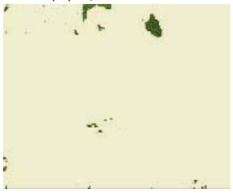


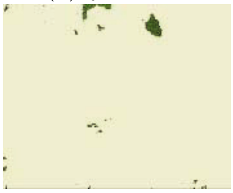
(b) Training samples

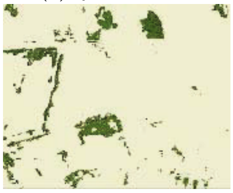

(c) $\zeta_{\bullet}^{\bullet\bullet}$ =0.43

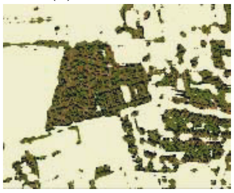

(d) $\zeta_{\bullet}^{\bullet\bullet}$ =0.48

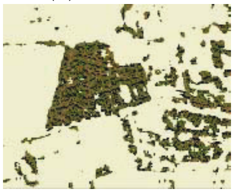

(e) $\zeta_{\bullet}^{\bullet\cdot\bullet\bullet}$ =0.24

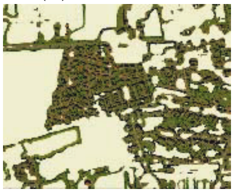

(f) $\zeta_{\bullet}^{\bullet\bullet}$ =0.24

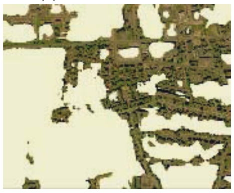

(g) $\zeta_{\bullet}^{\bullet\bullet}$ =0.35

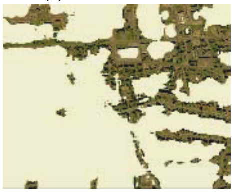

(h) $\zeta_{\bullet}^{\bullet\cdot\bullet\bullet}$ =0.16


(i) $\zeta_{\bullet}^{\bullet\cdot\bullet\bullet\bullet}$ =0.45


(j) $\zeta_{\bullet}^{\bullet\cdot\bullet\bullet}$ =0.47


(k) $\zeta_{\bullet}^{\bullet\cdot\bullet\bullet}$ =0.21


(l) $\zeta_{\bullet}^{\bullet\cdot\bullet\bullet}$ =0.47
$\chi^{\bullet}$-distance


(m) $\zeta_{\bullet}^{\bullet\bullet\cdot\bullet\bullet}$ =0.49
fidelity measure

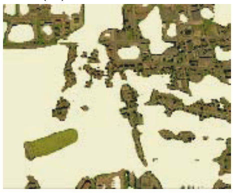
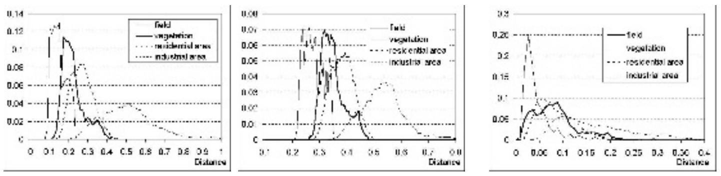

(n) $\zeta_{\bullet}^{\bullet\bullet\cdot\bullet\bullet}$ =0.22
QF distance

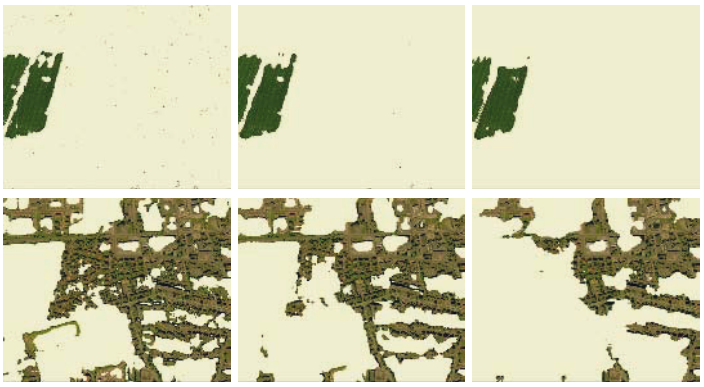
Fig. 3. Comparison using different distance measures: $\chi^{\bullet}$-distance (left), fidelity measure (middle), and QF distance(right) with different training samples: "field" (second row), "vegetation" (third row), "residential area" (fourth row), and "industrial area" (fifth row), $|\mathbf{N}| = 16$.

ferent distance measures with the different training samples demonstrated in (c)-(n) of Fig. 3. But QF is too time consuming so that it practically cannot be applied when $|\mathbf{N}| > 16$. Furthermore, we can see that the performance for the homogeneous training samples "field" and "vegetation", and for weakly ho-



(a) $\chi^{\bullet}$-distance $0 \le d(\mathbf{f}_a, \mathbf{f}_b) \le 2$ (b) fidelity measure $0 \le d(\mathbf{f}_a, \mathbf{f}_b) \le 1$ (c) quadratic form (QF) $0 \le d(\mathbf{f}_a, \mathbf{f}_b) \le \sqrt{2}$

Fig. 4. Empirical distance distributions of different training samples in (b) of Fig. 3 using different distance measures. $W_x \times W_y = 17 \times 17$, $|\mathbf{N}| = 16$, $|\mathbf{A}| = 1$.



$W_x \times W_y = 11 \times 11$ $W_x \times W_y = 17 \times 17$ $W_x \times W_y = 25 \times 25$

Fig. 5. Results with different size of moving window $W_x \times W_y$ for "field" (top row) and "industrial area" (bottom row) using $\chi^{\bullet}$-distance measure $|\mathbf{N}| = 16$.

mogeneous "residential area" are very good, except for a little border problem for "vegetation" because of its too small training area. For the inhomogeneous "industrial area", it still can get reasonably good performance, but some homogeneous sub-regions such as the large area roofs which exist in the training sample cannot be detected, the reason is that the CCH for the chosen clique family from the whole training sample has not reflect the one over these homogeneous sub-regions. Some roads are detected both for 'residential area" and "industrial area" because similar road information exists in both training samples.

Our experiments show that more colours and large number of characteristic families cannot significantly improve the performance. Generally the suitable range of the colours number N from the texture analysis viewpoint can be $16 \le N \le 32$.

Figure 5 shows the detecting results with different size of moving window $W_x \times W_y$ using χ^2-distance measure. For homogeneous textures such as "field", the results vary insignificantly with the different size of moving window within this range, and the border problem can be worse under the large size of moving window; however, the bigger size can get better performance for inhomogeneous textures such as "industrial area".

In total, the proposed algorithm effectively detects a given colour texture on an arbitrary background.

Acknowledgements

This work was supported by the Royal Society of New Zealand Marsden Fund under Grant 3600771/9143 (UOA122).

References

1. S. K. Chang and A. Hsu. Image information systems: Where do we go from here? *IEEE Trans. Knowledge and Data Engineering*, 4(5):431–442, 1992.
2. D. De Ridder, D. M. J. Tax, and R. P. W. Duin. An experimental comparison of one-class classification methods. In *Proc. of Advanced School for Computing and Imaging*, Delft, June 1998.
3. G. L. Gimel'farb. *Image Textures and Gibbs Random Fields.* Kluwer Academic Publishers, Dordrecht, 1999.
4. J. Hafner, H. Sawhney, W. Equitz, M. Flickner, and W. Niblack. Efficient color histogram indexing for quadratic form distance functions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(7):729–736, 1995.
5. J. Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Information Theory*, 37(1):145–151, 1991.
6. M. M. Moya and D. R. Hush. Network constraints and multi-objective optimization for one-class classification. *Neural Networks*, 9(3):463–474, 1996.
7. M. Nolle. Distribution distance measures applied to 3-d object recognition - a case study. In *Proc. the 25th Pattern Recognition Symposiom of the German Association for Pattern Recognition*, Magdeburg, Germany, Sept. 2003.
8. J. Puzicha, Y. Rubner, C. Tomasi, and J. Buhmann. Empirical evaluation of dissimilarity measures for color and texture. In *Proc. IEEE Int. Conf. on Computer Vision (ICCV-1999)*, pages 1165–1173, 1999.
9. Y. Rui, T. S. Huang, and S. Chang. Image retrieval: current techniques, promising directions and open issues. *J. Visual Communication and Image Representation*, 10(4):39–62, 1999.
10. A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content based image retrieval at the end of the early years. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.
11. R. E. Sánchez-Yáñez, E. V. Kurmyshev, and F. J. Cuevas. A framework for texture classification using the coordinated clusters representation. *Pattern Recognition Letters*, 24:21–31, 2003.
12. L. Yu and G. Gimel'farb. Pairwise pixel interactions in texture pyramids. In *Image and Vision Computing New Zealand*, pages 209–214, Auckland, New Zealand, November 2002.
13. L. Yu and G. Gimel'farb. Image retrieval using colour co-occurrence histograms. In *Image and Vision Computing New Zealand*, pages 42–47, Palmerston North, New Zealand, November 2003.
14. L. Yu and G. Gimel'farb. Separating a colour texture from an arbitrary background. In *7th Biennial Australian Pattern Recognition Society Conference on Digital Image Computing: Techniques and Applications*, volume 1, pages 489–498, Sydney, Australia, December 2003.

Clustering Based Under-Sampling for Improving Speaker Verification Decisions Using AdaBoost

Hakan Altınçay and Cem Ergün

Advanced Technology Research and Development Institute
Eastern Mediterranean University, Gazi Mağusa KKTC, Mersin 10, Turkey
{hakan.altincay,cem.ergun}@emu.edu.tr

Abstract. The class imbalance problem naturally occurs in some classification problems where the amount of training samples available for one class may be much less than that of another. In order to deal with this problem, random sampling based methods are generally used. This paper proposes a clustering based sampling technique to select a subset from the majority class involving much larger amount of training data. The proposed approach is verified in designing a post-classifier using AdaBoost to improve the speaker verification decisions. Experiments conducted on NIST99 speaker verification corpus have shown that in general, the proposed sampling technique provides better equal error rates (EER) than random sampling.

1 Introduction

Class imbalance where the training data available for some pattern classes is much less than that of the others, naturally occurs in pattern classification problems such as speaker or face verification. Consider a speaker verification experiment where a post-classifier is to be applied on the verification output scores for optimal decision making [1]. In obtaining the training data for the post-classifier, target tests where the tested identity is the same as the claimed is limited to the number of speakers Q, whereas $Q \times (Q-1)$ impostor tests where the tested identity is different than that of the tested can be obtained. It should be noted that the imbalance increases in proportion to the number of identities considered in the verification experiment.

A generally accepted fact which is also observed in other research domains such as text classification is that the developed classifiers may provide much less accuracy for the minority classes having much less amount of training data [2]. Several explanations are already available in the literature. For instance, in Ref. [3] it is argued that this is mainly due to the fact that the a priori probabilities bias the learning procedure in favor of the majority class. Also, due to the insufficiency of the training data available, the minority class models may not be accurate enough.

There are several approaches proposed to deal with the imbalance problem. For instance, *over-sampling* the minority class to make its training data set cardinality same as the majority class or *under-sampling* the majority class. In general, over-sampling is implemented by inserting replicas of the available data points and under-sampling is implemented by taking into account a random subset of the majority class. These techniques have some disadvantages. In the case of over-sampling, the computational load is increased and overtraining may occur due to the replicated samples. Under-sampling does not take into account all available training data which corresponds to loss of available information. Moreover, it is not guarantied that the subset of data includes sufficient amount of critical samples close in the regions where classes overlap. A common drawback of these techniques is that the best test accuracy is not guaranteed for cardinally equal training sets since the class probabilities are highly likely to be different during test phase leading to a larger number of test samples from the majority class. Moreover, a priori probability information is lost after sampling. In fact, using more majority samples than minority during training is shown to provide better test accuracies.

AdaBoost algorithm is an iterative multiple classifier system development tool which is shown to provide improved classification accuracies for many different data sets compared to the best individual classifier. In each iteration, a new classifier is trained on a subset of the training data where the weight of each training sample is taken into account in this process. In fact, this sample selection mechanism corresponds to the simultaneous application of under-sampling and over-sampling.

In this paper, the performance of the AdaBoost algorithm in the class imbalance case is investigated. Having observed its poor performance, under-sampling and over-sampling techniques are applied and a better performance is achieved. The use of *k-means* clustering based centroids in the training set of the AdaBoost algorithm is proposed as alternative under-sampling technique. In the proposed approach, the AdaBoost algorithm is also modified so as to take into account the class a priori probabilities. Each centroid is used as a representative of its neighborhood where the misclassification of one centroid is considered as more costly than another if the number of the training samples in that cluster is more. Experiments on speaker verification which is basically a 2-class classification problem have proven the effectiveness of the proposed approach compared to the random under-sampling.

2 AdaBoost in Class Imbalance

The original AdaBoost (**Ada**ptive **Boost**ing) algorithm is an ensemble creation technique which was introduced in [4]. The sequential structure of the algorithm allows to create new classifiers which are more effective on the training samples that the current ensemble has a poor performance. In order to achieve this, weighting is applied on the training samples where a training sample with a higher weight has a larger probability of being used in the training of the next classifier. The algorithm summarized in Figure 1. $d_m(n)$ denotes the weight of the nth training sample in $\mathcal{S} = \{(x_n, y_n)\}$, $n = 1, \ldots, N$ initialized to $1/N$ and $\mathcal{C}$ denotes the classifier ensemble where $\mathcal{C}_m$ is the classifier obtained at the

mth iteration. At the end of each iteration, the weights of the samples that are correctly classified (misclassified) by the new classifier are decreased (increased). Increasing the weight of a misclassified sample corresponds to increasing the probability of its inclusion in the training set of the next classifier, probably more than once if its weight is high enough.

```
1. for $m = 1, ..., M$
    1.1 Build classifier $\mathcal{C}_m$ using sample set $\mathcal{S}_m$ from $\mathcal{S}$ using distribution $d_m$.
    1.2 Compute the weighted error using $\epsilon_m = \sum_{n\cdot\,\cdot}^{N} d_m(n)(1 - q_{n,m})$ where
        $q_{n,m} = 1$ if $x_n$ is correctly classified by $\mathcal{C}_m$ and zero otherwise.
    1.3 Compute $\alpha_m = \frac{\cdot}{\cdot}\ln(\frac{\cdot-\epsilon_m}{\epsilon_m})$, $\epsilon_m \in (0, 0.5)$ and update the weights using,

        $$d_{m\cdot\,\cdot}(n) = \frac{d_m(n)}{Z_m}\begin{cases} e^{-\alpha_m} & \text{if } C_m(x_n) = y_n \\ e^{\alpha_m} & \text{if } C_m(x_n) \neq y_n \end{cases}$$

        where $Z_m$ is a normalization factor so that $d_{m\cdot\,\cdot}$ is a distribution.
2. The joint output of the classifier ensemble is computed using

        $$\mathcal{C}(x) = \sum_{m\cdot\,\cdot}^{M} \alpha_m \mathcal{C}_m(x).$$
```

Fig. 1. The AdaBoost algorithm.

In order to evaluate the AdaBoost algorithm in class imbalance case, experiments are conducted on the "phoneme" data set from the ELENA database which involves 3818 and 1586 samples respectively for the first class, w_0 and second class, w_1. 2500 and 100 samples are used for training providing an imbalance ratio of 25 : 1. 1300 samples from each class are used for testing. In the under-sampling case, 100 training samples are selected from w_0 to be considered for model training whereas in the over-sampling case, the training samples of w_1 are replicated for 24 times so that both classes have the same amount of training data. The experiments are repeated for 10 times and the results are averaged. A quadratic discriminant classifier (QDC) from the PRTOOLS toolbox for MATLAB is used as the base classifier [5].

Figure 2 illustrates the training error achieved as a function of the classifiers in the ensemble. As seen in the figure, the performance on the training data in the case of imbalanced classes is much better than the sampling based approaches. However, it is evident from Figure 3 that this is mainly due to training inaccurate models that almost always predict the majority class. The poor test performance of AdaBoost indicates that the algorithm is not well suited for imbalanced data sets. In other words, the inherently available sample weight based under-sampling and over-sampling mechanism in AdaBoost may not be helpful. Moreover, the test performance achieved by the sampling techniques provide their efficiency also for the AdaBoost algorithm where the under-sampling performance is slightly better than over-sampling.

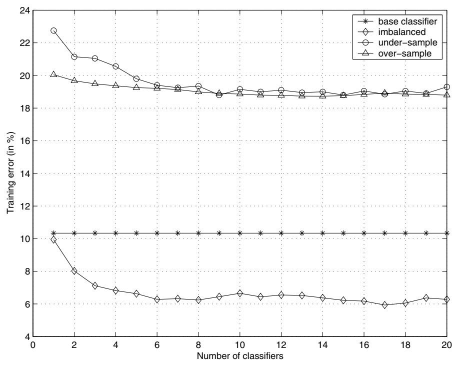


Fig. 2. Training error for different number of base classifiers.

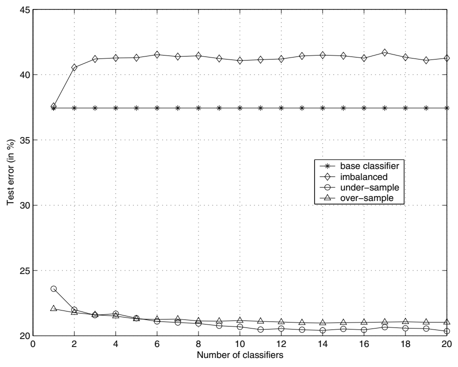


Fig. 3. Test error for different number of base classifiers.

3 Proposed Approach

Let N_t and N_i denote the number of training samples from minority and majority classes respectively where $N_i >> N_t$. Let μ_k, $k = 1, \ldots, N_t$ denote the centroids obtained by applying the k-means clustering algorithm to the training samples from the *majority class* so that the same number of training samples as in the minority class is obtained. Assume that c_k denote the number of training samples which are closest to μ_k where $\sum_{k=1}^{N_t} c_k = N_i$ and c_{avg} is the average number of training samples in the clusters. The selection of N_t centroids from the majority training data set balances the training data such that the same number of data points are used for both the minority and majority classes. The use of centroids instead of a random subset as in under-sampling approach has some advantages. For instance, the selection of samples which are very close to each other and have similar classification difficulties is avoided. Also, different costs can be associated with each misclassification. For instance, the misclassification of a centroid with a large number of training data can be considered to have a high cost. Having computed the centroids, the training data, $\mathcal{S}$ involving the N_t centroids from the majority class and all N_t training data from the minority are considered as the training set, $\mathcal{S} = \{(x_n, y_n)\}$, $n = 1, \ldots, 2N_t$.

There are some drawbacks in applying the AdaBoost algorithm using the centroids. Firstly, the subset selection mechanism in Step 1 does not take into account the relative cardinalities of the training sets of minority and majority classes. Moreover, the cost of misclassifying a highly crowded centroid is not differentiated from a less crowded one in computing the model error, ϵ_m. It should be noted that the term "cost" does not denote the relative importance of correct classification among different classes as used in various cost sensitive boosting algorithms [6]. Instead, it is assumed that a misclassified centroid corresponds to the misclassification of all the majority training vectors closest to it and hence, the term "cost" stands for the contribution to ϵ_m by a misclassified centroid.

In order to avoid the problems specified above, a balancing based AdaBoost algorithm (AdaBoost-B) is proposed as illustrated in Figure 4. The proposed algorithm is based on the SSTBoost algorithm [7]. However, as stated above, the definition of cost and error are not based on relative importance of correct classification among different classes as in SSTBoost. It should be noted that the weight update in AdaBoost-B algorithm is the same as AdaBoost when $cost_n = 1$, $\forall n$. Also, different orders of scaling on the majority class are applied depending on the number of training vectors belonging to the centroids. The weights of the misclassified centroids that are more crowded are increased more than those of less crowded and the weights of the correctly classified centroids that are more crowded are decreased less than those of less crowded. The computation of the weighted error is also modified so as to take into account the fact that each centroid is a representative for a cluster of training data. The contribution to the weighted error by all vectors in a given cluster μ_n is proportional to the number of samples in that cluster. Hence, a weighted distribution should be computed as $w_m(n) = c_n \times d_m(n)/\gamma_m$ to take into account the contribution to the error by all available majority class samples. As a matter of fact, the a priori probabilities are incorporated in the classifier creation which is lost in the under-sampling and over-sampling cases. The scaling factor γ_m is used to make the scaled weights a valid distribution and $c_n = 1$ for the minority class.

For the minority class, $cost_n = 1$, meaning that the standard weighting used in the AdaBoost algorithm is applied where different costs are not associated with correct classification or misclassification.

The performance evaluation of identity verification systems is usually based on the Receiver Operating Characteristic (ROC). The cost of misclassification for different classes determine the operating point on the curve which is generally set using a threshold on the output scores. In summary, the k-means clustering based under-sampling approach helps to select a subset of training samples which represent the underlying distribution more accurately than the random selection approach. Moreover, information about the dense and sparse regions in the input space are included in the iterations so that misclassified centroids corresponding to sparse regions are defined as less costly than those representing the dense regions.

```
- Define $cost_n = \begin{cases} 1 & \text{if } x_n \in \ minority \\ \frac{c_n}{c_{avg}} & \text{if } x_n \in \ majority \end{cases}$
1. for $m = 1, ..., M$
        1.1 Build classifier $\mathcal{C}_m$ using sample set $\mathcal{S}_m$ from $\mathcal{S}$ using distribution $W_m$.
        1.2 Compute $\gamma_m = \sum_{n\bullet\,\bullet}^{\bullet\,N_t} d_m(n)c_n$ where $c_n$ is selected as 1 for the minority
    class.
        1.3 Compute the weighted error using $\epsilon_m = \sum_{n\bullet\,\bullet}^{\bullet\,N_t} (\frac{c_n}{\gamma_m})d_m(n)(1 - q_{n,m})$ where
    $q_{n,m} = 1$ if $x_n$ is correctly classified by $\mathcal{C}_m$ and zero otherwise.
        1.4 Compute $\alpha_m = \frac{\bullet}{\bullet}\ln(\frac{\bullet-\epsilon_m}{\epsilon_m})$, $\epsilon_m \in (0, 0.5)$ and update the weights using,

            $d_{m\bullet\,\bullet}(n) = \frac{d_m(n)}{Z_m} \begin{cases} e^{-\alpha_m \bullet\bullet - cost_n \bullet} & \text{if } C_m(x_n) = y_n \\ e^{\alpha_m cost_n} & \text{if } C_m(x_n) \neq y_n \end{cases}$

    where $Z_m$ is a normalization factor so that $d_{m\bullet\,\bullet}$ is a distribution.
2. The joint output of the classifier ensemble is computed using

            $\mathcal{C}(x) = \sum_{m\bullet\,\bullet}^{M} \alpha_m \mathcal{C}_m(x).$
```

Fig. 4. The balancing based AdaBoost algorithm, AdaBoost-B.

4 Speaker Verification and Experimental Setup

In Speaker Verification (SV), the aim is to decide whether the tested speech utterance belongs to the claimed identity or it is an impostor [8]. In the state-of-the-art SV systems, the output is composed of two likelihood scores where the decision is based on the likelihood ratio obtained as the difference of the log likelihoods of the outputs,

$$\mathcal{L}_{\mathcal{R}} = \mathcal{L}(X|\lambda_i) - \mathcal{L}(X|\lambda_B) \tag{1}$$

where λ_i denotes the claimant model, λ_B denotes the reference model and X denotes the tested utterance. The decision to accept or reject is based on comparing the likelihood ratio to a threshold, Θ such that

$$\mathcal{L}_{\mathcal{R}} \geq \Theta \implies \text{target speaker} \tag{2}$$

$$\mathcal{L}_{\mathcal{R}} < \Theta \implies \text{impostor} \tag{3}$$

Universal Background Models (UBM) are generally used as the reference models where each UBM is a Gaussian Mixture Model (GMM) having a large number of mixtures trained to represent speaker-independent distribution of the feature vectors [9]. The claimant models, λ_i are also GMMs which are trained using Bayesian adaptation from the UBM. The short-time spectral information extracted from the speech utterances, Mel-frequency cepstral coefficients (MFCC) are used as the feature vectors. Sixteen MFCCs and their Delta's [8] are computed in every 80 samples for the spectral representation of each Hamming windowed speech frame of length 160 samples.

A subset of the corpus is excluded from verification tests and is used for training the reference model. Approximately one hour of speech is used to train a UBM for male and another one hour of speech for training a female UBM. Each UBM involves 1024 mixtures. Then, these UBMs are combined to obtain a single 2048 mixture joint UBM to be used as a reference model. Excluding the speakers used for UBM training, 396 speakers (245 female and 151 male) are considered during the verification experiments. The training data of each speaker is split into 6 equal parts for 6-fold cross validation to obtain the training data for the multiple classifier based decision boundary. During the cross-validation, the speech segment which is not included in the model training and kept outside for validating the models had impostor attacks on all the other speakers. During the testing phase, non-overlapping speech segments of length 10s are used. The target tests and impostor attacks are performed using the standard setup defined for this corpus.

The output scores corresponding to the tested speaker and the joint UBM are the treated as the inputs for the classifier ensemble to be created using AdaBoost. The number of training and test samples for the minority class (target tests) and the majority class (impostor attacks) are given in Table 1. Due to 396 speakers involved in the SV experiment, the ratio of impostor to target training samples is high as 395 : 1. This ratio represents a rather high imbalance ratio. However, it naturally occurs in practice for SV problem.

Table 1. The number of training and test samples for the minority and the majority classes.

class	number of training samples	number of test samples
majority	938520	22071
minority	2376	1479

5 Results

In the experiments, two independent multiple classifier systems are implemented. The first one, Ar corresponds to the application of the original AdaBoost algorithm on a random subset of the majority class which involves the same number of training samples as the minority class. The second system, ABc corresponds to the use of cluster centroids of the majority class equal to the number of minority samples and the AdaBoost-B algorithm. Using Ar, the performance of AdaBoost algorithm in improving the verification decision for SV systems is investigated. Using ABc, it is aimed to examine whether alternative sampling techniques can improve the verification accuracy or not. Two different versions of the boosting algorithm are considered, aggressive and conservative. In [10], the given form of the boosting algorithm in Figure 1 is referred as *Aggressive boosting* since the weights of both correctly and incorrectly classified samples are modified. Alternatively, in *Conservative boosting*, either the weights of misclassified samples are increased or the weights of correctly classified samples are decreased. In the conservative implementations in this study, only the weights of correctly classified samples are updated in both *Ar* and *ABc*. The over-sampling approach is not considered due to the heavily increased computational load after over-sampling in our case.

Two different base classifiers are considered, namely quadratic discriminant classifier (QDC) and an MLP neural network consisting of one hidden layer with 10 neurons trained for 300 iterations using fast backpropagation algorithm (NNET). The experiments are conducted for 10 times and the results are averaged. In the experiments, the total number of classifiers in each ensemble is selected as 20.

The Equal Error Rate (EER) provided by the baseline *SV* system based on a linear Bayes decision boundary is 15.28%. The EER's obtained using AdaBoost are presented in Table 2. As seen in the table, both of the systems considered in this study provide significant improvements in the verification accuracy. *ABc* provides better accuracies in both conservative and aggressive types of the boosting for the QDC type of base classifier. *ABc* provides better accuracy also in the conservative boosting of the NNET classifiers. However, the aggressive boosting of NNET classifiers in *Ar* yields a better performance than *ABc*.

There are some points that should be emphasized. Firstly, the imbalance ratio may be so large that, irrespective of the problems that may occur during the learning process, training an ensemble of classifiers may be infeasible from the computational point of view. Secondly, under-sampling based ensemble creation is observed to provide significant improvements. Moreover, taking into account the distribution of the impostor scores during sampling is valuable for further improvement. In fact, we mainly observe that the clustering based sampling and cluster dependent scaling of the training samples provide better accuracies than random selection in majority of the cases.

Table 2. Experimental results for two different base classifiers (in %).

	base classifier: QDC		base classifier: NNET	
MCS	Conservative	Aggressive	Conservative	Aggressive
Ar	13.87	14.00	13.46	13.33
ABc	13.72	13.86	13.39	13.40

6 Conclusions

In this study, the class imbalance problem is addressed and it is observed that the under-sampling approach is a simple but effective method where the AdaBoost algorithm based multiple classifier approach trained using the under-sampled training data provided significant improvements. The use of *k-means* clustering based centroids in the training set of the AdaBoost algorithm is proposed as an alternative under-sampling technique. In the proposed approach, the AdaBoost algorithm is also modified so as to take into account the class a priori probabilities. Each centroid is used as a representative of its neighborhood where the misclassification of one centroid is considered as more costly than another if the number of the training samples in that cluster are more. Experimental results have shown that the proposed approach may be effective in majority of the cases.

References

1. S. Bengio and J. Mariethoz. Learning the decision function for speaker verification. *IEEE-ICASSP Proceedings*, 2001.
2. M.C. Monard and G.E.A.P.A. Batista. Learning with Skewed Class Distribution. In J.M. Abe and J.I. da Silva Filho, editors, *Advances in Logic, Artificial Intelligence and Robotics*, pages 173–180. IOS Press, 2002.
3. G.M. Weiss and F. Provost. The effect of class distribution on classifier learning: An empirical study. *Technical Report ML-TR-44, Department of Computer Science, Rutgers University*, August 2001.
4. Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Second European Conference on Computational Learning Theory*, March 1995.
5. R.P.W. Duin. PRTOOLS (version 3.0). A Matlab toolbox for pattern recognition. *Pattern Recognition Group, Delft University, Netherlands*, January 2000.
6. Kai Ming Ting. A comparative study of cost-sensitive boosting algorithms. In *Proc. 17th International Conf. on Machine Learning*, pages 983–990. Morgan Kaufmann, San Francisco, CA, 2000.
7. S. Merler, C. Furlanello, B. Larcher, and A. Sboner. Automatic model selection in cost-sensitive boosting. *Information Fusion*, 4(1):3–10, March 2003.
8. D.A. Reynolds. Speaker identification and verification using Gaussian mixture speaker models. *Speech Communication*, 17:91–108, 1995.
9. D.A. Reynolds, T.F. Quateri, and R.B. Dunn. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10:19–41, 2000.
10. L.I. Kuncheva and C.J. Whitaker. Using diversity with three variants of boosting: Aggressive, conservative, and inverse. *MCS2002*, 2002.

Control of Sparseness for Feature Selection

Erinija Pranckeviciene[1,2], Richard Baumgartner[1], Ray Somorjai[1], and Christopher Bowman[1]

[1] Institute for Biodiagnostics, National Research Council Canada, 435 Ellice Avenue Winnipeg ,Canada
{Richard.Baumgartner,Ray.Somorjai, Christopher.Bowman}@nrc-cnrc.gc.ca
[2] Kaunas University of Tecnology, Studentu 50, LT 3031, Kaunas, Lithuania
Erinija.Pranckeviciene@ktu.lt

Abstract. In linear discriminant (LD) analysis high sample size/feature ratio is desirable. The linear programming procedure (LP) for LD identification handles the curse of dimensionality through simultaneous minimization of the L1 norm of the classification errors and the LD weights. The sparseness of the solution – the fraction of features retained - can be controlled by a parameter in the objective function. By qualitatively analyzing the objective function and the constraints of the problem, we show why sparseness arises. In a sparse solution, large values of the LD weight vector reveal those individual features most important for the decision boundary.

1 Introduction

In a high-dimensionality / small sample size scenario, many linear classification rules are possible. When the sample to feature ratio (SFR) is low, we face the problem of overfitting - many perfect classification rules for the training data and poor generalization on the test data. Achieving the proper ratio between number of features and available sample size is of great interest [1],[17]. Conventional dimensionality reduction techniques [2], [3] are not very useful if retaining the original feature positions is important. For high-dimensional situations, methods producing sparse solutions are in demand. Sparse means that only a few solution coefficients have large values. The linear programming (LP) technique of identifying a linear discriminant function belongs to the category of methods producing sparse solutions. Its usefulness in feature selection has been demonstrated [4]. There are case studies showing the potential of the technique in microarray analysis [5] and in face recognition [6]. This LP technique is a variant of linear support vector machine (SVM), the only difference being in the objective function. Selecting the value of a parameter in the objective function will force sparseness on the linear discriminant solutions of LP. The sparseness of the solution depends on the geometrical configuration of the data points. Although there exist studies on SVM via linear programming [8] and [7] there is a lack of systematic analysis on how the sparse solution is obtained, and what factors govern the sparseness. A deeper insight is also missing concerning the characteristics properties of the features the sparse solution identifies. Our analysis concerns the objective function of the LP formulation for linear discriminant and constraints imposed by the dataset. In the following, variables denoting vectors will be bold. We consider a 2-class classification problem. Our dataset consists of the vectors $\mathbf{x}_i \in X$ having components $\mathbf{x}_i = [x_i^1, x_i^2, ..., x_i^p]$, labeled by $y_i \in \{+1,-1\}$, where $i = 1,..., N_1 + N_2$ are the number of samples in the classes and p is data dimensionality. Our problem is to find a linear discriminant function classifying the samples into one of the two classes ω_1, ω_2:

$$g(\mathbf{x}_i) = \mathbf{w}^T \mathbf{x}_i + w_0, \quad \begin{matrix} g(\mathbf{x}_i) \geq 0 \Rightarrow \mathbf{x}_i \in \omega_1 \\ g(\mathbf{x}_i) < 0 \Rightarrow \mathbf{x}_i \in \omega_2 \end{matrix}. \tag{1}$$

We may formulate this problem as a system of linear equations:

$$\mathbf{Xw} = \mathbf{y}. \tag{2}$$

If there are more equations than unknowns, then (2) represents a system of over determined equations. We may obtain the solution for the weight vector by least squares or by minimizing the total absolute error [10]. When there are more unknowns (features) than equations, the system (2) is underdetermined and has many solutions.

During the last decade, SVM or maximal margin classifiers were used extensively. This learning algorithm is not parametric and implements an approximation of the unknown functional relationship between training data and class label/target. The unknown discriminant weight vector is found by optimizing a functional derived in learning theory [11], [12]:

$$J(\mathbf{w}) = G * \|\mathbf{w}\|_{Lp} + C * \|\xi\|_{Lp}, \tag{3}$$

where L_p denotes the p-norm, $\mathbf{w}$ is the vector of the weights of the linear discriminant to be found, ξ is the vector of errors between the actual output and the desired output: $\xi = \mathbf{Xw} - \mathbf{y}$. The choice of norm and the values of the constants in (3) span a range of criteria for regression and classification problems [12], [13] and [14]. Different criteria implement different methods to get the solution. Some instances of the criterion function with different choices of norm and parameter values are summarized in Table 1. The criterion for LD identification by the linear programming technique producing sparse solution is:

$$J(\mathbf{w}) = \|\mathbf{w}\|_{L1} + C * \|\xi\|_{L1}. \tag{4}$$

Two terms are minimized: the total absolute error and the sum of the components of the linear discriminant. The constraints are the same as for SVM in the linearly not separable case, and are given by (5). For some values $C > C_{max}$, we get the maximal margin classifier, identical to linear SVM. If the value of C is in the interval $0 < C < C_{max}$, then we get a sparse solution for the weight vector. For different datasets, the value of C_{max} is different. Our goal is to show how C influences the objective function and why small values of C lead to sparse solutions.

Table 1. Criteria and solution methods spanned by different norms and values of the parameters in the objective function (3). When G>0 the function (3) is studied in [16].

Norm	**Values of the parameters**		
	G = 1, C > C_{max}	**G= 0, C = 1**	**G = 1, 0 < C < C_{max}**
L_1	SVM, Linear programming method	Minimization of $\lvert Xw - y \rvert$. Least absolute error problem, usually solved by LP method.	Sparse solutions for linear discriminant function. Implements SVM by Linear programming method. The value of C, controlling sparseness, depends on dataset configuration. C is the upper bound on variables in the dual problem.
L_2	SVM, Quadratic programming method	Minimization of $(Xw - y)^T (Xw - y)$. Least Squares problem.	Sparse solutions for linear discriminant function. Implements SVM by Quadratic programming method. C is the upper bound of Lagrange multipliers.

2 Analysis of the Constraints and Objective Function

2.1 Formulation of the Problem for the General Linear Program Solver

The optimal solution minimizing (4) is usually obtained by using general linear program solvers [8]. SVM imposes the constraints onto the separating hyperplane. It has to be at the desired distance from the training points and have maximal margin with respect to vectors of the opposite classes [9]:

$$y_i \left(w_1 x_i^1 + \ldots + w_p x_i^p + w_0 \right) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, N_1 + N_2 . \tag{5}$$

The inequalities in (5) define the region of feasible solutions for (4). For an arbitrary hyper plane, the actual distance of the points x_i from it is:

$$d_i = \frac{\mathbf{x}_i \mathbf{w}^T + w_0}{\left\| \mathbf{w} \right\|_2} . \tag{6}$$

The desired distance imposed by the constraints is not less than Δ :

$$\Delta = \frac{1}{\left\| \mathbf{w} \right\|_2} . \tag{7}$$

The quantities ξ_i in (5) are proportional to the differences between the desired and actual distances:

$$\frac{\xi_i}{\left\| \mathbf{w} \right\|_2} = \Delta - y_i d_i . \tag{8}$$

The value of ξ_i shows the position of the data point with respect to the separating hyperplane: $\xi_i = 0$ means that the data point is exactly at distance Δ from the separating hyperplane, $\xi_i < 0$ means that the data point is correctly classified, $\xi_i > 0$ means that the data point is misclassified or closer to the hyper plane than Δ. Minimizing the second term in (4) means minimizing the empirical risk/classification errors. The constraint $\xi_i \geq 0$ in (5) restricts the feasible region of the weights of linear discriminant vector $\mathbf{w}$ to the region where classification errors occur or where the data point is closer than the desired distance. In order to present the minimization problem in a form suitable for a general linear program solver, the variables in the objective function should be positive. Thus, each weight component variable is modeled as a difference of two non-negative variables, as is common in linear programming [15], page 32:

$$w_j = u_j - v_j, \tag{9}$$

and the absolute value of the weight is:

$$|w_j| = u_j + v_j. \tag{10}$$

The pair u_j, v_j satisfying (9) and (10) is unique. Only three choices are possible simultaneously satisfying (9) and (10): 1) $u_j = 0 \quad v_j = 0$, 2) $u_j = 0 \quad v_j \neq 0$ and 3) $u_j \neq 0 \quad v_j = 0$. The constraints (5) now are:

$$\begin{array}{c} y_1\left(x_1^1 u_1 + \ldots + x_1^p u_p - x_1^1 v_1 - \ldots - x_1^p v_p + u_0 - v_0\right) + 1 * \xi_1 + \ldots + 0 * \xi_N \geq 1 \\ \vdots \\ y_N\left(x_N^1 u_1 + \ldots + x_N^p u_p - x_N^1 v_1 - \ldots - x_N^p v_p + u_0 - v_0\right) + 0 * \xi_1 + \ldots + 1 * \xi_N \geq 1 \end{array}, \tag{11}$$

and

$$u_j \geq 0, v_j \geq 0, \xi_i \geq 0, \quad j = 0, \ldots, P, \quad i = 1, \ldots, N, N = N_1 + N_2. \tag{12}$$

The objective function (4) is transformed to:

$$J(\mathbf{u}, \mathbf{v}, \xi) = \sum_{j=1}^{P} (u_j + v_j) + C \sum_{i=1}^{N} \xi_i. \tag{13}$$

We need to find:

$$(\mathbf{u}^*, \mathbf{v}^*, \xi^*) = \arg\min J(\mathbf{u}, \mathbf{v}, \xi). \tag{14}$$

One basic feasible solution of (13) subject to (11) and (12) is $\mathbf{u} = \mathbf{0}, \mathbf{v} = \mathbf{0}, \xi = \mathbf{1}$ not useful for classification. In (13) the empirical risk is minimized, thus only non-negative ξ_i are considered and the modulus of ξ_i in (4) is equivalent to a positive ξ_i in (13). If the exact objective function (4) is minimized, then each ξ_i should be modeled by two positive variables as were the components of linear discriminant $\mathbf{w}$ in (9) and (10). More details on the SVM formulations with different norms can be found in [7]. The slacks in (11) are decoupled from the weights of the linear discriminant, although, strictly speaking, slacks and weights depend on each other.

2.2 What Is the Origin of the Sparseness?

The sparseness of the optimal solutions (13) subject to (11) and (12) depends on the value of C. In available SVM software, this parameter is set to the default value, or is determined by cross validation [16]. To discover the origin of sparseness, we analyze qualitatively the dependence of the shape of the objective function (4) on the parameter C. With real instances of the weight vector and a given set of data points, the slack variables equal to the deviations from the target:

$$\xi_i = 1 - y_i \left(w_1 x_i^1 + \ldots + w_p x_i^p + w_0\right). \tag{15}$$

Substituting (15) directly into (4), we express (4) as a function:

$$J(w_1,\ldots,w_p,w_0) = \sum_{j=1}^{p} \left|w_j\right| + C \sum_{i=1}^{N_1+N_2} \left|1 - y_i \left(w_1 x_i^1 + \ldots + w_p x_i^p + w_0\right)\right|. \tag{16}$$

The function (16) has two parts:

$$J(\mathbf{w}, w_0) = R(\mathbf{w}) + C * A(\mathbf{w}, w_0), \tag{17}$$

R is called a regularizer and A is the empirical risk or penalty and loss in [16]. The objective function (16) is piecewise linear. Convex piecewise linear functions of the type (16) are analyzed in depth in [15]. In a constrained optimization problem, the optimal solutions for the objective function lie in the feasible region defined by the constraints. Here this region is fixed and determined by the data points as defined by (5). The objective function is controlled by the values of C leading to the different optimal solutions. When C is large, the term A dominates in the objective function. When C is small, the term R dominates in the objective function. When C is approaching zero, the objective function becomes flat and balanced. The minimum point of function (16) is forced to approach the zero origin point by small C. This narrows the set of possible optimal solutions to the points of feasible region lying near the origin. We illustrate this statement graphically by using a one-dimensional example. It is depicted in Figure 1. In higher dimensions, visualization of the concepts becomes intractable. Let the data consist of three points: (x1=0.5, y1=1), (x2= -2, y2=-1) and (x3=5, y3=-1). Let $w_0 = 0$ in the example. The linear discriminant w in the example is a scalar. The function (16) with these values is:

$$J(w) = |w| + C(|1 - 0.5w| + |1 - 2w| + |1 + 5w|). \tag{18}$$

It is a sum of convex functions and is convex itself. The coefficients 0.5, 2 and 5 can be interpreted as the influence of the data on the objective function. Higher values of the data-dependent coefficients increase the slopes of the components of the objective function and dominate the total sum. The constraints are:

$$\begin{aligned} &\xi_1 \ge 1 - 0.5w, \quad \xi_1 \ge 0, \\ &\xi_2 \ge 1 - 2w, \quad \xi_2 \ge 0, \\ &\xi_3 \ge 1 + 5w, \quad \xi_3 \ge 0. \end{aligned} \tag{19}$$

The functions e1(w)=1-0.5w, e2(w)=1-2w and e3(w)=1+5w for a given dataset represent the functional relationship (8) for all values of linear discriminant w. The interval of w values, satisfying all (19) constraints (feasible region, which does not change) is $w \in [-0.2 \quad 0.5]$. However the shape of the objective function is determined by C. In Fig.1 we illustrate the difference of the objective functions J1(w), J2(w) and J3(w) given in (18) corresponding to different values of C: C=0.2, C=1.5 and C=5. Function J1(w) attains its minimum at the point $w = 0$ (sparse solution), which is forced by the C=0.2.

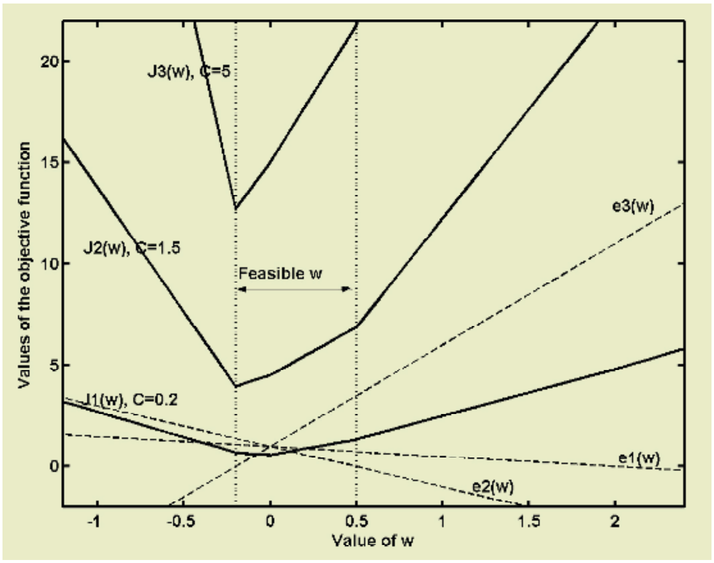


Fig. 1. The influence of the parameter C on the objective function. Solid lines represent the objective functions J1(w), J2(w) and J3(w) of (18) for different values of C: C=0.2, C=1.5 and C=5. The feasible region is shown by dotted lines. The functions e1(w), e2(w) and e3(w) are represented by dashed lines.

The simple one-dimensional example illustrates the effect of small values of C on the objective function of the form (4). In the high-dimensional case, many data points form a complicated convex surface for the feasible region. The objective function of form (4) is a superposition of hyperplanes defined by constraints plus a regularization term. When C approaches zero, the objective function is flattened. The minimum value of this function is forced to lie near the coordinate origin. Since all variables in the minimization problem (13) subject to (11) and (12) are constrained to be non-negative, the feasible region is restricted to the positive half of the high-dimensional space where minimization takes place. For C approaching zero, the optimal solutions of (13) will be at the points where the hyperplane of the objective function encounters the borders of the positive half of coordinate space. The level of sparseness depends on the dataset, determining the orientations of the constraints.

If we take the expression of ξ_i in (15) and substitute it into (13), then express the components of the weight vector $\mathbf{w}$ through $\mathbf{u}$ and $\mathbf{v}$ using (9) and (10) and rearrange the terms, we express (13) as a linear combination of the components of the vectors $\mathbf{u}$ and $\mathbf{v}$:

$$J_1(\mathbf{u},\mathbf{v},u_0,v_0)=\sum_{j=1}^{p}u_j(1-Ck_j)+\sum_{j=1}^{p}v_j(1+Ck_j)-(u_0-v_0)C\sum_{i=1}^{N}y_i+CN\cdot \qquad (20)$$

The term

$$k_j=\sum_{i=1}^{N}y_i x_i^j \qquad (21)$$

is the data-dependent term. In expression (20) coefficients (1-Ck$_j$) , (1+Ck$_j$) j =1…p, represent the coordinates of the normal vector of the hyper plane of the objective function (20) which is equivalent to (13). They determine the direction in which the function decreases. For the positive coordinates of a normal vector, the decreasing direction of the objective function hyperplane is towards the origin. As C vanishes, more coefficients become positive, depending on the term (21). C should be $C<1/|k_j|$ in order to set the corresponding normal coefficient positive. If all normal coefficients are positive, the optimal minimum value of (20) is zero $\mathbf{u}=\mathbf{0},\mathbf{v}=\mathbf{0}$. The analysis of (20) reveals how sparse solutions evolve and the type of influence the data has on the solutions. Noting that the empirical mean of the observations is:

$$\hat{m}_j=\frac{1}{N}\sum_{i=1}^{N}x_i^j\ , \qquad (22)$$

For (21) we have:

$$k_j=N_1\hat{m}_j^{\omega_1}-N_2\hat{m}_j^{\omega_2}\,. \qquad (23)$$

The data term k$_j$ is the difference between the weighted centroids of the two classes. (20) and (23) show that the last retained non-zero component of the sparse solution corresponds to the feature that has the largest distance between the centroids of the two classes.

3 Classification Example

We illustrate the geometrical property of the sparse solution induced by small C on a simple artificial example of linearly separable data. In order to compare with other methods, we present several decision boundaries obtained by LP with different values of C and linear SVM, least squares presented in Fig 2. Sparse solutions of the weight vector have zero components. The interpretation of zero components is that they identify unimportant features. “Unimportance” means that individual features, corresponding to zero components of the linear discriminant, do not contribute to the decision boundary. The geometrical property of **unimportant features** is that their **centroids for the two classes are closer** than those of the important features.

4 Conclusions

We presented the analysis of a particular example of the objective function used in the LP method for identification of a linear discriminant. Our analysis is qualitative, aim-

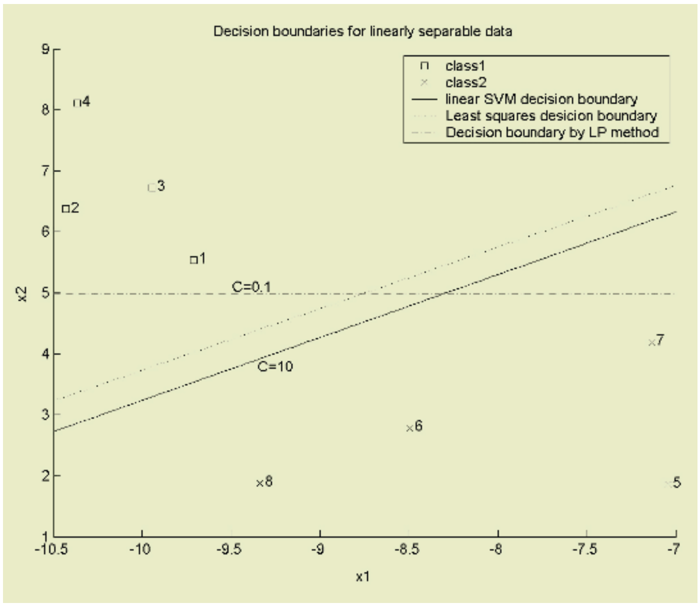


Fig. 2. Decision boundaries for the linearly separable problem for different values of C. C=0.1 gives sparse decision, where x1 is an unimportant feature. C=10 gives identical separation boundary to that of a linear SVM.

ing at a better understanding of the relationships between data, constraints and shape of the objective function. We show that we can control the sparseness of the solution by the parameter C. Small values of C induce sparseness, making the objective function flat and moving its extreme points towards zero. The solutions of the weight vector are affected by the changes in the objective function. The *practical effect* is that for individual features, with centroids for the two classes close (in the Euclidean sense), the corresponding components of the weight vector are very small. In the high dimension/small sample scenario, the method is useful for finding subsets of individual features that contribute to the class separation. However, the value of the sparseness-controlling parameter C for different sets must be identified experimentally. We are currently investigating the impact of the parameter C on the solution of the L1 norm classification problem in real life applications of high-dimensional biomedical spectra.

References

1. Raudys, S.: Statistical and neural classifiers, Springer Verlag, (2001)
2. Chen, L., Liao, H.M., Ko, M., Lin, J. and Yu, G.: A new lda-based face recognition system which can solve the small sample size problem. Pattern recognition, Vol. 33.(2000) 1713-1726

3. Howland, P., Jeon, M. and Park, H.: Structure preserving dimension reduction for clustered text data based on the generalized singular value decomposition. SIAM Journal on Matrix Analysis and Applications, 25-1,(2003) 165 - 179
4. Bradley, P., Mangasarian, O. and Street, W.: Feature selection via mathematical programming. INFORMS Journal on Computing, 10(2), (1998) 209 - 217
5. Bhattacharyya, C., Grate, L.R., Rizki, A. et al. : Simultaneous relevant feature identification and classification in high-dimensional spaces: application to molecular profiling data. Signal Processing, Vol. 83, Issue 4, (2003) 729 - 743
6. Guo, G.D and Dyer, C.: Simultaneous Feature Selection and Classifier Training via Linear Programming: A Case Study for Face Expression Recognition. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, Madison, Wisconsin, 18-20 June ,(2003) 346 - 352
7. Pedroso J.P., Murata N. Support vector machines with different norms: motivation, formulations and results, Pattern recognition letters, Vol. 12,Issue 2, (2001) 1263-1272
8. Kecman, V. and Hadzic, I.: Support vectors selection by linear programming. Proc. IJCNN 2000, Vol. 5, (2000) 193 - 198
9. Vapnik, V.: Introduction to statistical learning theory, Springer, (2001)
10. Rosen, J.B., Park, H., Glick, J. and Zhang, L.: Accurate solutions to overdetermined linear equations with errors using L1 norm minimization. Computational optimization and applications, Vol. 17, (2000) 329 - 341
11. Szeliski, R.: Regularization in neural nets. In: Mathematical perspectives on neural networks, Eds: P.Smolensky at.al, Lawrence Erlbaum Associates, (1996) 497 - 532
12. Poggio, T. and Smale, S.: The mathematics of learning:dealing with data. Notices of the American Mathematical Society(AMS), Vol. 50,No. 5, (2003) 537 - 544
13. Saunders, C., Gammerman, A.and Vovk, V.: Ridge regression learning algorithm in dual variables. Proceedings of the 15th International Conference on Machine Learning, (1998)
14. Mika, S., Ratsch, G., Weston, J., Schoelkopf, B., Smola, A. and Muller, K.R.: Constructing descriptive and discriminative nonlinear features: Rayleigh coefficients in kernel feature spaces. IEEE PAMI, Vol. 25, No. 5, (2003) 623 - 628
15. Arthanari, T.S and Dodge, Y.: Mathematical programming in statistics. John Willey and sons, (1981)
16. Hastie T., Rosset S., Tibshirani R. and Zhu J.: The entire regularization path for the support vector machine (2004)
17. Figureido M.: Adaptive sparseness for supervised learning. IEEE PAMI, Vol. 25, No. 9, (2003) 1150 - 1159

On Prediction Mechanisms in Fast Branch & Bound Algorithms

Petr Somol[1,2], Pavel Pudil[2,1], and Jiří Grim[1]

[1] Dept. of Pattern Recognition, Institute of Information Theory and Automation
Academy of Sciences of the Czech Republic, 182 08 Prague 8
somol@utia.cas.cz
www.utia.cas.cz/user_data/PR_dept

[2] Faculty of Management of the Prague University of Economics
377 01 Jindřichův Hradec, Czech Republic
pudil@fm.vse.cz
www.fm.vse.cz

Abstract. The idea of using the Branch & Bound search for optimal feature selection has been recently refined by introducing additional predicting heuristics that is able to considerably accelerate the search process while keeping the optimality of results unaffected. The heuristics is used most extensively in the so-called Fast Branch & Bound algorithm, where it replaces many slow criterion function computations by means of fast predictions. In this paper we investigate alternative prediction mechanisms. The alternatives are shown potentially useful for simplification and speed-up of the algorithm. We demonstrate the robustness of the prediction mechanism concept on real data experiments.

Keywords: subset search, feature selection, search tree, optimal search, subset selection, dimensionality reduction.

1 Introduction

The problem of optimal feature selection (or more generally of subset selection) is difficult especially because of its time complexity. Any known optimal search algorithm has an exponential nature. The only alternative to the exhaustive search is the *Branch & Bound* (B&B) algorithm [5, 2] and ancestor algorithms based on a similar principle. Two of the recent algorithm versions utilize a concept of prediction mechanism that enables considerable acceleration of the search process without affecting the optimality of results. The Branch & Bound with Partial Prediction (BBPP) [7] uses a prediction mechanism to avoid many criterion evaluations that are unavoidable in older algorithm versions for finding efficient ordering of features inside the search tree. The Fast Branch & Bound (FBB) [6] extends the prediction mechanism to enable bypassing of non-prospective branch sections. The speed-up of BBPP and FBB over older algorithms depends strongly on the efficacy of the underlying prediction mechanisms. In this paper we define and investigate alternative prediction mechanisms in addition to the original one. We show that alternative mechanisms may further improve the FBB speed and also simplify its implementation.

1.1 Preliminaries

All the algorithms addressed in this paper require the criterion function fulfilling the *monotonicity condition.* Consider a problem od selecting d features from an initial set of D measurements using objective function J as a criterion of subset effectiveness. The Branch & Bound approach aims to solve this search problem by making use of the monotonicity property of certain feature selection criterion function. Let $\bar{\chi}_j$ be the set of features obtained by removing j features $y_1, y_2, \cdots, y_j$ from the set Y of all D features, i.e.

$$\bar{\chi}_j = \{\xi_i | \xi_i \in Y, 1 \leq i \leq D; \xi_i \neq y_k, \forall k\}$$

The *monotonicity condition* assumes that for feature subsets $\bar{\chi}_1, \bar{\chi}_2, \cdots, \bar{\chi}_j$, where

$$\bar{\chi}_1 \supset \bar{\chi}_2 \supset \cdots \supset \bar{\chi}_j$$

the criterion function J fulfills

$$J(\bar{\chi}_1) \geq J(\bar{\chi}_2) \geq \cdots \geq J(\bar{\chi}_j). \tag{1}$$

Each B&B algorithm constructs a search tree where each node represents some set of "candidates". The root represents the set of all D features and leafs represent target subsets of d features. While traversing the tree down to leafs the algorithm successively removes single features from the current "candidate"set ($\bar{\chi}_k$ in the k-th level) and evaluates the criterion value. In leafs the information about both the currently best subset $\mathcal{X}$ and the '*bound*' $X^* = J(\mathcal{X})$ is updated. Anytime the criterion value in some internal node is found to be lower than the current *bound*, due to the condition (1) the whole sub-tree may be cut-off and many computations may be omitted. For details see [1, 2, 5].

Several improvements of this scheme are known. The "Improved" B&B algorithm [2] combined with the "minimum solution tree" [9] concept can be considered the fastest non-predicting algorithm. This algorithm (to be referred as IBB) improves the search speed by optimising the tree topology and bypassing redundant computations in paths leading to leafs. The Fast Branch & Bound includes both of these improvements and incoroporates additional mechanisms to further reduce the impact of some of the principal B&B drawbacks [6] what makes it approximately 2 to 20 times faster than IBB in feature selection tasks, depending on data and criterion properties. As all optimal algorithms yield equal results at a cost of (in principle) exponential computational time, speed becomes the most important property to compare. It should be noted, that releasing the rigor of result optimality in sub-optimal algorithms is the only way to achieve fundamentally higher computational speed of polynomial nature.

2 Fast Branch & Bound

Let the *criterion value decrease* be the difference between the criterion value for the current feature subset and the value after removal of one feature. The FBB uses *criterion value decreases* estimates for future predictions of the criterion values. Prediction is used only in non-leaf nodes and can not trigger a node cut-off. If a predicted criterion value remains significantly higher than the current *bound*, it may be expected that even the real value would not be lower and therefore the corresponding sub-tree could not be cut-off. In this situation the FBB proceeds to the consecutive tree level. But, if the predicted value drops below the *bound*, the actual criterion value must be computed to evaluate the cut-off chance. Sub-trees may be cut-off only if true criterion values prove to be lower than the current *bound*, what preserves the optimality of the final result. Note that the only impact of possibly inaccurate predictions is prolonging some branches. However, this drawback is usually strongly outweighed by the overall criterion computation savings.

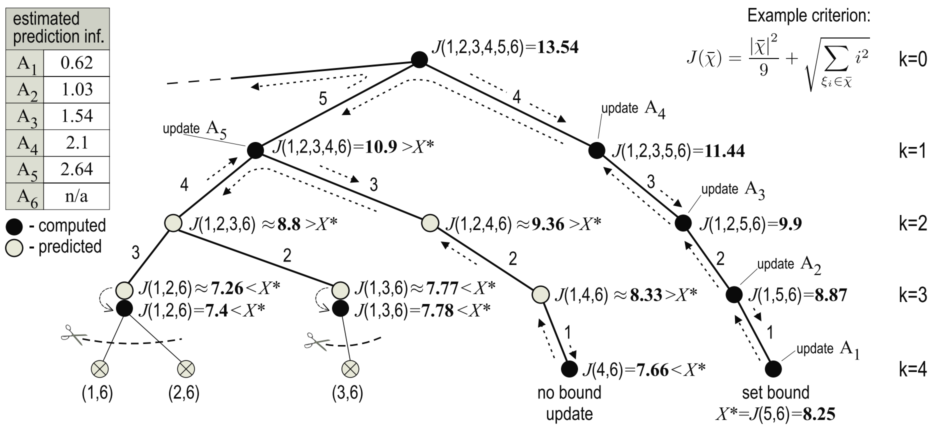


Fig. 1. Illustration of the prediction mechanism in Fast Branch & Bound for a problem of selecting $d = 2$ features out of $D = 6$ to maximise a synthetic criterion function.

See Fig. 1 for an illustration of the search process. The Figure shows initial stages of the search process on a synthetic problem where $d = 2$, $D = 6$. The prediction mechanism learns whenever two subsequent criterion values are computed (here for simplicity $A_i = J(\bar{\chi}) - J(\bar{\chi} \setminus \{i\})$ for $i = 4, 3, 2, 1, 5$) and later uses this information to replace criterion evaluation by a simple subtraction (in white nodes $J(\bar{\chi} \setminus \{i\}) \approx J(\bar{\chi}) - A_i$). The predicted values, being only approximations of the true criterion values, do not suffice to cut-off sub-trees and must be verified by true criterion evaluation whenever tree cutting seems possible (see nodes representing subsets 1,2,6 and 1,3,6) to preserve the optimality of the results.

2.1 Fast Branch & Bound with the Default Prediction Mechanism

The FBB *default prediction mechanism* is based on averaging feature contributions individually for each feature, independently on current tree level k. Averages are kept in 'contribution' vector $\mathbf{A} = [A_1, A_2, \ldots, A_D]^{\mathrm{T}}$, while the number of averaged values is stored in 'counter' vector $\mathbf{S} = [S_1, S_2, \ldots, S_D]^{\mathrm{T}}$. Both vectors are initially zeroed. Whenever FBB removes some feature y_i from the current "candidate" subset and computes the corresponding true criterion value $J(\bar{\chi}_k \setminus \{y_i\})$ at k-th tree level, and if also the predecessor value $J(\bar{\chi}_k) \equiv J(\bar{\chi}_{k-1} \setminus \{y_j\})$ (after previous removal of some feature y_j) had been computed (as indicated by $T_{k-1,y_j} =$"C"), the prediction mechanism vectors are updated as follows:

$$A_{y_i} = \frac{A_{y_i} \cdot S_{y_i} + J_{k-1,y_j} - J(\bar{\chi}_k \setminus \{y_i\})}{S_{y_i} + 1}, \quad S_{y_i} = S_{y_i} + 1 \tag{2}$$

For the formal FBB description we shall use the notion adopted from [1]:

$\bar{\chi}_k = \{\xi_j \mid j = 1, 2, \cdots, D-k\}$ – current "candidate" set at k-th tree level,

q_k – number of current node descendants (in consecutive tree level),

$\mathcal{Q}_k = \{Q_{k,1}, Q_{k,2}, \ldots, Q_{k,q_k}\}$ – ordered set of features assigned to edges leading to current node descendants (note that "candidate" subsets $\bar{\chi}_{k+1}$ are fully determined by features $Q_{k,i}$ for $i = 1, \cdots q_k$),

$\mathbf{J}_k = [J_{k,1}, J_{k,2}, \ldots, J_{k,q_k}]^{\mathrm{T}}$ – vector of criterion values corresponding to current node descendants in consecutive tree level ($J_{k,i} = J(\bar{\chi}_k \setminus \{Q_{k,i}\})$ for $i = 1, \cdots, q_k$),

$\Psi = \{\psi_j \mid j = 1, 2, \cdots, r\}$ – control set of r features being currently available for search-tree construction, i.e. for building the set $\mathcal{Q}_k$; set Ψ serves for maintaining the search tree topology,

$\mathcal{X} = \{x_j \mid j = 1, 2, \cdots, d\}$ – current best feature subset,

X^* – current *bound* (crit. value corresponding to $\mathcal{X}$).

$\delta \geq 1$ – *minimum number of evaluations, by default*$= 1$,

$\gamma \geq 0$ – *optimism, by default*$= 1$,

$\mathbf{T}_k = [T_{k,1}, T_{k,2}, \ldots, T_{k,q_k}]^{\mathrm{T}}$, $T_{k,i} \in \{$"C","P"$\}$ for $i = 1, \cdots, q_k$ – criterion value *type vector* (records the type of $J_{k,i}$ values–computed or predicted),

$\mathbf{V} = [v_1, v_2, \ldots, v_{q_k}]^{\mathrm{T}}$ – temporary sort vector,

Remark: values q_j, sets $\mathcal{Q}_j$ and vectors $\mathbf{J}_j$, $\mathbf{T}_j$ are to be stored for all $j = 0, \cdots, k$ to allow backtracking.

The Fast Branch & Bound Algorithm

Initialization: $k = 0$, $\bar{\chi}_0 = Y$, $\Psi = Y$, $r = D$, $\delta = 1$, $\gamma = 1$, $X^* = -\infty$.

STEP 1: *Select descendants of the current node to form the consecutive tree level:* First set their number $q_k = r - (D - d - k - 1)$. Construct $\mathcal{Q}_k$, $\mathbf{J}_k$ and $\mathbf{T}_k$ as follows: for every feature $\psi_j \in \Psi, j = 1, \cdots, r$ **if** $k + 1 < D - d$ (nodes are not leafs) and $S_{\psi_j} > \delta$ (prediction allowed), **then** $v_j = J_{k-1,q_{k-1}} - A_{\psi_j}$, i.e., predict by subtracting the appropriate prediction value based on ψ_j feature from the criterion value obtained in the parent node, **else** (nodes are leafs or prediction not allowed) the value must be computed, i.e., $v_j = J(\bar{\chi}_k \setminus \{\psi_j\})$. After obtaining all v_j values, sort them in the ascending order, i.e.,

$$v_{j_1} \leq v_{j_2} \leq \cdots \leq v_{j_r}$$

and for all $i = 1, \cdots, q_k$:

set $Q_{k,i} = \psi_{j_i}$ and
if v_{j_i} records a computed value, then set $J_{k,i} = v_{j_i}$ and $T_{k,i} =$"C"
else set $J_{k,i} = J_{k-1,q_{k-1}} - \gamma \cdot A_{\psi_{j_i}}$ and $T_{k,i} =$"P".
To avoid duplicate testing set $\Psi = \Psi \setminus \mathcal{Q}_k$ and $r = r - q_k$.
STEP 2: *Test the right-most descendant node (connected by the Q_{k,q_k}-edge):* if $q_k = 0$, then all descendants were tested and go to **Step 4** (backtracking). If $T_{k,q_k} =$"P" and $J_{k,q_k} < X^*$, then compute the true value $J_{k,q_k} = J(\bar{\chi}_k \setminus \{Q_{k,q_k}\})$ and mark $T_{k,q_k} =$"C". If $T_{k,q_k} =$"C" and $J_{k,q_k} < X^*$, then go to **Step 3**, else let $\bar{\chi}_{k+1} = \bar{\chi}_k \setminus \{Q_{k,q_k}\}$. If $k + 1 = D - d$, then a leaf has been reached and go to **Step 5**, else go to next level: let $k = k + 1$ and go to **Step 1**.
STEP 3: *Descendant node connected by the Q_{k,q_k}-edge (and its sub-tree) may be cut-off:* return feature Q_{k,q_k} to the set of features available for tree construction, i.e. let $\Psi = \Psi \cup \{Q_{k,q_k}\}$ and $r = r + 1$, $\mathcal{Q}_k = \mathcal{Q}_k \setminus \{Q_{k,q_k}\}$ and $q_k = q_k - 1$ and continue with its left neighbour; go to **Step 2**.
STEP 4: *Backtracking:* Let $k = k - 1$. If $k = -1$, then the complete tree had been searched through; stop the algorithm, else return feature Q_{k,q_k} to the set of "candidates": let $\bar{\chi}_k = \bar{\chi}_{k+1} \cup \{Q_{k,q_k}\}$ and go to **Step 3**.
STEP 5: *Update the bound value:* Let $X^* = J_{k,q_k}$. Store the currently best subset $\mathcal{X} = \bar{\chi}_{k+1}$ and go to **Step 2**.

3 Prediction Mechanisms

Here we define a set of alternative prediction mechanisms. Technically we change only the A_{y_i} estimation, i.e. formula (2). The estimation takes place under the same conditions as described in the preceeding section. The use of the alternative A_{y_i} values inside FBB instead of the default is principally the same, taking place in **STEP 1** only. For a list of defined mechanisms see Table 1.

The simplest *last-value* mechanism directly re-uses only the last computed contribution value. It is based on assumption that feature behaviour does not change too dramatically in local context.

The *level-based averaging* predictor should reduce the impact of *criterion value decrease* estimation errors with criterion functions yielding values strongly dependent on feature set size. As the estimation takes place separately for each tree level, this predictor may become compromised by the delay of prediction start and by the relatively lower number of true values available for learning when compared to the default, global averaging predictor.

The *maximising* and *minimising* predictors are likely to be outperformed by the others as they obviously yield biased predictions. Their purpose is to test the FBB vulnerability to "optimistic" (in case of *minimising*) and "pessimistic" (in case of *maximising*) errors caused by the predictor. The $A_{y_i}^{Max}$ and $A_{y_i}^{Min}$ values are expected here to cause similar effect as setting the optimism parameter γ either > 1 or < 1. In case of too pessimistic behaviour the accelerating effect of prediction mechanism on the FBB algorithm as a whole deteriorates, but the maximum number of search tree nodes remains equal to that of the IBB. In case

Table 1. List of considered prediction mechanisms.

Description	Definition	Comment
averaging	formula (2)	Default.
last-value	$A^{L}_{y_i} = J_{k-\bullet,y_j} - J(\bar{\chi}_k \setminus \{y_i\})$	Uses only the last value.
maximising	Let $A^{Max}_{y_i} = J_{k-\bullet,y_j} - J(\bar{\chi}_k \setminus \{y_i\})$ only if $A^{Max}_{y_i} < J_{k-\bullet,y_j} - J(\bar{\chi}_k \setminus \{y_i\})$	Uses the maximum value obtained so-far.
minimising	Let $A^{Min}_{y_i} = J_{k-\bullet,y_j} - J(\bar{\chi}_k \setminus \{y_i\})$ only if $A^{Min}_{y_i} > J_{k-\bullet,y_j} - J(\bar{\chi}_k \setminus \{y_i\})$	Uses the minimum value obtained so-far.
(max+min)/2	$A^{Mid}_{y_i} = (A^{Max}_{y_i} + A^{Min}_{y_i})/2$	"Middle" value.
level-based averaging	$A^{Lev}_{y_i,k} = \frac{A^{Lev}_{y_i,k} \cdot S_{y_i}\bullet\ J_{k-1,y_j} - J\bullet\chi_k \setminus \{y_i\}\bullet}{S_{y_i}\bullet\ \bullet}$	Same as default, but separately for each subset size.
individual	$A^{Ind}_{y_i} = J(\{y_i\})$	Constant predictor.
reverse individual	$A^{Rev}_{y_i} = J(Y) - J(Y \setminus \{y_i\})$	Constant predictor.

of too optimistic behaviour the FBB algorithm can unwantedly track the tree branches deeper than IBB what could result in worse performance loss then in the pessimistic case (for details see [6]). The *(max+min)/2* value is used as an alternative predictor as well.

The *individual* value predictor uses constant individual criterion values for each feature. It is defined to demonstrate the fact that individual feature evaluation often is not sufficient to estimate the value of feature sets. The *reverse individual* predictor analogously uses only the constant individual feature contributions with respect to the full set Y.

4 Experiments

The different predictors in the FBB algorithm were tested on a number of different data sets. Here we show results computed on 2-class mammogram Wisconsin Diagnostic Breast Center (WDBC) data (30 features, 357 benign and 212 malignant samples) and WAVEFORM data (40 features of which 19 represent noise, 1692 class 1 and 1653 class 2 samples) obtained via the UCI repository (ftp.ics.uci.edu)and 2-class SPEECH data originating at British Telecom (15 features, 682 word "yes" and 736 word "no" samples). We used the Bhattacharyya, Divergence and Patrick-Fischer distances. The Patrick-Fischer distance is considered difficult for use in B&B because of its strong dependence on evaluated set size. We used Pentium4-2,6Ghz CPU for all tests. As all the algorithms are optimal, they yield identical subsets identified by the same maximum criterion value. The only important difference between considered algorithms is therefore in computational time or in number of true criterion evaluations. Due to limited space we present only the graphs of computational time. It should be only noted, that practically all experiments proved a straightforward dependence between the number of criterion evaluations and overall time. However, it is difficult to describe this dependence precisely as the number of evaluations does not depend on criterion computational complexity while the overall computational time does.

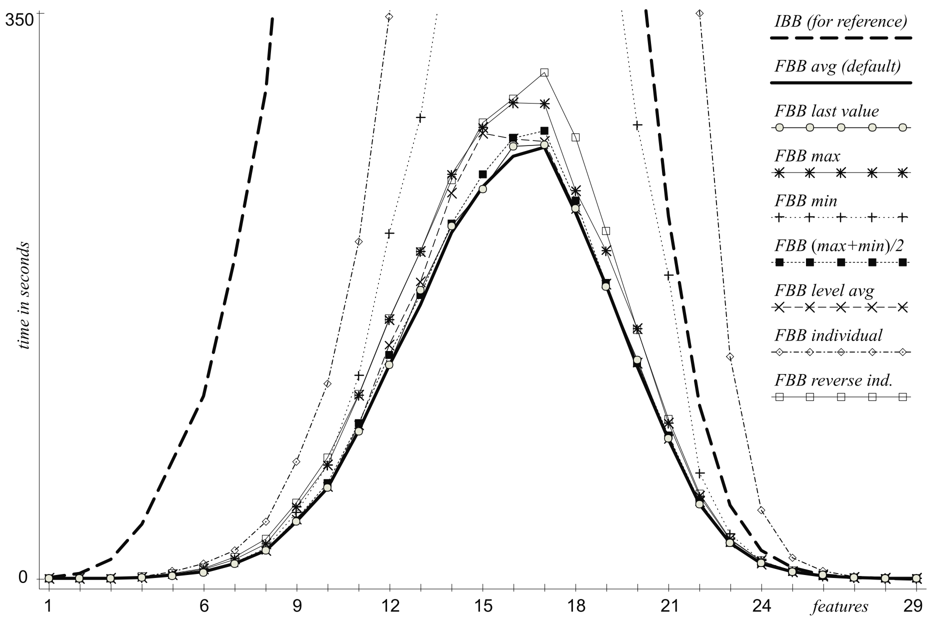


Fig. 2. FBB prediction efficiency – Bhattacharyya distance, WDBC data.

The results in Figures 2,3 and 4 show that the default *averaging* mechanism in FBB remains the best choice for general purpose. It markedly outperforms the referential IBB in all cases. Figures 2 and 3 show unexpectedly good performance of the simplest *last-value* predictor that becomes the fastest one in isolated cases. However, it becomes totally compromised in combination with the Patrick-Fischer criterion (Fig. 4). This illustrates the limits of using local information for generalization depending on criterion properties. The *level-based averaging* predictor performs comparably to the default *averaging*. It performs slightly worse in easier cases (Figs. 2,3) and slightly better in the difficult case (Fig. 4). It can be expected that the overall *level-based averaging* performance would improve with increasing problem sizes where more data becomes available for learning. A better than expected performance is observed for the *(max+min)/2* predictor that proves to be sufficiently good in simpler cases and excellent in the difficult case. It becomes a meaningful alternative to the default *averaging*, showing a good generalization ability.

The performance of the *maximising* predictor in Figs. 2 and 3 is noticeably worse than that of the default *averaging* but better in Fig. 4. This is the result of the invoked "pessimistic" algorithm behaviour, which slows-down the search in easy cases but helps to reduce the negative effect of "optimistic" errors if the learning process is compromised by noise in data or criterion properties. Similar or worse behaviour can be observed for the *reverse individual* predictor. The *minimising* and *individual* value predictors are shown to have a strongly negative impact on FBB performance. In case of the *minimising* predictor it confirms

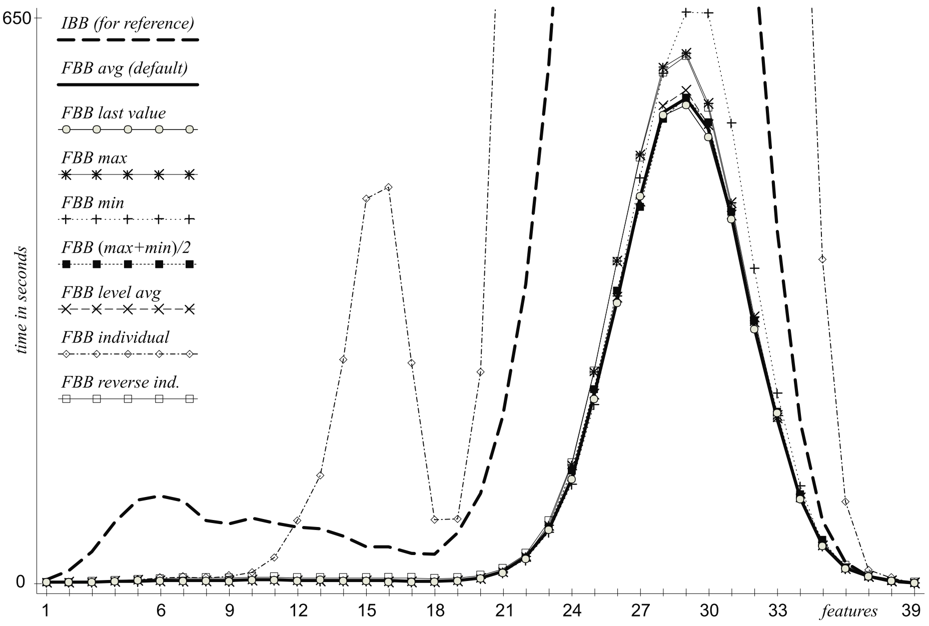


Fig. 3. FBB prediction efficiency – Divergence, WAVEFORM data.

the assumption that "optimistic" algorithm bahaviour (tracking branches deeper than necessary) can strongly deteriorate the FBB performance. The weak *individual* value predictor performance confirms that individual feature importance does not represent well its importance with respect to other features in a set. Remark: A more detailed study of additional aspects affecting the general B&B performance can be found in [8].

5 Conclusion

We have discussed in detail the recent Fast Branch & Bound optimal feature selection algorithm with respect to its core concept of prediction mechanism. Alternative predictors have been defined and investigated. The original *averaging* prediction mechanism has been verified to be the good option for general purpose. However, the simple *last-value* predictor shows to perform equally well for some criteria while being simpler to implement and requiring less computational overhead. The *level-based averaging* predictor is to be recommended especially for high-dimensional tasks and/or criteria where the feature contributions are known to be subset-size dependent. The *(max+min)/2* predictor has proved to be worth consideration as an alternative for general purpose. Regardless the differences between these prediction mechanisms the performance of the respective FBB algorithm versions is generally better than that of the older optimal search algorithms like the Improved Branch & Bound. This demonstrates the robustness of the prediction mechanism concept as such.

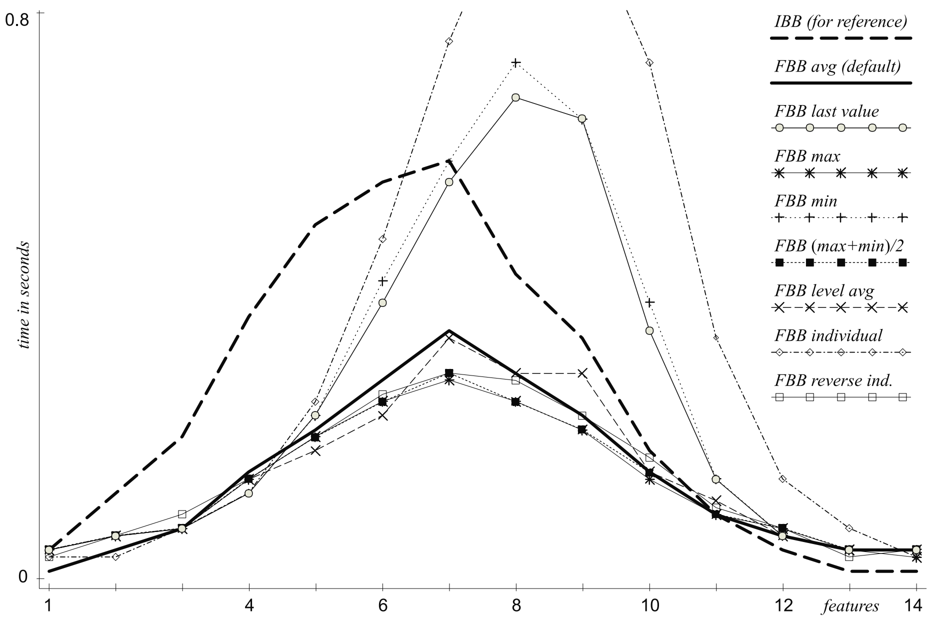


Fig. 4. FBB prediction efficiency – Patrick-Fischer distance, SPEECH data.

References

1. P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice-Hall, 1982.
2. K. Fukunaga. *Introduction to Statistical Pattern Recognition: 2nd edition.* Academic Press, Inc., 1990.
3. Y. Hamamoto, S. Uchimura, Y. Matsuura, T. Kanaoka and S. Tomita. Evaluation of the branch and bound algorithm for feature selection. *Pattern Recognition Letters*, 11(7): 453–456, July 1990.
4. M. Kudo and J. Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33(I): 25–41, January 2000.
5. P.M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, C-26: 917–922, September 1977.
6. P. Somol, P. Pudil, F. J.Ferri and J. Kittler. Fast Branch & Bound Algorithm in Feature Selection. *Proc. 4th World Multiconference on Systemics, Cybernetics and Informatics SCI 2000, Orlando, Florida*, Vol VII, Part 1: 646–651, 2000.
7. P. Somol, P. Pudil and J. Grim. Branch & Bound Algorithm with Partial Prediction For Use with Recursive and Non-Recursive Criterion Forms. *Lecture Notes in Computer Science*, Springer Verlag, Vol. 2013, 230–238, 2001.
8. P. Somol, P. Pudil, and J. Kittler. Fast Branch & Bound Algorithms in Feature Selection. *To appear in IEEE Transactions on PAMI*, July 2004.
9. B. Yu and B. Yuan. A more efficient branch and bound algorithm for feature selection. *Pattern Recognition*, 26: 883–889, 1993.

Structures of Covariance Matrix in Handwritten Character Recognition

Šarūnas Raudys[1] and Masakazu Iwamura[2]

[1] Vilnius Gediminas Technical University, Saulėtekio 11, Vilnius, Lithuania
raudys@ktl.mii.lt
[2] Tohoku University, Aoba 05, Aramaki, Aoba-ku, Sendai, 980-8579 Japan
masa@aso.ecei.tohoku.ac.jp

Abstract. The integrated approach is a classifier established on statistical estimator and artificial neural network. This consists of preliminary data whitening transformation which provides good starting weight vector, and fast training of single layer perceptron (SLP). If sample size is extremely small in comparison with dimensionality, this approach could be ineffective. In the present paper, we consider joint utilization of structures and conventional regularization techniques of sample covariance matrices in order to improve recognition performance in very difficult case where dimensionality and sample size do not differ essentially. The techniques considered reduce a number of parameters estimated from training set. We applied our methodology to handwritten Japanese character recognition and found that combination of the integrated approach, conventional regularization and various structurization methods of covariance matrix outperform other methods including optimized Regularized Discriminant Analysis (RDA).

1 Introduction

One of characteristic elements of modern pattern classification tasks is extremely large number of features that are of the similar origin. An example is classification of handwritten Japanese characters. Since the features are mutually correlated, one cannot ignore the correlations for designing the pattern classification algorithm. To reduce c*omplexity/sample size problems*, one needs to structurize covariance matrix (CM), i.e. describe it by small number of parameters. Two decades ago such approach has been used for classification of time series [1, 2], 2D remote sensing image classification [3-7]. Structurization approach has been utilized also in recognition of handwritten Japanese characters, too.

In many real world problems, distributions density functions of single features have clear deviation from Gaussian law. Promising way to solve such pattern recognition tasks is utilization of artificial neural networks based methods which do not require assumptions about type of distribution density functions of input features. In case of successful training, often one obtains good results. There are two main difficulties to apply such methods. First, results obtained depend on initial conditions (weight vector). Secondly, if input features are highly correlated, in high-dimensional situations the data becomes almost singular. This makes training become very slow.

A way to diminish the perceptron initialization problem and singularity of the data is the integrated approach of statistical and neural networks based methods [8-10]. Instead of using statistical estimate of CM to design the statistical classifier (denoted by CL_s), we use CM for data whitening transformation. In subsequent training of SLP, this strategy leads classifier CL_s just after the first bach-mode training with zero valued initial weight in the transformed feature space.

If the assumption of structure of the CM is truth and *sample size/complexity relationship* is sufficiently high, we have a good start to train the perceptron further. Good initialization leads to high-quality result if one stops training in a right moment [11]. Moreover, data whitening speeds up training process.

The integrated approach has been derived with the assumption that CM's of both classes are the same. This approach could be ineffective because of unequal CM's. It also could be ineffective when the assumptions of the structures of the CM are far from reality, due to use of wrong covariance structures or use of unreliable estimates which are calculated from small samples for the dimensionality. To improve effectiveness of the integrated approach, one can introduce additional regularization of the CM. An objective of the present paper is to investigate joint application of the CM regularization, standard and special CM structures designed for 2D spatial image recognition to the integrated approach for discrimination of handwritten characters. We performed experiments with similar pairs of Japanese characters (Fig. 1), however, our methodology is not application specific.

鳥 烏	采 釆	伸 仲	乎 平	東 束	熊 態	帥 師
栗 粟	ぽ ぼ	麿 磨	棒 捧	問 間	椎 推	肋 助

Fig. 1. Fourteen pairs of similar Japanese characters.

The standard CM structures are widely used structures, and the special ones are prepared with taking into account nature of feature vectors of 2D images: distant pixels in the 2D image have less important correlations. The covariance matrices of similar classes are expected to be similar as well as the postulated correlations structures be truthful. We performed our investigation of 2 class discrimination in very difficult condition where the number of sum of sample sizes, $n=N_1+N_2$, and dimensionality, p, are approximately equal. N_i is training sample size of class i.

2 Integrated Approach of Statistical Estimators and Artificial Neural Networks

2.1 Standard Fisher Linear Discriminant Function

The standard Fisher linear discriminant function is the most important rule to classify two categories, and offered the opportunity to give birth to the integrated approach. Suppose both pattern classes share a common covariance matrix. Denote the pooled sample covariance matrix by **S** and the sample mean vectors of two classes by $\bar{\boldsymbol{x}}^{(1)}$ and

$\bar{\boldsymbol{x}}^{(2)}$. Then, allocation of a p-variate vector $\boldsymbol{x} = (x_1, \cdots, x_p)^T$ is performed according to a sign of discriminant function (DF)

$$g(\boldsymbol{x}) = \left(\boldsymbol{x} - \frac{1}{2}\left(\bar{\boldsymbol{x}}^{(1)} + \bar{\boldsymbol{x}}^{(2)}\right)\right)^T \mathbf{S}^{-1}\left(\bar{\boldsymbol{x}}^{(1)} - \bar{\boldsymbol{x}}^{(2)}\right). \tag{1}$$

Instead of **S**, a "better" (simplified) estimate of the covariance matrix (say $\mathbf{S}_\mathrm{S}$) could convert DF (1) into (another) statistical classifier CL_s with possibly enhanced small sample properties.

2.2 Integrated Approach

In the integrated approach, the learning process consists of two stages: data whitening transformation by statistically estimated CM, and subsequent learning of SLP. Recognition is performed with trained SLP in transformed space.

Preliminarily, all the samples (including test ones) are moved so that the mean of the training set becomes at the origin of the coordinate (i.e., $\bar{\boldsymbol{x}}^{(1)} + \bar{\boldsymbol{x}}^{(2)} = 0$).

Data Whitening Transformation

Let $\mathbf{\Lambda}$ and $\mathbf{\Phi}$ be the eigenvalues matrix and eigenvectors matrix of sample estimate of simplified covariance matrix $\mathbf{S}_\mathrm{S}$, i.e., $\mathbf{S}_\mathrm{S} = \mathbf{\Phi\Lambda\Phi}^T$. All the test and training samples are transformed by $\boldsymbol{y} = \mathbf{\Lambda}^{-1/2}\mathbf{\Phi}^T\boldsymbol{x}$. This transformation makes zero valued weights be good initial ones for subsequent learning of SLP.

Learning of SLP

Let the initial weight be zero vector. Then, the perceptron is trained by gradient descent method. After the first batch iteration, we obtain discriminant function $g(\boldsymbol{y}) = \left(\boldsymbol{y} - \tfrac{1}{2}\left(\bar{\boldsymbol{y}}^{(1)} + \bar{\boldsymbol{y}}^{(2)}\right)\right)^T \left(\bar{\boldsymbol{y}}^{(1)} - \bar{\boldsymbol{y}}^{(2)}\right)k_\mathrm{E}$, where $\bar{\boldsymbol{y}}^{(1)}$ and $\bar{\boldsymbol{y}}^{(2)}$ are the sample mean vectors in the transformed space, and k_E is a scalar constant. This DF is equal to transformed DF (1), i.e. $g(\boldsymbol{y}) = \left(\boldsymbol{y} - \tfrac{1}{2}\left(\bar{\boldsymbol{y}}^{(1)} + \bar{\boldsymbol{y}}^{(2)}\right)\right)^T \left(\bar{\boldsymbol{y}}^{(1)} - \bar{\boldsymbol{y}}^{(2)}\right)$. The data whitening transformation gives good initial weights for training of SLP as long as the both classes share common CM and the distributions are well-estimated. For more details, see book [10]. Theoretically and practically, in subsequent training, SLP outperforms Fisher classifier if samples are non-Gaussian.

3 Structurized and Regularized Estimates of Covariance Matrix

In our research work, nine kinds of covariance matrix models are used (Fig. 4). First of all, three models without clear structurization of CM were considered. They are statistical estimators rather than structurization methods. Model **FULL** use regularized pooled CM $\mathbf{S}_{\text{FULL}} = \mathbf{S}$. In model **FULL**, we consider all $p(p$-1)/2 correlations (off-diagonal elements of CM). Model **NO** uses identity matrix. Note all CM models in this section become **NO** when $\lambda_0 = 1$. Model **SQDF** is a method which bases statistical classifier SQDF [12]. Unlike to other models, class-dependent CM's are separately calculated and then pooled. Small eigenvalues of each CM are replaced by a constant. The constant is estimated by the maximum likelihood estimation. Here, the number of eigenvalues which are not constant is 5.

Based on the assumption that *most correlations between distant pixels are low*, three types of fixed structure models specialized for feature vectors of 2D image were used (see the description for the feature vector at the very beginning of Section 4). The fixed structure models used block-diagonal CM: **49B4** has 49 independent 4×4 blocks, **4B49** has 4 blocks of size 49×49, and **4B&49B** covers both regions of **49B4** and **4B49**. Because both **49B4** and **4B49** are rather restrictive ones, less restrictive and more sophisticated model, **4B&49B**, is designed.

In addition to fixed structure models, we investigated three adaptive structured models. As a "dumb" model, we employ **LARG**, where much smaller correlations of CM are ignored so that the CM becomes close to diagonal. The second one is standard first-order tree dependence model **TREE1** [8, 9]. Here, it is postulated that each feature depends only on one other feature. Therefore, an inverse of the matrix , $(\mathbf{S}_{\text{TREE1}})^{-1}$, however, has 2$p$-1 non-zero elements. In general case, however, the inverse of non-structurized CM has p×p non-zero elements. In the previous research studies [9], most often this model appeared as a best one in moderate sample size situations. The last one is **EBD** which block-diagonalizes CM after exchanging elements of the matrix in order that sub-covariance matrices contain larger elements [13]. 14 is used for the number of sub-matrices.

In the earlier stage of the investigation, we also considered scaled rotation regularization [10, 14]. In experiments with 196-dimensional data and relatively small learning sets (100 samples), this regularization method was too complex and ineffective.

If the number of training samples is too small, stucturized estimate of CM, $\mathbf{S}_{\text{S}}$, is unreliable. For more reliable estimation, we are obliged to introduce additional regularization, i.e. $\mathbf{S}_{\text{S\&RDA}} = (1-\lambda)\mathbf{S}_{\text{S}} + \lambda\mathbf{I}$, where λ is a parameter for regularization. If $\lambda = 0$, we have no regularization. If $\lambda = 1$, we have no structurization (case **NO**). Intermediate values of parameter λ could improve accuracy of determination of the weight vector obtained after the first batch iteration. Because good initial weight vector leads good result if training would be stopped in a right moment, proper additional regularization should assist in reducing generalization error.

Note, if regularization is applied to conventional sample estimate of CM (in case of **FULL**), in dependence on number of iterations we have RDA ($\mathbf{S}_{\text{RDA}} = (1-\lambda)\mathbf{S} + \lambda\mathbf{I},$) with different λ (e.g., see Eq.(4.9) in [10]). RDA is known as one of the best classification methods in statistical pattern recognition.

4 Experiments

In the experiments, 196-dimensional directional element feature [15] was used to represent handwritten Japanese characters in database ETL9B. Preliminary to extracting the feature vector, a character image was normalized nonlinearly [16] to fit in a 64×64 box. Then, skeleton were extracted, and line segments of vertical, horizontal and slanted at ±45 degrees were extracted. An image is divided into 49 sub-areas of 16×16 dots (see Fig.3). Sum of each segment in a region is an element of feature vector.

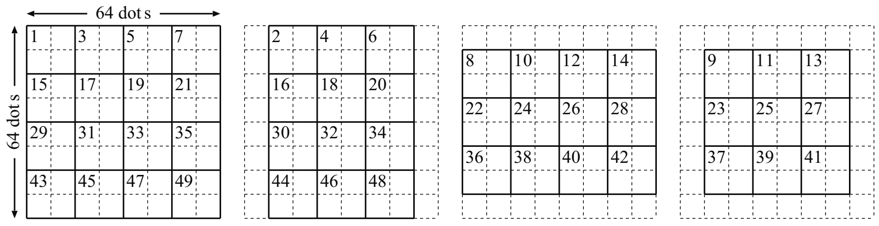


Fig. 2. 49 sub-areas of feature vector.

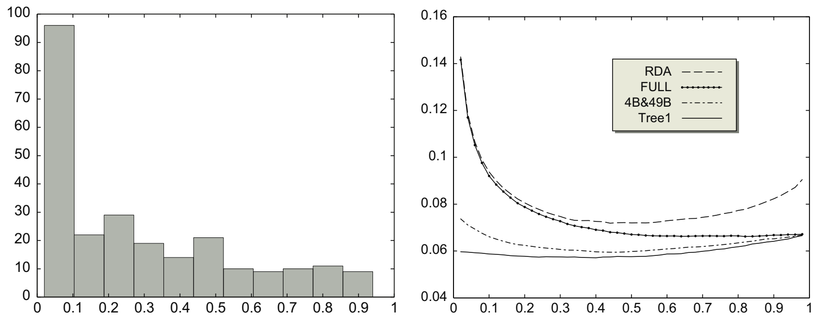
Our purpose is to investigate potential possibilities of each of pattern classification method (strategy) in very difficult case where training set size $n=N_1+N_2\approx p$. Therefore, for 196-dimensional feature vector, we considered $N = N_1= N_2 = 30, 50, 100, 150$. This is a really critical situation of small sample/high dimensionality problem. In each experiment, we used test sets to find optimal regularization parameter which achieves minimum error rate. Each time we permuted 200 vectors in each pattern class. In each category, N samples were selected for training and remaining 200-N ones were for testing.

Preliminary experiments demonstrated that test error estimates depend on value λ notably. Optimal values of λ depend on CM structurization method, training set size and also on random split of data into training and test sets. In Fig. 3a, we present typical histogram of distribution in 250 experiments for model **Tree1**. In Fig. 3b, we have generalization errors as function of λ for five CM structurization methods (**RDA, FULL**, **4B&49B** and **Tree1**) calculated from 250 experiments.

We analysed bivariate distributions of optimal values in Fig.3a. We found there is no or very small correlations between two distinct CM models considered. This means for each CM structurization model, one needs utilize its own (best) value of λ. Accordingly, optimal λ is CM structurization method dependent. For this reason, for each pair and CM structurization method for all handwritten character pairs in Fig.1 (named as **A** to **N** from upper left pair), we performed ten preliminary experiments to evaluate approximately a fixed value of optimal λ to be used in the main experiments.

Average results obtained in 100 experiments for character pairs are presented in Table 1. We see that joint utilization of prior information in form of postulated structure and additional regularization of CM are useful even when parameter λ is determined approximately. We found that there is no single CM structure best for all handprinted character pairs. Most often, fixed structure models such as **49B4**, **4B49** and **4B&49B** were the best. In several cases, statistical structure models such as **Tree1** and **SQDF** outperformed the fixed structure models.

Experiments with different training set sizes are shown in Table 2. This also confirmed usefulness of joint utilization of the CM structurization and regularization. The generalization error decreases uniformly with training set size *N*. The best two CM structurization models do not change with an increase in training set size.



(a) Distribution of optimal λ (b) Generalization errors as function of λ

Fig. 3. Optimal λ and generalization errors (Pair **M**, *N*=100, in 250 experiments): (a) distribution of optimal regularization parameter λ for model **Tree1**, and (b) generalization errors as functions of λ for **RDA**, **FULL**, **4B&49B** and **Tree1**.

Table 1. Average generalization errors for different character pairs of **FULL**, and relative ratios of generalization error of each method to generalization error of **FULL** (right 9 columns). The very last rows in the table are average values of the column.

Pair	FULL	NO	SQDF	49B4	4B49	4B&49B	LARG	TREE1	EBD	RDA
A	0.1955	0.9974	**0.9514**	0.9974	1	1.0051	0.9949	0.9923	0.9974	1.0691
B	0.1475	1.0034	0.9695	0.9864	0.9966	0.9831	1.0102	**0.9322**	1.0034	1.0508
C	0.0900	1.0056	1	0.9944	1	1	0.9944	**0.9889**	1.0056	1.1556
D	0.1405	0.9929	1.0036	**0.9893**	**0.9893**	**0.9893**	1.0071	1.0036	0.9929	1.0747
E	0.1465	1.0034	1.0137	1	1.0102	1.0068	1.0171	1.0068	**0.9863**	1.0819
F	0.0410	1.0366	**0.9390**	1.0244	0.9634	0.9756	0.9634	0.9756	1.0122	1.0854
G	0.0690	1.0290	**0.9783**	1.0290	1	1.0072	1.0145	1	1.0217	1.0217
H	0.0810	1.0556	1.0309	1.0556	**0.9506**	0.9815	1.0556	0.9753	1.0617	1.0864
I	0.1515	1.0033	1.0033	0.9967	**0.8515**	0.8746	0.967	0.9703	0.9934	1.0627
J	0.0725	1.0138	0.9862	0.9034	0.9862	**0.9241**	0.9724	1.0069	1.0138	1.0828
K	0.0950	0.9895	0.9842	0.9368	0.9579	0.9211	0.9211	**0.9158**	0.9632	1.0368
L	0.0785	1.0191	**0.9554**	1.0191	1.0127	1.0127	1.0255	1.0127	0.9873	1.1274
M	0.0640	1.0391	1.0156	1.0078	0.9531	0.9531	0.8828	**0.7891**	1.0313	1.1172
N	0.0450	1.0111	0.9889	**0.9778**	1	1	1.0111	1.0111	0.9889	1.0667
Mean	0.1012	1.0143	0.9871	0.9942	0.9765	0.9739	0.9884	**0.9700**	1.0042	1.0799

Table 2. Average generalization errors for different training set sizes, *N*, and diverse CM structurization methods (Pair **M**).

N	FULL	NO	SQDF	49B4	4B49	4B&49B	LARG	TREE1	EBD	RDA
30	0.1151	0.1145	0.1146	0.1118	0.1089	0.1052	0.1116	**0.0965**	0.1145	0.1250
50	0.0895	0.0902	0.0914	0.0883	0.0846	0.0820	0.0865	**0.0769**	0.0904	0.0980
100	0.0624	0.0646	0.0658	0.0633	0.0595	0.0582	0.0614	**0.0565**	0.0648	0.0691
150	0.0537	0.0543	0.0549	0.0515	0.0491	0.0487	0.0490	**0.0452**	0.0550	0.0608

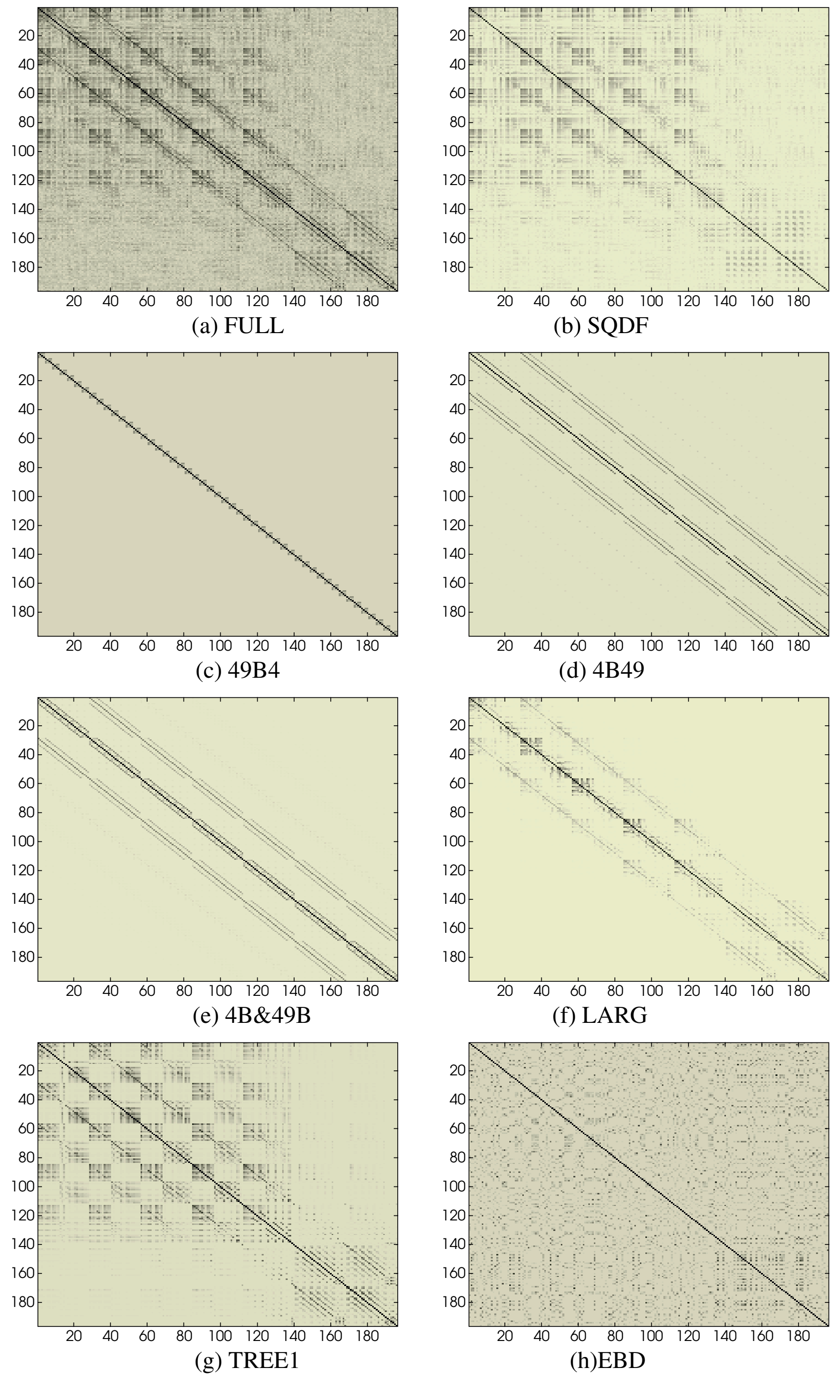


Fig. 4. Elements of structurized covariance matrices except model **NO**. Darker pixel stands for larger absolute value.

5 Concluding Remarks

In the current paper, we considered performance of the integrated approach of statistical estimators and neural networks. The main purpose is to investigate potential possibilities of this approach combined with various strategies under very difficult condition where the number of sum of training vectors is almost dimensionality. We used similar pairs of handwritten Japanese characters. This aggravates more difficult situation.

The strategies we used were 1) utilization of prior information in form of postulated structure of covariance matrix; 2) regularization of CM; 3) solution of the perceptron is regularized by early stopping before a minimum of the cost function. As prior information, nine kinds of structurization methods (models) were used. They also could be grouped as statistical models, fixed structure ones and adaptive structure ones. Fixed structure models are designed for feature vector of 2D spatial image. Regularization of CM directly improves data transformation which gives initialization of the perceptron. The number of learning steps of SLP decides complexity of pattern classification algorithm, too.

In experiments, utilization of structure models allowed us to reduce generalization error for most of the character pairs. For all 14 character pairs considered, error of “the best method” was on average 1.15 times smaller in comparison with the optimized regularized discriminant analysis. It was 1.05 times smaller than that of SLP with regularized maximum likelihood covariance matrix (model **FULL**) utilized for preliminary data transformation. Results of our research pointed out that joint utilization of structurization and conventional regularization of CM has a potential to improve efficacy of the integrated approach in designing pattern classifiers. The experiments show no structure is the best for all pairs. Therefore, the best structure and regularization parameter have to be selected for each pair and sample size respectively.

All three regularization techniques are acting simultaneously in the same directions. Thus, each of them can influence (reduce) effectiveness of other two. The effects of CM structures were also aggravated by the fact that most of distributions of the input features are highly asymmetric or bimodal, i.e. assumptions about Gaussian distributions were violated markedly [17]. In future research, the effects of factors aggravating positive effects have to be considered in detail. Practical techniques to select proper values of regularization parameters and optimal iteration should be developed.

Acknowledgments

The authors thank Assoc. Prof. Shinichiro Omachi, Prof. Hirotomo Aso, Dr. Ausra Saudargiene and Giedrius Misiukas for useful discussions, shared with us data sets and Matlab codes.

References

1. D. Morgera, D.B. Cooper. Structurized estimation: Sample size reduction for adaptive pattern classification. *IEEE Trans. Information Theory*, 23:728-741,1977.

2. V. Kligys. On the classification of multivariate Markov sequences. *Statistical Problems of Control*, Inst. of Math. and Cyb. Press, Vilnius, (S. Raudys, ed.), 50:57-75, 1981 (in Russian).
3. D.A. Landgrebe. The development of a spectral-spatial classifier for earth observational data. *Pattern Recognition*, Vol. 12:185-175, 1980.
4. D. Morgera. Linear, structured covariance estimation: An application to pattern classification for remote sensing. *Pattern Recognition Letters*, 4(1): 1-7, 1986.
5. G. Palubinskas. Spatial image recognition. *Statistical Problems of Control*, Inst. of Math. and Cyb. Press, Vilnius, (S. Raudys, ed.), 74:104-113, 1986 (in Russian).
6. G. Palubinskas. A comparative study of decision making algorithms in images modeled by Gaussian random fields. *Int. J. of Pattern Recognition and Artificial Intelligenc*e. Vol. 2(4):621-639, 1988.
7. G. Palubinskas. A review of spatial image recognition methods. *Statistical Problems of Control*, Inst. of Math. and Cyb. Press, Vilnius, (Raudys S., ed.), 93:215-231, 1990 (in Russian).
8. S. Raudys, A. Saudargiene. Structures of the covariance matrices in the classifier design. *Lecture Notes in Computer Science*, Springer-Verlag, 1451:583–592, 1998.
9. S. Raudys, A. Saudargiene. Tree type dependency model and sample size - dimensionality properties. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 23(2):233-239, 2001.
10. S. Raudys. *Statistical and Neural Classifiers: An integrated approach to design*. Springer, NY, 2001.
11. S. Raudys, S. Amari. Effect of initial values in simple perception. *Proceedings 1998 IEEE World Congress on Computational Intelligence, IJCNN'98*, 1530-1535, 1998.
12. S. Omachi, F. Sun, H. Aso. A new approximation method of the quadratic discriminant function. *Lecture Notes in Computer Science*, 1876: 601-610, 2000.
13. F. Sun, S. Omachi, N. Kato, H. Aso, S. Kono, T. Takagi. Two-stage computational cost reduction algorithm based on Mahalanobis distance approximations. *Proceedings 15th Int. Conf. on Pattern Recognition* (ICPR2000), IEEE Press, 2:700-703, 2000.
14. S. Raudys. Scaled rotation regularization. *Pattern Recognition* 33:1989–1998, 2000.
15. N. Sun, Y. Uchiyama, H. Ichimura, H. Aso, M. Kimura: Intelligent recognition of characters using associative matching technique. *Proc. Pacific Rim Int'l Conf. Artificial Intelligence* (PRICAI'90), 546-551, 1990.
16. H. Yamada, K. Yamamoto, T. Saito. A nonlinear normalization method for handprinted kanji character recognition - line density equalization. *Pattern Recognition*, 23(9):1023-1029, 1990.
17. S. Raudys, M. Iwamura. Multiple classifiers system for reducing influences of atypical observations. *Lecture Notes in Computer Science* (Proceedings of *Multiple Classification Systems*; MCS 2004).

Statistical Machine Translation Decoding Using Target Word Reordering

Jesús Tomás and Francisco Casacuberta

Institut Tecnològic d'Informàtica, Universidad Politécnica de Valencia
46071 Valencia, Spain
{jtomas,fcm}@upv.es

Abstract. In the field of pattern recognition, the design of an efficient decoding algorithm is critical for statistical machine translation. The most common statistical machine translation decoding algorithms use the concept of partial hypothesis. Typically, a partial hypothesis is composed by a subset of source positions, which indicates the words that have been translated in this hypothesis, and a prefix of the target sentence. Thus, the target sentence is generated from left to right obtaining source words in an arbitrary order. We present a new approach, where the source sentence is translated from left to right and the possible word reordering is performed at the target prefix. We implemented this approach using a multi-stack decoding technique for a phrase-based model, and compared it with both a conventional approach and a monotone approach. Our experiments show how the new approach can significantly reduce the search time without increasing the search errors.

1 Introduction

Statistical methods have proven to be valuable in tasks such as automatic speech recognition, and they present a new opportunity for automatic translation. The goal of statistical machine translation is to translate a given source language sentence $\mathbf{f}= f_1^{|\mathbf{f}|} = f_1 ... f_{|\mathbf{f}|}$ to a target sentence $\mathbf{e}= e_1^{|\mathbf{e}|} = e_1 ... e_{|\mathbf{e}|}$. The pattern recognition methodology most commonly used [2] is based on the definition of a function Pr(**e**|**f**) that returns the probability of translating a given source sentence **f** into a target sentence **e**. Once this function is estimated, the problem can be formulated as the search for a sentence **e** that maximizes the probability Pr(**e**|**f**) for a given **f**. Using Bayes' theorem, we can decompose the initial function in the language model and the inverse translation model:

$$\mathbf{e}' = \arg\max_{\mathbf{e}} (\Pr(\mathbf{e}) \Pr(\mathbf{f} \mid \mathbf{e})) \quad (1)$$

Nearly all statistical translation models try to establish the correspondence between source and target words by introducing the hidden variable of alignment [2]. Once the concept of alignment is formalized (see section 3 for a description), we introduce the variable of alignment, **a**, into the previous equation in order to obtain the sum of all possible alignments between **e** and **f**. However, the search is usually performed using the so-called maximum approximation [8]:

$$\mathbf{e}' = \arg\max_{\mathbf{e}} \left(\Pr(\mathbf{e}) \sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \right) \approx \arg\max_{\mathbf{e}} \left(\Pr(\mathbf{e}) \max_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \right) \tag{2}$$

There are different approaches to define the concept of alignment. The most common approaches are the single-word-based alignment models. Models of this kind assume that an input word is generated by only one output word [1][2]. This assumption does not correspond to the nature of natural language.

One initiative for overcoming the above-mentioned restriction is known as the template-based approach [7]. In this approach, an entire group of adjacent words in the source sentence may be aligned with an entire group of adjacent target words. A template establishes the reordering between two sequences of word classes. However, the lexical model continues to be based on word-to-word correspondence.

Recently, a simple alternative to these models has been proposed, the phrase-based (PB) approach [5][10][13]. That is commented in the section 3.

2 Decoding Algorithms

In this section, we describe the most common statistical decoding algorithms. With the exception of the greedy algorithm [4], the rest of them use the concept of partial translation hypothesis to perform the search [1][8][12]. In a partial translation hypothesis, some of the source words have been used to generate a target prefix. Each hypothesis is scored according to the translation and language model. The most typical partial hypothesis comprises:

- $w \subset \{1..|\mathbf{f}|\}$: The coverage set indicates the positions of the source sentence that has been translated by this hypothesis.
- e_1^{lk} : The target prefix, where lk is the prefix length.
- g: The score of the partial hypothesis.

The translation procedure can be described as follows: The system maintains a large set of hypotheses, each of which has a corresponding translation score. This set starts with an initial empty hypothesis. The system selects one partial hypothesis to extend. The extension consists of selecting one or more untranslated words in the source sentence and also selecting one or more target words that are attached to the existing output prefix. In the new hypothesis, the source words are marked as translated and the probability cost of the hypothesis is updated. The extension of a partial hypothesis can generate hundreds of new partial hypotheses. The output of the search is the hypothesis that has the highest score and no untranslated source words (final hypothesis).

The strategies for selecting one partial hypothesis to be extended can be grouped into three types: If the best-first search is used, the A* algorithm is obtained; if the deep-first search is used, the multi-stack decoding algorithm is obtained. If the breadth-first search is used, the dynamic-programming algorithm is obtained.

2.1 A* Algorithm

The A* algorithm [4][8][12] is also known as the stack-decoding algorithm. All search hypotheses are managed in a priority queue (stack) ordered by their scores. This algorithm has one main drawback: short hypotheses have higher scores, therefore, the algorithm has a tendency to expand the short hypotheses first and the search is very slow. To solve this problem, we need to introduce the heuristic function [8][12]. This function estimates the probability of completing a partial hypothesis.

2.2 Multi-stack Decoding Algorithm

This algorithm, which was proposed in [1], tries to solve the problem of the previous algorithm by using a different approach. It uses a different stack to store the hypothesis depending on which words in the source sentence have been translated (the coverage set). Therefore, we may need up to a total of $2^{|\mathbf{f}|}$ stacks. This procedure allows us to force the expansion of hypotheses with a different degree of completion. In each iteration, the algorithm covers all stacks with some hypotheses and extends the best one for each. After the first iteration, there is at least one final hypothesis. [1] proposes using the best final hypothesis, after each iteration, to establish a pruning criterion in order to erase the partial hypotheses that cannot improve the best final hypothesis.

2.3 Dynamic-Programming Algorithm

In this algorithm, the partial hypotheses are stored in the nodes of a search graph. This graph is explored in a breadth-first manner, that is, when a hypothesis is extended, all their predecessors[1] have also been extended.

The search space can be very large, so some authors make simplifying assumptions about the search space. [3] and [6] propose algorithms for IBM model 2, and [9] proposes algorithms for a monotone model. However, the most frequent strategy for making dynamic-programming feasible is beam search [9][13]. Beam search is based on histogram pruning, that is, in each node of graph, we extend only the H better hypotheses.

3 The Phrase-Based Model

In this approach the probability of a sequence of words in a source sentence being translated to another sequence of words in the target sentence is explicitly learnt. To define the PB model, we segment the source sentence $\mathbf{f}$ into K phrases ($\tilde{f}_1^K$) and the target sentence $\mathbf{e}$ into K phrases ($\tilde{e}_1^K$). A uniform probability distribution over all possible segmentation is assumed ($\alpha(\mathbf{e})$).

[1] h' is a predecessor of h if the words translated by h' are a subset of the words translated by h.

$$\Pr(\mathbf{f}\,|\,\mathbf{e}) = \alpha(\mathbf{e}) \sum_{K} \sum_{\tilde{e}_1^K : \tilde{e}_1^K = \mathbf{e}} \sum_{\tilde{f}_1^K : \tilde{f}_1^K = \mathbf{f}} \Pr(\tilde{f}_1^K \,|\, \tilde{e}_1^K) \tag{3}$$

If we assume a monotone alignment, that is, the target phrase in position *k* is produced only by the source phrase in the same position [10], we can write:

$$\Pr(\tilde{f}_1^K \,|\, \tilde{e}_1^K) = \prod_{k=1}^{K} p(\tilde{f}_k \,|\, \tilde{e}_k) \tag{4}$$

Where the parameter $p(\tilde{f}\,|\,\tilde{e})$ estimates the probability that the phrase, $\tilde{e}$, be translated to the phrase $\tilde{f}$. These are the only parameters of this model. A phrase can be comprised by a single word. Thus, the conventional, word-to-word statistical dictionary is included.

If we permit the reordering of the target phrases, a hidden phrase level alignment variable, $\tilde{\mathbf{a}}$, is introduced. In this case, we assume that the target phrase in position *k* is produced only by the source phrase in position $\tilde{a}_k$.

$$\Pr(\tilde{e}_1^K \,|\, \tilde{f}_1^K) = \sum_{\tilde{\mathbf{a}}} \prod_{k=1}^{K} p(\tilde{a}_k \,|\, \tilde{a}_1^{k-1})\, p(\tilde{e}_k \,|\, \tilde{f}_{\tilde{a}_k}) \tag{5}$$

For the distortion model, we assume a first-order alignment that depends only on the distance of the two phases [7]:

4 Monotone Algorithm

In this section, we present a search algorithm for the monotone PB model (equation **4**) based on multi-stack decoding. This algorithm is similar to the one in [1] but takes advantage of the sequentiality of the model. We define a hypothesis search as the triple (mk, e_1^{lk}, g), where *mk* is the length of the source prefix we are translating in the hypothesis (that is f_1^{mk}). The sequence of *lk* words e_1^{lk} is the target prefix that has been generated. And *g* is the score of the hypothesis.

Following the multi-stack decoding approach, hypotheses are stored in different priority queues according to their value of *mk*. This results in the following algorithm:

```
Create priority queues from $Q_0$ to $Q_{|\mathbf{f}|}$
Initialize $Q_0$ with the empty hypothesis ($mk$=0, $lk$=0, $g$=1)
Repeat $max\_expan$ times or until no more hypotheses to extend
  For each queue from $Q_0$ to $Q_{|\mathbf{f}|-1}$:
    Pop the hypothesis with the highest score; $h=(mk, e_1^{lk}, g)$
    For $|\tilde{f}|$=1 to $|\mathbf{f}|-mk$
      $\tilde{f} = f_{mk+1}^{mk+|\tilde{f}|}$
      For each $\tilde{e}$ with $p(\tilde{f}\,|\,\tilde{e})$>0
        Push in $Q_{mk+|\tilde{f}|}$ the hypothesis:
          $(mk+|\tilde{f}|,\ e_1^{lk}\tilde{e},\ g\cdot\Pr(\tilde{e}\,|\,e_1^{lk})\cdot p(\tilde{f}\,|\,\tilde{e}))$
The hypothesis of $Q_{|\mathbf{f}|}$ with the highest score is the output
```

Fig. 1. Multi-stack decoding monotone algorithm for PB model.

The hypothesis extension consists of: selecting a phrase $\tilde{f}$ from the source sentence, starting with the last word translated and with any length; selectng $\tilde{e}$, a possible translation of $\tilde{f}$; and appending it to the target prefix.

The parameter *max_expan* limits the number of iterations of the algorithm. A typical value used is 10; results do not improve with greater values. The introduction of this parameter permits a very fast search. We can translate several hundred words in a second.

To improve the translation speed, pruning criteria is introduced to erase hypotheses with a low probability of becoming the best output. [1] proposes using a threshold by stack, which is calculated from the best final hypothesis up to that point. In our implementation, we use a very simple solution. If a hypothesis has a score that is higher than the best output, the hypothesis is discarded. The same pruning criterion is used in the two algorithms described below.

5 Source Word Reordering Algorithm

The monotone algorithm does not permit the reordering of the target phrases. In this section, we present a search algorithm for the nonmonotone model (equation **5**). This algorithm is similar to the one described in [1].

We define a hypothesis search as the triple (w, e_1^{lk} , g), where $w \subset \{1..|\mathbf{f}|\}$ is the coverage set that defines which positions of source words have been translated; e_1^{lk} is the target prefix that has been generated; and g is the score of the hypothesis. A similar definition of a hypothesis search can be found in [1][12][13].

For a better comparison of hypotheses, [1] proposes storing each hypothesis in different priority queues according to their value of w. The number of possible queues can be very high ($2^{|\mathbf{f}|}$); thus, the queues are created on demand. Here is the algorithm:

```
Initialize $Q_\varnothing$ with the null hypothesis ($w=\varnothing$, $lk=0$, $g=1$)
Repeat $max\_expan$ times or until no more hypotheses to expand
  For each queue in order of index cardinality
    Pop the highest scored hypothesis; $h=(w, e_1^{lk}, g)$
    For $|\tilde{f}|=1$ to $|\mathbf{f}|$
      Let $j_{init}= \min_{n=1..|\mathbf{f}|} (n \notin w)$
        For $j=j_{init}$ to $j_{init}+win\_search-1$ with $w \cap \{j,..,j+|\tilde{f}|-1\}=\varnothing$
          $\tilde{f} = f_j^{j+|\tilde{f}|-1}$
          For each $\tilde{e}$ with $p(\tilde{f}\,|\,\tilde{e})>0$
            Push in $Q_{w \cup \{j,\dots,j+|\tilde{f}|-1\}}$ the hypothesis:
              $(w \cup \{j,\dots,j+|\tilde{f}|-1\},\ e_1^{lk}\,\tilde{e}\ ,\ g\cdot\Pr(\tilde{e}\,|\,e_1^{lk})\cdot p(\tilde{f}\,|\,\tilde{e}))$
The hypothesis of $Q_{\{1,..,|\mathbf{f}|\}}$ with the highest score is the out-
put
```

Fig. 2. Multi-stack decoding source word reordering algorithm for PB model.

The principal difference of this algorithm with the monotone algorithm is the hypothesis extension. Now, the source phrase $\tilde{f}$ can be selected starting at any position *j*, with the restriction of the words of $\tilde{f}$ which have not been translated in the hypothesis to be extended ($w \cap \{j,..,j+|\tilde{f}|-1\}=\emptyset$). This algorithm uses an additional restriction proposed in [1]. The initial source position *j* is selected only through the first *win_search* positions, beginning at the first nontranslated word.

This algorithm is much slower than the monotone algorithm. First, it introduces an additional order of magnitude selecting the position of the input phrase. Second, it extends a hypothesis from each stack created. Afterwards, there are $|\mathbf{f}|$ stacks and there can be now reach up to $2^{|\mathbf{f}|}$.

6 Target Word Reordering Algorithm

In order to improve the search time, we propose an algorithm that has a structure that is similar to the monotone algorithm. We take the words of **f** from left to right, and we introduce a possible word reordering at the output prefix.

Similar to the monotone algorithm, we define a hypothesis search as the triple (*mk*, e_1^{lk}, *g*). This hypothesis indicates that the source prefix f_1^{mk} has been translated by the target prefix e_1^{lk} with a score of *g*. Different to the previous case, we can introduce the special token <nul> at the output prefix. The meaning of this token is that in a future expansion, the token <nul> must be replaced by a sequence of words. See Figure 3 for an example. In principle, a hypothesis could include an arbitrary number of <nul> tokens; however, in our implementation, we allow only one. Therefore, we can distinguish between two classes of hypotheses. A hypothesis is closed if it does not contain the token <nul>, and it is open if it contains this token. In the latter case, if <nul> token is at position *i*, we can represent the target prefix as e_1^{i-1} <nul> e_{i+1}^{lk}.

(0, *“”*, *1*) → (1, *“el”*, g_1) → (2, *“el <nul> de configuración”*, g_2) → (3, *“el programa de configuración”*, g_3)

Fig. 3. Example of a sequence of hypotheses that permits translating the Spanish sentence ‘*el programa de configuración*’ from the English sentence ‘*the configuration program*’.

The hypothesis extension begins selecting $\tilde{f}$, starting at the position *mk*+1 of **f**. Then, $\tilde{e}$ is selected as a possible translation of $\tilde{f}$. If the hypothesis to be extended is closed, two new hypotheses are created; one is created by putting $\tilde{e}$ at the right of the target prefix, and the other is created by putting <nul> $\tilde{e}$ at the right of the target prefix. If the hypothesis is open, four new hypotheses are created; one that closes the hypothesis, replacing the token <nul> by $\tilde{e}$, and three that keep the new hypotheses open, putting $\tilde{e}$ at the left or right of <nul>, or putting $\tilde{e}$ at right of the target prefix.

This algorithm uses a different approach to calculate the distortion probability. A similar approach for the previous algorithm is not possible. When we open a hypothesis, we do not know the final position that a phrase will take. Thus, we have a different parameter distortion for each type of extension. If the hypothesis is closed, we use de probabilities p_m to keep it closed and 1-p_m to open it. If the hypothesis is open, we have four different extension types with the probabilities p_c, p_i, p_d and 1-p_c-p_i-p_d.

Another problem we need to solve is the evaluation of the language model in the partial hypotheses. In the two other algorithms, the target prefix is created from left to right, so we can easily use an n-gram model. In this algorithm, if we have an open hypothesis, we cannot calculate the language model contribution of the right part of the prefix after the <nul> token, since we do not know which words will be replaced by <nul>. To solve this problem, we compute an estimation of the language model contribution. It consists of assigning the probability of its unigram to the word at the right of <nul>. We assign the probability of the bigram to the next word, etc. When a hypothesis is closed, this estimation is replaced by the true language model contribution.

```
Create priority queues from $Q_0$ to $Q_{|\mathbf{f}|}$
Initialize $Q_0$ with the null hypothesis ($mk$=0, $lk$=0, $g$=1)
Repeat $max\_expan$ times or until no more hypotheses to extend
  For each queue from $Q_0$ to $Q_{|\mathbf{f}|-1}$
    Pop the hypothesis with the highest score; $h=(mk,\ e_1^{lk},\ g)$
    For $|\tilde{f}|=1$ to $|\mathbf{f}|-mk$
      $\tilde{f} = f_{mk+1}^{mk+|\tilde{f}|}$
      For each $\tilde{e}$ with $p(\tilde{f}\,|\,\tilde{e})>0$
        Push in $Q_{mk+|\tilde{f}|}$ the hypotheses:
        if $h$ is closed:
          $(mk+|\tilde{f}|,\ e_1^{lk}\ \tilde{e},\ g\cdot\Pr(\tilde{e}\,|\,e_1^{lk})\cdot p(\tilde{f}\,|\,\tilde{e})\cdot p_m)$
          $(mk+|\tilde{f}|,\ e_1^{lk}\ \text{<nul>}\ \tilde{e},\ g\cdot\Pr(\tilde{e}\,|\,\text{<nul>})\cdot p(\tilde{f}\,|\,\tilde{e})(1-p_m))$
        if $h$ is open:
          $(mk+|\tilde{f}|,e_1^{lk}\ \tilde{e},\ g\cdot\Pr(\tilde{e}\,|\,\text{<nul>}\ e_{i+1}^{lk})\cdot p(\tilde{f}\,|\,\tilde{e})(1-p_c-p_i-p_d))$
          $(mk+|\tilde{f}|,e_1^{i-1}\ \tilde{e}\ e_{i+1}^{lk},\ g\cdot\Pr(\tilde{e}\ e_{i+1}^{lk}\,|\,e_1^{i-1})\cdot p(\tilde{f}\,|\,\tilde{e})/\Pr(e_{i+1}^{lk}\,|\,\text{<nul>})\cdot p_c)$
          $(mk+|\tilde{f}|,e_1^{i-1}\ \tilde{e}\ \text{<nul>}\ e_{i+1}^{lk},\ g\cdot\Pr(\tilde{e}\,|\,e_1^{i-1})\cdot p(\tilde{f}\,|\,\tilde{e})\cdot p_i)$
          $(mk+|\tilde{f}|,e_1^{i-1}\ \text{<nul>}\ \tilde{e}\ e_{i+1}^{lk},g\cdot\Pr(\tilde{e}\ e_{i+1}^{lk}\,|\,\text{<nul>})\cdot p(\tilde{f}\,|\,\tilde{e})/\Pr(e_{i+1}^{lk}\,|\,\text{<nul>})\cdot p_d)$
Closed hypothesis of $Q_{|\mathbf{f}|}$ with highest score is the output
```

Fig. 4. Multi-stack decoding target word reordering algorithm for the PB model.

7 Experimental Results

In order to compare the three decoding algorithms introduced above, several experiments were carried out. We use the Hansards task which consists of proceedings of the Canadian parliament. The training corpus consists of 50,000 sentences (2.1 million French and 1.9 million English words). The vocabulary size is 29,479 for French and 37,554 for English. With this corpus we train a phrase-based model [11] using a maximum length of phrase of 4 words. We obtain 916,414 parameters in this model. We used a test comprised of 240 bilingual sentences, uniformly distributed across the source lengths 5, 10, 15, …, 60. We use two criteria to evaluate the algorithms, the translation accuracy, measured in word error rate, (WER)[8][9] and the translation speed, measured in words per second.

Table 1 shows the effect *max_expand* parameter in the three algorithms. Monotone algorithm obtains a WER a little worst than nonmonotone algorithms. In other tasks, such as a reduced domain task or a task between Romanic languages, it obtains the same results as nonmonotone algorithms. With respect the search speed, the monotone algorithm is very fast. It obtains the best results with a value of *max_expand*=8. For this value, it can translate 37 words per second. Source word reordering (SWR) algorithm obtains somewhat better results than monotone algorithm. To obtain this improvement, it needs a high value of *max_expand* and it is very slow. We cannot use this algorithm in real time tasks. Target word reordering (TWR) algorithm obtains similar results to the SWR algorithm. However, it is faster. It obtain the best results with a value of *max_expant*=32. For this value, it can translate 4.2 words per second.

Table 1. Effect of *max_expand* parameter on WER and translation speed for the three decoders (*win_search*=6).

	max_expand	1	2	4	8	16	32	64
monotone	WER	74.49	74.54	74.46	74.46	74.40	74.39	74.40
	word/sec.	268	142	73.0	37.2	18.8	8.5	3.1
source word reordering	WER	74.60	74.45	74.41	74.35	74.38	74.13	-
	word/sec.	2.2	1.0	0.49	0.23	0.11	0.05	-
target word reordering	WER	74.49	74.54	74.39	74.40	74.31	74.12	74.14
	word/sec.	160	80.3	39.3	19.3	8.6	4.2	2.2

Table 2 shows the effect of sentence length on the three decoding algorithms. Usually, sort sentences are better translated than long sentences; however, the difference is small. Except for sentences with 5 words, the translation speed is independent of the sentence length in monotone and TWR algorithms. However, in SWR algorithm, the translation speed decreases with the sentence length in sentences with less than 25 words.

Table 2. Effect of sentence length on WER and translation speed for the three decoders (*max_expand*=20, *win_search*=6).

Sentence length		5	10	15	20	25	30	35	40	45	50	55	60
monotone	WER	63.0	69.5	79.9	75.2	79.8	76.2	71.9	74.9	72.0	69.9	74.9	75.8
	word/sec.	31.7	13.9	15.5	14.0	15.7	12.8	14.3	14.7	14.5	14.5	14.5	14.3
source word reordering	WER	63.0	73.1	78.7	72.8	79.1	75.0	71.7	74.8	71.2	69.3	73.9	74.5
	word/sec.	1.2	0.31	0.17	0.12	0.14	0.11	0.12	0.13	0.12	0.12	0.11	0.13
target word reordering	WER	62.0	68.5	79.0	74.0	79.2	75.0	71.9	74.6	71.0	69.3	73.6	74.9
	word/sec.	11.2	6.4	7.5	6.8	7.7	6.4	7.2	7.4	7.8	7.5	7.2	7.5

8 Conclusions

In this paper, we have described three search algorithms based on multi-stack decoding techniques. We have used them with the phrase-based model [10][13], and we think these algorithms can be easily adapted to other models like template-based [7] or IBM models [1][2].

In our experiments, monotone search obtained results in WER that were comparable to non monotone search. However, it is much faster. We obtain similar results in other tasks and in other models (like template-based) which are not reported in this paper [11]. For nonmonotone search, we have proposed a new approach, where the source sentence is translated from left to right and the word reordering is computed at the target prefix. We have compared it with the conventional approach. The experiments show that for similar search results the new approach is faster and requires less memory for stacks.

In the future, we are interested in applying the dynamic-programming technique to these algorithms, and comparing them with the stack-decoding technique. We are also interested in using these algorithms with other models, such as the IBM models.

Acknowledgements

This work has been partially supported by the Spanish project TIC2003-08681-C02-02 and the IST Program of the European Union under grant IST-2001-32091.

References

1. Berger, A.L., Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Gillett, J.R., Lafferty, J.D., Mercer, R.L., Printz, H., Ures, L.: Language Translation Apparatus and Method Using Context-Based Translation Models. United States Patent No. 5,510,981 (1996)
2. Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Mercer, R.L.: The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics, vol. 19 no. 2, (1993) 263–311
3. Garcia-Varea, I., Casacuberta, F.,: An Iterative, DP-based Search Algorithm for Statistical Machine Translation. Proceedings of the ICSLP-98, Sidney, Australia (1998)
4. Germann, U., Jahr, M., Knight, K., Marcu, D. Yamada, K: Fast Decoding and Optimal Decoding for Machine Translation. Proceedings of ACL 2001. Toulouse, France (2001)
5. Marcu, D., Wong W.: A Phrase-Based, Joint Probability Model for Statistical Machine Translation. Proc. of the Conference on Empirical Methods in Natural Language Processing, Philadelphia, PA, July (2002)
6. Niessen, S., Vogel, S., Ney, H., Tillmann, C.: A DP based Search Algorithm for Statistical Machine Translation. Proceedings of 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Montreal, Canada (1998)

7. Och, F.J., Ney, H.: Improved Statistical Alignment Models. Proc. of the 38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, China, October (2000)
8. Och, F.J., Ueffing, N., Ney, H.: An Efficient A* Search Algorithm for Statistical Machine Translation. Proc. of Data-Driven Machine Translation Workshop, Toulouse, France (2001)
9. Tillmann, C., Vogel, S., Ney, H.: A DP based Search Using Monotone Alignments in Statistical Translation. Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, Madrid, Spain (1997)
10. Tomás, J., Casacuberta, F.: Monotone Statistical Translation using Word Groups. Proceedings of the Machine Translation Summit VIII, Santiago de Compostela, Spain (2001)
11. Tomás, J., Casacuberta, F.: Combining Phrase-Based and Template-Based alignment models in Statistical Translation. Lecture Notes in Computer Science 2652, (2003)
12. Wang, Y.Y., Waibel, A.: Fast Decoding for Statistical Machine Translation. Proceedings of the International Conference on Speech Language Processing, (1998) 2775-2778
13. Zens, R., Och, F.J., Ney, H.: Phrase-Based Statistical Machine Translation. Proceedings of the Conference on Empirical Methods for Natural Language Processing (2002)

Classifier Design for Population and Sensor Drift

Keith Copsey and Andrew Webb

QinetiQ, St. Andrews Road, Malvern, WR14 3PS
{k.copsey,a.webb}@signal.qinetiq.com

Abstract. The basic assumption in classifier design is that the distribution from which the design sample is selected is the same as the distribution from which future objects will arise: i.e., that the training set is representative of the operating conditions. In many applications, this assumption is not valid. In this paper, we discuss sources of variation and possible approaches to handling it. We then focus on a problem in radar target recognition in which the operating sensor differs from the sensor used to gather the training data. For situations where the physical and processing models for the sensors are known, a solution based on Bayesian image restoration is proposed.

Keywords: classification; generalisation; sensor drift; population drift; Bayesian inference; target recognition.

1 Introduction

In classifier design, we often have a design or training set that is used to train a classifier; a validation set (used as part of the training process) for model selection or termination of an iterative learning rule; and an independent test set, which is used to measure the *generalisation performance* of the classifier: the ability of the classifier to generalise to future objects. These datasets are often gathered as part of the same trial and it is common that they are, in fact, different partitions of the same dataset.

In many practical situations, the operating conditions may differ from those prevailing at the time of the test data collection, particularly in a sensor data analysis problem. For example, sensor characteristics may drift with time or environmental conditions may change. These effects result in changes to the distributions from which patterns are drawn. This is referred to as *population drift* [4].

The circumstances and degree of population drift vary from problem to problem. This presents a difficulty for classifier performance evaluation since the test set may not be representative of the operating conditions and thus the generalisation performance quoted on the test set may be overly optimistic.

Designing classifiers to accommodate population drift is problem specific. In Section 2 we review some of the causes of population drift and the approaches that can be taken to mitigate against distributional changes. In section 3, a generic Bayesian approach is described and in section 4, this is applied in a target recognition example in which the probability densities of interest can be calculated using knowledge of the properties of the imaging radars.

2 Population Drift

2.1 Sensor Drift

Pattern recognition techniques need to be developed for drifting sensors. An example is an electronic nose, a device that contains a number of different individual sensors whose response characteristics depend on the chemical odour present. These have applications in quality control, bioprocess monitoring and defence. All chemical sensors are affected by drift, stability problems and memory effects. Data processing techniques are required to handle these effects autonomously. This may be simple preprocessing to remove shifts in zero points of responses, and changes in sensitivity handled by a gain control, but there may be more complex effects.

2.2 Changes in Object Characteristics

In medical applications there may be drift in the patient population (changes in patient characteristics) over time. Population drift also occurs in speech recognition when a new speaker is presented. There are various approaches including analysing a standard input from a new speaker and using this to modify stored prototypes. In credit scoring, the behaviour of borrowers is influenced by short-term pressures (for example, Budget announcements by the Chancellor of the Exchequer) and classification rules will need to be changed quite frequently [5].

In radar target recognition, classifiers need to be robust to changes in vehicle equipment fit which can give rise to large changes in the radar reflectivity [1]. Within a Bayesian density estimation framework for classification, one approach is to introduce hyperpriors to model target variability. In condition monitoring, the healthy state of an engine will change with time. In object recognition in images, it is important that the classifier has some invariance to object pose (translational/rotational invariance).

In each of the examples above, it is an advantage if the classification method can be dynamically updated and does not need to be re-computed from scratch (using new sets of training data) as the conditions change.

2.3 Environmental Changes

The training conditions may only approximate the expected operating conditions and a trained classifier will need some modification [6]. The signal-to-noise ratio of the operating conditions may be different from the (controlled) training conditions and may possibly be unknown. In order to derive a classifier for noisier operating conditions, several approaches may be adopted including noise injection in the training set and modifying the training procedure to minimise a cost function appropriate for the expected operating conditions [8]. Errors-in-variables models are relevant here.

In target recognition, the ambient light conditions may change. The environmental conditions, for example the clutter in which the target is embedded, will differ from the training conditions. Sea clutter is time-varying.

2.4 Sensor Change

For various reasons, it may not be possible to gather sufficient information with the operating sensor to train a classifier: it might be too expensive or too dangerous. However, measurements can be made in more controlled conditions using a different sensor and a classifier can be designed using this set of measurements. In this type of problem, a classifier needs to be designed using sensor-independent features or a means of translating operating sensor data to the training domain must be developed. This is discussed further in section 3.

2.5 Variable Priors

Prior probabilities of class membership are likely to change with time. Thus, although class conditional densities do not change, decision boundaries are altered due to varying priors. This requires high-level modelling, but often there is little data available to model the dependencies and Bayesian networks are constructed using expert opinion [2].

Costs of misclassification are also variable and unknown. The consequence is that the optimisation criterion used in training (for example, minimum cost) may be inappropriate for the operating conditions.

2.6 Conclusions

Uncertainties between training and operating conditions mean that there is a limit beyond which it is not worth pushing the development of a classification rule [4]. In some cases, population drift is amenable to treatment, but this is problem dependent.

3 Joint Density Modelling

3.1 Introduction

Let us denote measurements in the training conditions by the variable $\boldsymbol{x}$ and measurements in the operating conditions by the variable $\boldsymbol{z}$. Assuming a probabilistic relationship between the measurements of an object in the operating conditions and the measurements of the (same) object that would have been obtained in the training conditions (denoted by $p(\boldsymbol{x}|\boldsymbol{z})$), inference proceeds by considering expectations of functions $g(\boldsymbol{x})$ of $\boldsymbol{x}$, conditional on the measurements $\boldsymbol{z}$:

$$E[g(\boldsymbol{x})|\boldsymbol{z}] = \int_{\boldsymbol{x}} g(\boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{x} \qquad (1)$$

There are a number of special cases.

Classification. Suppose that we have designed a classifier using a design set $\mathcal{D} = \{\boldsymbol{x}_i, i = 1. \dots, N\}$ of samples gathered under the training conditions. Denote the class posterior probabilities estimated by the classifier for a measurement $\boldsymbol{x}$ by $p(C = j|\boldsymbol{x}, \mathcal{D})$, $j = 1, \dots, J$. Setting $g_j(\boldsymbol{x}) = p(C = j|\boldsymbol{x}, \mathcal{D})$, for $j = 1, \dots, J$ and substituting into (1) gives

$$E[p(C = j|\boldsymbol{x}, \mathcal{D})|\boldsymbol{z}] = \int_{\boldsymbol{x}} p(C = j|\boldsymbol{x}, \mathcal{D}) p(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{x} \tag{2}$$

These expectations provide an estimate of the posterior class probabilities for the $\boldsymbol{z}$ data, based on a classifier trained using $\boldsymbol{x}$ data.

Regression. Another special case is if $g(\boldsymbol{x})$ is a regression function. The training set consists of $\{(\boldsymbol{x}_i, \theta_i), i = 1, \dots, N\}$ and a regression function is constructed to provide an estimate of θ given $\boldsymbol{x}$. Conditioned on a value $\boldsymbol{z}$, we have (with $g(\boldsymbol{x}) = E[\theta|\boldsymbol{x}]$)

$$E\left[E[\theta|\boldsymbol{x}]|\boldsymbol{z}\right] = \int E[\boldsymbol{\theta}|\boldsymbol{x}] p(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{x} \tag{3}$$

The expectations in (3) provide an estimate of θ for the $\boldsymbol{z}$ data, based upon a regression function designed using $\boldsymbol{x}$ data.

3.2 The Conditional Density, $p(x|z)$

Specification of the conditional density, $p(\boldsymbol{x}|\boldsymbol{z})$ may be done in a number of ways. There may be prior knowledge of the form of the discrepancy; we have a physical model that characterises the difference between training and operating conditions. An example is a point scatterer model of a target in a radar target classification problem or a facet model of an object which we can use to simulate the image of the object under different illumination. The key to dealing with both of these examples is to first invert the operational data $\boldsymbol{z}$ to an underlying representation $\boldsymbol{\sigma}$ (e.g. the point scatterer model), and to then convert the underlying representation to measurements consistent with the training conditions (e.g. the different illumination).

4 Radar Target Classification Example

4.1 Introduction

The remainder of this paper concentrates on the development of procedures that enable a classifier designed using data gathered from one sensor to be applied to data gathered from a different sensor (provided that appropriate sensor measurement models are available). A motivating application is to use automatic target recognition (ATR) systems trained on readily available ground-based Inverse Synthetic Aperture Radar (ISAR) data to classify objects imaged by an airborne Doppler Beam Sharpened (DBS) radar seeker. It is intended that this will be done by inverting each operational DBS image to an underlying radar cross section (RCS). A Bayesian image restoration (inverse model based) approach, which estimates the distribution of the underlying RCS, is proposed for this task. Given the distribution of the RCS we can then use the ISAR sensor measurement model to obtain the distribution of the ISAR image given the DBS image. ISAR images sampled from this model generated distribution can then be classified by an ATR system trained using ISAR data, thus providing the classification for the operational DBS image. The advantage to such a procedure is that it is easier to collect ISAR training data (by imaging targets on turntables) than it is to collect DBS data. Throughout the paper the DBS data is referred to as the $\boldsymbol{z}$ sensor data, and the ISAR data as the $\boldsymbol{x}$ sensor data.

4.2 Inverting the Data from the Operational Sensor

The Bayesian framework used to model the operational ($\boldsymbol{z}$) sensor data measurement process is illustrated by the model in Fig. 1.

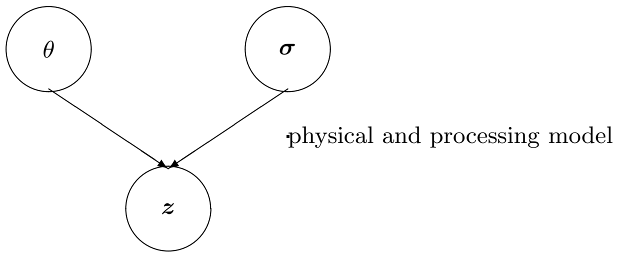


Fig. 1. Sensor model for $\boldsymbol{z}$ sensor data.

We assume that a physical and processing model is available that transforms the underlying RCS $\boldsymbol{\sigma}$ to a set of sensor measurements $\boldsymbol{z}$. The model depends on parameters θ, which may be unknown. The parameters θ will contain information on the sensor characteristics (beam shape, pulse width etc) and sensor platform dynamics (e.g. for an airborne sensor the speed, acceleration, roll).

Using Bayes' theorem, along with Fig. 1, the posterior distribution is:

$$p(\boldsymbol{\sigma}, \theta | \boldsymbol{z}) \propto p(\boldsymbol{z} | \boldsymbol{\sigma}, \theta) p(\boldsymbol{\sigma}) p(\theta). \tag{4}$$

Integrating over the parameters of the sensor measurement model produces the marginal posterior distribution $p(\boldsymbol{\sigma}|\boldsymbol{z})$. The components of the posterior distribution are the prior distribution for the RCS, $p(\boldsymbol{\sigma})$, the prior distribution for θ, $p(\theta)$, and the conditional distribution for the sensor data given the measurement model parameters and the RCS, $p(\boldsymbol{z}|\boldsymbol{\sigma}, \theta)$. These distributions are examined in more detail below.

The form of the prior distribution for $\boldsymbol{\sigma}$ will depend on the representation of $\boldsymbol{\sigma}$. For our example, $\boldsymbol{\sigma}$ is represented as a two-dimensional grid of values, $\boldsymbol{\sigma} = \{\sigma_{i,j}, 1 \leq i, j \leq d\}$, where $d \times d$ is the dimensionality of the grid. The prior distribution for $\boldsymbol{\sigma}$ is taken to be multivariate Gaussian (having concatenated the rows of the grid into a single vector), although this would not be appropriate in many situations. Ideally, the prior distribution should reflect the RCS grids that would be expected for the range of targets to be identified (i.e. determined using expert knowledge).

A prior distribution $p(\theta)$ is required for the imaging model parameters θ. This will depend on the exact form of θ, which is determined by the physical model transforming the RCS to the sensor measurements. In many cases each of the variables that comprise θ (e.g. pulse width) will be known to within a tolerance, so independent Gaussian distributions would be appropriate.

The form of the conditional distribution $p(\boldsymbol{z}|\boldsymbol{\sigma},\theta)$ depends on the physical and processing model for the sensor data generation and the noise in that physical model. Typically, the model will be assumed to consist of a (known) deterministic function of $\boldsymbol{\sigma}$ and θ, together with additive noise. For our example, the operational sensor measurement process is taken to consist of the application of a 3×3 point spread function (PSF) to the underlying RCS grid, followed by the addition of independent Gaussian noise to each pixel (although the presented Bayesian algorithm would be unchanged by the addition of multivariate Gaussian noise on the whole image). Edge effects from the application of the PSF are dealt with by adding a temporary boundary layer of zeros to the RCS grid. The additive noise for each pixel is drawn independently from a zero mean Gaussian distribution, $N(0,\psi_z^2)$. The variable θ is therefore given by the noise variance ψ_z^2 and the matrix psf_z defining the PSF. The documented example assumes that the elements of θ are known (and correct), corresponding to a point mass prior distribution for $p(\theta)$. The resulting sensor measurement distribution is:

$$p(z|\boldsymbol{\sigma},\theta) = \prod_{i=1}^{d}\prod_{j=1}^{d} N(z_{i,j};\mathrm{psf}(\boldsymbol{\sigma},psf_z)_{i,j},\psi_z^2) \tag{5}$$

where $z = \{z_{i,j}, 1 \leq i,j,\leq d\}$ are the pixels of the measurement z, and the (i,j)-th element of the image created by applying the point spread matrix psf to the RCS grid $\boldsymbol{\sigma}$ is represented by $\mathrm{psf}(\boldsymbol{\sigma},psf)_{i,j}$.

It turns out that for the example in this paper, the distribution $p(\sigma|\boldsymbol{z})$ can be expressed analytically as a multivariate Gaussian distribution. This would not be the case for the majority of sensor measurement models and prior distributions. Indeed, calculation of the normalisation constant for the posterior distribution is usually not tractable, and for most physical and processing models, statistics of interest (such as the mean and covariance) will not be available analytically. In such cases, rather than making simplifications to allow analytic inference on the posterior distribution, a full Bayesian approach to the problem is maintained by drawing samples from the posterior. All inferences can then be made through consideration of these samples. In most circumstances it will not be possible to sample directly from the posterior distribution, in which case a Markov chain Monte Carlo (MCMC) algorithm [3, 7] is used.

4.3 The Conditional Density $p(\boldsymbol{x}|\boldsymbol{z})$

The density $p(\boldsymbol{x}|\boldsymbol{z})$ used in the equations of Section 3 can be expressed as:

$$p(\boldsymbol{x}|\boldsymbol{z}) = \int_{\boldsymbol{\sigma}} p(\boldsymbol{x},\boldsymbol{\sigma}|\boldsymbol{z})\, d\boldsymbol{\sigma} = \int_{\boldsymbol{\sigma}} p(\boldsymbol{x}|\boldsymbol{\sigma})p(\boldsymbol{\sigma}|\boldsymbol{z})\, d\boldsymbol{\sigma} \tag{6}$$

The density $p(\boldsymbol{x}|\boldsymbol{\sigma})$ represents the physical and processing model for the training sensor measurements, under the assumption that the imaging model parameters are known. For our example, the same measurement process as for the operational sensor is used (see (5)), but with a different matrix psf_x defining the PSF, and a potentially different additive noise variance ψ_x^2.

For our documented example, the density $p(\boldsymbol{x}|\boldsymbol{z})$ in (6) turns out to be multivariate Gaussian. Thus, samples can be drawn easily from this distribution and used to approximate the expectations of Section 3. Given samples $\{\boldsymbol{x}^{(s)}, s = 1,\ldots,N\}$ from $p(\boldsymbol{x}|\boldsymbol{z})$, the posterior classification probabilities defined in (2) become:

$$E[p(C=j|\boldsymbol{x},D)|\boldsymbol{z}] \approx \frac{1}{N}\sum_{s=1}^{N} p(C=j|\boldsymbol{x}^{(s)},D) \tag{7}$$

More complicated operational and training sensor measurement models and prior distributions require the use of MCMC samples from $p(\boldsymbol{\sigma},\theta|\boldsymbol{z})$. In particular, we can sample from $p(\boldsymbol{x}|\boldsymbol{z})$ by passing the RCS samples through the physical and processing model for sensor data $\boldsymbol{x}$ (i.e. by sampling from $p(\boldsymbol{x}|\boldsymbol{\sigma})$ for each MCMC sample for $\boldsymbol{\sigma}$).

4.4 Description of Experiment

A two class problem has been used to illustrate the approach, with the targets defined on underlying $d \times d$ RCS grids, where $d = 5$. The PSF matrices for the sensors were taken to be:

$$psf_z = \begin{pmatrix} 0.3 & 0.3 & 0.3 \\ 0.3 & 1.0 & 0.3 \\ 0.3 & 0.3 & 0.3 \end{pmatrix} \qquad psf_x = \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 1.0 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} \tag{8}$$

The different PSF widths mimic the effects of different sensor resolutions, with the data from the operational sensor being more distorted than that from the training sensor. The standard deviations of the additive noise applied to the sensor data were $\psi_x = \psi_z = 0.1$.

The sensor data were created by generating underlying RCS grids and then simulating the sensor measurement processes. The RCS grids were generated by sampling from multivariate Gaussian distributions (after concatenation of the rows of the grid). For both classes, the Gaussian covariance matrix was diagonal, with the same standard deviation $\psi_\sigma = 0.5$ across all components. The means were dependent on the class and are displayed graphically in Fig. 2.

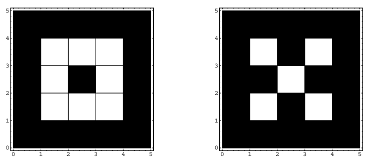


Fig. 2. Mean values for the target RCS grids (class 0 to the left, class 1 to the right). Dark pixels correspond to value 0, light pixels to value 1.

A multivariate Gaussian classifier (outputting posterior class probabilities) was selected for the $\boldsymbol{x}$ sensor data classifier. The classifier was trained by generating $n_{tr} = 100$ $\boldsymbol{x}$ sensor data measurements from each class. The test/operational data were obtained by generating $n_{te} = 1000$ $\boldsymbol{z}$ sensor data measurements from each class. Thus, the test data came from a different sensor to that used in the design phase of the classifier.

Within the Bayesian algorithm, the mean of the Gaussian prior distribution for $\boldsymbol{\sigma}$ was set to be zero around the outer layer, and 0.5 within the central 3×3 grid. The covariance matrix was set to be diagonal with the standard deviations for each grid location set to $\varsigma_{\sigma} = 0.5$. We note that the prior distribution incorporates expert knowledge that the values in the outer layer of the RCS grid (for both targets) are likely to be smaller than the inner values.

For each test data $\boldsymbol{z}$ measurement, 500 samples were drawn from the distribution $p(\boldsymbol{x}|\boldsymbol{z})$, and used in (7). Class decisions were made by selecting the class with the maximum expected posterior class probability.

4.5 Experimental Results

To assess the performance of the Bayesian approach, three additional sets of classification results have been obtained (all based on Gaussian classifiers).

C1) A classifier applied directly to the $\boldsymbol{z}$ sensor data. $n_{tr} = 100$ $\boldsymbol{z}$ sensor data measurements from each class were made available for training.
C2) A classifier applied directly to the $\boldsymbol{x}$ sensor data. Test data for the $\boldsymbol{x}$ sensor measurements were made available.
C3) The classifier for the $\boldsymbol{x}$ sensor data applied directly to the $\boldsymbol{z}$ sensor data.

Classifiers C1 and C2 rely on data that is not available under the proposed scenario, so provide an indication of the the performance that could be expected in idealised situations, rather than a baseline performance.

Figure 3 displays the classification rates obtained for the Bayesian approach and the three additional classifiers. The poorer performance of classifier C3 relative to the other classifiers indicates that, if ignored, the change in sensor between operational and training conditions does (as would be expected) reduce the classifier performance. There is little to choose between classifiers C1, C2 and the Bayesian approach, indicating that (given appropriate sensor measurement models) we have provided a mechanism for dealing with situations where the operating sensor differs from the sensor used to gather the training data.

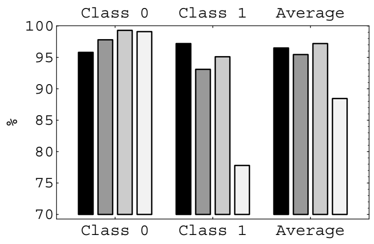


Fig. 3. Classification rates. From left to right, within each set of bars, the results are for the Bayesian image restoration based approach, classifiers C1, C2 and C3 respectively.

5 Conclusions

The basic assumption in classifier design is that the distribution from which the design sample is selected is the same as the distribution from which future objects will arise. This paper examines the validity of that assumption and considers a problem in radar target recognition in which the operating sensor differs from the sensor used to gather the training data. A Bayesian image restoration based solution is proposed for situations where the physical and processing models for the sensors are known. The approach is illustrated on a simplified problem.

Acknowledgements

This research was sponsored by the UK MOD Corporate Research Programme.

References

1. K.D. Copsey and A.R. Webb. Bayesian approach to mixture models for discrimination. *Advances in Pattern Recognition, Springer Lecture Notes in Computer Science*, 1876:491–500, August 2000.
2. K.D. Copsey and A.R. Webb. Bayesian networks for incorporation of contextual information in target recognition systems. *Advances in Pattern Recognition, Springer Lecture Notes in Computer Science*, 2396:709–717, August 2002.
3. W.R. Gilks, S. Richardson, and D.J. Spiegelhalter. *Markov chain Monte Carlo in Practice.* Chapman and Hall, 1996.
4. D.J. Hand. *Construction and Assessment of Classification Rules.* John Wiley, Chichester, 1997.
5. M.G. Kelly, D.J. Hand, and N.M. Adams. The impact of changing populations on classifier performance. *Proc. of the 5th ACM SIGKDD Conf, San Diego*, pages 367–371, 1999.
6. M.A. Kraaijveld. A Parzen classifier with an improved robustness against deviations between training and test data. *Pattern Recognition Letters*, 17:679–689, 1996.
7. A.R. Webb. *Statistical Pattern Recognition.* John Wiley & Sons, Chichester, 2nd edition, August 2002.
8. A.R. Webb and P.N. Garner. A basis function approach to position estimation using microwave arrays. *Applied Statistics*, 48(2):197–209, 1999.

A Cost-Sensitive Paradigm for Multiclass to Binary Decomposition Schemes

Claudio Marrocco and Francesco Tortorella

Dipartimento di Automazione, Elettromagnetismo
Ingegneria dell'Informazione e Matematica Industriale
Università degli Studi di Cassino
03043 Cassino (FR), Italy
{marrocco,tortorella}@unicas.it

Abstract. An established technique to face a multiclass categorization problem is to reduce it into a set of two-class problems. To this aim, the main decomposition schemes employed are *one vs. one*, *one vs. all* and *Error Correcting Output Coding*. A point not yet considered in the research is how to apply these methods to a cost-sensitive classification that represents a significant aspect in many real problems. In this paper we propose a novel method which, starting from the cost matrix for the multi-class problem and from the code matrix employed, extracts a cost matrix for each of the binary subproblems induced by the coding matrix. In this way, it is possible to tune the single two-class classifier according to the cost matrix obtained and achieve an output from all the dichotomizers which takes into account the requirements of the original multi-class cost matrix. To evaluate the effectiveness of the method, a large number of tests has been performed on real data sets. The experiments results have shown a significant improvement in terms of classification cost, specially when using the ECOC scheme.

1 Introduction

A diffused technique to face a classification problem with many possible classes is to decompose the original problem into a set of two-class problems. The rationale of this approach rely on the stronger theoretical roots and better comprehension characterizing two class classifiers (*dichotomizers*) such as Perceptrons or Support Vector Machines. Moreover, with this method it becomes possible to employ in multi class problems some dichotomizers which are very effective in two-class problems but are not capable to directly perform multi-class classification. To this aim, the main decomposition schemes employed are *one vs. one*, *one vs. all* and *Error Correcting Output Coding*. However, a point not yet considered is how to devise a decomposition scheme for cost-sensitive classification. This is a significant point in many real problems such as automated disease diagnosis, currency recognition, speaker identification, and fraud detection in which different classification errors frequently have consequences of very different gravity. For this reason, the classification systems used in such situations must take into account the different costs and benefits (collected in a cost matrix) which the different decisions can provide and thus should be tuned accordingly. In mult-class classifiers, such task is usually accomplished by modifying the learning algorithm used during the training phase of the classifier or by tuning the classifier after the learning phase.

In this paper we propose a cost-sensitive paradigm for multiclass to binary decomposition schemes. Starting from the cost matrix for the multi-class problem and from the code matrix employed, a cost matrix is derived for each of the binary problems induced by the columns of the code matrix. In this way it is possible to tune the single dichotomizer according to the cost matrix obtained and achieve an output from the dichotomizers which takes into account the requirements of the original multi-class cost matrix.

In the rest of the paper we describe, after a short description of the main decomposition schemes, how to decompose the original multi-class cost matrix in more two-class cost matrices. A conclusive section describes the results obtained from experiments performed on real data set.

2 Output Coding for Multi-class Problems

In order to introduce the main decomposition methods, let us consider a problem with n classes to be reduced to a set of L binary problems. For each problem, we employ a dichotomizer, i.e. a classifier able to discriminate between two mutually exclusive classes that can be generically called *Positive* (P) class (labelled by +1) and *Negative* (N) class (labelled by -1).

The most immediate approach is to create one binary problem for each class i, in which all samples labelled i are considered positive samples while all other samples are considered negative. Such method is frequently defined *one vs. all* (OVA) and involves the definition of $L = n$ binary problems. Another approach, suggested by Hastie and Tibshirani [1], is to define as many binary problems as the possible pairs of different classes. For a given pair of distinct classes $\{i, j\}$, the corresponding binary problem considers positive the samples belonging to i and negative the samples belonging to j; all other samples are ignored. This is the *one vs. one* (OVO) approach; in this case, we have to define $L = n(n-1)/2$ binary problems.

A further technique that has emerged for its good generalization capabilities is the *Error Correcting Output Coding* (ECOC), introduced by Dietterich and Bakiri in [2], which is based on a $n \times L$ coding matrix $M = \{M_{hk}\}$, where $M_{hk} = \pm 1$. Each row of M corresponds to a bit string, called *codeword*, that represents a class label, while each column corresponds to a binary problem. Usually, it is chosen $L > n$, so as to make the Hamming distance between every pair of strings as large as possible. In this way, if d_m is the minimum Hamming distance between any pair of codewords, the code is able to correct at least $\lfloor (d_m - 1)/2 \rfloor$ single bit errors.

The coding matrix can be used to model also the OVA approach: this kind of decomposition, in fact, is represented by a square coding matrix M in which $M_{kk} = +1$ and $M_{hk} = -1\ \forall\ h \neq k$. Moreover, we can generalize the coding matrix as suggested in [3] and assume that M_{hk} can be also 0, thus indicating that the k-th binary problem does not consider samples of the class h. In this way, the OVO approach can be represented by a $n \times n(n-1)/2$ coding matrix. In such case, if we suppose that the k-th dichotomizer must decide between class i and class j, we will have $M_{ik} = +1$, $M_{jk} = -1$ (or vice versa) and $M_{hk} = 0$, $\forall\ h \neq i,j$.

It is worth noting that both OVA and OVO decomposition schemes do not provide the same robustness given by the ECOC to errors made by the dichotomizers. In fact, while the ECOC matrix is built so as to guarantee a large value of distance between different codewords, the OVA scheme entails a fixed distance of 2 between different codewords. The situation is even worse for the OVO scheme for which the distance between distinct codewords is unitary because the 0 values in the matrix act as *don't care* and thus are not considered in the evaluation of the distance.

Once the coding matrix has been defined and the dichotomizers have been trained, to classify a new sample x, a vector of binary decisions is computed by applying each of the learned dichotomizers to x; to decode the resulting vector, i.e. to pass from the binary to the multi-class problem, the most common approach consists in evaluating the Hamming distances between the vector and the codewords of the matrix and choose for the nearest code word, i.e. for the minimum Hamming distance. Other decoding rules have been proposed which are based on a Least Squares approach [4] or on the loss function employed in the training algorithm of the dichotomizer [3], but we will not consider them in this paper.

3 Evaluating the Binary Costs from the Multi-class Costs

Before analyzing cost-sensitive classification in the multi-class case, it is convenient to focus preliminarily on the two-class problem. In this case, a sample can be assigned to one of two mutually exclusive classes that we have defined n the previous section as *Positive* class and *Negative* class. The set of samples classified as "positive" by the dichotomizer will contain some actually-positive samples correctly classified and some actually-negative samples incorrectly classified. Hence, two appropriate performance figures are given by the *True Positive Rate* (*TPR*), i.e. the fraction of actually-positive cases correctly classified, and by the *False Positive Rate* (*FPR*), given by the fraction of actually-negative cases incorrectly classified as "positive". In a similar way, it is possible to evaluate the *True Negative Rate* (*TNR*) and the *False Negative Rate* (*FNR*). It is worth noting that only two indices are actually necessary because the following relations hold:

$$FNR=1-TPR \quad TNR=1-FPR\ . \tag{1}$$

In cost sensitive applications, every decision taken by the classifier involves a cost which estimates the penalty (benefit) produced by an error (by a correct decision). In many applications the two kinds of error (false positive and false negative) are not equally costly as well as the value of the benefit obtained can depend on the class of the sample correctly identified. Hence, we have to consider a cost matrix similar to the one described in table 1. It is worth noting that, while *CFN* (Cost for a False Negative) and *CFP* (Cost for a False Positive) have positive values, *CTP* (Cost for a True Positive) and *CTN* (Cost for a True Negative) are negative costs since they actually represent a benefit.

Table 1. Cost matrix for a two class problem.

		True class	
		N	*P*
Predicted Class	*N*	*CTN*	*CFN*
	P	*CFP*	*CTP*

With such assumptions an estimate of the effectiveness of a dichotomizer working in a cost sensitive application can be given by the expected classification cost (EC) defined as:

$$\begin{aligned} EC = p(P)\cdot CFN\cdot FNR + p(N)\cdot CFP\cdot FPR + \\ p(P)\cdot CTP\cdot TPR + p(N)\cdot CTN\cdot TNR \end{aligned} \tag{2}$$

where $p(P)$ and $p(N)$ are the a priori probabilities of the positive and negative classes.

It is easy to show [5] that the cost matrix in table 1 is equivalent to the cost matrix in table 2 which depends only on the *cost ratio* $\rho = \dfrac{CFN - CTP}{CFP - CTN}$.

Table 2. Cost matrix for a two class problem.

		True class	
		N	*P*
Predicted Class	*N*	0	$\frac{CFN-CTP}{CFP-CTN}$
	P	1	0

As a consequence, the expressions of the expected cost changes in:

$$EC = p(N)\cdot FPR + p(P)\cdot FNR\cdot \rho \tag{3}$$

Let us now consider a multi class problem with n classes to be reduced to L binary problems by using a $n\times L$ coding matrix $M=\{M_{hk}\}$ where M_{h*} is the codeword for the class h and M_{hk} is the label assumed by a sample belonging to the class h in the binary problem induced by the k-th column. Moreover, let us assume that the costs of the multi-class problems are described by a $n\times n$ cost matrix $C = \{C_{ij}\}$ where $C_{ij} > 0$ represents the cost produced by assigning to the class j a sample actually belonging to the class i; the cost for a correct classification is null, i.e. $C_{ii} = 0 \;\; \forall i$.

Let us define $N^{(k)} = \{h \mid M_{hk} = -1\}$ the set of classes labelled with -1 and $P^{(k)} = \{h \mid M_{hk} = +1\}$ the set of classes labelled with +1 in the k-th binary problem. In an analogous way, let us call $C_{FP}^{(k)}$ and $C_{FN}^{(k)}$ the cost produced in the operative phase by the dichotomizer trained on the k-th problem when it erroneously assigns to $P^{(k)}$ a sample belonging to $N^{(k)}$ and vice versa.

To establish the values of $C_{FP}^{(k)}$ and $C_{FN}^{(k)}$, let us consider which are the consequences on the multi-class problem of an error made by the k-th dichotomizer. A false positive error moves one unit away from the true codewords containing a -1 in the k-th position toward the erroneous codewords containing a +1 in the same position. In particular, if r and s are two classes such that $r \in N^{(k)}$ and $s \in P^{(k)}$, a false positive error made by the k-th dichotomizer on a sample belonging to r will move one unit from the correct codeword of r, M_{r*}, toward the codeword of s, M_{s*}. Let us call $\mathrm{d}(M_{r*},M_{s*})$ the Hamming distance existing between M_{r*} and M_{s*}; if there were errors also on the other $\mathrm{d}(M_{r*},M_{s*})$ bits in which the two codewords differ, an error (with a cost equal to C_{rs}) would be generated in the multi-class problem. The contribution to such error given by the false positive produced by the k-th dichotomizer can be hence estimated as $\dfrac{1}{\mathrm{d}(M_{r*},M_{s*})}$; as a consequence, the cost of the false positive related to the possible misclassification between r and s can be estimated as $\dfrac{C_{rs}}{\mathrm{d}(M_{r*},M_{s*})}$.

Actually, the false positive moves the M_{r*} toward all the codewords belonging to $P^{(k)}$ and thus the cost related to all the possible misclassifications involving the class r can be estimated as:

$$\sum_{s \in P^{(k)}} \frac{C_{rs}}{\mathrm{d}(M_{r*},M_{s*})} \tag{4}$$

Eventually, we have to extend such evaluation to all the classes belonging to $N^{(k)}$. The conclusion is that the cost for a false positive made by the k-th dichotomizer is related to the risk of misclassifying one of the classes belonging to $N^{(k)}$ with one from $P^{(k)}$ and an estimate of its value is:

$$C_{FP}^{(k)} = \sum_{r \in N^{(k)}} \sum_{s \in P^{(k)}} \frac{C_{rs}}{\mathrm{d}(M_{r*},M_{s*})} \tag{5}$$

Likewise, it is possible to estimate the cost for a false negative made by the k-th dichotomizer since it is related to the risk of misclassifying one of the classes belonging to $P^{(k)}$ with one from $N^{(k)}$:

$$C_{FN}^{(k)} = \sum_{u \in P^{(k)}} \sum_{v \in N^{(k)}} \frac{C_{uv}}{\mathrm{d}(M_{u*},M_{v*})} \tag{6}$$

In this way, we can define for the k-th dichotomizer a cost matrix similar to the one shown in table 2 with cost ratio:

$$\rho_{ECOC}^{(k)} = \frac{\sum_{u \in P^{(k)}} \sum_{v \in N^{(k)}} \frac{C_{uv}}{\mathrm{d}(M_{u*}, M_{v*})}}{\sum_{r \in N^{(k)}} \sum_{s \in P^{(k)}} \frac{C_{rs}}{\mathrm{d}(M_{r*}, M_{s*})}} \tag{7}$$

It is easy to see that the conditions for a realistic cost matrix (i.e. $0 < \rho^{(k)} < +\infty$) are satisfied since $C_{ij} > 0\ \forall\ i \neq j$ and $\mathrm{d}(M_{r*},M_{s*}) \neq 0\ \forall r \neq s$.

Taking into account the generalized form of the coding matrix introduced in [3], eq. (7) is valid for each of the decomposition methods previously described. However it takes simpler expressions when a OVA or a OVO method is used. In the first case the expressions for $C_{FP}^{(k)}$ and $C_{FN}^{(k)}$ become:

$$C_{FP}^{(k)} = \sum_{r \neq k} \frac{C_{rk}}{\mathrm{d}(M_{r*}, M_{k*})} = \frac{1}{2} \sum_{r \neq k} C_{rk} \qquad C_{FN}^{(k)} = \sum_{v \neq k} \frac{C_{kv}}{\mathrm{d}(M_{k*}, M_{v*})} = \frac{1}{2} \sum_{v \neq k} C_{kv} \tag{8}$$

while the corresponding cost ratio is:

$$\rho_{OVA}^{(k)} = \frac{\sum_{v \neq k} C_{kv}}{\sum_{r \neq k} C_{rk}} \tag{9}$$

In the same way, it is possible to obtain the expressions of the costs for the binary problems induced by the OVO decomposition scheme. Let us suppose, without loss of generalization, that $M_{ik} = +1$, $M_{jk} = -1$ and $M_{hk} = 0$, $\forall\ h \neq i,j$, i.e. that the k-th dichotomizer must decide between class i and class j, with $P^{(k)} = \{i\}$ and $N^{(k)} = \{j\}$. Taking into account that $\mathrm{d}(M_{i*},M_{j*}) = 1\ \forall\ i \neq j$, the corresponding costs are:

$$C_{FP}^{(k)} = \frac{C_{ji}}{\mathrm{d}(M_{j*}, M_{i*})} = C_{ji} \qquad C_{FN}^{(k)} = \frac{C_{ij}}{\mathrm{d}(M_{i*}, M_{j*})} = C_{ij} \tag{10}$$

while the cost ratio is:

$$\rho_{OVO}^{(k)} = \frac{C_{ij}}{C_{ji}} \tag{11}$$

4 Experimental Results

To evaluate the effectiveness of the proposed approach we have made several experiments on different data sets and compared the classification costs obtained with the different decomposition schemes. To this aim, a comparison technique has been devised to assure that the outcomes obtained were statistically significant.

The data sets used are publicly available at the UCI Machine Learning Repository [6]; all of them have numerical input features and a variable number of classes. The features were previously rescaled so as to have zero mean and unit standard deviation. More details of data sets are given in table 3. The table provides also the type of ECOC coding matrix used for each data set. We choose an exhaustive code [2] for the sets that have a number of classes lower than 8 and a BCH code [2] for those having a number of classes greater or equal to 8. In particular, for Vowel set we adopted ECOC codes 15-11 available at http://web.engr.oregonstate.edu/~tgd.

Table 3. Data sets and coding matrices used in the experiments.

data Set	classes	feat.	samples	train. set	test set	valid. set	ECOC matrices	ECOC length
Ann-thyroid	3	21	7200	5040	1080	1080	Exhaustive	3
Dermatology.	6	34	358	252	54	52	Exhaustive	31
Glass	6	9	214	149	32	33	Exhaustive	31
Sat Image	6	36	6435	4505	965	965	Exhaustive	31
Segmentation	7	18	2310	1617	350	343	Exhaustive	63
Optdigits	10	62	5620	3935	844	841	BCH 31-21	31
Pendigits	10	16	10992	7696	1647	1649	BCH 31-21	31
Vowel	11	10	990	693	154	143	Diett 15-11	15

The dichotomizer employed is a Support Vector Machine with a RBF kernel; it has been implemented by SVM^{light} tool [7] available at http://svmlight.joachims.org. In order to build dichotomizers tuned on the cost matrices determined according the method described in Section 3, we have adopted a *post-learning scheme* [5] which evaluates a threshold t to be imposed on the output of the SVM, so as to attribute the sample to be classified to the class N if the SVM output is less than t and to the class P otherwise. The threshold is chosen so as to minimize the expected classification cost on a validation set.

We have compared each other the different results obtained with the decomposition schemes previously described. In particular, we have considered both standard versions of the ECOC (E-N), of the OVA (A-N) and of the OVO (O-N) which use dichotomizers without any cost-sensitive tuning and the corresponding cost-sensitive versions (E-C, A-C, O-C), where the dichotomizers are tuned according to the cost matrices built as seen in Section 3.

To avoid any bias in the comparison, 12 runs of a multiple hold-out procedure were performed on all data sets. In each run, the data set was split in three subsets: a training set (containing 70% of the samples of each class), a validation set and a test set (each containing 15% of the samples of each class); the final size of each of these sets is given in table 3. The validation set is used to evaluate the optimal threshold for the cost-sensitive tuning, while it is considered as part of the training set for the not cost-sensitive tuned dichotomizers.

The classification costs to be compared were evaluated on the test set, thus obtaining, for a given data set, 12 different values for each of the costs required. To establish a statistically significant comparison among all the decomposition schemes (both standard and cost-sensitive), we have used the Tukey's method [8] that tests all possible pairwise differences of means of distinct populations and determines if each difference is significantly lower than, higher than or undistinguishable from 0. All the results are provided with a level of significance equal to 0.05. To obtain a result unbiased with respect to the particular cost values, we apply the approach proposed in [9]: a hundred of different cost matrices have been used whose elements were randomly generated according to a uniform distribution over the range [1,10]. For each cost matrix, the test before described has been repeated.

In tables 4-11 the results of the comparisons performed are presented. We have a table in which the element *T*(*p*,*q*) on row *p* and column *q* indicates the number of runs (out of 100) for which the *p*-th scheme has produced a cost higher than the *q*-th scheme. For example, the element *T*(2,4) indicates the number of runs in which the A-N scheme gives a classification cost higher than the E-C scheme, while the symmetrical element *T*(4,2) reports the number of runs in which the E-C gives a higher cost. The number of runs in which the *p*-th and the *q*-th schemes provide costs not significantly different is given by 100-*T*(*p*,*q*)-*T*(*q*,*p*).

Table 4. Results for ANN Thyroid data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	0	0	100	100	100
A-N	0	0	0	100	100	100
O-N	0	0	0	100	100	100
E-C	0	0	0	0	0	26
A-C	0	0	0	0	0	27
O-C	0	0	0	0	0	0

Table 5. Results for Dermatology data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	0	0	100	100	96
A-N	0	0	0	100	100	96
O-N	0	0	0	100	100	97
E-C	0	0	0	0	0	8
A-C	0	0	0	52	0	40
O-C	0	0	0	30	8	0

Table 6. Results for Glass data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	5	0	68	37	62
A-N	0	0	0	63	17	50
O-N	28	57	0	84	60	77
E-C	0	0	0	0	0	0
A-C	10	6	6	45	0	37
O-C	0	0	0	0	1	0

Table 7. Results for Satimage data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	0	0	60	30	85
A-N	0	0	0	60	30	85
O-N	0	0	0	69	33	90
E-C	2	2	0	0	2	54
A-C	30	30	29	51	0	77
O-C	0	0	0	1	1	0

Table 8. Results for Segmentation data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	7	22	80	34	0
A-N	65	0	67	92	78	64
O-N	33	9	0	69	35	21
E-C	0	0	0	0	8	0
A-C	31	0	29	58	0	25
O-C	5	8	21	85	32	0

Table 9. Results for OptdiEgits data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	27	21	91	72	84
A-N	28	0	3	98	86	93
O-N	31	9	0	98	87	97
E-C	0	0	0	0	0	0
A-C	2	0	0	47	0	39
O-C	0	0	0	0	3	0

Table 10. Results for Pendigits data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	15	0	44	16	37
A-N	63	0	58	70	0	59
O-N	3	17	0	49	18	40
E-C	12	8	6	0	7	3
A-C	61	0	56	66	0	56
O-C	35	13	30	41	15	0

Table 11. Results for Vowel data set.

	E-N	A-N	O-N	E-C	A-C	O-C
E-N	0	13	10	19	13	8
A-N	60	0	43	63	0	36
O-N	41	9	0	23	9	2
E-C	30	13	0	0	13	0
A-C	60	0	45	64	0	36
O-C	58	28	32	54	27	0

It is worth noting that each table can be divided in four quadrants (evidenced by thick lines), each corresponding to a distinct kind of comparison: in particular, the top right (bottom left) quadrant reports the number of runs in which a cost-sensitive scheme works better (worse) than a non cost-sensitive scheme, while the top left (bottom right) quadrant contains the results of comparisons among the non cost-sensitive (cost-sensitive) schemes.

Let us firstly compare the cost-sensitive and non cost-sensitive versions of the decomposition schemes. We can note that cost-sensitive ECOC gives always better results than its non cost-sensitive counterpart except for Vowel data set. A similar behavior is shown by the OVA scheme, even though there are four cases (Glass, Satimage, Pendigits and Vowel) in which the two versions of the scheme are equivalent and by the OVO scheme that is equivalent to the standard version on Segmentation and Pendigits while performs significantly worse on Vowel data set.

If we focus only on cost-sensitive schemes, it is evident the superiority of the ECOC scheme with respect to OVA in all the cases except the ANN-Thyroid set, where the two schemes are equivalent. On the other side, the comparison between ECOC and OVO shows that the first scheme is better in half of the considered sets, while OVO outperforms ECOC in the ANN-Thyroid and Satimage sets. ECOC and OVO appear to be equivalent in Glass and Optdigits sets.

In summary, the experiments show that the proposed method achieves an improvement in terms of classification cost in cost-sensitive applications. This is specially true when using an ECOC decomposition scheme, which exploits its major robustness with respect to OVA and OVO schemes.

References

1. Hastie, T., Tibshirani, R.: Classification by Pairwise Coupling. The Annals of Statistics 26 (1998) 451-471
2. Dietterich, T.G., Bakiri, G.: Solving Multiclass Learning Problems via Error Correcting Output Codes. Journal of Artificial Intelligence Research 2 (1995) 263-286
3. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. Journal of Machine Learning Research 1 (2000) 113-141
4. Windeatt, T., Ghaderi, R.: Coding and Decoding Strategies for Multi Class Learning Problems. Information Fusion 4 (2003) 11-21
5. Tortorella, F.: An Empirical Comparison of In-Learning and Post-Learning Optimization Schemes for Tuning the Support Vector Machines in Cost-Sensitive Applications. Proc. 12th Int. Conf. on Image Anal. and Proc., IEEE Computer Society Press (2003), 560-565
6. Blake, C., Keogh, E., Merz, C.J.: UCI Repository of Machine Learning Databases. (1998) [www.ics.uci.edu/~mlearn/MLRepository.html]
7. Joachims, T.: Making Large-Scale SVM Learning Practical, in Schölkopf, B., Burges, C.J.C., Smola, A.J., eds., Advances in Kernel Methods, MIT Press (1999) 169-184
8. NIST/SEMATECH e-Handbook of Statistical Methods. http://www.itl.nist.gov/div898/handbook/ (2003)
9. Margineantu, D.D., Dietterich, T.G., Bootstrap Methods for the Cost-Sensitive Evaluation of Classifiers. Proc. Int. Conf. Machine Learning ICML-2000 (2000), 582-590

Cost-Based Classifier Evaluation for Imbalanced Problems

Thomas Landgrebe, Pavel Paclík, David M.J. Tax,
Serguei Verzakov, and Robert P.W. Duin

Elect. Eng., Maths and Comp. Sc., Delft University of Technology, The Netherlands
{t.c.w.landgrebe,p.paclik,d.m.j.tax,s.verzakov,r.p.w.duin}@ewi.tudelft.nl

Abstract. A common assumption made in the field of Pattern Recognition is that the priors inherent to the class distributions in the training set are representative of the true class distributions. However this assumption does not always hold, since the true class-distributions may be different, and in fact may vary significantly. The implication of this is that the effect on cost for a given classifier may be worse than expected. In this paper we address this issue, discussing a theoretical framework and methodology to assess the effect on cost for a classifier in imbalanced conditions. The methodology can be applied to many different types of costs. Some artificial experiments show how the methodology can be used to assess and compare classifiers. It is observed that classifiers that model the underlying distributions well are more resilient to changes in the true class distribution than weaker classifiers.

1 Introduction

Many typical discrimination problems can be expressed as a *target* versus *non-target* class problem, where the emphasis of the problem is to recover *target* examples amongst outlier or *non-target* ones. ROC analysis is often used to evaluate a classifier [9], depicting the operating characteristic in terms of the fraction of *target* examples recovered (True Positive rate or TP_r), traded off against the fraction of *non-target* examples classified as *target* (False Positive rate or FP_r). The ROC curve is a useful tool to optimise the trade-off between TP_r and FP_r. A loss matrix is often applied to these types of problems in an attempt to specify decision boundaries well suited to the problem, as discussed in [2], and [1]. Both TP_r and FP_r are invariant to changes in the class distributions [10].

TP_r and FP_r are not always the only costs used in assessing classifier performance. Some applications are assessed with other cost measurements, typical ones including *accuracy*, *purity/precision*, and *recall*, discussed in [6], and [8]. An example of this could be in automatic detection of tumours in images, where a human expert is required to make a final decision on all images flagged by the classifier as *target* (called the Positive fraction or $POSfrac$). In this application we could expect the number of *target* images (images actually depicting a tumour) to be less abundant than *non-target* images, and consequently the recognition system would be expected to minimise the amount of manual inspection

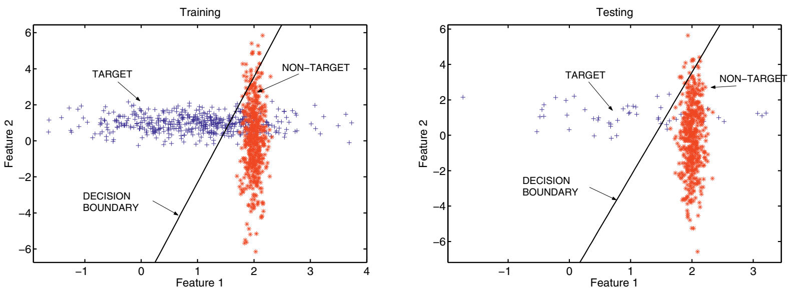


Fig. 1. Scatter diagrams to illustrate the example for balanced training conditions (left plot), but where the true class distribution is imbalanced (a prior of ÷ with respect to the *target* class simulating a real class distribution), shown in the right plot. Data is generated by the Highleyman distribution as in [4], proposed in [7].

required by the expert post-classification. A high $POSfrac$ would result in an inefficient automatic system, resulting in a high manual cost. Cost-based measurements such as $POSfrac$ and *purity* are dependent on true class distributions (as opposed to TP_r and FP_r) [5]. In some cases the actual distributions may be impossible to estimate or predict, implying that these costs may vary. It has been found that when *non-target* classes outnumber *target* classes, the effect on the costs of interest may be worse than expected [11], [8]. An example of this is shown in Figure 1 – here a balanced dataset is used for training (equal priors), but the right plot depicts the situation that arises when the true class distribution differs. Here the absolute number of *non-target* examples misclassified as *target* becomes comparable to the number of *target* examples correctly classified.

In this paper we discuss the evaluation with respect to cost of two-class discrimination problems between *target* and *non-target* classes in which the true class distribution is imbalanced and may vary (i.e. the abundance of examples for the two classes differ, called skewing). This is an important practical question that often arises, discussed and demonstrated here using synthetic examples in which the costs can easily be understood and compared. The objective is to formulate a procedure for evaluating classification problems of this nature. We show that in some situations where the true class distribution is extremely skewed in favour of the *non-target* class, the costs measurements could degrade considerably. As an example of how the proposed rationale can be applied, we choose two costs that are important to many applications, namely TP_r and $POSfrac$. Their relation is computed in conjunction with the ROC curve. These $POSfrac$ representations can be used to quickly, intuitively, and fairly, assess the outcome of the classifier for a given class distribution, or for a range of hypothetical class distributions (if it is unknown or varying). In a similar way, *Purity* or another cost measure could be used as part of the assessment procedure.

This paper is organised as follows: Section 2 introduces a theoretical framework, and shows how a classifier for the *target* versus *non-target* problem can be evaluated, discussing the construction of operating characteristics for the

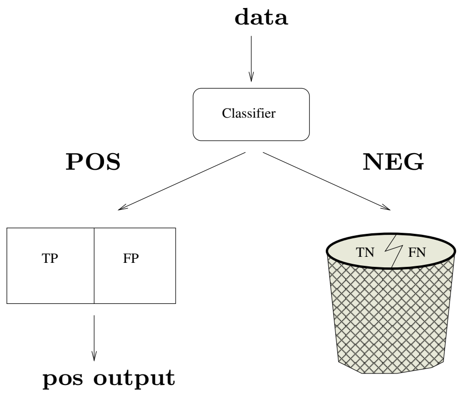


Fig. 2. A representation of the classification scheme discussed in this paper, showing a 2-class problem between a *target* and *non-target* class.

two costs emphasised here, namely TP_r and $POSfrac$. Section 3 consists of a discussion of the effect that skewed class-priors can have on the costs. A few experiments are performed in section 4 to illustrate the concepts discussed in the paper, showing direct application of the proposed methodology. Conclusions are given in section 5.

2 Classifier Evaluation between Two Classes

2.1 Notation and Problem Formulation

Consider the representation of a typical classification problem in Figure 2. Here it can be seen that a trained classifier analyses each incoming example, and labels each one as either positive (POS) or negative (NEG).

After classifying objects, four different object classifications can be distinguished (see Table 1). Data samples labeled by the tested classifier as *target* (the $POSfrac$) fall into two categories: true positives TP (true targets) and false positives FP (true non-targets). Corresponding true positive and false positive ratios TP_r and FP_r are computed by normalising TP and FP by the total amount of true targets N_t and non-targets N_n respectively.

Table 1. Defining a confusion matrix.

		estimated labels	
		target	non-target
true labels	target	TP	FN
	non-target	FP	TN

Data samples labeled by the classifier as non-target also fall in two categories, namely true negatives TN and false negatives FN. Note that $TN_r = 1 - FP_r$, and $FN_r = 1 - TP_r$. The examples labeled by the classifier as *target* are denoted POS, and those classified as *non-target* are denoted NEG. The fraction of objects which are positively labeled is the ‘Positive Fraction’, $POSfrac$, defined in equation 1.

$$POSfrac = \frac{POS}{N} = \frac{POS}{POS + NEG} \tag{1}$$

where N is the total number of objects in the test set. The fraction of examples in the POS output of the classifier that are really *target* is called the *purity/precision* of the classifier, defined as $Purity = \frac{TP}{POS}$. TP_r can be written as ($TP_r = \frac{TP}{TP+FN}$), and is equivalent to $Recall$. A TP_r of 90% implies that 90% of all the target objects will be classified positive by the classifier (classified in the left branch in Figure 2). The $POSfrac$ tends to increase with TP_r for overlapping problems, indicating a fundamental trade-off between the costs. If $P(C_t)$ is very low it would be expected that the $POSfrac$ will be very small. However it will be shown in the experiments that this is not always the case – overlapping classes and weak classifiers can result in a very undesirable $POSfrac$, depending on the operating condition.

2.2 ROC Analysis and *POSfrac* Analysis

Given a two class problem (*target* vs *non-target*), a trained density-based classifier and a test set, the ROC curve is computed as follows: the trained classifier is applied to the test set and the aposteriori probability is estimated for each data sample. Then, a set of thresholds Θ is applied to this probability estimate and corresponding data labelings are generated (this can be conceptualised as shifting the position of the decision boundary of a classifier across all possibilities). The confusion matrix is computed between each estimated set of labels and the true test-set labeling. The ROC curve now plots the TP_r as a function of the FP_r (see the left plot in Figure 3).

Note that the ROC curve is completely insensitive to the class priors, depending only on the class conditional probabilities. When the prior of one of the classes is increased (and therefore the probability of the other class is decreased), both the TP_r and the FP_r stay exactly the same (for a fixed classifier), although the absolute number of target and non-target objects change. Costs dependent on class distribution such as $POSfrac$ are considered, the ROC curve alone is not sufficient to assess performance.

In order to compute the corresponding $POSfrac$ operating characteristic for the classifier, the same set of thresholds Θ are used. Equation 1 can be written as equation 2, which can then be posed in terms of the ROC thresholds as in equation 3.

$$POSfrac = \frac{TP + FP}{N} = \frac{TP + FP}{TP + FP + FN + TN} = \frac{TP_r N_t + FP_r N_n}{N} \tag{2}$$

$$POSfrac(\Theta) = \frac{TP_r(\Theta) N_t + FP_r(\Theta) N_n}{N} \tag{3}$$

Similarly the *Purity* cost can be derived as in Equation 4.

$$Purity(\Theta) = \frac{TP(\Theta)}{TP(\Theta) + FP(\Theta)} = \frac{TP_r(\Theta) N_t}{TP_r(\Theta) N_t + FP_r(\Theta) N_n} \tag{4}$$

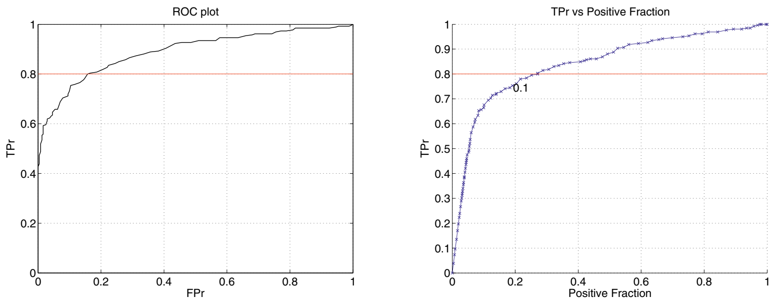


Fig. 3. ROC and $POSfrac$ plots for a linear discriminant classifier applied to Highleyman data, where $P(C_t) = 0.1$. Plots are made with respect to the *target* class.

3 The Effect of Class Imbalance on Classifier Performance

$POSfrac$ and *Purity* are two costs that are dependent on the skewness (imbalance) of the true class distribution. Define the true prior probability for the *target* class as $P(C_t)$, and for the *non-target* class as $P(C_n)$. Equations 3 and 4 can be written in terms of prior probabilities as shown in equations 5 and 6 respectively. Note that $P(C_t) = \frac{N_t}{N}$, $P(C_n) = \frac{N_n}{N}$, and $\frac{P(C_n)}{P(C_t)} = \frac{N_n}{N_t} = skewratio$.

$$POSfrac(\Theta) = \frac{TP_r(\Theta)N_t}{N} + \frac{FP_r(\Theta)N_n}{N} = P(C_t)TP_r(\Theta) + P(C_n)FP_r(\Theta) \quad (5)$$

$$Purity(\Theta) = \frac{TP_r(\Theta)}{TP_r(\Theta) + \frac{P(C_n)}{P(C_t)}FP_r(\Theta)} \quad (6)$$

Interestingly it is clear from equation 5 that the $POSfrac$ tends to FP_r as the skew ratio increases, and thus for extremely low $P(C_t)$ the ROC representation could be used alone in depicting the trade-off between costs. Now given an actual class-distribution and an operating condition, the $POSfrac$ and *Purity* can be calculated. For example, the corresponding ROC and $POSfrac$ curves for the example in Figure 1 is shown in Figure 3 for a linear discriminant classifier. Here $P(C_t) = 0.1$. It can be seen that for this condition, a TP_r of 80% would result in a $POSfrac$ of just under 30%. However, it could be that the exact class distribution is not known, and in fact sometimes the class distribution can vary from very small to extremely high levels. In these cases it can still be possible to evaluate and compare classifiers by investigating the operating characteristics across a range of priors in implementation. For example in the overlapping class problem in Figure 1, if the actual class distribution is completely unknown, it could be of value to investigate the TP_r and $POSfrac$ operating characteristic for a range of operating conditions. In Figure 4 the operating characteristic is shown for $P(C_t) = 0.5$, 0.1, and 0.001. The results for the linear classifier are shown in the left plot (the ROC plot remains constant as in the left plot of Figure 3). The right plot then shows the operating characteristic for a quadratic

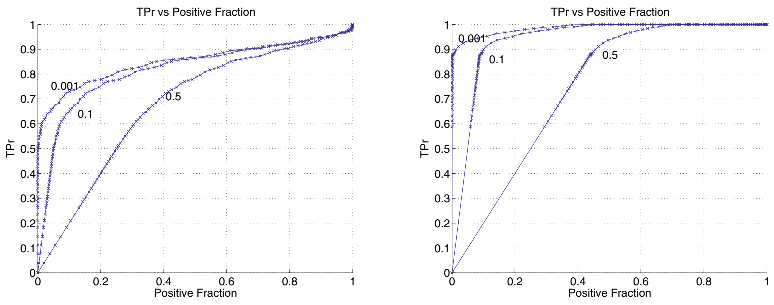


Fig. 4. $POSfrac$ plots for a linear discriminant (left plot) and quadratic discriminant (right plot) classifier applied to Highleyman data, where $P(C_t) = 0.5$, 0.1, and 0.001. The values on each curve represent different true class distributions.

classifier under the same conditions. It is clear that this classifier is much more resilient to changes in the true class distribution. Clearly as the linear classifier reaches a TP_r of 75% for a $POSfrac$ of 0.1, the quadratic classifier is substantially better at over 90%. Considering the case at which the classifiers are set to operate at a TP_r of 80%, a balanced dataset results in a $POSfrac$ of 54.9%, and a *Purity* of 77.9% for the linear classifier. These may be acceptable results. However when the class distribution changes such that $P(C_t) = 0.01$ (1 example in 100 is a *target*) the results become much worse. Whereas it could be expected that the $POSfrac$ should also drop by two orders of magnitude since the priors did so, for the linear classifier the $POSfrac$ only dropped to 24.8%. Conversely the quadratic classifier shows much better performance and resilience to changes of the true class distribution (in this case).

The simple example discussed shows that even if the true class distribution is unknown (or varies), two classifiers can still be compared to an extent. This becomes more useful when the underlying structure of the data is unknown and the choice of classifier less obvious. One example of this type of problem is in the field of geological exploration where the prior probabilities of different minerals change geographically, and often only a range of true class distributions is known.

4 Experiments

A number of experiments on artificial data are carried out in order to illustrate the effect (and indeed severity) that skewed data can have on costs in a number of situations. Four different classifiers are implemented for each data set, trained using a 30-fold cross-validation procedure. Each classifier is trained with equal training priors. For each classifier an ROC plot is generated such as in Figure 3, as well as the TP_r versus $POSfrac$ relation for the same thresholds as the ROC plot, generated for the following class distributions (as in Figure 4): $P(C_t) = 0.5$, 0.1, and 0.001. Thus each classifier is assessed from the case of balanced class distribution, to cases where the sampling is extremely skewed. In order to easily compare results a case study is performed for each classifier to estimate

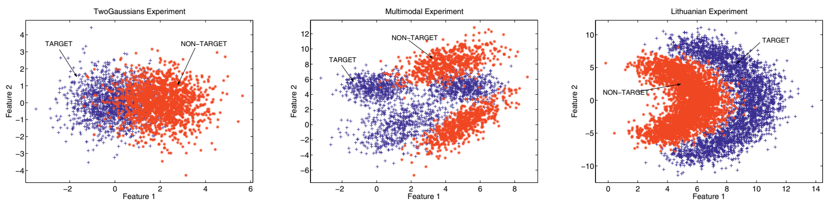


Fig. 5. Scatter diagrams of the 2-class experiments (the *Highleyman* plot is shown in Figure 1).

the effect on cost when the TP_r operating condition is fixed at 80% (in the same way $POSfrac$ or *Purity* could be used as the independent variable as the specification for the classifier). This allows each classifier to be compared easily, but still keeping cognisant that the results are for a specific operating condition only[1]. Note here that the objective of the classifier is to maintain the specified TP_r, and at the same time minimise the $POSfrac$ (and in some cases it could be more important to maximise *Purity*, but they are inversely dependent, i.e. a low $POSfrac$ results in a high *Purity*, since it can be shown that $POSfrac = \frac{TP}{N}\frac{1}{Purity}$).

The following classifiers are trained and evaluated for each data set: Linear Discriminant (LDC), Quadratic Discriminant (QDC), Mixture of Gaussian (GAUSSM), trained using the standard Expectation-Maximisation procedure, and the Parzen Classifier, as in [3] where the width parameter is optimised by maximising the log-likelihood with a leave-one-out procedure. The classifiers range from low to high complexity, the complex ones hypothetically capable of handling more difficult discrimination problems. These are all density-based classifiers, capable of utilising prior-probabilities directly, and can be compared fairly. The datasets are illustrated in Figure 5, corresponding to the following experiments, where 1500 examples are generated for each class: *TwoGaussians* with two overlapping homogenous Gaussian classes with equal covariance matrices and a high Bayes error is high at around 15.3%; *Highleyman* consisting of two overlapping Gaussian classes with different covariance matrices according to the Highleyman distribution (as in the *prtools* toolbox [4]); *Multimodal*, a multi-modal dataset with two modes corresponding to the first class, and three to the second; *Lithuanian* where two rather irregular classes overlap. All are computed under the *prtools* library [4].

4.1 Results of Experiments

For conciseness only the case study results are presented, showing the effect on $POSfrac$ cost for a single operating point, across a number of different class distributions. However the ROC and $POSfrac$ representations for the linear classifier in the *Highleyman* experiment have been shown before in the left plots

• A different operating point could for example be in favour of a different classifier.

Table 2. *POSfrac* results (with standard deviations shown) for the four data sets, where the classifiers are fixed to operate to recover 80% of *target* examples. Results are shown for three different class distributions.

Experiment	**P(C$_t$)**	**LDC**	**QDC**	**MOGC**	**PARZEN**
TwoGaussians	0.5	$46.08 \pm 0.51\%$	$45.99 \pm 0.50\%$	$47.86 \pm 0.77\%$	$45.58 \pm 0.54\%$
	0.1	$18.94 \pm 0.92\%$	$18.78 \pm 0.90\%$	$22.14 \pm 1.38\%$	$18.05 \pm 0.98\%$
	0.001	$12.22 \pm 1.02\%$	$12.05 \pm 1.00\%$	$15.78 \pm 1.53\%$	$11.24 \pm 1.08\%$
Highleyman	0.5	$52.14 \pm 1.23\%$	$40.00 \pm 0.00\%$	$40.00 \pm 0.00\%$	$40.00 \pm 0.00\%$
	0.1	$29.85 \pm 2.22\%$	$8.00 \pm 0.00\%$	$8.00 \pm 0.00\%$	$8.00 \pm 0.00\%$
	0.001	$24.33 \pm 2.46\%$	$0.08 \pm 0.00\%$	$0.08 \pm 0.00\%$	$0.08 \pm 0.00\%$
Multimodal	0.5	$69.00 \pm 1.06\%$	$50.25 \pm 0.94\%$	$41.20 \pm 0.22\%$	$40.93 \pm 0.21\%$
	0.1	$60.21 \pm 1.90\%$	$26.46 \pm 1.68\%$	$10.15 \pm 0.40\%$	$9.68 \pm 0.38\%$
	0.001	$58.03 \pm 2.11\%$	$20.57 \pm 1.87\%$	$2.47 \pm 0.45\%$	$1.94 \pm 0.42\%$
Lithuanian	0.5	$46.81 \pm 0.81\%$	$42.10 \pm 0.33\%$	$40.16 \pm 0.04\%$	$40.06 \pm 0.04\%$
	0.1	$20.26 \pm 1.45\%$	$11.77 \pm 0.59\%$	$8.28 \pm 0.07\%$	$8.11 \pm 0.07\%$
	0.001	$13.68 \pm 1.61\%$	$4.27 \pm 0.65\%$	$0.39 \pm 0.08\%$	$0.21 \pm 0.08\%$

of Figures 3 and 4 respectively. Table 2 shows the *POSfrac* results for the three different class distributions for the chosen operating point. In the *TwoGaussians* all the classifiers show an extreme effect on cost as the skewness is increased. The *POSfrac* remains above 10% even when the distribution is such that only one example in a thousand are *target*. Here the overlap between the classes does not allow for any improvement (a high Bayes error). In the *Highleyman* experiment it can be seen that only the linear classifier is severely affected by different class distributions for the given operating condition. This experiment suggests that an inappropriate or weak classifier can be disastrous in extreme prior conditions, even though it may have seemed acceptable in training (classifiers are often trained assuming balanced conditions). The *Multimodal* experiment presented a multimodal overlapping problem, and as expected the more complex classifiers fared better. Whereas the linear and quadratic classifiers showed a *POSfrac* of over 20% with extreme priors, the mixture-model and parzen classifiers performed a lot better. Similar results were obtained in the *Lithuanian* experiment.

5 Conclusion

This paper discussed the effect of imbalanced class distributions on cost, concentrating on 2-class problems between a class of interest called a *target* class, and a less interesting *non-target* class. A methodology was proposed in order to compare classifiers with respect to cost under conditions in which training conditions are fixed (often balanced), and true class distributions are imbalanced or varying.

It was shown that even though costs such as the true and false positive rates are independent of the true class distributions, other important costs such as *POSfrac* and *Purity* are dependent on it. Thus for these types of problems we proposed that in addition to traditional ROC analysis, a similar analysis of the other costs should be made simultaneously, evaluating the effect of a changing class distribution.

Following some simple experiments, it was observed that in some cases, classifiers that appeared to perform well on balanced class distribution data failed completely in imbalanced conditions. Conversely some classifiers showed resilience to the imbalance, even when extreme conditions were imposed. Thus we conclude that especially in cases in which the underlying data structure is complex or unknown, an analysis of the effect of varying and imbalanced class distributions should be included when comparing and evaluating classifiers.

Acknowledgements

This research is/was supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Ministry of Economic Affairs. Support was also provided by TSS Technology Research, De Beers Consolidated Mines, South Africa.

References

1. C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press Inc., New York, first edition, 1995.
2. R. Duda, P. Hart, and D. Stork. *Pattern Classification.* Wiley - Interscience, second edition, 2001.
3. R.P.W. Duin. On the choice of smoothing parameters for parzen estimators of probability density functions. *IEEE Trans. Computing*, 25:1175–1179, 1976.
4. R.P.W. Duin. *PRTools Version 3.0, A Matlab Toolbox for Pattern Recognition.* Pattern Recognition Group, TUDelft, January 2000.
5. P. Flach. The geometry of roc space: understanding machine learning metrics through roc isometrics. *ICML-2003 Washington DC*, pages 194–201, 2003.
6. D. J. Hand. *Construction and Assessment of Classification Rules, ISBN 0-471-96583-9.* John Wiley and Sons, Chichester, 1997.
7. W. Highleyman. Linear decision functions, with application to pattern recognition. *Proc. IRE*, 49:31–48, 1961.
8. M. Kubat and S. Matwin. Addressing the curse of imbalanced data sets: One-sided sampling. *Proceedings, 14th ICML, Nashville*, pages 179–186, July 1997.
9. C. Metz. Basic principles of roc analysis. *Seminars in Nuclear Medicine*, 3(4), 1978.
10. F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42:203–231, 2001.
11. G.M. Weiss and F. Provost. The effect of class distribution on classifier learning: an empirical study. *Technical report ML-TR-44, Department of Computer Science, Rutgers University*, August 2 2001.

A Two-Stage Classifier with Reject Option for Text Categorisation

Giorgio Fumera, Ignazio Pillai, and Fabio Roli

Dept. of Electrical and Electronic Eng., University of Cagliari
Piazza d'Armi, 09123 Cagliari, Italy
{fumera,pillai,roli}@diee.unica.it

Abstract. In this paper, we investigate the usefulness of the reject option in text categorisation systems. The reject option is introduced by allowing a text classifier to withhold the decision of assigning or not a document to any subset of categories, for which the decision is considered not sufficiently reliable. To automatically handle rejections, a two-stage classifier architecture is used, in which documents rejected at the first stage are automatically classified at the second stage, so that no rejections eventually remain. The performance improvement achievable by using the reject option is assessed on a real text categorisation task, using the well known Reuters data set.

1 Introduction

Text categorisation (TC) systems are key components of many applications of document managing, like document retrieval, routing, and filtering. With the increased availability of documents in digital form over the last decade, and the consequent need of automatic document management systems, TC has become an active research topic in the machine learning field. TC can be viewed as a classification task, in which a document in natural language must be labeled as belonging or not to thematic categories from a predefined set, on the basis of its content [10]. Accordingly, in recent years, several researches investigated the use of statistical pattern recognition techniques applied to TC (a comprehensive review is given in [10]). In particular, different kinds of classification techniques have been evaluated and compared, like neural networks, k-nearest neighbors, support vector machines, naïve Bayes, and multiple classifier systems [5, 12, 6, 9], as well as feature extraction and selection techniques [11].

In this work, we focus on the reject option, which is a technique used to improve classification reliability in pattern recognition systems. The reject option has been formalised in the context of statistical pattern recognition, under the minimum risk theory, in [1, 2]. It consists in withholding the automatic classification of a pattern, if the decision is considered not sufficiently reliable. Rejected patterns must then be handled by a different classifier, or by a human operator. This requires to find a trade-off between the achievable reduction of the cost due to classification errors, and the cost of handling rejections (costs are obviously application-dependent). Although the reject option turns out to be very useful in many pattern recognition systems, its use in TC systems has not been considered in the literature so far.

In a previous work, summarised in Sect. 3, we investigated how the reject option can be implemented in a TC system, and whether it can improve its reliability [3]. We implemented the reject option by allowing a text classifier to reject any subset of category assignments, for a given document. Using three different kind of classifiers (neural networks, k-nearest neighbors and support vector machines), we experimentally observed remarkable performance improvements, at the expense of small rates of rejected assignments. However, the rejected category assignments turned out to be spread across a large fraction of documents, making it impractical to handle them manually.

In this paper, we investigate whether the documents with rejected category assignments can be automatically handled. To this aim, we implement a two-stage classifier, based on the multi-stage architecture defined in [8] for pattern recognition systems. In our classifier, described in Sect. 4, documents can be either classified or rejected at the first stage. All the documents rejected at the first stage are then classified at the second stage, so that no rejections eventually remain. The effectiveness of this approach is evaluated by preliminary experiments carried out on the well known Reuters data set. The experimental results are presented in Sect. 5.

2 Text Categorisation

In TC systems, a document is typically represented as a vector of weights $d = (w_1, \ldots, w_T)$, where each w_k is associated to one of the T words that occur in training documents (*bag of words* approach). Weights can be computed in several ways, and are usually related to the frequency of the corresponding words, both in the document and in the whole training set [10]. While traditional classification problems are single-label, TC is a multi-label problem, i.e. each document can belong to any subset of C predefined categories $c_1, \ldots, c_C$. Given an input document d, a text classifier usually provides a score s_i for each category c_i, denoting the likelihood that d belongs to c_i. Several strategies can then be used to decide which categories d should be assigned to, given the scores. One of the most used strategies consists in determining a threshold τ_i for each c_i, after the training phase of the classifier, using a separate validation set. In the classification phase, each score is compared with the corresponding threshold: if $s_i \geq \tau_i$ ($s_i < \tau_i$), then d is labeled as (not) belonging to c_i [13, 10]. For instance, if neural networks are used as base classifiers, C output units can be used, each one related to one category, and their output values are taken as the scores s_i. Instead, the k-nearest neighbors (k-NN) classifier is implemented by first retrieving the k training documents most similar to an input document d. The similarity between two documents d and d' is computed using the cosine measure $\frac{d^{\mathrm{T}} \cdot d'}{\|d\| \cdot \|d'\|}$. Then, the score s_i for each category c_i is computed as the sum of the similarity measures between d and the training documents belonging to c_i, among the k nearest neighbors of d [10].

The performance measures used in TC are based on *precision* and *recall*, derived from the field of information retrieval. Precision π_i, for the i-th category, is defined as the fraction of documents that belong to c_i, among the ones that are assigned to c_i by the classifier. Recall ρ_i is defined as the fraction of documents that are correctly assigned by the classifier to c_i, among the ones that belong to c_i. Denoting with TP_i (True Positive) and FP_i (False Positive) the number of documents, out of a given set, correctly and erroneously labeled as belonging to c_i by the classifier, and with FN_i (False Negative) the number of documents erroneously labeled as not belonging to c_i, we have:

$$\pi_i = \frac{TP_i}{TP_i + FP_i}, \quad \rho_i = \frac{TP_i}{TP_i + FN_i} \ . \tag{1}$$

A global performance measure over all categories can be obtained either by micro- or macro-averaging the above category-related values, depending on application requirements. Micro- and macro-averaged values are defined respectively as follows:

$$\pi^\mu = \frac{\sum_{i=1}^C TP_i}{\sum_{i=1}^C (TP_i + FP_i)}, \quad \rho^\mu = \frac{\sum_{i=1}^C TP_i}{\sum_{i=1}^C (TP_i + FN_i)} \ , \tag{2}$$

$$\pi^{\mathrm{M}} = \frac{1}{C}\sum_{i=1}^C \pi_i, \quad \rho^{\mathrm{M}} = \frac{1}{C}\sum_{i=1}^C \rho_i \ . \tag{3}$$

The values of precision and recall lie in the range $[0, 1]$, and are not independent on each other: by using different classifier parameters (for instance, different thresholds τ_i) higher values of precision can be achieved at the expense of a lower recall, and vice-versa. Often, the combined measure F_1 is used, defined as the harmonic mean of precision and recall: $F_1^\mu = \frac{2\pi^\mu\rho^\mu}{\pi^\mu+\rho^\mu}, \quad F_1^{\mathrm{M}} = \frac{1}{C}\sum_{i=1}^C \frac{2\pi_i^{\mathrm{M}}\rho_i^{\mathrm{M}}}{\pi_i^{\mathrm{M}}+\rho_i^{\mathrm{M}}}$. Also the F_1 measure takes on values in the range $[0, 1]$.

3 Text Categorisation with Reject Option

For single-label classification problems, the meaning of rejecting a pattern is usually to withhold automatically deciding the class to which it should be assigned [1, 2, 8]. The human operator (or the classifier) that handles rejections has thus to decide among the whole set of classes. A slightly different approach was used in [4]: a pattern can be automatically labeled as *not* belonging to any subset of classes, while it is rejected only by the remaining classes. Accordingly, the final decision should be taken among the latter classes only. Anyway, a rejected pattern must be assigned to only one class. To the best of our knowledge, the concept of rejecton has not been extended so far to multi-label problems.

Taking into account that multi-label TC problems involving C categories are usually viewed as C independent two-class problems (each one consisting in deciding whether a document should be assigned or not to the corresponding category), in [3] we proposed to implement the reject option as follows: given a document d, a text classifier can automatically label d as belonging or not to any subset of the C categories, while it rejects d from the remaining categories, i.e. no decision is taken about these latter categories. Using the decision strategy without reject option, based on category-related thresholds τ_i (see Sect. 2), we experimentally found that the distribution of the scores s_i corresponding to incorrect category assignments, was peaked around the threshold τ_i. In other words, for any given category c_i, lower values of $|s_i - \tau_i|$ correspond to less reliable decisions. Accordingly, to implement the reject option as described above, we used two threshold values for each category c_i, τ_{L_i} and τ_{H_i} (with $\tau_{\mathrm{L}_i} \leq \tau_{\mathrm{H}_i}$), and the following decision strategy:

$$\begin{array}{ll} \text{if } s_i \leq \tau_{\mathrm{L}_i}, & \text{then } d \text{ is labeled as not belonging to } c_i; \\ \text{if } \tau_{\mathrm{L}_i} < s_i < \tau_{\mathrm{H}_i}, & \text{then } d \text{ is rejected from category } c_i; \\ \text{if } s_i \geq \tau_{\mathrm{H}_i}, & \text{then } d \text{ is labeled as belonging to } c_i. \end{array} \tag{4}$$

As the thresholds τ_i for the case without reject option, also the thresholds $\tau_{\mathrm{L}_i}, \tau_{\mathrm{H}_i}, i = 1, \ldots, C$, should be computed after the training phase of the classifier, on a separate validation set, by maximising the chosen performance measure. However, defining a performance measure for a TC problem with reject option is not straightforward. The measure typically used in statistical pattern recognition is the expected value (named expected risk), of the classification cost of a pattern, for a given decision rule. Different costs are defined for correctly classified patterns and for misclassified ones. When the reject option is used, it is straightforward to take into account the costs of rejections in the definition of the expected risk. In this case, it turns out that minimising the expected risk is equivalent to finding the best trade-off between the misclassification and rejection rates, depending on the corresponding costs [1, 2]. Instead, for TC problems, the precision and recall measures are not based on classification costs. It is thus not straightforward to generalise them to take into account the costs of rejections. Moreover, even defining the cost of rejections is not easy, since they are strongly application-dependent: in general, the cost for manually handling a document with some rejected category assignments could depend both on the time required by a person to read that document, and on the number of rejected assignments.

Nevertheless, when the reject option in a TC system is implemented as described above, it still makes sense to evaluate the performance through the same measures used without reject option, based on precision and recall, provided that the rejected category assignments are not taken into account when computing TP_i, FP_i and FN_i in (1) and (2). Such measure should be considered together with a measure related to the cost of documents with rejected assignments. Given the above difficulties in defining the cost of rejections, in [3] we chose to consider simply the rate of rejected decisions, i.e. the percentage of rejected category assignments over all test documents,[1] which will be denoted in the following as the "reject rate".

[1] Given D documents and C categories, the total number of category assignments is $D \cdot C$.

In [3], we experimentally evaluated the effectiveness of such implementation of the reject option, on the well known Reuters data set. We used neural networks, k-nearest neighbors and support vector machines as base classifiers. We considered all the main performance measures, i.e. the precision-recall curve and the F_1 measure, both micro- and macro-averaged. The C pairs of threshold values $\tau_{\mathrm{L}_i}, \tau_{\mathrm{H}_i}, i = 1, \ldots, C$, were computed by maximising the considered performance measure, while keeping the reject rate below a predefined value. Values of the reject rate in the range $[0, 0.15]$ were considered. For all the classifiers and all the performance measures considered, we found remarkable performance improvements, at the expense of small values of the reject rate. For instance, using the neural network classifier, the values of F_1^{M} increased from 0.39 to 0.47, as the reject rate increased from 0 to 0.03 (i.e. up to 3% of the total category assignments were rejected for test documents), while F_1^{μ} increased from 0.83 to 0.94, as the reject rate increased from 0 to 0.04. However, the analysis of these results showed that such performance improvements were always achieved by rejecting a small number of category assignments from a large fraction of documents. In particular, for most documents (50% to 75% of all test documents, depending on the reject rate) only one category assignment was rejected. Moreover, the number of documents with n rejected assignments decreased for increasing n. This could be a serious problem from the practical viewpoint, if documents with rejected assignments are manually handled, and if the cost of handling them depends mainly on the time required by a person to read a document, rather than on the number of rejected assignments. In this case, handling a single document rejected from ten categories could be much faster than handling ten documents, each rejected from one category, even if the number of rejected assignments is the same. It is worth noting that TC tasks involve usually several dozen categories (in the version of the Reuters data set we used, $C = 90$).

In conclusion, although we found that the reject option can lead to remarkable performance improvements at the expense of a small percentage of rejected category assignments, it turned out that such rejections are spread across a large fraction of documents: handling them manually is thus likely to be impractical for real applications. A possible solution to this problem is to automatically handle the documents with rejected category assignments, through the use of a second-stage classifier. This approach is investigated in the rest of this paper.

4 A Two-Stage Classifier with Reject Option

For pattern recognition applications in which a rejection is not acceptable as a final result, a multi-stage classifier architecture was proposed in [8] to automatically treating the rejects. At all stages, but the last one, a pattern can be either classified or rejected. Rejected patterns are fed into the next stage. At the final stage, a decision is taken in any case, so that no rejections eventually remain. The rationale is that each stage should use more informative, and thus more costly measurements than the previous stage. An interesting implementation of such approach has been recently proposed in [7]: a "global" and fast neural network classifier is used at the first stage, followed by a "local" and slower nearest neighbor classifier at the second stage. To speed up the response time of the second stage, it is trained on a subset of the training patterns of the first stage. More precisely, only patterns rejected at the first stage (possibly using a narrower rejection criterion) are used. Moreover, when a test pattern is rejected at the first stage, the second stage decides only among the top-h ranking classes returned by the first stage, using only the training patterns belonging to such classes. In this work, we chose to investigate the two-stage approach proposed in [7], after modifying it to fit the characteristics of a multi-label problem. This approach was implemented as follows.

Let $\mathbb{C}$ denotes the set of all categories $\{c_1, \ldots, c_C\}$, and T' the training set of the first stage classifier. The decision thresholds, that in the following will be denoted as $\tau'_{\mathrm{L}_i}, \tau'_{\mathrm{H}_i}, i = 1, \ldots, C$, are computed using a separate validation set V', by maximising the chosen performance measure, while keeping the reject rate below a predefined value r', and also enforcing that the fraction of documents rejected from each category does not exceed r'. The algorithms we used are described in [3]. In the classification phase, each document is classified according to decision rule (4), using the scores s'_i provided by the first stage classifier, and the thresholds $\tau'_{\mathrm{L}_i}, \tau'_{\mathrm{H}_i}$. For rejected categories, the final decision is taken by the second stage, that we implemented as a modified k-NN classifier.

The training set of the second stage classifier is $\mathrm{T}'' = \bigcup_{i=1}^{C} \mathrm{T}''_i$, where T''_i is the subset of documents of T', for which the decision for category c_i was rejected at the first stage, using a narrower rejection criterion, i.e. using a new set of thresholds $\tau''_{\mathrm{L}_i}, \tau''_{\mathrm{H}_i}$, such that $\tau''_{\mathrm{L}_i} \leq \tau'_{\mathrm{L}_i}$, and $\tau''_{\mathrm{H}_i} \geq \tau'_{\mathrm{H}_i}, i = 1, \ldots, C$. Such thresholds are computed starting from the values of $\tau'_{\mathrm{L}_i}, \tau'_{\mathrm{H}_i}$, and imposing that exactly a fraction $r'' \geq r'$ of documents of T' are rejected from each category, at the first stage. In other words, we require that $|\mathrm{T}''_i| = r'' \cdot |\mathrm{T}'|$. Note that, in general, $\mathrm{T}''_i \cap \mathrm{T}''_j \neq \emptyset$, since a document can be rejected from more than one category. The validation set V'' of the second stage is obtained from V' analogously.

Now, let $\mathbb{C}_{\mathrm{R}}(d) \subseteq \mathbb{C}$ denotes the set of categories for which the decision has been rejected at the first stage, for a given test document d. For each category $c_i \in \mathbb{C}_{\mathrm{R}}(d)$, the final decision is taken at the second stage by first computing a score s''_i, and then comparing it with a single threshold τ''_i: if $s''_i \geq \tau''_i$ ($s''_i < \tau''_i$), then d is labeled as (not) belonging to c_i. The score s_i is computed as follows. Let $\mathrm{T}''_i(d)$ denotes the documents that belong to category c_i, among the k documents of T''_i nearest to d, according to the cosine similarity measure described in Sect. 2. The score s''_i for d is computed as the sum of the similarity measures between d and all the documents in $\mathrm{T}''_i(d)$. The thresholds $\tau''_i, i = 1, \ldots, C$, are computed on the validation set V'', by maximising the same performance measure used at the first stage. We point out that this is a modified version of the k-NN classifier described in Sect. 2: indeed, the score for each category is computed using a different subset of the training set, instead of using the whole training set.

It is worth noting that this approach does not require to modify the performance measures based on precision and recall, to take into account the cost of rejections, since rejections are automatically handled. Instead, it could be necessary to find a trade-off between the achievable performance improvement, and the increased computational complexity of the two-stage architecture. This issue is discussed in the next section.

5 Experimental Results

In this section, we present the results of a first set of experiments aimed at evaluating whether the performance of a TC system can be improved using the two-stage classifier with reject option described in Sect. 4. The experiments have been carried out on the Reuters-21578 data set ("Mod-Apté" version), a standard benchmark for TC systems [10]. This data set consists of newswire stories classified under categories related to economics. After discarding unlabeled documents, and retaining only categories with at least one document both in the training set and in the test set, we obtained 7,769 training documents and 3,019 test documents belonging to $C = 90$ categories, with a vocabulary (extracted from the training set) of $T = 16{,}635$ words, after stemming and stop-word removal. We represented each document using the bag of words approach. The weights were computed using the well known TF-IDF strategy [10]. The 75% of documents of the original training set, randomly extracted, were used as the training set for the first stage classifier, T', while the remaining documents were used as validation set V'. Feature selection was performed on the training set, using the Information Gain criterion [10]. For these experiments, we used two different classifiers at the first stage: a multi-layer perceptron neural network (MLP), and a Naïve Bayes classifier (NB). For the MLP classifier, we used a number of input units equal to the number of document weights, and one output unit for each category. The MLP was trained with the standard back-propagation algorithm. The number of hidden neurons and of features (weights) was determined using the validation set, and was set respectively to 50 and 1,000. For the NB classifier, the number of features was set to 250. For the k-NN classifier at the second stage, we used 2,500 features, and a value of k equal to 10.

In Table 1 we report the first results obtained using the micro- and macro-averaged F_1 performance measures. The reported values of F_1 refer to the test set, and are average values over ten runs of the experiments, carried out using ten randomly generated training sets T'. We considered two values of the reject rate r', 0.05 and 0.10, and three different values of r'' (see Table 1), to evaluate the effect of different sizes of the training and validation sets at the second stage. The first row of Table 1, shows the F_1 values obtained by the first stage classifier without the reject option ($r' = r'' = 0$). The results obtained using the MLP and NB classifier at the first stage are reported respectively in columns "MLP+k-NN", and "NB+k-NN". For comparison, we also show the results achieved using a standard k-NN classifier at the second stage, trained using the same training set of the MLP at first stage ("MLP+k-NN*"), and a single standard k-NN classifier without the reject option ("k-NN"). For the two-stage classifier implemented as described in Sect. 4, the use of the reject option lead to an increase of the

Table 1. Average test set micro- and macro-averaged $F_\bullet$ values, obtained by two-stage classifiers with reject option (columns "MLP+k-NN", "MLP+k-NN*" and "NB+k-NN"), and by a standard k-NN classifier without reject option (column "k-NN"). The reject rate on the test set (r') is reported in the first column.

reject rates		k-NN		MLP+k-NN		MLP+k-NN*		NB+k-NN	
r'	r''	$F_\bullet^\mu$	$F_\bullet^\bullet$	$F_\bullet^\mu$	$F_\bullet^\bullet$	$F_\bullet^\mu$	$F_\bullet^\bullet$	$F_\bullet^\mu$	$F_\bullet^\bullet$
0	0	0.820	0.532	0.846	0.447	0.843	0.395	0.700	0.214
0.05	0.05			0.842	0.468	0.822	0.445	0.722	0.195
0.05	0.10			0.853	0.474	0.824	0.450	0.752	0.355
0.05	0.15			0.856	0.474	0.819	0.450	0.753	0.356
0.10	0.10			0.848	0.474	0.824	0.462	0.738	0.345
0.10	0.15			0.853	0.479	0.821	0.463	0.786	0.430
0.10	0.20			0.854	0.481	0.816	0.463	0.825	0.456

micro-averaged F_1 from 0.846 to 0.854 (using an MLP at the first stage), and from 0.700 to 0.825 (NB at the first stage). The macro-averaged F_1 increased from from 0.447 to 0.481 (MLP), and from 0.214 to 0.456 (NB). In particular, although the performance of the NB classifier without the reject option was quite poor when using the micro-averaged F_1, the reject option lead to a percentage improvement greater than 200%.

As one can expect, for the same value of the reject rate r', higher improvements are achieved using a larger training set at the second stage (i.e., higher values of r''). In particular, for small values of r'', the values of F_1 can decrease with respect to the case without reject option. This can be due to the small size of the training and validation sets of the second stage.

It is interesting to note that, in some cases, the two-stage classifier with the reject option outperformed the standard k-NN classifier without the reject option. This happened for the micro-averaged F_1, when a MLP was used at the first stage, and also for the NB classifier, but only for the highest values of r' and r''. Moreover, using a standard k-NN classifier at the second stage, with the same training set of the first stage ("MLP+k-NN*"), always lead to a worse performance than that achieved implementing the second stage as in Sect. 4 ("MLP+k-NN").

Finally, we point out that the performance obtained using the two-stage classifier, on the whole test set, was worse than that achieved by the first stage on only the *accepted* category assignments. For instance, for the MLP classifier at the first stage, such improvement was about 0.08 for both the micro- and macro-averaged F_1, for a reject rate $r' = 0.10$. This is not surprising, since, obviously, not all rejected category assignments are turned into correct assignments by the second stage classifier, as they could, in principle, if rejections were manually handled.

The above preliminary results show that the use of the reject option can improve the reliability of a TC system, even when rejections are automatically handled by a second stage classifier. These results indicate that higher improvements can be achieved by using higher reject rates r' at the first stage, and larger sizes of the training set of the second-stage classifier. In particular, it should be noted that the increase in computational complexity, due to the use of a larger training set for the k-NN classifier at the second stage, can be limited by the fact that most documents are rejected by only one category, as we experimentally observed.

References

1. Chow, C.K.: An optimum Character Recognition System Using Decision Functions. IRE Trans. on Electronic Computers **6** (1957) 247–254
2. Chow, C.K.: On Optimum Error and Reject Tradeoff. IEEE Trans. on Information Theory **16** (1970) 41–46
3. Fumera, G., Pillai, I., Roli, F.: Classification with Reject Option in Text Categorisation Systems. In: Proc. 12th International Conference on Image Analysis and Processing. IEEE Computer Society (2003) 582–587
4. Ha, T.M.: The Optimum Class-Selective Rejection Rule. IEEE Trans. on Pattern Analysis and Machine Intelligence **19** (1997) 608–615
5. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In: Proc. 10th European Conference on Machine Learning (1998) 137–142
6. Li, Y.H., Jain, A.K.: Classification of Text Documents. The Computer Journal **41** (1998) 537–546
7. Giusti, N., Masulli, F., Sperduti, A.: Theoretical and Experimental Analysis of a Two-Stage System for Classification. IEEE Trans. on Pattern Analysis and Machine Intelligence **24** (2002) 893–904
8. Pudil, P., Novovicova, J., Blaha, S., Kittler, J.: Multistage Pattern Recognition with Reject Option. In: Proc. 11th IAPR Int. Conf. on Pattern Recognition, Vol. 2. (1992) 92–95
9. Schapire, R.E., Singer, Y.: BoosTexter: a Boosting-Based System for Text Categorization. Machine Learning **39** (2000) 135–168
10. Sebastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys **34** (2002) 1–47
11. Yang, Y., Pedersen, J.O.: A Comparative Study on Feature Selection in Text Categorization. In: Proc. 14th Int. Conf. on Machine Learning (1997) 412–420
12. Yang, Y., Liu, X.: A Re-Examination of Text Categorization Methods. In: Proc. 22nd ACM Int. Conf. on Res. and Dev. in Inf. Retrieval (1999) 42–49
13. Yang, Y.: A Study on Thresholding Strategies for Text Categorization. In: Proc. 24th ACM Int. Conf. on Res. and Dev. in Inf. Retrieval (2001) 137–145

Misclassification Probability Estimations for Linear Decision Functions

Victor Mikhailovich Nedel'ko

Institute of Mathematics SB RAS, Laboratory of Data Analysis
660090 Novosibirsk, Russia
nedelko@math.nsc.ru
http://math.nsc.ru/LBRT/i1/nedelko/index.html

Abstract. The work is devoted to a problem of statistical robustness of deciding functions, or risk estimation. By risk we mean some measure of decision function prediction quality, for example, an error probability. For the case of discrete "independent" variable the dependence of average risk on empirical risk for the "worst" distribution ("strategies of nature") is obtained. The result gives exact value of empirical risk bias that allows evaluating an accuracy of Vapnik–Chervonenkis risk estimations. To find a distribution providing maximum of empirical risk bias one need to solve an optimization problem on function space. The problem being very complicate in general case appears to be solvable when the "independent" feature is a space of isolated points. The space has low practical use but it allows scaling well-known estimations by Vapnik and Chervonenkis. Such scaling appears to be available for linear decision functions.

1 Introduction

There is well known fact that decision function quality being evaluated by the training sample appears much better than its real quality. To get true risk estimation in data mining one uses a testing sample or moving test. But these methods have some disadvantages. The first one decreases a volume of sample available for building a decision function. The second one takes extra computational resources and is unable to estimate risk dispersion.

So one need a method that allows estimating a risk by training sample directly, i. e. by an empirical risk. This requires estimating first an empirical risk bias.

The problem was solved by Vapnik and Chervonenkis [1]. They introduced a concept of capacity (growth function) of a decision rules set. This approach is quite powerful, but provides pessimistic decision quality estimations. Obtained bias estimation is very rough, because of performed by authors replacement of a probability of a sum of compatible events by the sum of its probabilities.

The goal of this paper is to evaluate an accuracy of these estimations. We shall find out how far they are off by considering a case of discrete feature that allows obtaining an exact value of empirical risk bias.

2 Problem Definition

Let X – an "independent" variable values space, Y – a goal space of forecasting values, and C – a set of probabilistic measures on $D = X \times Y$. A measure $c \in C$ will be $P_c[D]$. The set C contains all the measures for those a conditional measure $P_c[Y/x]$ exists $\forall x \in X$.

Hereinafter square parentheses will hold a set on which σ-algebra of subsets the measure assigned, but parentheses — the measure of the set (probability of event).

A deciding function is a correspondence $f : X \to Y$.

For the determination of deciding functions quality we need to assign a function of losses: $L : Y^2 \to [0, \infty)$.

By a risk we shall understand an average loss:

$R(c, f) = \int L(y, f(x)) dP_c[D]$.

Another form is:

$R(c, f) = \int R_x(c, f)\, dP_c[X]$, where $R_x(c, f) = \int L(y, f(x))\, dP_c[Y/x]$.

For building a deciding function there is a random independent sample $v_c = \left\{ (x^i, y^i) \in D \;\middle|\; i = \overline{1, N} \right\}$ from distribution $P_c[D]$ used.

An empirical risk will mean a sample risk estimation: $\tilde{R}(v, f) = \frac{1}{N} \sum_{i=1}^{N} L\left(y^i, f\left(x^i\right)\right)$.

For the all practically used algorithms building deciding functions an empirical risk appears a biased risk estimation, being always lowered, as far as algorithms minimize an empirical risk.

So, estimating this bias is actual.

Enter indications:

$$F(c, Q) = ER\left(c, f_{Q,v}\right), \qquad \tilde{F}(c, Q) = E\tilde{R}\left(c, f_{Q,v}\right).$$

Here $Q : \{v\} \to \{f\}$ is an algorithm building deciding functions, and $f_{Q,v}$ – a deciding function built on the sample v by algorithm Q.

Expectation is calculated over the all samples of volume N.

Introduce an extreme bias function:

$$S_Q\left(\tilde{F}_0\right) = \hat{F}_Q\left(\tilde{F}_0\right) - \tilde{F}_0, \tag{1}$$

where $\hat{F}_Q\left(\tilde{F}_0\right) = \sup\limits_{c : \tilde{F}(c,Q) = \tilde{F}_0} F(c, Q)$.

One of the results of this work consists in finding the dependency $S_Q\left(\tilde{F}_0\right)$ for the multinomial case when X is discrete and Q minimizes an empirical risk in each $x \in X$.

3 Discrete Case

Let X be discrete, i. e. $X = \{1,\ldots,n\}$.

Then $R(c,f)=\sum_{x=1}^{n} p_x R_x(\xi_x, f(x))$, where $R_x(\xi_x, v_x)=\int L(y, f_{v_x}(x))\,dP_c[Y/x]$ – a conditional risk in the point x, $p_x = P_c(x)$, ξ_x – a short indication of conditional measure $P_c[Y/x]$, $v_x = \left\{(x^i, y^i)\in v \,\middle|\, x^i = x\right\}$, $f_{v_x}(x)$ – a decision built on sub-sample v_x.

Let the deciding function minimize an empirical risk: $f_v^*(x)=\arg\min_{y\in Y}\tilde{L}(y, v_x)$, $\tilde{L}(y,v_x)=\frac{1}{|v_x|}\sum_{v_x} L(y, y^i)$.

Determine values which average risks depend on:

$F(c,Q)=ER(c, f_v^*)=\sum_{x=1}^{n} F_x(p_x,\xi_x)$, where $F_x(p_x,\xi_x)=p_x ER_x(\xi_x, f_{v_x}^*(x))$.

Similarly:

$\tilde{F}(c,Q)=\frac{1}{N}E\tilde{L}(v, f_v^*)=\sum_{x=1}^{n}\tilde{F}_x(p_x,\xi_x)$, where $F_x(p_x,\xi_x)=E\tilde{L}_x(v_x, f_{v_x}^*(x))$.

Introduce the function

$\hat{F}_x(p_x,\tilde{F}_x^0)=\sup_{\xi_x:\tilde{F}_x(p_x,\xi_x)=\tilde{F}_x^0} F_x(p_x,\xi_x)$.

Now

$$\hat{F}_Q(\tilde{F}_0)=\max\sum_{x=1}^{n}\hat{F}_x(p_x,\tilde{F}_x^0), \tag{2}$$

where maximum is taken over the all p_x and $\tilde{F}_x^0$, $x=\overline{1,n}$ with restrictions: $p_x\ge 0$, $\tilde{F}_x^0\ge 0$, $\sum_{x=1}^{n}p_x=1$, $\sum_{x=1}^{n}\tilde{F}_x^0=\tilde{R}_0$.

Initial extreme problem is rather simplified, being split into two parts: finding $\hat{F}_x(p_x,\tilde{F}_x^0)$ and finding the maximum of function on the simple area of Euclid space.

Function $\hat{F}_x(p_x,\tilde{F}_x^0)$ may be easily approximated by calculation on a grid.

However it is impossible to solve a problem (2) directly by digital methods, because the dimensionality of space on that the maximization is performed is $2n$ where n may be great (twenty and the more).

4 Deciding the Extreme Problem

Let's reformulate a problem (2) in abstract indications:

$$\sum_{x=1}^{n} \Phi\left(z^x\right) \to \max_{z^x}, \tag{3}$$

$$z^x = \left(z_1^x, \ldots, z_m^x\right),\ z_j^x \geq 0,\ \sum_{x=1}^{n} z_j^x = 1,\ j = \overline{1,m}.$$

In (2) Φ corresponds to $\hat{F}_x$, m = 2, $z_1^x = p_x$, $z_2^x = \frac{\tilde{F}_x^0}{\tilde{F}_0}$.

Suppose a space of values of vector z^x to be made discrete: $z^x \in \left\{t^1, \ldots, t^l\right\}$.

Then problem (3) now may be put in the equivalent form:

$$\sum_{i=1}^{l} \Phi\left(t^i\right)\kappa^i \to \max_{\kappa^i}, \tag{4}$$

$$\kappa^i \geq 0,\ \sum_{i=1}^{l} t_j^i \kappa^i = 1,\ j = \overline{1,m},\ \sum_{i=1}^{l} \kappa^i = 1, \text{ where } \kappa^i \in \left\{\tfrac{\nu}{l} \,\middle|\, \nu = \overline{0,l}\right\}.$$

Solve a problem without the last restriction on κ^i.

This is a linear programming problem with a decision in a vertex of values area. This means that only m+1 of κ^i -s may be nonzero.

Now it is easy to show that the effect of discreteness κ^i restriction has an influence like $\frac{m}{l}$ and one may neglect it.

As far as the conclusion on the number of nonzero κ^i -s does not depend on the step of sampling a space of vector z^x values, the result may be propagated to the initial problem.

Theorem 1. Decision of problem (3) includes not more then m+1 different vectors.

Thereby the dimensionality of space on which maximizing is performed decreases to m (m + 1).

Applying to the problem (2) this comprises 6, and problem may be easily solved numerically.

5 Results

Offered method allows finding a dependency $S_Q\left(\tilde{F}_0\right)$ by any parameters n and N.

However there is the most illustrative and suitable for the comparison with other approaches the asymptotic case: $\frac{N}{n} = M = \text{const},\ N \to \infty,\ n \to \infty$.

This asymptotic approximation is wholly acceptable already by n = 10, herewith it has only one input parameter M.

For illustrations consider a problem of categorization (two classes).

The loss function is: $L(y, y') = \begin{cases} 0, & y = y' \\ 1, & y \neq y' \end{cases}$.

Now we are to calculate an accuracy of Vapnik–Chervonenkis evaluations for the considered case of discrete *X*, as far as we have found an exact dependency of average risk on the empirical risk for the "worst" strategy of nature.

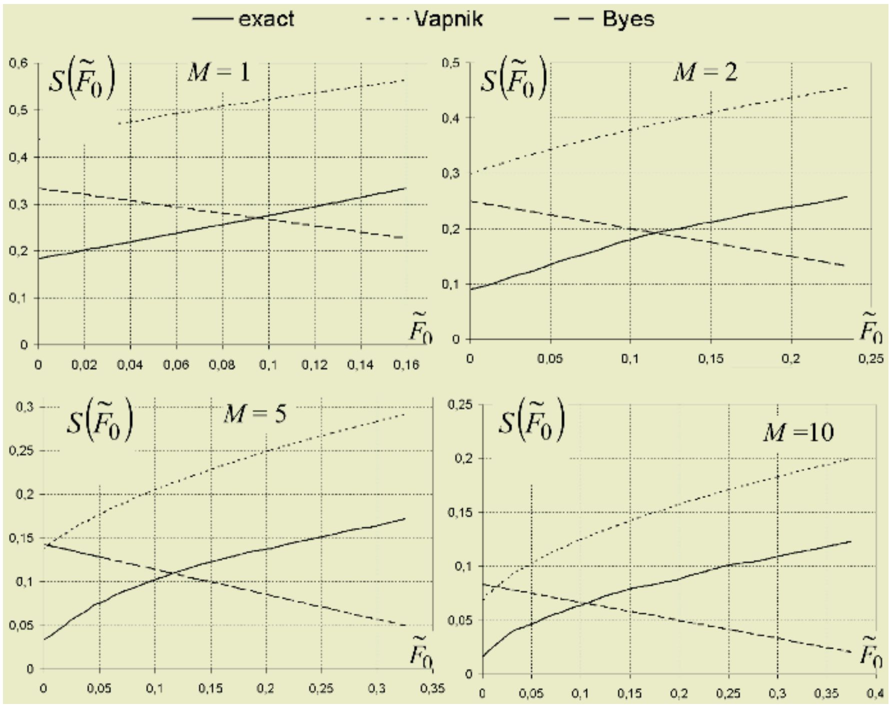


Fig. 1. The exact dependency of supreme risk bias on empirical risk and its estimations are shown by different *M*. Solid line shows difference between expected misclassification error and empirical risk, the difference maximized over the all distributions providing expected empirical risk $\tilde{F}_0$.

For $S(\tilde{F}_0)$ in [1] there is reported an estimation $S'_V(\tilde{F}_0)=\tau$, as well as an improved estimation: $S'_V(\tilde{F}_0)=\tau^2\left(1+\sqrt{1+\frac{2\tilde{F}_0}{\tau^2}}\right)$, where τ asymptotically tends to $\sqrt{\frac{\ln 2}{2M'}}$, $M'=M/\left(1-e^{-M}\right)$.

On figure 1 for the case of two classes by different *M* there are drawn the dependency $S(\tilde{F}_0)$ and its estimation $S_V(\tilde{F}_0)$. Plots demonstrate significant greatness of the last. Note that the accuracy of Vapnik–Chervonenkis estimation falls since the sample size (parameter *M*) decreases.

By $M \leq 1$ the "worst" distribution (that provides maximal bias) is uniform on *X* and the results obtained is consistent with results for multinomial case reported in [2] (table 3.7). By $M > 1$ and restricted $\tilde{F}_0$ the "worst" distribution is not uniform on X and this fact is new.

The third line on graphs (straight) presents a variant of dependency obtained by Byesian approach, based on assuming the uniform distribution on strategies of nature [3,5].

In work [5] there is reported the result: $S_B\left(\tilde{F}_0\right) = \dfrac{1 - 2\tilde{F}_0}{M + 2}$.

It may be a surprised fact that $S\left(\tilde{F}_0\right)$ on the significant interval appears to be greatly less than $S_B\left(\tilde{F}_0\right)$, though Byesian approach implies an averaging by the all nature strategies of nature, but our approach — the choice of the worst *c*.

6 Linear Decision Functions

Let us compare risk bias values obtained for discrete case with bias for linear decision functions.

For simplifying there was considered uniform distribution on features for both classes. For such c misclassification probability equals to 0.5 for every decision function, but empirical risk appears to be much lower.

We can evaluate an accuracy of Vapnik-Chervonenkis risk estimations for the case of discrete *X*, as far as we know an exact dependency of average risk on the empirical risk for the "worst" strategy of nature (distribution).

On fig. 2 for the case of two classes and discrete *X* there are drawn the dependency $S(M) = 0.5 - \tilde{F}(c_U)$ and its estimation $S_V(M) = \sqrt{\frac{\ln 2}{2M'}}$ by Vapnik and Chervonenkis. Here $\tilde{F}(c_U)$ – expected empirical risk by uniform distribution; $\frac{N}{n} = M = \text{const}$, $N \to \infty$, $n \to \infty$; $M' = M\big/\left(1 - e^{-M}\right)$ is sample size divided by VC-capacity of decision functions class in discrete case. Note that the uniform distribution on *D* provides maximum of empirical risk bias since we put no restrictions on $\tilde{F}_0$.

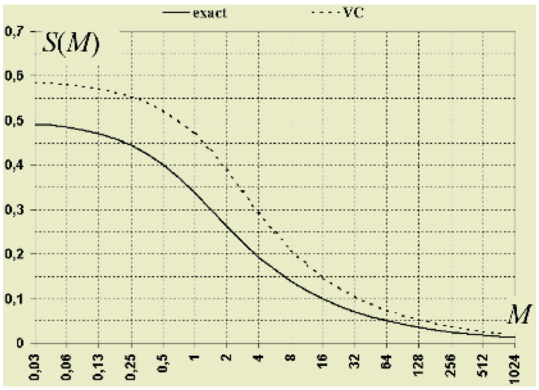


Fig. 2. Risk bias and VC–estimation. Discrete case, E*R* = 0.5.

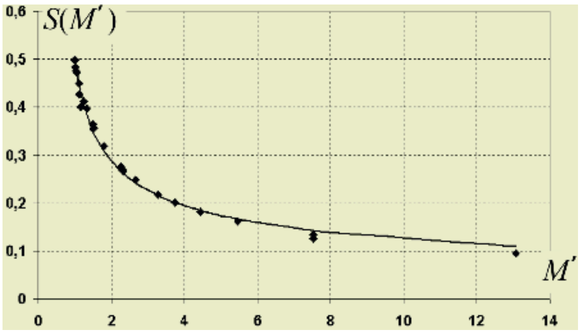


Fig. 3. Risk biases for multinomial and linear decision functions.

Graphics demonstrate significant greatness of Vapnik-Chervonenkis estimation.

To find a dependence $S(M)$ for linear deciding functions in $X = [0,1]^d$ a statistical modeling was used. By the modeling there was for each combination of parameters a hundred of samples drawn from uniform distribution on D, for each sample the best linear classifier built by exhaustive search.

A table shows the result of modeling. Here d – features space X dimensionality, N – sample size, $M' = \dfrac{N}{\log_2 \mathbf{C}}$ – sample size divided by VC-capacity of linear functions class ($\mathbf{C}$ is a total number of possible decision assignments to sample points by using linear decision functions), S – risk bias.

The same results are shown (by markers) on fig. 3 in comparison with $S(M')$ for discrete case (solid line).

Obtained results show that bias dependence on M' for linear functions is close to dependence for discrete (multinomial) case.

d	*N*	*M′*	*S*
1	3	1.16	0.4
1	5	1.5	0.36
1	10	2.31	0.27
1	20	3.75	0.20
1	50	7.53	0.13
1	100	13.08	0.095
2	4	1.05	0.47
2	10	1.53	0.357
2	20	2.33	0.27
2	50	4.44	0.18
2	100	7.53	0.13
3	5	1.019	0.48
3	10	1.247	0.41
3	20	1.79	0.32
3	50	3.28	0.22
3	100	5.46	0.162
4	6	1.008	0.498
4	10	1.111	0.45
4	20	1.5	0.355
4	50	2.66	0.249
5	7	1.0033	0.499
5	10	1.044	0.476
5	20	1.33	0.398
5	50	2.27	0.275

7 Conclusion

For the considered case of a discrete feature the exact maximum of empirical risk bias have been obtained. The comparison conducted shows that risk estimations by Vapnik and Chervonenkis may increase an expected risk up to several times from its true maximum. This means that these estimations may be essentially improved.

Obtained improvement is applyable also for continuous space, e.g. linear decision functions.

Practical use of the result consists in that one can apply obtained scaling of VC-estimations to real tasks. The results obtained for multinomial case may be propagated on continuous one by using VC-capacity of decision function class instead of n.

The work is supported by RFBR, grant 04-01-00858-a.

References

1. Vapnik V.N., Chervonenkis A. Ja. Theory of pattern recognition. Moscow “Nauka”, 1974. 415p. (in Russian).
2. Raudys S., Statistical and neural classifiers, Springer, 2001.
3. Lbov G.S., Startseva N.G. Logical deciding functions and questions of statistical stability of decisions. Novosibirsk: Institute of mathematics, 1999. 211 p. (in Russian).
4. Nedelko V.M. An Asymptotic Estimate of the Quality of a Decision Function Based on Empirical Risk for the Case of a Discrete Variable. // Pattern Recognition and Image Analysis, Vol. 11, No. 1, 2001, pp. 69-72.
5. Berikov V.B. On stability of recognition algorithms in discrete statement. // Artificial Intelligence. Ukraine, 2000, N. 2. P. 5–8. (in Russian).

Generalized Variable-Kernel Similarity Metric Learning

Johannes J. Naudé[1,2], Michaël A. van Wyk[3], and Barend J. van Wyk[3]

• Rand Afrikaans University, Auckland Park, Johannesburg, South Africa
• Kentron Dynamics, Centurion, South Africa
hannes.naude@kentron.co.za
• French South-African Technical Institute in Electronics
Tshwane University of Technology
Staatsartillerie Road, Pretoria, South Africa
mavw@fsatie.ac.za
ben.van.wyk@fsatie.ac.za

Abstract. Proximity-based classifiers such as RBF-networks and nearest-neighbour classifiers are notoriously sensitive to the metric used to determine distance between samples. In this paper a method for learning such a metric from training data is presented. This algorithm is a generalization of the so called Variable-Kernel Similarity Metric (VSM) Learning, originally proposed by Lowe and is therefore known as Generalized Variable-Kernel Similarity Metric (GVSM) learning. Experimental results show GVSM to be superior to VSM for extremely noisy or cross-correlated data.

1 Introduction

In a classification problem, we are given T training observations belonging to I distinct classes. Each training observation consists of a D-dimensional feature vector $\overline{x_t} = (x_{t1}, \ldots, x_{tD}) \in \mathbb{R}^d$ and the known class label $L_t, t = 1, \ldots, T$. The goal is to predict the class label of a previously unseen query $\overline{x_0}$. The K Nearest neighbour classification method is a simple and appealing approach to this problem: it finds the K nearest neighbours of x_0 in the training set and then predicts the class label of x_0 as the one that occurs most frequently in the K neighbours. Nearest neighbour classifiers allow rapid incremental learning from new instances and controlled removal of outdated or invalid training samples. Also, they are much easier to interpret than neural networks. Unfortunately their generalization performance is often inferior to competing methods [1].

As Lowe [1] has pointed out, the performance of these methods is highly dependent on the similarity metric that is used to select the neighbours. One example of a situation where an appropriate metric is important is the case where one or more of the features are irrelevant. Such a case is depicted in fig.1. Based on this Lowe [1] proceeded to develop a technique known as Variable Similarity Metric (VSM) learning to automatically scale the features appropriately and demonstrated that the generalization performance of this technique is state of the

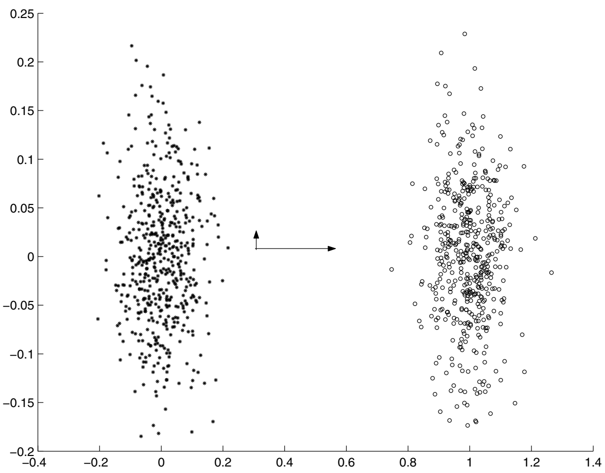


Fig. 1. VSM learns a metric that accentuates differences along the x axis and suppresses those along the y axis. The arrows represent the kind of metric that VSM will typically learn for this case. (Classes are widely separated for illustrative purposes.)

art. VSM learning can be run as a black box with no problem-specific parameters to be set by the user.

It was found that while VSM learning performs well for cases where the input features are uncorrelated, simple feature scaling is not powerful enough once the noise added to different features becomes cross-correlated (as depicted in fig. 2). One can reasonably expect that this will be the case in many pattern recognition applications, since all the features are derived from the same original input.

The solution proposed here is a generalized form of VSM learning (GVSM). GVSM optimizes more parameters in order to obtain a metric, the principal axes of which are not necessarily aligned with the coordinate axes. Unfortunately, as always, more degrees of freedom implies more vulnerability to overfitting.

In Section 2 we introduce the notation used for the remainder of the paper, while in Section 3 the equations governing K-nearest neighbour classification is discussed in more detail. In order to optimize the metric used, we need a measure of the performance of a metric and the partial derivatives of this measure with respect to the variables to be optimized, these are introduced in Section 4. Nearest neighbour methods are often perceived as more computationally expensive than competing methods, and for this reason Section 5 discusses some of the speed-up techniques that can fruitfully be applied. Section 6 reports some results obtained when GVSM was compared to VSM on synthetic data. We discuss the alternative interpretation of GVSM as a transform design method in Section 7 and our conclusions and suggestions for future research appear in the last section.

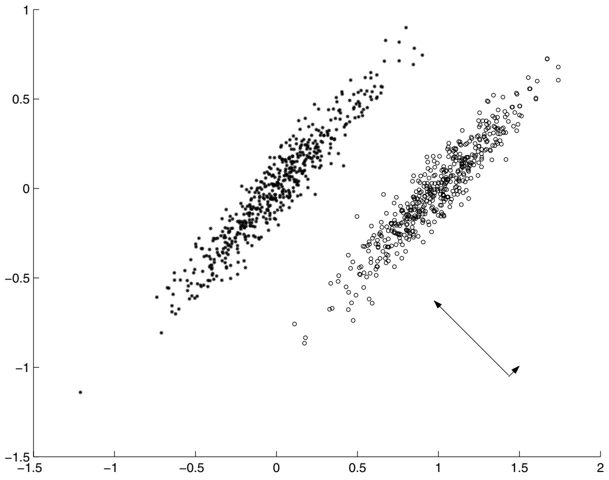


Fig. 2. VSM is unable to learn an appropriate distance measure for cases where the noise affecting different features is correlated. The arrows represent the kind of metric that GVSM will typically learn for this case.

2 Preliminaries

For the rest of the paper s_{ti} will denote the known probability (i.e. 1 or 0) that sample number $t \in 1, 2, \ldots, T$ falls in class $i \in 1, 2, \ldots, I$ and p_{ti} will denote the estimated probability that sample number t falls in class i based on the training set excluding sample t. Similarly s_{tki} will denote the known probability that the k-th nearest neighbour of sample number t from the training set falls in class i. $\overline{x_t}$ and $\overline{c_{tk}}$ will denote the feature vectors of the t-th sample and its k-th nearest neighbour respectively.

3 *K*-Nearest Neighbour Classification and VSM

The K-nearest neighbour technique uses the following expression to determine the probability that a sample belongs to class i

$$p_i = \frac{\sum_{k=1}^{K} n_k s_{ki}}{\sum_{k=1}^{K} n_k}.$$

In the most basic form of the method all n_k coefficients are set to 1 and it becomes a simple vote. A slightly more sophisticated method attaches more importance to closer neighbours by determining the weight n_{tk} assigned to each neighbour using a kernel centered at x_t. In this case we will use a Gaussian kernel,

$$n_k = e^{-\frac{d_k^2}{2\sigma^2}}.$$

The width of this kernel is determined by σ. If σ is too small, the truly nearest neighbour will dominate the decision and generalization will be poor. If it is too large, the method will fail to capture significant changes in the output. In general σ may be chosen smaller, the more densely the data is sampled. Since the density of data varies over the input space, fixed values of σ will not normally perform well. In order to make the width of the window vary with the density of available training samples, σ is set to some multiple of the average distance to the M nearest neighbours. It is better if only a fraction (e.g. $M = \frac{K}{2}$) of the nearest neighbours are used so the kernel becomes small even when only a few neighbours are close to the input. Sigma is therefore given by

$$\sigma = \frac{r}{M} \sum_{m=1}^{M} d_k$$

where r can be fixed or determined using an optimization routine. The difference between VSM and GVSM lies in a single equation. While VSM uses the expression

$$d_k^2 = \sum_{d=1}^{D} w_d^2 (x_d - c_{kd})^2$$

to define the distance between a sample x and its k-th nearest neighbour c_k, GVSM uses the more general matrix norm

$$d_k^2 = (\overline{x} - \overline{c_k})^T A (\overline{x} - \overline{c_k}) \tag{1}$$

where A is a positive definite symmetric matrix. For the case where A is a diagonal matrix, this is exactly equivalent to VSM.

4 GVSM Optimization

The first complication that arises when attempting to optimize a matrix norm is the fact that the matrix must be constrained to be symmetric positive definite. A necessary condition for a matrix A to be a symmetric positive definite matrix, is that it can be expressed as $A = L^T L$ where L is upper triangular with positive diagonal elements. A sufficient condition for A to be symmetric positive definite, is that it can be written as $A = L^T L$ where L can be any non-singular matrix. Therefore if and only if L is a non-singular upper triangular matrix will $A = L^T L$ be positive definite.

Expressing L as

$$L = \begin{bmatrix} L_{11} & L_{12} & L_{13} & \dots & L_{1d} \\ 0 & L_{22} & L_{23} & \dots & L_{2d} \\ 0 & 0 & L_{33} & \dots & L_{3d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & L_{dd} \end{bmatrix}$$

allows us to optimize the various elements L_{uv} assured in the knowledge that A will be a symmetric positive definite matrix and that all symmetric positive definite matrices can be obtained in this way. Although VSM learning uses conjugate-gradient descent to minimize the cross validation error over the training set we have only implemented a gradient descent method with a primitive form of line search for use with GVSM. However, convergence is still attained within a reasonable time for most problems.

The cross validation error is defined as

$$E = \sum_t \sum_i (s_{ti} - p_{ti})^2.$$

The derivative of this error can be computed with respect to each of the parameters to be optimized to obtain

$$\frac{\partial E}{\partial L_{uv}} = -2 \sum_t \sum_i (s_{ti} - p_{ti}) \frac{\partial p_{ti}}{\partial L_{uv}}$$

where

$$\frac{\partial p_{ti}}{\partial L_{uv}} = \frac{\sum_k (s_{tki} - p_{ti}) \partial n_{tk} / \partial L_{uv}}{\sum_k n_t k}$$

and

$$\frac{\partial n_{tk}}{\partial L_{uv}} = \frac{n_{tk}}{2\sigma^2} \left(\frac{d_{tk}^2 r}{M\sigma} \sum_{m=1}^{M} \frac{1}{d_{tm}} \frac{\partial d_{tm}^2}{\partial L_{uv}} - \frac{\partial d_{tk}^2}{\partial L_{uv}} \right), \tag{2}$$

$$\frac{\partial d_{tk}^2}{\partial L_{uv}} = \sum_o \sum_p (x_{to} - c_{tko})(x_{tp} - c_{tkp}) \frac{\partial A_{op}}{\partial L_{uv}},$$

$$\frac{\partial A_{op}}{\partial L_{uv}} = \begin{cases} 2L_{uv} & \text{if } o = p = v \\ L_{uo} & \text{if } o \neq p = v \\ L_{up} & \text{if } p \neq o = v \\ 0 & \text{if } p \neq o \neq v \end{cases}.$$

In order to optimize the parameter r we simply use the derivative of n_{tk} with respect to r, namely

$$\frac{\partial n_{tk}}{\partial r} = \frac{n_{tk} d_{tk}^2}{r\sigma^2}$$

instead of equation 3. For a d-dimensional input space GVSM optimizes $\frac{d(d+1)}{2} + 1$ parameters as opposed to the $d + 1$ parameters optimized by VSM, which implies more power to select an appropriate metric, but also more potential for overfitting the data set. Thus VSM would be more appropriate if the size of the data set is small relative to the dimensionality of the input. On the other hand GVSM optimizes very few parameters compared to an equivalent neural net, which seems to indicate that the overtraining problem should not be excessive.

5 Improving Run-Time Performance

Nearest neighbour methods are sometimes criticized for slow run-time performance. However, with the correct optimizations nearest neighbour methods can actually outperform other algorithms. For example we can use the distributive law to expand our expression for distance

$$d_k^2 = (\overline{x} - \overline{c_k})^T A (\overline{x} - \overline{c_k})$$

to

$$d_k^2 = \overline{x}^T A \overline{x} - 2\overline{x}^T A \overline{c_k} + \overline{c_k}^T A \overline{c_k}.$$

The last term in this expansion is a constant that can be calculated at design-time for each exemplar in the database. The first term is a constant that will be the same for all candidate neighbours and can thus be calculated once off and added to all candidates. However, we can save computation by only calculating the pseudo-distance

$$\tilde{d}_k^2 = -2\overline{x}^T A \overline{c_k} + \overline{c_k}^T A \overline{c_k},$$

then selecting the K nearest neighbours based on pseudo-distance and adding the $\overline{x}^T A \overline{x}$ term to each of these to obtain the true distances to the nearest neighbours. Since K is typically much smaller than the number of exemplars this saves a lot of addition. If we calculate the training vectors in the database as follows

$$\tilde{c}_k = \left[-2c_k^T A \; c_k^T A c_k \right],$$

and concatenate them all into a database matrix

$$C = \begin{bmatrix} \tilde{c}_1 \\ \tilde{c}_2 \\ \vdots \\ \tilde{c}_T \end{bmatrix},$$

we may calculate the pseudo distance between the query vector and each of the samples in the database by the simple operation of augmenting the query vector

$$\tilde{x} = \left[x_1 \; x_2 \ldots x_D \; 1 \right]^T,$$

and performing the matrix multiplication

$$\overline{\tilde{d}_j} = C\tilde{x}.$$

For problems with very large databases this matrix multiplication may take too long. In such cases the *k-d* tree algorithm [4,5] may be used to obtain further speedup. However, the performance gain of this algorithm diminishes with increasing dimensionality of the input. Large databases with high dimensionality may require the use of approximate methods such as Best Bin First [6]. Using one or more of these speedup techniques very often results in better run-time performance than competing methods.

6 Simulation Results

The GVSM algorithm was compared to VSM on the synthetic data set originally used by Lowe in [1] to demonstrate the working of VSM learning. The task is to solve a noisy XOR problem in which the first two real-valued inputs are randomly assigned values of 0 or 1 and the binary output class is determined by the XOR-function of these inputs. Noise was added to these 2 inputs drawn from a normal distribution with standard deviation of 0.3. The cross-correlation between the noise signals added to these two inputs α is the parameter against which we plot our results and varies from 0 to 0.95. The next two inputs were assigned the same initial 0 or 1 values as the first two, but with noise with standard deviation of 0.5 also correlated according to α. Finally another 4 inputs were added that had zero mean values with a standard deviation of 2.

The training set consisted of 100 samples and the test set of 1000 samples. The presence of irrelevant features as well as less-important features makes this a very difficult task for classical nearest-neighbour classifiers to solve. As can be seen from fig. 3 both VSM and GVSM fare very well with GVSM gaining a slight edge as the correlation increases.

An interesting result is that as the noise levels increase, GVSM becomes superior to VSM even for data in which the noise is not correlated. This result is shown in fig. 4 and was obtained by adding noise with a standard deviation of 0.55 and 0.7 respectively to the first two pairs of inputs.

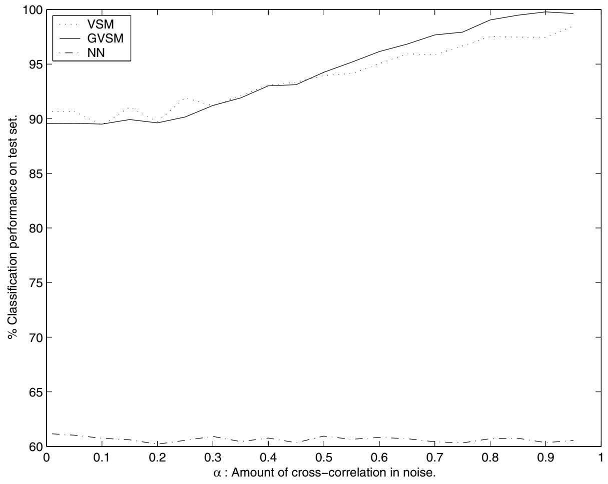


Fig. 3. VSM, GVSM and untrained nearest-neighbour generalization performance on the synthetic data set plotted against the amount of correlation in the noise. Standard deviations of 0.3 and 0.5 were used.

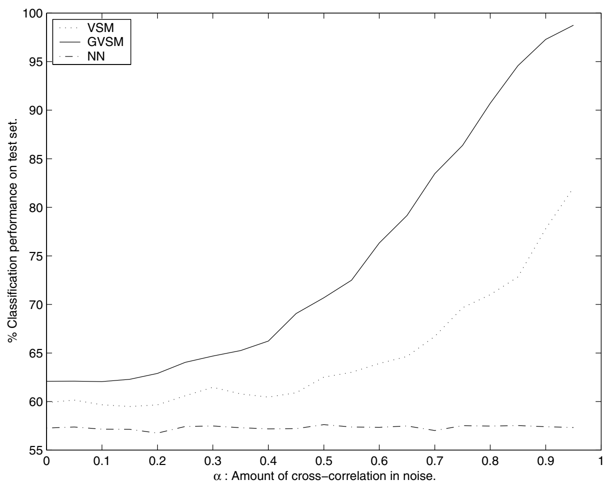


Fig. 4. VSM, GVSM and untrained nearest-neighbour generalization performance on the synthetic data set plotted against the amount of correlation in the noise. Standard deviations of 0.5 and 0.7 were used.

7 Conclusion and Future Directions

A generalized method for learning a metric suitable for use with proximity-based classifiers was presented. The superior performance of GVSM-learning on the synthetic XOR data set for increasing correlation seems to suggest that GVSM-learning can be a valuable tool to improve the generalization performance of proximity based classifiers. However, to justify the use of GVSM on any data set one must first verify that the conditions that GVSM was designed to exploit do indeed occur in the data.

While GVSM consistently obtains better training performance than VSM the drop from training to test performance is also much bigger. This indicates that the poor test performance is due to overtraining, but why the overtraining penalty on GVSM should be so much larger than that experienced by a neural net with the same amount of parameters is not clear. Another promising approach that is currently being investigated is the use of VSM or GVSM to learn different norms for the different classes in a problem.

References

1. Lowe, D. G., Similarity Metric Learning for a Variable-Kernel Classifier, Neural Computation, 7(1)(1995)72-85.
2. Cover, T.M., and Hart, P.E., Nearest neighbour pattern classifification., IEEE Transactions on Information Theory, 13(1)(1967)21-27.
3. Duda, R.O., and Hart, P.E., Pattern Classification and Scene Analysis., New York, Wiley, 1973.

4. Friedman, J.H., Bentley, J.L., and Finkel R.A., An algorithm for finding best matches in logarithmic expected time., ACM Trans. Math. Software, 3(1973)209-226.
5. Sproull, R.F., Refinements to nearest-neighbour searching in $k-d$ trees., Algorithmica, 6(1973)579-589.
6. Beis, J.S., Lowe, D.G., Shape indexing using approximate nearest neighbour in high-dimensional spaces., Conference on Computer Vision and Pattern Recognition, Puerto Rico, 1997, 1000-1006.

Approximate Gradient Direction Metric for Face Authentication

Josef Kittler and Mohammad T. Sadeghi

Centre for Vision, Speech and Signal Processing
School of Electronics and Physical Sciences
University of Surrey, Guildford GU2 7XH, UK
{J.Kittler,M.Sadeghi}@surrey.ac.uk
http://www.ee.surrey.ac.uk/CVSSP/

Abstract. In pattern recognition problems where the decision making is based on a measure of similarity, the choice of an appropriate distance metric significantly influences the performance and speed of the decision making process. We develop a novel metric which is an approximation of the successful Gradient Direction (GD) metric. The proposed metric is evaluated on a face authentication problem using the Banca database. It outperforms the standard benchmark, the normalised correlation. Although it is not as powerful as GD metric, it is ten times faster.

1 Introduction

In certain pattern recognition applications the training sets are notoriously small. A typical example is biometric person recognition where only a few training data points are available for each individual. An extreme case of the small sample set situation arises in image and video database retrieval, where only a single exemplar is available to define the class of objects of interest.

The usual approach to such problems is to base the decision making on some form of similarity measure, or scoring function, which relates unknown patterns to the query object template. If the degree of similarity exceeds a prespecified threshold, the unknown pattern is accepted to be the same as the query object. Otherwise it is rejected. The similarity concept can also be used in recognition scenarios where the unknown pattern would be associated with that class, the template of which is the most similar to the observed data.

The similarity score is computed in a suitable feature space. Commonly, similarity would be quantised in terms of a distance function, on the grounds that similar patterns will lie physically close to each other. Thus smaller the distance, the greater the similarity of two entities. The role of the feature space in similarity measurement is multifold. First of all the feature space is selected so as to maximise the discriminatory information content of the data projected into the feature space and to remove any redundancy. However, additional benefits sought after from mapping the original pattern data into a feature space is to simplify the similarity measure deployed for decision making. A classical example of this is the use of the Euclidean distance metric in Linear Discriminant Analysis

(LDA) feature spaces as the within class covariance matrix in the LDA space becomes an identity matrix and such metric becomes theoretically optimal.

Recently we have shown that in some applications, namely personal identity verification based on face biometrics, other scoring functions perform better, even in the LDA space, than the Euclidean distance. One of the examples is the normalised correlation, but the most promising scoring function appears to be the Gradient Direction metric [3]. This has been further generalised in [6]. The main problem with the Gradient Direction metric is its computational complexity.

In this paper we develop an approximation to the Gradient Direction metric which is defined as the difference between the mean (template) of the claimed identity and the local mean of other identities representing the anti-class (impostors). Although not as powerful as the Gradient Direction method, we show that this approximate Gradient Direction metric gives good performance, in comparison with normalised correlation and is significantly simpler to implement than the Gradient Direction metric method.

The paper is organised as follows. In the next section the Gradient Direction metric is reviewed and its approximation is introduced. The experimental set up adopted for the study is detailed in Section 3. The results of experiments are presented in Section 4. A discussion of the results as well as the main conclusions can be found in Section 5.

2 Computationally Efficient Gradient Direction Metric

In a face verification system, a matching scheme measures the similarity or distance of the test sample, $\mathbf{x}$ to the template of the claimed identity, $\boldsymbol{\mu}_i$. Note that $\mathbf{x}$ and $\boldsymbol{\mu}_i$ are the projection of the test sample and class mean into the feature space respectively. As the simplest solution, a matching score, s for the probe and the ith client mean can be defined as the Euclidean distance between the two vectors, i.e.

$$s_E = \sqrt{(\mathbf{x} - \boldsymbol{\mu}_i)^T (\mathbf{x} - \boldsymbol{\mu}_i)} \tag{1}$$

In [3], it has been demonstrated that a matching score based on Normalised Correlation (NC) is more efficient. The measure is defined as

$$s_N = \frac{||\mathbf{x}^T \boldsymbol{\mu}_i||}{\sqrt{\mathbf{x}^T \mathbf{x} \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i}} \tag{2}$$

Note that the NC is a similarity measure not a distance metric. The normalised correlation projects the probe vector onto the mean vector of the claimed client identity, emanating from the origin. It effectively uses just one dimensional space onto which the test data is projected. The magnitude of projection is normalised by the length of the mean and probe vectors. The normalised correlation tessellates the probe space into hyper cones or hyper frustums with the axes passing through the origin. It is apparent that the normalised correlation score will be insensitive to probe movements in the radial direction defined by the class mean.

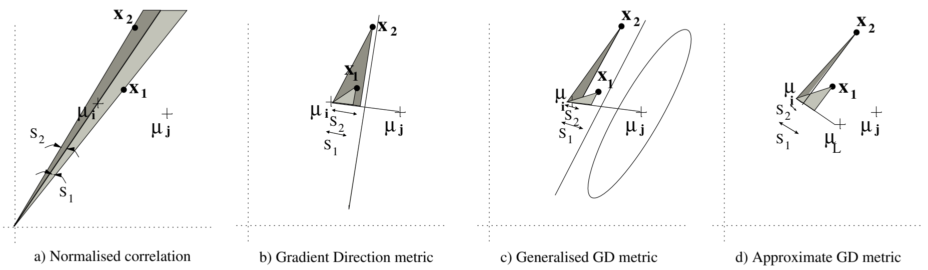


Fig. 1. Metrics for matching score definition.

However, the score will drop in value if the probe moves away from this direction angularly. A threshold on the normalised correlation then defines the acceptance region for each client.

Notwithstanding the improvements afforded by the normalised correlation, one can still voice some misgivings. The main drawback of the normalised correlation is that the axes of symmetry of the client acceptance cells are constrained to pass the origin. Inspecting Figure 1-a, this score will not produce the most effective separation of client i from potential imposter j.

In [3] an innovate metric called the *Gradient Direction metric* (GD) has been proposed. In this method the distance between a probe image $\mathbf{x}$ and the i-th client mean vector $\boldsymbol{\mu}_i$ is measured along the direction of the gradient of the i-th class aposteriori probability function $P(i|\mathbf{x})$. A mixture of Gaussian distributions with the identity covariance matrix has been assumed as the density function of the classes. Note that, this is the same direction along which one should measure distances when using the nearest neighbour decision rule [7]. However the motivation for projecting data on the gradient direction in the case of the nearest neighbour rule is completely different from the above argument for the nearest mean rule used here and we shall therefore not pursue this analogy any further. A conceptually similar direction has been adopted in the nearest feature line method proposed by Li *et al.* [4].

In [6], we revisited the theory of the Gradient Direction metric and proposed a Generalised Gradient Direction metric. We demonstrated that applying GD metric using either a general covariance matrix derived from the training data or an isotropic covariance matrix with a variance of the order of the variation of the image data in the feature space is even more efficient than the NC scoring function. The proposed optimal matching score has been defined as

$$s_O = \frac{||(\mathbf{x} - \boldsymbol{\mu}_i)^T \nabla_O P(i|\mathbf{x})||}{||\nabla_O P(i|\mathbf{x})||} \tag{3}$$

where $\nabla_O P(i|\mathbf{x})$ refers to the gradient direction. In the generalised form of the GD metric, the optimal direction would be [6]

$$\nabla_G P(i|\mathbf{x}) = \boldsymbol{\Sigma}^{-1} \sum_{\substack{j=1 \\ j \neq i}}^{m} p(\mathbf{x}|j)(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i) \tag{4}$$

where $p(\mathbf{x}|j)$ is the j-th client measurement distribution. Considering an isotropic structure for the covariance matrix, i.e. $\mathbf{\Sigma} = \sigma\mathbf{I}$, equation 4 can be simplified as:

$$\nabla_I P(i|\mathbf{x}) = \sum_{\substack{j=1 \\ j \neq i}}^{m} p(\mathbf{x}|j)(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i) \tag{5}$$

Note that as the length of the gradient vector will have to be normalised anyway the constant on the rhs of the equation can be ignored, however the magnitude of the σ will affect the direction through the values of $p(\mathbf{x}|j)$. Figure 1-b and 1-c illustrate the geometric differences between these scoring functions. The main drawback of the GD metric is that the method is computationally more expensive than NC function which is a consequence of dependency of the gradient direction on the test data.

Now suppose that the density function $p(\mathbf{x}|j)$ of each impostor was very flat (i.e. a large standard deviation). Then the weight factor in (5) defined by the density would effectively be constant and the gradient direction would lose its dependence on the observed probe image. In this case, the optimal direction of projection would be

$$\nabla P(i|\mathbf{x}) = \sum_{\substack{j=1 \\ j \neq i}}^{m} (\boldsymbol{\mu}_j - \boldsymbol{\mu}_i) \tag{6}$$

which can be expressed as

$$\nabla P(i|\mathbf{x}) = \boldsymbol{\mu} - \boldsymbol{\mu}_i \tag{7}$$

where $\boldsymbol{\mu}$ is the global mean. If the global mean is zero, then the gradient direction is simply defined by the i-th client mean. In the other extreme case, when the standard deviation is very small and the gradient vector is completely dominated by the nearest imposter, the direction of projection will be defined completely by the difference of the i-th client mean and the mean of the nearest imposter.

In practice, none of the above assumptions about the standard deviation is realistic. A reasonable compromise is to consider a small number of the closest neighbours to the claimed identity template to define the Approximate Gradient Direction (AGD) metric as

$$\nabla_A P(i|\mathbf{x}) \cong \sum_{\substack{j=1 \\ j \neq i \text{ and } j \in \mathcal{N}^L_{\boldsymbol{\mu}_i}}}^{L} (\boldsymbol{\mu}_j - \boldsymbol{\mu}_i) = \frac{1}{L} \sum_{j \in \mathcal{N}^L_{\boldsymbol{\mu}_i}} \boldsymbol{\mu}_j - \boldsymbol{\mu}_i = \boldsymbol{\mu}_L - \boldsymbol{\mu}_i \tag{8}$$

where $\mathcal{N}^L_{\boldsymbol{\mu}_i}$ refers to the set of L such neighbouring templates.

The main advantage of the above formulation is that the direction does not depend on the value of the test sample, $\mathbf{x}$. It will highly speed up the score evaluation process as the gradient direction for each client can be calculated in advance. In the next section the Normalised Correlation and Gradient Direction scores will be compared experimentally on the BANCA database [1].

3 Experimental Design

In this section the face verification experiments carried out on images of the BANCA database are described. The BANCA database is briefly introduced first. The main specifications of the experimental setup are then presented.

3.1 BANCA Database

The BANCA database has been designed in order to test multi-modal identity verification systems deploying different cameras in different scenarios (Controlled, Degraded and Adverse). The database has been recorded in several languages in different countries. Our experiments were performed on the English section of the database. Each section contains 52 subjects (26 males and 26 females). Experiments can be performed on each group separately.

Each subject participated to 12 recording sessions in different conditions and with different cameras. Sessions 1-4 contain data under *Controlled* conditions while sessions 5-8 and 9-12 contain *Degraded* and *Adverse* scenarios respectively. Figure 2 shows a few examples of the face data. Each session contains two recordings per subject, a true client access and an informed imposter attack. For the face image database, 5 frontal face images have been extracted from each video recording, which are supposed to be used as client images and 5 impostor ones. In order to create more independent experiments, images in each session have been divided into two groups of 26 subjects (13 males and 13 females). Thus, considering the subjects' gender, each session can be divided into 4 groups.

In the BANCA protocol, 7 different distinct experimental configurations have been specified, namely, Matched Controlled (MC), Matched Degraded (MD), Matched Adverse (MA), Unmatched Degraded (UD), Unmatched Adverse (UA), Pooled test (P) and Grand test (G). Table 1 describes the usage of the different sessions in each configuration. "T" refers to the client training while "C" and "I" depict client and impostor test sessions respectively. As we mentioned, 4 groups of data can be considered in each session. The decision function can be trained using only 5 client images per person from the same group and all client images from the other groups. More details about the database and experimental protocols can be found in [1].

3.2 Experimental Setup

The original resolution of the image data is 720×576. The experiments were performed with a relatively low resolution face images, namely 64×49. The results reported in this article have been obtained by applying a geometric face registration based on manually annotated eyes positions. Histogram equalisation was

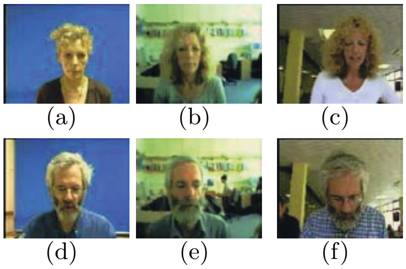


Fig. 2. Examples of the database images. *a,d:* Controlled, *b,e:* Degraded and *c,f:* Adverse scenarios.

Table 1. The usage of the different sessions in the BANCA experimental protocols.

	1	2	3	4	5	6	7	8	9	10	11	12
MC	TI	CI	CI	CI								
MD					TI	CI	CI	CI				
MA									TI	CI	CI	CI
UD	T				I	CI	CI	CI				
UA	T								I	CI	CI	CI
P	TI	CI	CI	CI	I	CI	CI	CI	I	CI	CI	CI
G	TI	CI	CI	CI	TI	CI	CI	CI	TI	CI	CI	CI

used to normalise the registered face photometrically. Linear Discriminant Analysis (LDA) is used for the feature extraction. The XM2VTS database [5] was used for calculating the LDA projection matrix. The thresholds in the decision making system have been determined based on the Equal Error Rate criterion, i.e. where the false rejection rate (FRR) is equal to the false acceptance rate (FAR). The thresholds are set either globally (*GT*) or using the client specific thresholding (*CST*) technique [2]. As we mentioned earlier, in the training sessions of the BANCA database 5 client images per person are available. In the case of global thresholding method, all these images are used for training of the clients template. The other group data is then used to set the threshold. While using the client specific thresholding strategy, only two images are used for the template training and the other three along with the other group data are used to determine the thresholds. Moreover, in order to increase the number of data used for training and to take the errors of the geometric normalisation into account, 24 additional face images per each image are generated by perturbing the location of the eyes position around the annotated positions.

4 Experimental Results and Discussion

The performance of different decision making methods based on the Normalised Correlation (NC), the Isotropic Gradient Direction (GD) and the Approximate Gradient Direction (AGD) metrics is experimentally evaluated on the BANCA database using the configurations discussed in the previous section.

Table 2 contains a summary of the results obtained on the test set using the NC and GD methods along with the global and client specific thresholding methods. The values in the table indicate the FAR, FRR and Total Error Rates (TER), i.e. the sum of false rejection and false acceptance rates. In the GD experiments, an isotropic structure $\mathbf{\Sigma} = \sigma\mathbf{I}$ with σ, of the order of the standard deviation of the input data (gray level values) was considered for the impostor distributions. Thus equation 5 is adopted for calculating the gradient directions.

A comparison of the results obtained using the Global and Client Specific thresholding methods indicates that the CST technique is superior in the

Table 2. ID verification results on BANCA protocols using Normalised Correlation and Gradient Direction methods with Global and Client Specific Thresholding techniques.

	Global Thresholding						Client Specific Thresholding					
	NC			GD			NC			GD		
	FAR	FRR	TER	FAR	FRR	TER	FAR	FRR	TER	FAR	FRR	TER
MC	5.48	8.08	13.56	2.40	5.90	8.30	2.98	5.77	8.75	1.25	3.97	**5.22**
MD	6.35	7.18	13.53	8.36	9.61	17.97	4.14	8.20	12.34	1.25	7.05	**8.30**
MA	8.94	9.36	18.30	7.40	7.82	15.22	5.96	10.00	15.96	1.35	6.53	**7.88**
UD	13.65	13.72	27.37	13.94	15.26	**29.2**	1.92	32.95	34.87	1.34	40.64	41.98
UA	20.19	21.92	42.12	16.06	16.15	**32.21**	1.82	42.44	44.26	0.96	47.95	48.91
P	14.01	14.23	28.24	11.57	10.64	**22.21**	2.18	31.45	33.63	0.99	32.48	33.47
G	4.84	4.61	9.45	2.18	3.33	5.51	8.20	3.33	11.54	2.02	1.58	**3.60**

matched scenario while the GT method gives a better performance on the unmatched protocols. The reason is that, as we mentioned earlier, images in each session have been divided into two groups of 26 subjects. In the GT method, the global threshold of each test group is calculated using the other group of test data. It means that the evaluation and test data have always the same image quality. In the case of the CST technique, we need to have available the client and impostor scores of each client individually. Thus we divided the training images into two subsets, client template training and client scores evaluation images. The impostor scores are then calculated using the other group of the test data (the group which the client does not belong to). It means that in the threshold evaluation of the unmatched experiments, images with different quality are used for calculating the client and impostors scores, while in the test stage all probe images (clients and impostors) have the same quality as the ones which are used for the impostor scores evaluation. Note that P protocol which is a collection of the MC, UD and UA protocols can mainly be considered as an unmatched protocol while the G protocol which involves data from different scenarios for both training and test can be seen as a matched protocol.

These results demonstrate clearly that, overall, the best performance is achieved using the Gradient Direction metric.

In the next step, the performance of the Approximate Gradient Direction metric was investigated. The neighbourhood size, L is the most important parameter in the AGD metric. Figure 3 presents the plots of the TER (Total Error Rate) versus the neighbourhood size for the evaluation and test data of different BANCA protocols. Based on the above argument about the thresholding techniques, these results were obtained with the CST method in the case of the matched protocols and GT method in the unmatched cases.

These plots show that by increasing the neighbourhood size the TER first rapidly decreases. Then, for larger values of L, the TER remains relatively constant or increases gradually. From these plots, one can also see that the behaviour of TER versus L in the evaluation and test stages is almost consistent. Therefore, the proper neighbourhood size can be found in the evaluation step by looking for the point after which the performance of the system is not significantly improved

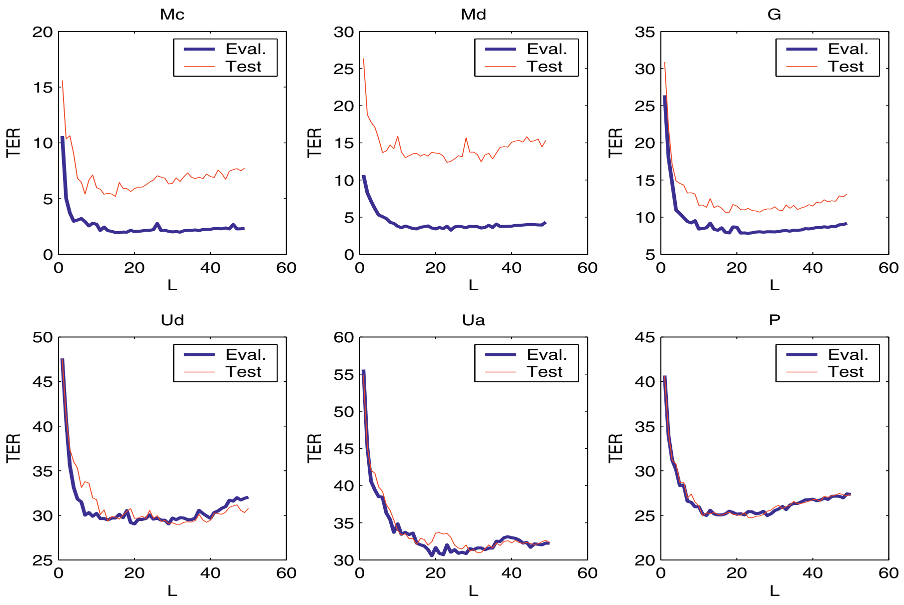


Fig. 3. The performance of the AGD metric versus the neighbourhood size for different BANCA protocols.

by increasing the neighbourhood size. Table 3 contains the results obtained using the proposed AGD method. For the sake of simplicity of comparison, the similar results using the isotropic GD method have also been reported in the table.

These results demonstrate that in the unmatched scenarios, the performance using the AGD method is comparable with the GD technique. However, in the matched scenarios the GD results are better. As mentioned earlier, the main advantage of the AGD method as compared to the GD method is its computational simplicity. The average verification time using the GD metric is around 0.04 CPU units while using the AGD metric it decreases to 0.006 CPU units.

Table 3. Verification results using the AGD and GD methods with global and client specific thresholding methods for unmatched and matched scenarios respectively.

	Approximate GD			GD		
	FAR	FRR	TER	FAR	FRR	TER
MC	2.212	3.205	5.417	1.25	3.97	5.22
MD	3.462	10.13	13.59	1.25	7.05	8.30
MA	4.904	9.103	14.01	1.35	6.53	7.88
UD	15.48	14.62	30.1	13.94	15.26	29.2
UA	16.35	15.77	32.12	16.06	16.15	32.21
P	12.5	12.61	25.1	11.57	10.64	22.211
G	6.923	4.274	11.2	2.02	1.58	3.60

A comparison of the results obtained using the NC and AGD methods also shows that overall the AGD method is superior, while the computational complexity of the methods (in the test mode) is similar.

5 Conclusions

A novel metric which is an approximation of the successful Gradient Direction metric was developed. The performance of face authentication systems in the LDA space using the proposed metric was experimentally compared with the original Gradient Direction (GD) and Normalised Correlation (NC) metrics. The results suggest that the Approximate GD metric outperforms the standard benchmark, the normalised correlation. Although it is not as powerful as GD metric, it is ten times faster.

Acknowledgements

The financial support from the EU project VAMPIRE is acknowledged.

References

1. E. Bailly-Baillière, S. Bengio, F. Bimbot, M. Hamouz, J. Kittler, J. Mariéthoz, J. Matas, K. Messer, V. Popovici, F. Porée, B. Ruiz, and J.-P. Thiran. The BANCA database and evaluation protocol. In *4th International Conference on Audio- and Video-Based Biometric Person Authentication, AVBPA*, pages 625–638. Springer-Verlag, 2003.
2. K. Jonsson, J. Kittler, Y. Li, and J. Matas. Support vector machines for face authentication. In *T. Pridmore and D. Elliman, editors, Proceedings of BMVC'99*, pages 543– 553, 1999.
3. J. Kittler, Y. P. Li, and J. Matas. On matching scores for LDA-based face verification. In M Mirmehdi and B Thomas, editors, *Proceedings of British Machine Vision Conference 2000*, pages 42–51, 2000.
4. S.Z. Li and J.W. Lu. Face recognition using the nearest feature line method. *IEEE Transactions on Neural Networks*, 10(2):439–443, 1999.
5. K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. XM2VTSDB: The extended M2VTS database. In *Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 72–77, March 1999.
6. M. Sadeghi and J. Kittler. Decision making in the LDA space: Generalised gradient direction metric. In *the 6th International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, May 2004.
7. R.D. Short and K. Fukunaga. The optimal distance measure for nearest neighbour classification. *IEEE Transactions on Information Theory*, 27(5):622–627, 1981.

The Imbalanced Training Sample Problem: Under or over Sampling?

Ricardo Barandela[1,2], Rosa M. Valdovinos[1],
J. Salvador Sánchez[3], and Francesc J. Ferri[4]

[1] Instituto Tecnológico de Toluca, Ave. Tecnológico s/n, 52140 Metepec, México
{rbarandela,li_rmvr}@hotmail.com
[2] Instituto de Geografía Tropical, La Habana, Cuba
[3] Dept. Llenguatges i Sistemes Informàtics, U. Jaume I, 12071 Castelló, Spain
sanchez@uji.es
[4] Dept. d'Informàtica, U. Valencia, 46100 Burjassot (Valencia), Spain
ferri@uv.es

Abstract. The problem of imbalanced training sets in supervised pattern recognition methods is receiving growing attention. Imbalanced training sample means that one class is represented by a large number of examples while the other is represented by only a few. It has been observed that this situation, which arises in several practical domains, may produce an important deterioration of the classification accuracy, in particular with patterns belonging to the less represented classes. In this paper we present a study concerning the relative merits of several re-sizing techniques for handling the imbalance issue. We assess also the convenience of combining some of these techniques.

1 Introduction

Design of supervised pattern recognition methods is usually based on a training sample (TS): a collection of examples previously analyzed by a human expert. There is a considerable amount of recent research on how to build "good" classifiers when the class distribution of the data in the TS is imbalanced. A TS is said to be imbalanced when one of the classes (the minority one) is heavily under-represented in comparison to the other (the majority) class. This issue is particularly important in those applications where it is costly to misclassify minority-class examples. For simplicity, and consistently with the common practice [8,13], only two-class problems are here considered. High imbalance occurs in real-world domains where the decision system is aimed to detect a rare but important case, such as fraudulent telephone calls [10], oil spills in satellite images of the sea surface [14], an infrequent disease [20], or text categorization [15].

Basic methods for reducing class imbalance in the TS can be sorted in 3 groups [12]:

a) Over-sampling (replicates examples in) the minority-class
b) Under-sampling (eliminates examples in) the majority class
c) Internally biasing the discrimination based process so as to compensate for the class imbalance [8,14]

As pointed out by many authors, overall accuracy is not the best criterion to assess the classifier's performance in imbalanced domains. For instance, in the thyroid data set used in [1], only 5% of the patterns belong to the minority class. In such a situation, labeling all new patterns as members of the majority class would give an accuracy of 95%. Obviously, this kind of system would be useless. Consequently, other criteria have been proposed. One of the most widely accepted criterion is the geometric mean, $g= (a^+ \cdot a^-)^{1/2}$, where a^+ is the accuracy on cases from the minority class and a^- is the accuracy on cases from the majority one [13]. This measure tries to maximize the accuracy on each of the two classes while keeping these accuracies balanced.

In previous studies [3-5], we have provided results of several techniques addressing the class imbalance problem. We have focused on under-sampling the majority class and also on internally biasing the discrimination process, as well as on combinations of both approaches. In the present paper, we present an experimental comparison of our results with those obtained with one method for over-sampling the minority class [6]. Our purpose is to illustrate the relative benefits of both basic techniques and to draw some conclusions about those situations in which one of them could be more useful than the other. We also present experimental results obtained with a combination of both resizing approaches. The experiments have been done with five real datasets using the Nearest Neighbor (NN) rule for classification and the geometric mean as the performance measure.

The NN rule is one of the oldest and better-known algorithms for performing supervised nonparametric classification. The entire TS is stored in the computer memory. To classify a new pattern, its distance to each one of the stored training patterns is computed. The new pattern is then assigned to the class represented by its nearest neighboring training pattern. Performance of NN rule, as with any nonparametric method, is extremely sensitive to incorrectness or imperfections in the TS. Nevertheless, the NN rule is very popular because of: a) conceptual simplicity, b) easy implementation, c) known error rate bounds, and d) potentiality to compete favorably in accuracy with other classification methods in real data applications.

2 Related Works

The two basic methods for resizing the TS cause the class distribution to become more balanced. Nevertheless, both strategies have shown important drawbacks. Under sampling may throw out potentially useful data, while over sampling increases the TS size and hence the time to train a classifier. In the last years, research has focused on improving these basic methods. Kubat and Matwin [13] proposed an under sampling technique that is aimed at removing those majority prototypes that are "redundant" or that "border" the minority instances. They assume that these bordering cases are noisy examples. However, they do not use any of the well-known techniques for cleaning the TS.

Chawla et al. [6] proposed a technique for over sampling the minority class and, instead of merely replicating prototypes of the minority class, they form new minority instances by interpolating between several minority examples that lie close together.

Pazzani et al. [16] take a slightly different approach when learning from an imbalanced TS by assigning different weights to prototypes of the different classes. On the other hand, Ezawa et al. [9] bias the classifier in favour of certain attribute relationships. Kubat et al. [14] use some counter-examples to bias the recognition process.

In an earlier study [3], we provided preliminary results of several techniques addressing the class imbalance problem. In that work, we focused on under sampling the majority class by using several editing and pruning techniques, conveniently adapted to the imbalance case. We proposed also a mechanism for internally biasing the discrimination-based process, and we evaluated the combination of this biasing mechanism with some under sampling methods. In [4], we have extended this idea with a modification of the Wilson's Editing [19] technique. This modification, that biases the editing procedure, allows a better and higher decrease in the number of prototypes of the majority class. We have also explored [5] the convenience of designing a multiple classification system for working in imbalanced situations. Instead of using a single classifier, an ensemble has been implemented. The idea is to train each one of the individual components of the ensemble with a balanced TS. In order to achieve this, as many training sub-samples as required to get balanced subsets are generated. The number of sub-samples is determined by the difference between the amount of prototypes from the majority class and that of the minority class.

3 Techniques to Be Evaluated

The main purpose of the present paper is to experimentally compare several techniques for handling the imbalance situation. Some of these techniques, corresponding to the under sampling and biasing approaches, have already shown important increases in the g value obtained in classification tasks. The experiments to be reported below, include now an over sampling method. All these techniques are explained hereafter.

3.1 Under Sampling Approach

As already explained in Section 2, we have experimented with several methods [3] aimed at reducing the size of the majority class. Out of concern for the possibility of eliminating useful information, we have employed well-known editing algorithms, in particular the already classical Wilson's proposal [19]. One of the contributions of [3] has been the application of this editing technique only to the majority class.

Wilson's Editing. Wilson's Editing corresponds to the first proposal to edit the NN rule. In a few words, it consists of applying the k-NN classifier to estimate the class label of all prototypes in the TS and discard those samples whose class label does not agree with the class associated with the largest number of the k neighbors.

Weighted Editing. Despite the important obtained results, it was observed in [3] that the editing technique did not produce significant reductions in the size of the majority class. Accordingly, the imbalance in the training sample is not diminished in an important way.

It is worthy to remember that Wilson's technique consists essentially in a sort of classification system. The corresponding procedure works by applying the k-NN classifier to estimate the class label of all prototypes in the TS, as explained above. Of course, this *k*-NN classifier is also affected by the imbalance issue. When applied to prototypes of the majority class, the imbalance in the TS will cause a tendency to find most of their *k* nearest neighbors into that majority class. Consequently, only a few of the majority class prototypes will be removed. This means that the majority class is not completely cleaned of atypical cases and also that the balance in the TS is far from being reached.

To cope with this difficulty, in [4] we introduced the employment of the weighted distance below mentioned, not only in the classification phase but also in editing the majority class in the TS. That is, we apply the Editing algorithm, but using the weighted distance instead of the Euclidean metric. In that way, the already explained tendency has been overturned.

A Pruning Technique: The Modified Selective Subset. The NN rule generalizes accurately for many real applications. However, since it must store all the available training patterns and search through all of them to identify a new pattern, it has large memory requirements and works slowly in the classification phase. Many proposals have been done to reduce the TS size, while trying to maintain accuracy rate in the classification phase of the NN rule. Hart's [11] idea of a *consistent* subset has become a milestone in this research line. But his algorithm to obtain this consistent subset suffers for several well-known drawbacks. That has stimulated a sequel of new algorithms attempting to remedy these faults. Particularly remarkable is the approach of Ritter et al. [17] with a clear and precise formulation of the desired goals and of the way to reach them (the Selective Subset).

According to Hart's statement, the Condensed Subset (CS) is a subset *S* of the TS such that every member of TS is closer to a member of *S* of the same class than to a member of *S* of a different class. Ritter et al. have changed this concept in their Selective Subset (SS) by defining it as that subset *S* such that every member of TS must be closer to a member of *S* of the same class than to a member of TS (instead of *S*) of a different class. Their purpose is to eliminate the order-dependence of the building algorithm. Instead of using a greedy algorithm, Ritter et al. use a kind of branch and bound algorithm that implicitly considers every solution. In fact, they define the SS as the smallest subset containing at least a *related* prototype for each of the original ones. In this context, related means that it is able to correctly classify the corresponding prototype. As Ritter et al. have recognized, their algorithm does not necessarily conduct to a unique solution. Moreover, although they stated the importance of selecting "samples near the decision boundaries", this requisite is not included in the criteria serving as a basis for their SS.

As obtaining a more accurate decision boundary is more important that achieving true minimality, the SS procedure has been modified in two main ways. First, the minimality criterion has been partially substituted by an explicit boundary proximity criterion. And second, the procedure has been converted into a greedy algorithm that ends scanning the TS only twice. This Modified Selective Subset (MSS [2]) turns out to be much simpler and usually obtains subsets with improved quality boundaries and with slightly larger sizes than the corresponding SS solutions.

In [3], we have discussed the usefulness of the MSS technique for handling the imbalanced situation. Here, this pruning algorithm in included only for reducing the TS size after it has been considerable increased by the over sampling method.

3.2 Biasing Mechanism

For internally biasing the discrimination procedure, we proposed in [3] a weighted distance function to be used in the classification phase. Let $d_E(\cdot)$ be the Euclidean metric, and let Y be a new pattern to be classified. Let x_0 be a training prototype from class i, let N_i be the number of prototypes from class i, let N be the TS size, and let m be the dimensionality of the feature space. Then, the weighted distance measure is defined as:

$$d_W(Y,x_0) = (N_i/N)^{1/m} \cdot d_E(Y,x_0) \qquad (1)$$

The basic idea behind this weighted distance is to compensate for the imbalance in the TS without actually altering the class distribution. Thus, weights are assigned, unlike in the usual weighted k-NN rule [7], to the respective classes and not to the individual prototypes. In such a way, since the weighting factor is greater for the majority class than for the minority one, the distance to positive minority class prototypes becomes much lower than the distance to prototypes of the majority class. This produces a tendency for the new patterns to find their nearest neighbor among the prototypes of the minority class.

3.3 Over Sampling Approach

Most of the proposed techniques for increasing the size of the minority class merely replicate some of the minority class prototypes. Inclusion of exact copies of some minority class examples means to raise the requirement in computational resources. Moreover, with this procedure, overfitting is likely to occur, particularly in some learning models like the decision trees [17]. To avoid the overfitting problem, Chawla et al. [6] form new minority class prototypes by interpolating between minority class prototypes that lie close together. The technique these authors proposed, takes each minority class prototype and introduces "synthetic" prototypes along the line joining any/all of the minority class nearest neighbors. Depending upon the amount of over sampling required, neighbors from the k nearest neighbors are randomly chosen. In the experiments they reported, k is set to five. When, for instance, the amount of over sampling needed is 200%, only two neighbors from the five nearest neighbors are chosen and one prototype is generated in the direction of each of these two neighbors. Synthetic prototypes are generated in the following way: take the difference between the feature vector (prototype) under consideration and its nearest neighbor. Multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration.

4 Experimental Results

All these techniques, as well as combinations of some of them, were assessed with experiments that were carried out with five datasets. Four of these datasets have been taken from the UCI Database Repository (http://www.ics.uci.edu/~mlearn/). The Mammography dataset was kindly provided by N. V. Chawla and it was reported in

[6] and in [20]. Five-fold cross validation was employed to obtain averaged results of the *g* criterion. To facilitate comparison with other published results, in the Glass dataset the problem was transformed for discriminate class 7 against all the other classes and in the Vehicle dataset the task was to classify class 1 against all the others. Satimage dataset was also mapped to configure a two-class problem: the training patterns of classes 1, 2, 3, 5 and 6 were joined to form a unique class and the original class 4 was left as the minority one. Phoneme and Mammography are two-class datasets.

Table 1. Mean values of the geometric mean.

Training sets	Phoneme	Satimage	Glass	Vehicle	Mammography
Original TS Euclidean Classif.	73.8	70.9	86.7	55.8	60.2
Original TS Weighted. Classif.	76.0	75.9	88.2	59.6	75.8
Under-sampling majority class					
Euclidean Editing & Classif.	74.9	73.0	86.2	64.0	63.9
Euclid. Edit.+Weighted Classif.	75.7	76.2	87.9	65.8	76.2
Weighted+Edit.+Euclid. Classif.	75.0	74.5	86.2	65.6	70.0
Weighted Editing & Classif.	75.3	77.8	87.9	67.2	78.7
Over-sampling minority class and processing both classes					
Synthetic prototypes	73.6	77.1	88.7	59.7	83.4
Synthetic &Wilson's Editing	74.9	78.5	86.4	64.5	86.8
Synthetic & Modif. Select. Subset	70.3	74.1	88.2	57.1	80.8
Synthetic & Wilson & MSS	74.8	76.2	85.9	62.7	86.0

The obtained experimental results are shown in Table 1. This table has three parts. In the first one, the results when employing the original TS, both with Euclidean and Weighted distance, are included for comparison purposes. In the second part, we present the geometric mean values observed when the TS was under sampled through Wilson's Editing and Weighted Editing. Here also, the classification was done twice with each edited TS, using the Euclidean and the Weighted distances. In the third part of the table, results of the over sampling technique are incorporated. In this case, no weighted distance for classification has been employed since balance in the TSs has been attained by the over sampling technique.

Fom the figures in Table 1, it is evident that the over sampling approach can not compete, in most of the datasets, with the combination of the Weighted Editing (for under sampling) and the Weighted classification (the biasing mechanism). The difference in the Glass dataset (88.7 vs. 87.9) was not statistically significant. The only exception is the Mammography dataset, where results obtained after over sampling excelled to those of all the other evaluated techniques.

The explanation for these, somehow contradictory, results is to be found in the amount of imbalance present in each dataset (see Table 2). When the imbalance in the TS is not very big (say, a majority/minority ratio less than 10), then the under sampling techniques, particularly the Weighted Editing, can be useful in reducing enough the imbalance as to produce an important enhancement in the performance of the classifier. However, when this ratio is greater, the degree of balance achieved is not satisfactory. With the employed under sampling techniques, we are very careful in not throwing away potentially useful information. Accordingly, not many majority class prototypes are removed. With greater ratios, it is much better to employ the over sampling technique, even at the cost of a considerable increase in the total TS size.

Table 2. Imbalance present in each training dataset (majority/minority ratio).

Training sets	Phoneme	Satimage	Glass	Vehicle	Mammography
Original TS	2.41	9.29	6.25	2.99	44.12
After under-sampling majority class					
Euclidean Editing	2.27	8.94	6.13	2.44	43.99
Weighted Editing	2.15	8.64	6.02	2.31	43.03
Over-sampling minority class and processing both classes					
Synthetic prototypes	1.20	1.03	1.04	1.49	1.01
Synthetic &Wilson's Editing	1.18	0.94	1.03	1.31	1.04
Synthetic & Modif. Select. Subset	1.01	1.49	1.42	1.42	2.12
Synthetic & Wilson & MSS	0.93	1.35	0.94	1.24	1.00

Table 3. Size of the TSs (Original and after application of the under and over sampling).

Training sets	Phoneme	Satimage	Glass	Vehicle	Mammography
Original TS	4322	5147	174	678	10062
After under-sampling majority class					
Euclidean Editing	4150.8	4971.6	171.2	584.8	10032.9
Weighted Editing	3997.8	4820.6	168.6	562.0	9818.5
Over-sampling minority class and processing both classes					
Synthetic prototypes	5590	9147	294	848	19572
Synthetic &Wilson's Editing	5185.2	8706.0	285.0	686.8	17725.3
Synthetic & Modif. Select. Subset	1201.2	1322.8	27.6	321.2	6931.9
Synthetic & Wilson & MSS	756.0	906.4	18.6	176.2	1599.5

This concern for the huge increase in the TS size produced by over sampling (almost twice the number of original prototypes), has been the motivation for exploring the convenience of applying preprocessing techniques after the formation of new minority class prototypes (see Table 3). As usual, the combined employment of Wilson's Editing and the pruning technique, MSS, has yielded a considerable decrease in the TS size and, in general, a classification performance better than before their application. Thus, another recommendation: in those cases where over sampling the TS is a must, it is convenient, afterwards, to try to clean the TS and to reduce its size.

5 Concluding Remarks

In many real-world applications, supervised pattern recognition methods have to cope with highly imbalanced TSs. Traditional learning systems such as the NN rule can be misled when applied to such practical problems. This effect can become softer by using procedures to resize (under sampling or over sampling) the TS. In the present paper we have assessed the relative merits of these two approaches for re-sampling the TS. Our results indicate that, when the imbalance is not very severe, techniques for appropriately under sampling the majority class are the best option. Only when the majority/minority ratio is very high it is required to over sampling the minority class. Convenience of using combinations of some techniques is also established. In particular, this combination is remarkable in those cases where over sampling is unavoidable. In these situations, cleaning of the TS and reduction of its size, after the over sampling is done, allows for a considerable decrease in the computational burden of the NN rule and for an increase in the classification performance of the system.

The present report is part of a more extensive research we are conducting to explore all the issues linked to the imbalanced TSs. At present, we are studying the convenience of applying genetic algorithms to reach a better balance among classes. We are also experimenting in situations with more than two classes, as well as doing some research about the convenience of using these procedures to obtain a better performance with other classifiers, such as the neural networks models.

Acknowledgements

This work has been partially supported by grants 32016-A from the Mexican CONACYT, 644.03-P from the Mexican Cosnet, and TIC2003-8496-C04 from the Spanish CICYT.

References

1. Aha, D., Kibler, D.: Learning Representative Exemplars of Concepts: An Initial Case Study, Proceedings of the Fourth International Conference on Machine Learning (1987) 24-30.
2. Barandela, R., Cortés, N., Palacios, A.: The Nearest Neighbor rule and the reduction of the training sample size, *Proc. 9th Spanish Symposium on Pattern Recognition and Image Analysis* **1** (2001) 103-108.
3. Barandela, R., Sánchez, J.S., García, V., Rangel, E.: Strategies for learning in class imbalance problems, *Pattern Recognition* **36(3)** (2003) 849-851.
4. Barandela, R., Sánchez, J.S., García, V., Ferri, F.J.: Learning from Imbalanced sets through resampling and weighting, *Lecture Notes in Computer Science* **2652** (2003) 80-88.
5. Barandela, R., Valdovinos, R. M., Sánchez, J.S.: New applications of ensembles of classifiers, *Pattern Analysis and Applications* **6(3)** (2003) 245-256.
6. Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P.: SMOTE: synthetic minority over-sampling technique, *Journal of Artificial Intelligence Research* **16** (2000) 321-357.
7. Dudani, S.A.: The distance-weighted *k*-nearest neighbor rule, *IEEE Trans. on Systems, Man and Cybernetics* **6** (1976) 325-327.
8. Eavis, T., Japkowicz, N.: A Recognition-based Alternative to Discrimination-based Multi-Layer Perceptrons, Workshop on Learning from Imbalanced Data Sets. *Technical Report WS-00-05,* AAAI Press (2000).
9. Ezawa, K.J., Singh, M., Norton, S.W.: Learning goal oriented Bayesian networks for telecommunications management, In: *Proc. 13th Int. Conf. on Machine Learning* (1996) 139-147.
10. Fawcett, T., Provost, F.: Adaptive fraud detection, *Data Mining and Knowledge Discovery* **1** (1996) 291-316.
11. Hart, PE.: The Condensed Nearest Neighbor rule. *IEEE Trans. on Information Theory* **6(4)** (1968) 515-516.
12. Japkowicz, N., Stephen, S.: The class imbalance problem: A systematic study, *Intelligent Data Analysis Journal* **6(5)** (2002) 429-450.

13. Kubat, M., Matwin, S.: Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. *Proceedings of the 14th International Conference on Machine Learning,* (1997) 179-186.
14. Kubat, M., Holte, R., Matwin, S.: Detection of Oil-Spills in Radar Images of Sea Surface. *Machine Learning,* **30** (1998) 195-215.
15. Mladenic, D., Grobelnik, M.: Feature selection for unbalanced class distribution and naïve Bayes, In: *Proc. 16th Int. Conf. on Machine Learning* (1999) 258-267.
16. Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., Brunk, C.: Reducing misclassification costs, In: *Proc 11th Int. Conf. on Machine Learning* (1994) 217-225.
17. Ritter, GI., Woodruff, HB., Lowry, SR., Isenhour, TL: An Algorithm for Selective Nearest Neighbor Decision Rule. *IEEE Trans. on Information Theory* **21(6) (1975)** 665-669.
18. Weiss, GM., Provost, F.: Learning when training data are costly: The effect of class distribution on tree induction, Journal of Artificial Intelligence Research **19** (2003) 315-354.
19. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited data sets, *IEEE Trans. on Systems, Man and Cybernetics* **2** (1972) 408-421.
20. Woods, K., Doss, C., Bowyer, K.W., Solka, J., Priebe, C., Kegelmeyer, W.P.: Comparative evaluation of pattern recognition techniques for detection of micro-calcifications in mammography, *International Journal of Pattern Recognition and Artificial Intelligence* **7** (1993) 1417-1436.

Automatic Preference Mining through Learning User Profile with Extracted Information

Kyung-Yong Jung[1], Kee-Wook Rim[2], and Jung-Hyun Lee[3]

[1] HCI Lab., Dept. of Computer Science & Information Engineering, Inha Univ., Korea
kyjung@gcgc.ac.kr
[2] Dept. of Knowledge Information & Industrial Engineering, Sunmoon Univ., Korea
rim@omega.sunmoon.ac.kr
[3] Dept. of Computer Science & Information Engineering, Inha Univ., Korea
jhlee@inha.ac.kr

Abstract. Previous Bayesian classification has a problem because of reflecting semantic relation accurately in expressing characteristic of web pages. To resolve this problem, this paper proposes automatic preference mining through learning user profile with extracted information. Apriori algorithm extracts characteristic of web pages in form of association words that reflects semantic relation and it mines association words from learning the ontological user profile. Our prototype personalized movie recommender system, WebBot, extracts information about movies from web pages to recommend titles based on training movie set supplied by an individual user. The proposed method was tested in database that users estimated the preference about web pages, and certified that was more efficient than existent methods.

1 Introduction

Recommender systems using information filtering accumulates a database of users preferences, and then uses them to make personalized recommendations for items such as books, music, clothing, and movies. The user's preference can be either explicit ratings or implicit usage history. Information filtering can help E-commerce in converting web surfers into buyers by personalization of the web interface. It can also improve cross-sell by suggesting other items the user might be interested. In a real world where an E-commerce recommender system's competitors are only a one click, gaining users loyalty is an essential business strategy. Recommender system using information filtering has been very successful in both practice and research. However, there still remain important research issues in overcoming two fundamental challenges for information filtering [5]. The first challenge is to improve the scalability of the information filtering algorithms. Existing information filtering algorithms can deal with thousands of users within a reasonable time, but the demand of modern E-Commerce system is to handle tens of millions of users. The second challenge is to improve the quality of the recommendations. Users need recommendations they can trust to help them find items they will like. If a user trusts a recommender system, purchases an item, but finds out he does not like the item, the user will be unlikely to use the recommender system again [18]. In this paper, we present automatic preference mining through learning the ontological user profile with extracted information to improve the scalability and the quality of the personalized movie recommender system [7]. Users provide actual 1-6 ratings for a selected training movies set; the system then learns an ontological user profile using Naïve Bayesian algorithm and produces a ranked list of the most recommended additional titles [6,10].

2 Extracted Information Based on Association Word Knowledge

We have been exploring the personalized movie recommender system by applying automated text categorization methods to semi-structured text extracted from the web pages [14]. Our current prototype system, WebBot: Web Robot Agent [6,7], uses a database of movie content information extracted from web pages at Internet Movie Database. Therefore, the personalized movie recommender system's content information about titles consists of textual meta-data rather than the actual text of the web pages. An IMDb subject search is performed to obtain a list of movie-description URLs of broadly relevant titles. WebBot then downloads each of these pages and uses a pattern-based information extraction system to extract the data about each title. Information extraction is the task of locating specific information from web pages, thereby getting useful structured related data from unstructured text. Specifically, it involves finding a set of sub-strings from the web pages, for each of a set of slots.

2.1 Building a Database with Extracting Information

A WebBot follows the IMDb link provided for every movie in the EachMovie dataset [9] and collects information from the various links off the main URL. We represent the content information of all movies as a set of slots. Each slot is represented as a bag of association words. This content information, after suitable preprocessing such as elimination of stop word etc., is collected into a vector of bag of association words, one bag for each feature describing the movie. IMDb produces the information about related directors and movie titles using information filtering: however, WebBot treats them as additional content information about the movie. Since the layout of IMDb's automatically generated web pages is so regular. A moderately simple extraction system is sufficient. The text in each slot is then processed into an unordered bag of words and all movies represented as a vector of bags of association words [6].

2.2 Expression of Web Pages Characteristics

To express the characteristics of web pages as either a bag-of-words or a bag-of-association-words [1], it is a necessary preprocess the web pages by analyzing its morphology. The system used in the morphological analysis is identical to the focused intelligent information retrieval system [17]. The Apriori algorithm [1] is used to mine related data from the words extracted from morphological analysis. The associated word mining algorithm, Apriori is used to find the associative rules of items out of the set of transactions. The mimed data, or the set of associated words from each web page, are represented as a related-word vector model. As a result, the web page $\{d_j\}$ is represented as Equation (1) in the form of a related-word vector model.

$$\{d_j\}=\{(w_{11}\&w_{12}\ldots\&w_{1(r-1)}\rightarrow w_{1r}),(w_{21}\&w_{22}\ldots\&w_{2(r-1)}\rightarrow w_{2r}),\ldots,\\ (w_{k1}\&w_{k2}\ldots\&w_{k(r-1)}\rightarrow w_{kr}),\ldots,(w_{p1}\&w_{p2}\ldots\&w_{p(r-1)}=>w_{pr})\}_j \quad (j=1,2,\ldots,m) \tag{1}$$

$(w_{k1}\&w_{k2}\ldots\&w_{k(r-1)})$ is antecedent of association word $(w_{k1}\&w_{k2}\ldots\&w_{k(r-1)}\rightarrow w_{kr})$ and w_{kr} is consequent of association word $(w_{k1}\&w_{k2}\ldots\&w_{k(r-1)}\rightarrow w_{kr})$. Here, each of $\{w_{k1},w_{k2},\ldots,w_{k(r-1)},w_{kr}\}$ in $(w_{k1}\&w_{k2}\ldots\&w_{k(r-1)}\rightarrow w_{kr})$ represents a word for composing association word. "*p*" represents the number of association words in a web page. "*r*" represents the number of words in an association word. The "&" shows that the words on each side are related. For the good results in extracting related words, the dataset must have a confidence of over 85 and a support of less than 25 [6].

2.3 Generating the Ontological User Profile

The association word mining is used to extract features from web pages. Most users have a habit of searching similar or same web pages. We generate an ontological user profile by extracting features from web pages user accesses continuously. Content-based filtering generates new user profiles by receiving relevance feedback on web pages that users access after receiving recommendations to the web pages. Related words gathered by relevance feedback on web pages the user has visited are stored in the user profile. The related words stored in the user profile, related words show a high frequency is given more weight. The weight means the ratio of each association word to all association words, which are extracted form web pages that user accesses. If wUP_a is a user profile with weight given to related words, then wUP_a can be expressed as Equation (2).

$$wUP_a = \{w_1 \cdot AW_1, w_2 \cdot AW_1, \ldots, w_t \cdot AW_t\} \tag{2}$$

In Equation (2), $\{w_1, w_2,\ldots, w_t\}$ is a weight vector that shows the weight of the related word, and t is the total number of related words within the ontological user profile. The ontological user profile is generated based on the web page features extracted from user's preference rating. The preference of association words expressed in features is indicated various values according to the weight.

3 Automatic Preferences Mining through Learning User Profile

Previous studies of automatic learning of profile include use of probability [4,8], use of statistics [13] and use of vector similarity [7,8]. Among them, learning user profile through Bayesian probability is effective method [10,15]. Since learning the ontological user profile through use of simple Naïve Bayesian classifier extracts all word appeared in web pages, it is hard to reflect characteristic of web pages accurately. Mistaken learning user profile caused by this reduces accuracy of classification. Because of this, Bayesian classification method [11] that uses TF·IDF to make it more accurate was suggested. The suggested method extracts characteristic of web pages through use of TF·IDF from web pages [10]. It also gives weight to characteristic extracted from web pages and so mistaken classification caused by noises is reduced more than simple Naïve Bayesian classifier. However, since characteristic of extracted web pages does not reflect semantic relation, it could not resolve the problem of mistaken classification caused by ambiguity of words [6]. To resolve this problem, this paper suggests automatic preference mining through learning user profile with extracted information. In suggested method, Apriori algorithm [1] extracts characteristic of web pages in the form of association words that reflects semantic relation between words.

3.1 Learning a User Profile Using Naïve Bayesian Classifier

The learner currently employed by WebBot [6,7] is a bag of association words Naïve Bayesian classifier [10] extended to handle a vector of bags rather than a single bag. WebBot does not attempt to predict the extract numerical rating of a movie title, but rather just a total ordering or ranking of titles in order of preference [14,15,16].

We use a multinomial text model, in which a web page is modeled as an ordered sequence of ordered sequence of word events drawn from the same vocabulary, V. The Naïve Bayesian assumption states that the probability of each word is dependent on the web page classes but independent of the word's context and position. For each class c_j, and word, w_k□V, the probability, $P(c_j)$ and $P(w_k|c_j)$ must be estimated from the training data. Then the posterior probability of each class given a web page, D, is computed using Bayes rule by Equation (3).

$$p(c_j \mid D) = \frac{P(c_j)}{P(D)} \prod_{i=1}^{|D|} P(a_i \mid c_j) \tag{3}$$

Where a_i is the ith word in the web page, and $|D|$ is the length of the page in words. Since for any given page, the prior $P(D)$ is a constant, this factor can be ignored if all that is desired is a rating rather than a probability estimate. A ranking is produced by sorting pages by their odds ratio, $P(c_1|D)/P(c_0|D)$, where c_1 represents the positive class and c_0 represents the negative class. A movie is classified as positive if the odds are greater than 1, and negative otherwise. In case, since movies are represented as a vector of "web pages", d_m, one for each slot s_m, the probability of each word given the category and the slot $P(w_k|c_j, s_m)$, must be estimated and the posterior category probability for a film, F, computed using Equation (4).

$$p(c_j \mid F) = \frac{P(c_j)}{P(F)} \prod_{m=1}^{|S|} \prod_{i=1}^{|d_m|} W_{nk} \cdot P(a_{m,i} \mid c_j,\ s_m) \tag{4}$$

Where S is the number of slots and $a_{m,i}$ is the ith word in the mth slot. The class with the highest probability determines the predicted rating. The *Laplace* smoothing [10] is used to avoid zero probability estimates for words that do not appear in the limited training movie set. Finally, calculation with logarithms of probabilities is used to avoid underflow.

Naïve Bayesian classifier through use of TF·IDF [13] makes morphological analysis of document to extract characteristic of web pages and extracts only nouns from its outcome. TF·IDF of all extracted nouns can be obtained through Equation (5).

$$W_{nk} = f_{nk} \cdot [Log_2 \frac{n}{DF} + 1] \tag{5}$$

In Equation (5), f_{nk} is relative frequency of word n_k against all words within all web pages and n is the number of web pages and DF is the number of learning web pages where word n_k appeared. If characteristic of web pages is $\{a_{1,}a_{2,...,}a_{i,...,}a_D\}$, Naïve Bayesian classifier classifies web pages into one class among $\{c_{1,}c_{2,...,}c_{i,...,}c_j\}$. Equation (6) is used to give probability of association word $(w_{\mathrm{k1}}\&w_{\mathrm{k2}}...\&w_{\mathrm{k(r-1)}} \rightarrow w_{\mathrm{kr}})$ within $c_{\mathrm{j}},s_{\mathrm{m}}$ is expressed as $p((w_{\mathrm{k1}}\&w_{\mathrm{k2}}...\&w_{\mathrm{k(r-1)}}) \rightarrow w_{\mathrm{kr}})|c_{\mathrm{j}},s_{\mathrm{m}})$. If an association word appears n times in a movie F_e, it is counted as occurring $a_{e1}n$ times in a positive movie and $a_{e0}n$ times in a negative movie. The parameters are estimated as Equation (6).

$$p((\mathrm{w}_{\mathrm{k1}} \,\&\, \mathrm{w}_{\mathrm{k2}} \ldots \&\, \mathrm{w}_{\mathrm{k(r-1)}} \rightarrow w_{k(r-1)}) \,|\, c_j, s_m) = \sum_{e=1}^{N} W_{nk} \cdot a_{ej} \cdot n_{kem} /(\sum\nolimits_{e=1}^{N} a_{ej} \,|\, d_m \,|) \tag{6}$$

Where n_{kem} is the count of the number of times word w_k, appears in movie F_e, in slot s_{m}, and denotes the total weighted length of web pages in category c_j and slot s_{m}.

3.2 Automatic Preferences Mining for Recommendations

To produce recommendations, WebBot [6,7] learns the ontological user profile, predicts the ratings for the un-rated movies, and finally ranks the movies by their ratings. Current WebBot uses one of the three methods to learn an ontological user profile. The first method is a binary Naïve Bayesian classifier. It treats movies rated 1-3 as negative instances, and those rated 4-6 as positive instances. The scores are ranked based on the natural log of the posterior odds of positive. A second method treats the 6 ratings as 6 distinct categories [8,14,15]. When predicting for a target movie, the personalized movie recommender system computes the posterior probability of each category given the target movie. Then the predicted rating of the posterior probability distribution for the categories is computed and used as the predicted ratings by Equation (7). Here, $P(i)$ is the posterior probability for category i.

$$predicted\ score = \sum\nolimits_{i=1}^{6} i \cdot P(i) \tag{7}$$

We use the predicted ratings rather than choosing the most probable category to better represent the continuity of the ratings. When using the 6-category classifier to predict a binary category, we classify all movies as positive category or negative category. The final method is used a weighted binary classifier that maps the user's actual 1-6 rating into a weight value. As with relevance feedback in information retrieval, this cycle can be repeated several times to recommend the good results [2,10,13,15].

4 Performance Evaluations

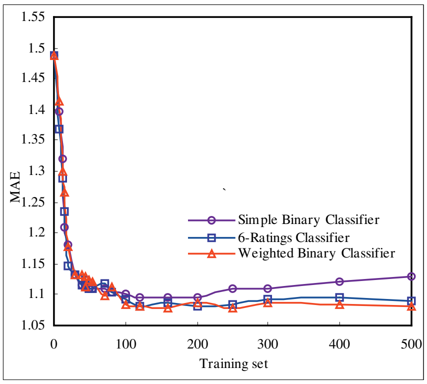
We use the user-movie ratings data provided by the EachMovie dataset and the movie details from the Internet Movie Database (www.imdb.com). We represent the content information of every movie as a set of slots. Each slot is represented as a bag of association words. The slots we use for the EachMovie dataset [9] are: movie title, cast, director, newsgroup reviews, genre, plot, summary, plot keywords, user comments, external reviews, and award. The reduced dataset has 299,997 ratings for 1,289 movies for which content information was available from IMDb. To evaluate various approaches of information filtering, we divided the rating dataset in test-set and training-set. To observe performance given varying amounts of training movie set, learning curves were generated by checking the recommender system after training on increasing subsets of the overall training data. A number of metrics were used to performance measures, mean absolute error (MAE) and rank scoring measure (RSM), both suggested by [3], F-measure are used to gauge performance. It is important to evaluate accuracy and recall in conjunction. To quantify this with single measure, we use F-measure in Equation (8), which is a weighted combination of accuracy and recall [6,8,10]. *P* indicates accuracy and *R* means recall and in that case, the higher F-measure the better classification. Beta indicates relative weight of recall against accuracy and if it is 1.0, weight of accuracy and recall is the same. "*a*" is the number of web pages, which appear in both classes. "*b*" is the number of web pages, which appear in class categorized by first method but not in class categorized by second method. "*c*" is the number of web pages, which appear in class categorized by second method but not in class categorized by first method. In this experiment, beta was designed as 1 to analysis the results of classification, and different results of F-measure according to change in value of beta from 0.5 to 1.4 were observed [6,7].

$$F-measure = \frac{(\beta^2+1)PR}{\beta^2 P + R} \qquad P = \frac{a}{a+b}100\% \qquad R = \frac{a}{a+c}100\% \tag{8}$$

Fig. 1 shows the result of MAE, RSM at varying the training movie set. For current experiments, this paper compares the following methods: a simple binary classifier, a 6-ratings classifier, and a weighed binary classifier.

We expected that the binary classifier would perform better on classification accuracy on the MAE, RSM since it is specifically designed for that task. Since users will often be willing to rate a small number of the training movie set, getting enough ratings to produce good performance from the 6-ratings classifier could often be impractical. However, as the weighted binary classifier performs comparable to the binary classifier, this proposes that the weighted binary classifier may be the good [15].

For evaluation, this paper uses all of the following methods: the proposed method using learning user profile with extracted information (LUP_EI), the collaborative filtering method using the Pearson correlation coefficient (P_Corr) [3], the recommendation method using only the content-based filtering (Content) [4], and naïve combined approach (N_Com) [16]. The naïve combined approach takes the average of the ratings generated by collaborative filtering and content-based filtering. These methods are compared by varying the number of clustered users. Also, the proposed



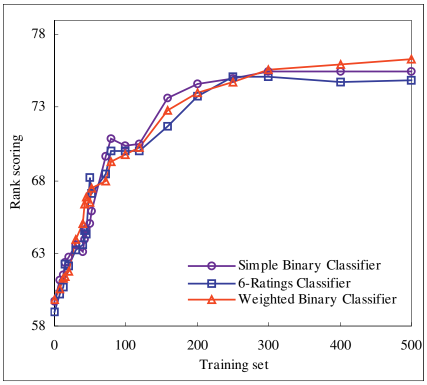
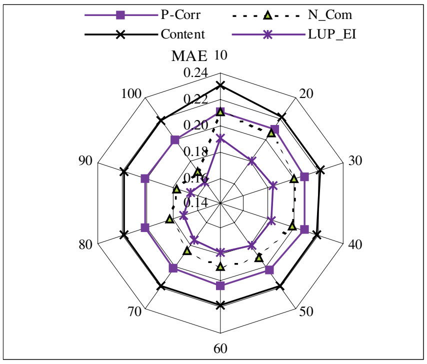


Fig. 1. MAE, RSM at varying the training movie set.



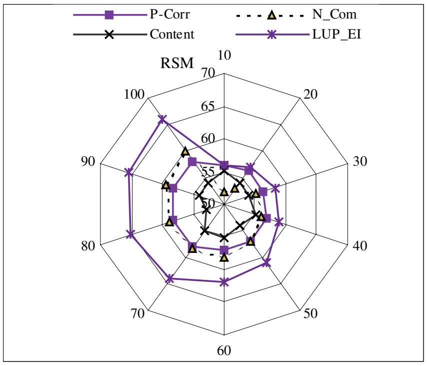


Fig. 2. MAE, RSM at varying the number of clustered users.

method (LUP_EI) was compared with the previous methods that use both collaborative filtering and content-based filtering by changing the number of user evaluations on items. The previous methods include the Pazzani method [12], Lee method [8], and Good method [5] using user profile.

Fig. 2 shows the MAE and RSM of LUP_EI, P_Corr, Content, and N_Com. Fig. 2, as the number of users increases, the performance of the LUP_IE, and the N_Com also increases, whereas P_Corr and Content show no notable change in performance. In terms of accuracy of prediction, it is evident that method LUP_EI is more superior to others. Fig. 3 shows the prediction speed of LUP_EI, Pazzani method, Lee method, and Good method when the number of user's evaluations is increased. Fig. 4 shows the result of F-measure at varying with change for with Beta. Fig. 3 indicates speed of prediction at *n*th rating about 1,892 movies. LUP_EI is the fastest 8.71sec followed by 10.84sec of Pazzani method, 14.7sec of Lee method, and 13.2sec of Good method. In terms of speed, LUP_EI and Pazzani are excellent in comparison with others and remainder records similar speed. Fig. 4 indicates analyzed performance of F-measure according to change of beta value from 0.5 to 1.4. LUP_EI represents a rising curve as beta value increases and so it records better performance in terms of accuracy than

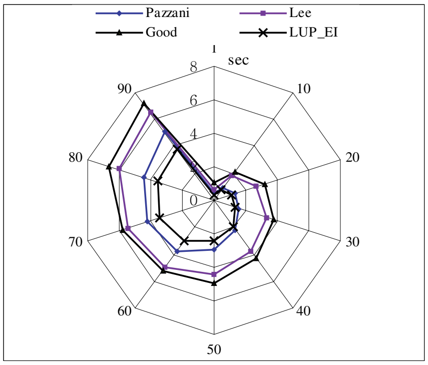


Fig. 3. Prediction speed at *n*th rating (sec).

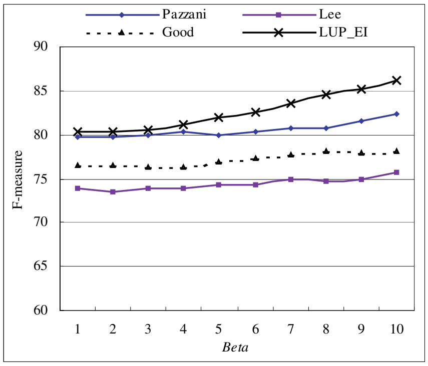


Fig. 4. F-measure at varying Beta.

recall. But in Pazzani method and Good method, change of beta value hardly affects F-measure value and so it has similar level of performance in terms of recall and accuracy. However, in Lee method, it has a bit higher performance in terms of recall than accuracy. On average, if beta is 1.0, LUP_EI has 1.31% higher performance than Pazzani method and 2.13% Good method and 4.79% Lee method.

5 Conclusions

Recommender systems improve access to relevant products and information by making personalized suggestions based on previous movies of a user's likes and dislikes. We have proposed automatic preference mining through learning user profile with Naïve Bayesian algorithm to efficiently handle set-valued features. And then information about movies extracted from web pages to recommend movie titles based on training movies supplied by an individual user. The suggested method has two advantages. First, it has refined association words so that Naïve Bayesian classifier effects accurate and speedy classification of web pages. Second, it removes confusion in meaning by expressing characteristic of web pages. As a result, the automatic preference mining through learning user profile with extracted information has 1.31% higher performance than Pazzani method and 2.13% higher than Good method and 4.79% than Lee method. The proposed method was compared with the existent methods rating preference automatically in recommender systems, and the existent methods for information filtering, which the naïve combined method. The proposed method shows higher performance than existent method in both comparisons. Our approach could provide a useful service to customers overwhelmed by the abundance of choice presented by the world.

Acknowledgements

The author would like to thank the anonymous reviewers whose comments helped improve the paper. This work was supported by INHA UNIVERSITY Research Grant. (INHA-31613)

References

1. R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," In Proc. of the 20th International Conference Very Large Data Bases, pp.487-499, Santiago, Chile, 1994.
2. D. Billsus, M. J. Pazzani, "Learning Collaborative Information Filters," In Proc. of the 15th International Conference on Machine Learning, pp.46-54, 1998.
3. J. S. Breese, D. Heckerman, C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," In Proc. of the Conference on Uncertainty in Artificial Intelligence, pp.43-52, Madison, WI, 1998.
4. D. D. Lewis, "Naive (Bayes) at forty: The Independence Assumption in Information Retrieval," Proc. of the 10th European Conference on Machine Learning, pp.4-15, 1998.
5. N. Good, et al., "Combining Collaborative Filtering with Personal Agents for Better Recommendations," In Proc. of the 16th National Conference on Artificial Intelligence, pp.439-446, 1999.
6. S. J. Ko, J. H. Lee, "Bayesian Web Document Classification through Optimizing Association Word," In Proc. of the 16th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, LNAI 2718, Springer-Verlag, pp.565-574, 2003.
7. K. Y. Jung, J. H. Lee, "Hybrid Collaborative Filtering and Content-based Filtering for Improved Recommender System," In Proc. of the International Conference on Computational Science, LNCS, Springer-Verlag, pp.299-306, 2004.
8. W. S. Lee, "Collaborative Learning for Recommender Systems," In Proc. of the 18th International Conference on Machine Learning, pp.314-321, 1997.
9. P. McJones, EachMovie dataset, URL: http://www.research.digital.com/SRC/eachmovie
10. T. Mitchell, Machine Learning, McGraw-hill, New York, pp.154-200, 1997.
11. Introduction to Rainbow URL: http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naïve-bayes.html.
12. M. J. Pazzani, "A Framework for Collaborative, Content-based and Demographic Filtering," Artificial Intelligence Review, pp.393-408, 1999.
13. T. Joachims, "A Probabilistic Analysis of the Rocchio Algorithm with TF·IDF for Text Categorization," In Proc. of the 14th International Conference on Machine Learning, pp.143-151, 1997.
14. P. Melville, et al., "Content-Boosted Collaborative Filtering for Improved Recommendations," In Proc. of the National Conference on Artificial Intelligence, pp.187-192, 2002.
15. R. Raymond, P. Bennett, L. Roy, "Book Recommending Using Text Categorization with Extracted Information," In Proc. of the AAAI'98/ICML Workshop on Learning Categorization, pp.49-54, Madison, WI, 1998.
16. I. Soboroff, C. Nicholas, "Combining Content and Collaboration in Text Filtering," In Proc. of the IJCAI'99 Workshop on Machine Learning in Information Filtering, pp.86-91, 1999.
17. P. N. Bennett, et al., "Probabilistic Combination of Text Classifiers using Reliability Indicators: Models and Results," In Proc. of the 25th Annual International ACM SIGIR Conference, pp.207-214, 2002.
18. K. Yu, et al., "Feature Weighting and Instance Selection for Collaborative Filtering: An Information-Theoretic Approach," Knowledge and Information Systems, Vol. 5, No. 2, pp. 201-224, Springer-Verlag, 2003.

A Computational Study of Naïve Bayesian Learning in Anti-spam Management

Zhiwei Fu and Isa Sarac

Virginia International University,
3957 Pender Drive, Fairfax, VA 22030 USA
{zfu,isarac}@viu.edu
http://www.viu.edu

Abstract. It has been argued that Bayesian learning can be used to filter unsolicited junk e-mail ("spam") and outperform other anti-spam methods, e.g., the heuristics approaches. We develop a Bayesian learning system, and conduct a computational study on a corpus of 10,000 emails to evaluate its performance and robustness, particularly the performances with different training-corpus sizes and multi-grams. Based on the computational results, we conclude that the Bayesian anti-spam approach is promising in anti-spam management as compared with other methods at the client side, and may need additional work to be viable at the corporate level in practice.

1 Introduction

As information technology fast advances forward, unsolicited commercial e-mail ("spam") has become an ever-increasing problem. The statistics has shown that 10.4 million spam emails are sent every minute worldwide, and the spam has at least quadrupled in the past two years. The problems caused by unsolicited commercial e-mail ("spam") go well beyond the annoyance spam causes to the public. These problems include the fraudulent and deceptive content of most spam messages, the sheer volume of spam being sent across the Internet, and the security issues raised because spam can be used to disrupt service or as a vehicle for sending viruses [1]. A recent Gartner Group survey revealed that 34% of business email is useless. The same study also revealed that employees spend an average of 49 minutes a day managing email.

President George W. Bush signed a landmark anti-spam bill into law in December 2003 that became effective Jan. 1, 2004, setting into motion the first national standards for sending bulk unsolicited commercial e-mail (UCE). Pre-empting many tougher state anti-spam laws, the Can Spam Act aims to curb the most egregious practices of spammers by targeting e-mail with falsified headers, but allows e-marketers to send UCE as long as the message contains an opt-out mechanism, a functioning return e-mail address, a valid subject line indicating the e-mail is an advertisement and the legitimate physical address of the mailer. However, some critics say that it legitimizes spam by allowing sending unsolicited commercial email as long as it has an opt-out mechanism, a functioning return e-mail address, a valid subject line indicating the email is an advertisement and the legitimate physical address of the mailer. Therefore, improved technological tools will be an essential part of any solution as well. Currently there are three major types of anti-spam approaches, 1) source-based approach of managing sender identity, e.g., Black lists, White lists, Real Time Black lists (RBL), Reverse DNS Lookups. Source-based approaches are perhaps most effective in levering system resources but they are likely to yield poor overall hit rate. 2) Rule-based content analysis, e.g., pattern matching, spam definitions, heuristics, and the approaches of this kind often use scoring techniques. However, rule-based approaches tend to go stale as spammers move on to new tricks. Keeping rule-based up-to-date and effective requires a great amount of human resources to come up with new rules. Rule-based is generally considered more accurate than source-based approaches. In this category, the heuristics approach applies multiple detection tests to provide greater confidence in identifying spam messages [2]. 3) Bayesian learning, a type of statistical approach to identify spam based on their characteristics that have been learned from existing emails categorized by users and then apply the knowledge to new incoming emails. Bayesian analysis has been considered most accurate approach, especially at the client side, to effectively tackle fast-changing spam. But the Bayesian approach consumes heavy computation.

As we emphasize on accuracy than cost, Bayesian analysis became the major focus of this paper, in comparison to the heuristics approach. In this paper, we will first describe naïve Bayesian learning in section 2, and our learning system, followed by our computational study, including experiment design and the computational results; and section 4 provide concluding remarks and some future work.

2 Bayesian Learning

Naïve Bayesian learning is the optimal classification method of supervised learning if the values of the attributes of an example are independent given the class of the example [3]. On many real-world example datasets Bayesian learning gives better test set accuracy than any other known method, including backpropagation [7] and C4.5 decision trees [6].

Let A_1 through A_k be attributes with discrete values used to predict a discrete class C. Given an example with observed attribute values a_1 through a_k, the optimal prediction is class value c such that $P(C = c \mid A_1 = a_1 \cap ... \cap A_k = a_k)$ is maximal. By Bayes' rule this probability equals

$$\frac{P(A_1 = a_1 \cap ... \cap A_k = a_k \mid C = c)}{P(A_1 = a_1 \cap ... \cap A_k = a_k)} \cdot P(C = c).$$

The background probability or base rate $P(C = c)$ can be estimated from training data easily. The example probability $P(C = c \mid A_1 = a_1 \cap ... \cap A_k = a_k)$ is irrelevant for decision-making since it is the same for each class value c. Learning is therefore reduced to the problem of estimating $P(A_1 = a_1 \cap ... \cap A_k = a_k \mid C = c)$ from training examples. Using Bayes' rule again, this class-conditional probability can be written as

$$P(A_1 = a_1 \mid A_2 = a_2 \cap ... \cap A_k = a_k, C = c) \cdot P(A_2 = a_2 \cap ... \cap A_k = a_k \mid C = c).$$

Recursively, the second factor above can be written as

$$P(A_2 = a_2 \mid A_3 = a_3 \cap ... \cap A_k = a_k, C = c) \cdot P(A_3 = a_3 \cap ... \cap A_k = a_k \mid C = c)$$

and so on. Now suppose we assume for each A_i that its outcome is independent of the outcome of all other A_j, given C. Formally, we assume that

$$P(A_1 = a_1 \mid A_2 = a_2 \cap ... \cap A_k = a_k, C = c) = P(A_1 = a_1 \mid C = c)$$

and so on for A_2 through A_k. Then $P(A_1 = a_1 \cap ... \cap A_k = a_k \mid C = c)$ equals

$$P(A_1 = a_1 \mid C = c) \cdot P(A_2 = a_2 \mid C = c) ... P(A_k = a_k \mid C = c)$$

Now each factor in the product above can be estimated from training data:

$$\hat{P}(A_j = a_j \mid C = c) = \frac{count(A_j = a_j \cap C = c)}{count(C = c)}$$

It can be shown that the above equation gives "maximum likelihood" probability estimates, i.e. probability parameter values that maximize the probability of the training examples. The above induction algorithm is called naïve Bayesian learning [4].

3 Computational Experiments

3.1 Experimental Design

We build our Bayesian learning system based on the above induction algorithm, i.e., the naive Bayesian learning. Our Bayesian learning system is developed to take the whole message into account. Each email is treated as a data record, and the score of the email is collectively determined by the spam characteristics of each and every word in the email. Specifically, the Bayesian learning system not only recognizes keywords that identify spam, but can also recognize words that denote valid mail. For example: not every email that contains the word "free" and "cash" is spam. Our learning system would be able to recognize the name of the business contact that sent the message and thus classify the message as legitimate, and therefore, allows words to "balance" each other. Our learning system also inherits Bayesian's self-adaptation to evolve itself by constantly learning from new spam and new valid outbound mails. For example, when spammers started using "f-r-e-e" instead of "free" they succeeded in evading keyword checking until "f-r-e-e" was also included in the keyword database. But our Bayesian learning system is able to automatically notice such tactics; in fact if the word "f-r-e-e" is found, it is an even better spam indicator. In addition, an initial training database of emails is formed based on inbound emails for our Bayesian learning system and the database is then updated based on the training results. The Bayesian learning will continuously learn the characteristics from the inbound emails and apply the learning to categorize the prospective emails.

We designed and developed our semantic Bayesian learning system to filter spam (junk and unsolicited commercial emails) from hams (the legitimate emails). Our experiments are intended to test the performance and robustness of our Bayesian approach as compared with some anti-spam heuristics scoring approach. We have collected about 10,000 emails to obtain unbiased results. In general, the resampling techniques provide reliable estimates of the true error rate, because nearly all the data points used for training, and all data points are used for testing, and the Bayesian classifier can therefore be reapplied to all data points. In our experiments, we use resampling techniques, i.e., repeated train-and-test partitions, to estimate the classification error rate. In particular, we use 10-fold cross-validation to make our results less prone to random variation and to further compare statistically across different anti-spam approaches [5].

3.2 Computational Results

In our experiments, we apply both rule-based heuristics approach and our Bayesian learning system on a corpus of the 10,000 emails. The overall distribution of spam and ham obtained from the heuristics approach given in Fig.1 indicates a non-trivial set separation of ham and spam for the heuristics approach. The modes of spam and ham in Fig.1 are mainly located in the middle of the scoring spectrum with tails of both "clearly" identified spam and ham extended to both ends. On the contrary, Bayesian learning system provides a desired "clear-cut" set separation of ham from spam (as given in Fig.2). The modes of spam and ham are located far apart at the ends of the entire spectrum of spam scores for all emails, while there are some overlaps (or the area with uncertain emails based on their spam scores) in the middle of the spectrum. Ideally, every email should be correctly classified with no uncertainty and there is no false positive and false negative, and then there would be no overlaps in the middle of the spectrum. Nevertheless, the distribution of spam/ham scores obtained by the Bayesian approach (as in Fig.2) apparently looks more promising in practice than that by the heuristics approach (as in Fig. 1).

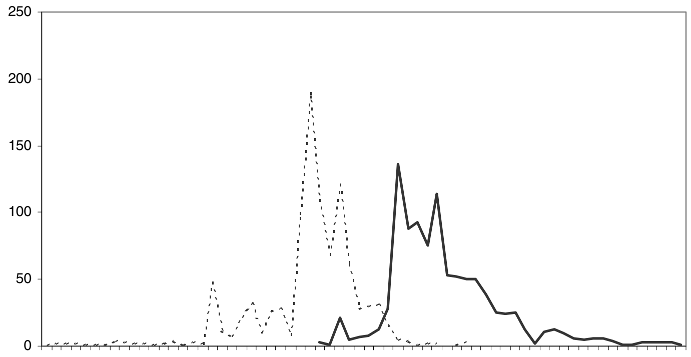


Fig. 1. Overall spam (the solid line) and ham (the dotted line) distribution obtained by the heuristics approach. The horizontal axis is the spectrum of spam scores for all emails, and the vertical axis is the frequency of emails by spam scores.

Anti-spam approaches such as Bayesian learning require some level of training, and almost all anti-spam filtering systems use unigrams (a single 1-word approach) to filter spam. However, the English language becomes more "structured" if we use bigrams and trigrams. Therefore, we study the performance of multi-grams, i.e., *n*-word approaches (*n* = 1,2,3) combined with the training sets of different sizes in our experiments. As given in Table 1, the heuristics approach provides good error estimates with average classification accuracy above 93%, FP (false positive) and FN (false negative) about 10.5% and 2.5%, respectively. In particular, 1000 training size provides the best performance for all training sizes. It indicates that a better quality training set with reasonable size would be more sufficient and effective than a less quality but larger training set, i.e., "garbage in, garbage out". On the other hand, multi-grams, trigrams and bigrams in our experiments have not demonstrated statistical significance in their performances over unigrams.

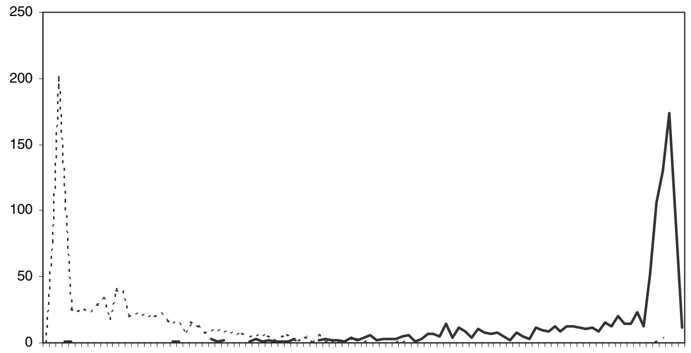


Fig. 2. Overall spam (the solid line) and ham (the dotted line) distribution obtained by our Bayesian learning approach. The horizontal axis is the spectrum of spam scores for all emails, and the vertical axis is the frequency of emails by spam scores.

Bayesian learning has demonstrated significantly better performances in all regards as given in Table 2 than the heuristics approach. The classification accuracy and FP by the Bayesian learning are on average 97% and 2%, respectively. We then conducted paired difference *t*-test, and the *p*-values of paired difference *t*-tests of overall classification accuracy for different combinations are given in Table 3 and Table 4. As shown, the computational results from both the heuristics approaches and our Bayesian approach also find no statistical significance among different training sizes. However, the performance of accuracy for *n*-word approaches has shown some improvements from unigrams and bigrams, to trigrams. As shown, multi-grams outperform unigrams for training size of 1000, but show no significance statistically for other training sizes. Although the experiments here may not be conclusive statistically, the results strongly indicate again that applying Bayesian unigram learning with a good quality training set with reasonable size should be sufficiently effective in anti-spam management at the client side.

Table 1. Computational results of classification accuracy, false positive (FP), and false negative (FN) obtained by the heuristics approach (an experiment with 1000 training set using 1-word approach is denoted by 1000 1-word).

	Accuracy	FP	FN
1000-1w ord	94.06%	7.99%	3.90%
1000-2w ord	94.91%	6.79%	3.40%
1000-3w ord	97.50%	2.20%	2.80%
2000-1w ord	92.85%	12.59%	1.70%
2000-2w ord	92.68%	13.24%	1.40%
2000-3w ord	92.85%	12.64%	1.65%
3000-1w ord	92.93%	12.46%	2.13%
3000-2w ord	92.56%	13.37%	2.00%
3000-3w ord	92.91%	12.54%	2.10%
Mean	93.69%	10.43%	2.34%
StdErr	0.005	0.012	0.003

Table 2. Computational results, classification accuracy, false positive (FP), and false negative (FN) obtained by our Bayesian approach.

	Accuracy	FP	FN
1000-1w ord	96.80%	1.80%	4.60%
1000-2w ord	97.75%	1.70%	2.80%
1000-3w ord	98.75%	0.90%	1.60%
2000-1w ord	96.78%	1.75%	4.70%
2000-2w ord	96.85%	2.10%	4.20%
2000-3w ord	96.90%	2.10%	4.10%
3000-1w ord	96.19%	2.87%	4.67%
3000-2w ord	96.37%	3.02%	4.20%
3000-3w ord	96.37%	2.94%	4.27%
Mean	96.97%	2.13%	3.90%
StdErr	0.003	0.002	0.003

Table 3. *p*-values of paired difference *t*-test of overall classification accuracy obtained by the heuristics approach (the results are computed based on row methods over column methods, and 1000-1word is denoted by 1k-1w).

vs.	1K-1w	1K-2w	1K-3w	2k-1w	2k-2w	2k-3w	3k-1w	3k-2w	3k-3w
1K-1w				0.0199	0.0120	0.0191	0.0074	0.0023	0.0055
1K-2w	0.0942			0.0026	0.0008	0.0023	0.0016	0.0005	0.0009
1K-3w	0.0006	0.0002		0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2k-1w					0.1044			0.1010	
2k-2w								0.3184	
2k-3w				0.5000	0.1207			0.1451	
3k-1w				0.3845	0.1735	0.3860		0.0078	0.4158
3k-2w									
3k-3w				0.4128	0.1843	0.4101		0.0126	

Table 4. *p*-values of paired difference *t*-test of overall classification accuracy obtained by the Bayesian approach (the results are computed based on row methods over column methods, and 1000-1word is denoted by 1k-1w).

vs.	1K-1w	1K-2w	1K-3w	2k-1w	2k-2w	2k-3w	3k-1w	3k-2w	3k-3w
1K-1w				0.4734	0.4486	0.3917	0.0488	0.0825	0.1084
1K-2w	0.0137			0.0319	0.0346	0.0455	0.0027	0.0021	0.0046
1K-3w	0.0000	0.0058		0.0003	0.0004	0.0004	0.0000	0.0000	0.0000
2k-1w					0.2796			0.1089	
2k-2w								0.0525	
2k-3w				0.1357	0.2950			0.0437	
3k-1w				0.0246	0.0103	0.0057		0.0934	0.0534
3k-2w									
3k-3w				0.1029	0.0480	0.0399		0.4997	

4 Concluding Remarks

Fighting spam is not a trivial undertaking, especially at the server side or corporate level. For example, the judgment on spam/ham is highly individual, and the unsolicited commercial emails that one recipient would accept/reject could be significantly different from another recipient based on the recipients' interests. As shown in our experiments, the Bayesian learning approach is advantageous as compared to other heuristics approaches, but it works best at client side. In addition, consumes significantly more computational resources, and another major concern resides in maintaining a good database for continuous training. As shown in the study, multi-grams look appealing but need more study for its significance. In our future work, we will research the robustness of implemented at gateway level for corporations and ways to improve the performance of our Bayesian learning.

References

1. Baker, S. (2003). The Taming of the Internet, Business Week Magazine, December 2003.
2. Conry-Murray, A. (2003). Fighting the Spam Monster - and Winning, Network Magazine, April 2003.
3. Elkan, C. (1997). Naïve Bayesian Learning, Technical Report No. CS97-557, University of California, San Diego. Baldonado, M., Chang, C-C.K., Gravano, L., Paepcke, A.: The Stanford Digital Library Metadata Architecture. Int. J. Digit. Libr. 1 (1997) 108–121
4. Gelman, A., J. Carlin, H. Stern, and D. Rubin (2000). Bayesian Data Analysis, New York, NY: Chapman & Hall/CRC.
5. Hastie, T., R. Tibshirani, and J. Friedman (2001). The Elements of Statistical Learning – Data Mining, Inference, and Prediction, New York, NY: Springer-Verlag.
6. Quinlan, J. R. (1993). C4.5: Programs for Machine Learning, San Mateo, CA: Morgan Kaufmann.
7. Riply, B. (1996). Pattern Recognition and Neural Networks, Cambridge, MA: Cambridge University Press.

Kernel-Based Non-linear Template Matching

Barend J. van Wyk, Michaël A. van Wyk, and Guillaume Noel

French South-African Technical Institute in Electronics,
Tshwane University of Technology
Staatsartillerie Road, Pretoria, South Africa
{ben.van.wyk,guillaume.noel,mavw}@fsatie.ac.za

Abstract. A new non-linear minimum norm template matching technique is introduced. Similar to the theory of Support Vector Machines the proposed framework is also based on Reproducing Kernel Hilbert Space principles. Promising results when applied to aerial image matching are reported and future work is highlighted.

1 Introduction

In this paper the problem of finding the location of a known reference image, or template, in a larger input image is addressed. Finding the location of the reference image can be done by searching through the input image using the normalized cross correlation as a similarity measure [1]. As the normalized cross correlation technique won't work when our reference image is rotated or scaled, several modifications were proposed. The Fourier and Mellin transforms can be combined [2], multiple templates can be used, or the reference and sub-images can be described in terms of invariant moments and then the correlations involving these moments can be used as a similarity measure [3]. As shown by Uenohara and Kanade [4] the multiple template approach can be made more efficient by implementing a dual decomposition using the Fourier and Karhunen-Loéve transforms. Ben-Arie and Rao [5] [6], on the other hand proposed non-orthogonal image expansions where the search area is represented by basis functions that are effectively the template translated to different positions.

The technique proposed in this paper is also based on the idea of having multiple templates, but differs from other popular methods in the way the templates are selected. Another differentiating factor is that it is non-linear, with the linear case as a special instance of the proposed framework.

2 Non-linear Template Matching Framework

Similar to the theory of Support Vector Machines (SVMs) our framework is also based on Reproducing Kernel Hilbert Space (RKHS) principles, in particular on the idea of an RKHS interpolator. For a more general discussion on RKHS interpolators, the reader is referred to [7], [8], [9], [10] and [11]. The following theorem, stated by Zyla and De Figueiredo [12] for Bochner spaces, but adapted here for our purposes, is the core of our methodology:

Theorem 1. *[12] Given that an input-output map $\widetilde{F}$ to be identified belongs to $\mathcal{H}_n$, an RKHS, and assuming that we are provided with a set of test input-output pairs*

$$\{(\mathbf{x}_i \in \mathbb{R}^N, y_i)\}_{i=1}^m \tag{1}$$

where $\mathbf{x}_i$, $i = 1, \ldots, m$, are linearly independent elements of $\mathbb{R}^N$, the problem has a unique minimum norm solution expressed by

$$\widetilde{F}(\mathbf{x}) = \sum_{i=1}^{m} C_i \, K(\mathbf{x}_i, \mathbf{x}) \tag{2}$$

where $K(\mathbf{x}_i, \cdot)$ is a reproducing kernel of the space $\mathcal{H}_n$. The coefficients C_i are given by the expression

$$\mathbf{C} = \mathbf{G}^{-1}\mathbf{y} \tag{3}$$

where

$$\mathbf{C} := (C_1, ..., C_m)^T,$$
$$\mathbf{y} := (y_1, ..., y_m)^T$$

and the Gram matrix, $\mathbf{G}$, is given by

$$\mathbf{G} := (G_{ij})$$

where

$$G_{ij} := K(\mathbf{x}_i, \mathbf{x}_j), \qquad i, j = 1, \ldots, m.$$

Theorem 1 will now be used for the derivation of our template matching scheme. To apply theorem 1 to template matching five factors need to be considered namely, defining the test input-output pairs, choosing a suitable kernel, calculating the interpolator coefficients, deriving a minimum norm template and implementing the matching process. From theorem 1 it is clear that once the test input-output pairs are defined and an appropriate kernel chosen, the interpolator constraints are obtained by simply inverting a Gram matrix. Refer to Luenberger [13] for conditions under which the Gram matrix is invertible. If the Gram matrix is found to be ill-conditioned or badly scaled one can resort to the pseudo-inverse. In the approach followed by De Figueiredo and Zyla [7] [12], every reproducing kernel $K(\mathbf{x}_i, \cdot)$ is associated with a specific norm on $\mathcal{H}_n$. It is important to note that because of this relationship *only* knowledge of the reproducing kernel is required when applying theorem 1. Refer to [14] to see how theorem 1 relates to the well-known representer's theorem.

The definition of the test input-output pairs is discussed in section 2.1, examples of suitable kernels are given in section 2.2, the derivation of minimum norm templates are discussed in section 2.3 and the matching process is detailed in section 2.4.

In short we will infer a *Minimum Norm Template* (MNT) based on Eq. 2 using k *Desirable Image Templates* (DITs) and $m-k$ *Undesirable Image Templates* (UITs) where $m \geq k$, as input-output pairs. For the application considered in this paper the DITs will be rotated and scaled versions of the region of interest as shown in figures 1 to 4, i.e. instances of the entity we want to find in a complex image. The UITs will be instances of objects or backgrounds we don't want to recognize as our region of interest such as undesirable complex backgrounds as shown in figure 5. For simplicity it will be assumed that the DITs and UITs are square, have equal dimension and are represented by $\mathbf{X}_i \in \mathbb{R}^{\overline{N}\times\overline{N}}$.

2.1 Test Input-Output Pairs $\{(\mathbf{x}_i \in \mathbb{R}^N, y_i)\}_{i=1}^m$

For the DITs (i.e. $i \leq k$) the y_i values in Eq. 1 are normally chosen equal to some positive value, say γ. The rest of the y_i values for the UITs are normally set to α, where α is zero or $-\gamma$. Each $\mathbf{x}_i$ is simply set equal to $\operatorname{vec}(\mathbf{X}_i) \in \mathbb{R}^N$ where $\operatorname{vec}(\cdot)$ is the matrix vectorization operator and $N = \overline{N}^2$.

2.2 Examples of Reproducing Kernels $K(\mathbf{x}, \mathbf{z})$

Although the theory is general enough to allow other reproducing kernels we will only focus on three types of kernels, namely the linear kernel

$$K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z}, \tag{4}$$

the polynomial kernel,

$$K(\mathbf{x}, \mathbf{z}) = \left(1 + \mathbf{x}^T \mathbf{z}\right)^d, \qquad d \geq 1, \tag{5}$$

and the polynomial kernel without cross terms

$$K(\mathbf{x}, \mathbf{z}) = \left(1 + \sum_{\beta=1}^{d} \left(x_1^\beta z_1^\beta + x_2^\beta z_2^\beta, ..., x_N^\beta z_N^\beta\right)\right), \quad d \geq 1. \tag{6}$$

2.3 The Minimum Norm Template

Once the interpolator coefficients are obtained an MNT can be inferred. When using the linear kernel it is easy to show that the MNT has the form

$$\widetilde{\mathbf{x}} = \sum_{i=1}^{m} C_i \mathbf{x}_i \tag{7}$$

and that $K(\widetilde{\mathbf{x}}, \cdot\,)$ will satisfy

$$K(\widetilde{\mathbf{x}}, \mathbf{x}_i\,) = \begin{cases} \gamma \text{ for } \; i = 1, ..., k \\ \alpha \text{ for } \; i > k \end{cases} .$$

When the DITs and UITs are linearly separable, the suitability of using $\operatorname{devec}(\widetilde{\mathbf{x}}) \in \mathbb{R}^{\bar{N}\times\bar{N}}$ as an object template is obvious. Here devec denotes the inverse of vec.

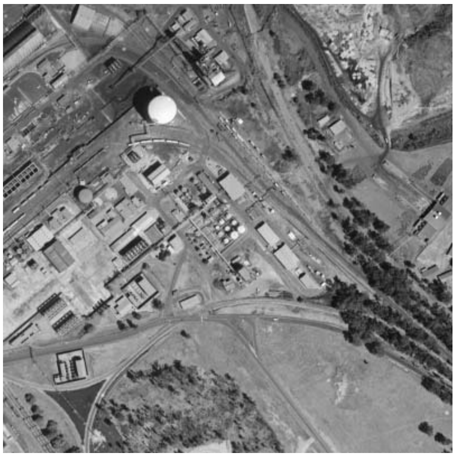


Fig. 1. Original image.

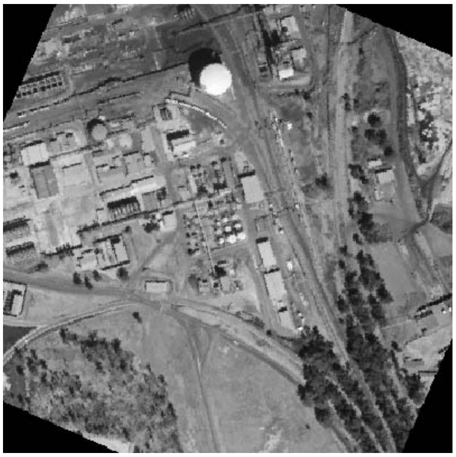


Fig. 2. Example of rotated (-24 degrees) and scaled (factor 20) input image.

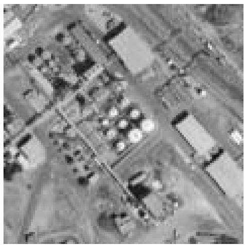


Fig. 3. Image template: Region searched for in input image.

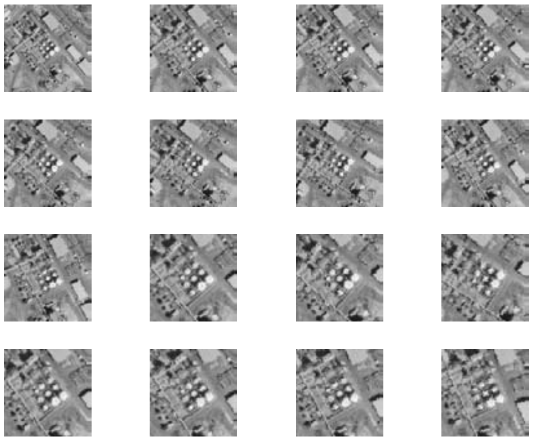


Fig. 4. Example of 16 Desirable Image Templates (DITs).

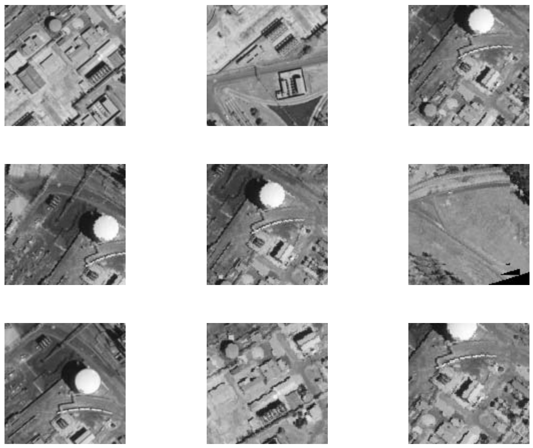


Fig. 5. Example of 9 Undesirable Image Templates (UITs).

When our input training pairs are not linearly separable we will resort to polynomial kernels. First consider the kernel given by Eq. 5. For simplicity we will consider the case where $d = 2$. By using similar arguments as for the linear kernel case it can be shown that

$$\widetilde{\mathbf{x}} = \sum_{i=1}^{m} C_i \bar{\mathbf{x}}_i, \tag{8}$$

where $\bar{\mathbf{x}}_i = \left[\left[1\ \mathbf{x}_i^T\right] \otimes \left[1\ \mathbf{x}_i^T\right]\right]^T$ and $\otimes$ denotes the *Kronecker Tensor Product*. Similar to the linear case,

$$\widetilde{\mathbf{x}}^T \bar{\mathbf{x}}_i = \begin{cases} \gamma \text{ for } \ i = 1, ..., k \\ \alpha \text{ for } \ i > k \end{cases} .$$

These results can be readily extended for cases where $d > 2$. When $d = 3$ for example we have

$$\widetilde{\mathbf{x}} = \sum_{i=1}^{m} C_i \left[\left[1\, \mathbf{x}_i^T \right] \otimes \left[1\, \mathbf{x}_i^T \right] \otimes \left[1\, \mathbf{x}_i^T \right] \right]^T .$$

The polynomial kernel without cross terms given by Eq. 6 can be seen as a compromise between the linear kernel and the polynomial kernel given by Eq. 5. The *Minimum Norm Template* (MNT) for this case can be expressed as a concatenated vector given by

$$\widetilde{\mathbf{x}} = \left[\sum_{i=1}^{m} C_i,\ \sum_{i=1}^{m} C_i \mathbf{x}_i^1,\ \sum_{i=1}^{m} C_i \mathbf{x}_i^2, ..., \sum_{i=1}^{m} C_i \mathbf{x}_i^d \right]^T \tag{9}$$

where $\mathbf{x}_i^d$ denotes that every element of $\mathbf{x}_i^T$ is raised to the power d. Once again

$$\widetilde{\mathbf{x}}^T \bar{\mathbf{x}}_i = \begin{cases} \gamma \text{ for } i = 1, ..., k \\ \alpha \text{ for } i > k \end{cases} ,$$

where $\bar{\mathbf{x}}_i := [1\, \mathbf{x}_i^1\ \mathbf{x}_i^2 ... \mathbf{x}_i^d]$.

2.4 The Matching Process

In summary the kernel-based template matching process involves the following steps:

1. Calculate $\widetilde{\mathbf{x}}$, the MNT, offline using the DITs and the UITs.
2. Once the MNT $\tilde{\mathbf{x}}$ has been obtained, $\tilde{\mathbf{x}}^T \bar{\mathbf{x}}_{kl}$ which serves as our similarity measure, is calculated for all (or selected) portions over an area of interest in the input image. When for example the polynomial kernel without cross terms is used, then from the previous section we have $\bar{\mathbf{x}}_{kl} := [1\, \text{vec}\mathbf{X}_{kl}^1\ \text{vec}\mathbf{X}_{kl}^2 ... \text{vec}\mathbf{X}_{kl}^d]$. Here $\mathbf{X}_{kl} \in \mathbb{R}^{\overline{N} \times \overline{N}}$, an $\overline{N} \times \overline{N}$ region centered at position (k, l) in the $K \times L$ input image and $\text{vec}\mathbf{X}^d$ denotes that every element of $(\text{vec}\mathbf{X})^T$ is raised to the power d. It is assumed that $K, L > \overline{N}$.
3. Position (k, l) in the input image where the value for $\tilde{\mathbf{x}}^T \bar{\mathbf{x}}_{kl}$ is a maximum is taken as the location of the object or region to be identified.

The template matching method presented here differs from conventional SVM classification strategies mainly in two aspects: *1)* Kernel evaluations are only performed to calculate the MNT. Once the MNT is calculated no further kernel evaluations are required during the execution of the matching process. *2)* The interpolator coefficients used to construct the MNT are obtained by simply inverting a Gram matrix. Complex optimization methods are not required.

2.5 Efficient Implementation

If the linear kernel is used the matching process is equivalent to 2D filtering, where the 2D filter coefficients are given by $\widetilde{\mathbf{X}} = \text{devec}\,(\widetilde{\mathbf{x}})$. By using the fact that discrete convolution in the spatial domain is equivalent to point-wise multiplication of discrete Fourier spectra in the frequency domain, the cost of searching for a match in an area of interest can be reduced by calculating two *Fast Fourier Transforms* (FFTs), performing a point-wise multiplication, and calculating the Inverse FFT (IFFT) of the result. It is assumed that sufficient zero padding is added to implement linear and not circular convolution. When using the polynomial kernel given by Eq. 5, a single-pass FFT technique is not feasible in general as 2D non-linear filtering is performed. However even when using the polynomial kernel,

$$\left(\sum_{i=1}^{m} C_i \left(1 + \mathbf{x}_i^T \mathbf{x}_{k,l}\right)^d\right),$$

sequential or parallel FFT processing can still be used to process the inner terms $\mathbf{x}_i^T \mathbf{x}_{k,l}$. When m, that is dependent on the number of DITs and UITs, is large, say > 10 which will usually be the case, all $\mathbf{x}_i$ associated with a near zero C_i can be ignored to save complexity. Alternatively a more optimal SVM approach can be followed to only obtain those $\mathbf{x}_i$ labelled as support vectors.

Most of the computational difficulties associated with the kernel given by Eq. 5 can be overcome by implementing the kernel given by Eq. 6. What is important to note in this case is that fast frequency domain algorithms developed for the linear template matching case can be adapted for use with the kernel given by Eq.6 by calculating d sub-templates and executing the linear algorithm d times. Since d is normally chosen as a small value, say 3 or 4, the computational load will still be manageable for most cases. However, performance versus complexity will in the end be dictated by the application.

3 Simulation Results

To test our kernel-based approach, figure 1 (400×400 pixels) was used as a reference image. The objective was to locate the (121×121 pixel) region depicted by figure 3, in figures such as figure 2. Here figure 2 is a rotated and scaled version of figure 3.

Thirty-three DITs were constructed by first rotating the reference template, i.e. figure 3, through -20 -16 -8 -4, 4, 8, 12, 16 and 20 degrees to obtain 9 additional templates. Each rotated template as well as the original template was then scaled by factors 26 and 52 to produce another additional 20 templates. A scale factor of 26 implies that a 26 pixel wide border was removed from the original template after which the cropped template was again made the same size as the original template using bilinear interpolation. Sixteen of the 30 DITs are shown in figure 4.

Sixteen UITs were obtained by extracting 121×121 pixel regions around the neighborhood of the zero rotated and scaled DIT and elsewhere from the reference image. Nine of the UITs are depicted in figure 5. The DITs and UITs were then used as described in section 2.4.

To test the performance of the different kernels, a series of input images was generated by first rotating the reference image by -21, -15, -9, -3, 3, 9, 15 and 21 degrees, and then scaling by factors 0, 8, 16, 24, 32, 40 and 48. For each input image the Euclidean distance (measured in pixels) from the midpoint of region with highest match, to the midpoint of the true location of the region searched for, was calculated. Note that the rotation and scale factors used to produce the input images *do not correspond* the rotation and scale factors used to generate the DITs. The results of the experiment for the *linear* kernel is reported in table 1. Note that ordinary normalized cross correlation is only of value when no rotation or scaling is involved. Both the polynomial kernel given by Eq. 5 with $d = 2$ and Eq. 6 with $d = 3$ yielded improved results. If a distance of more than 10 pixels is taken as a mismatch, then the results obtained using the polynomial kernel without cross terms reported in table 2 show that there is much to gain by using non-linear kernels.

Table 1. Euclidean distance measured in pixels from midpoint of region with highest match to midpoint of true region using the linear kernel.

Rotation and Scale	-21°	-15°	-9°	-3°	3°	9°	15°	21°
0	117	0	0	0	0	11	115	124
8	114	1	1	1	0	10	9	126
16	119	2	1	5	0	9	10	56
24	8	2	1	0	0	31	80	57
32	1	1	1	1	0	1	1	1
40	1	1	1	2	1	1	0	1
48	2	1	1	2	1	1	1	1

Table 2. Euclidean distance measured in pixels from midpoint of region with highest match to midpoint of true region using the polynomial kernel without cross terms.

Rotation and Scale	-21°	-15°	-9°	-3°	3°	9°	15°	21°
0	2	0	0	0	0	0	0	0
8	2	1	1	1	1	1	1	1
16	1	2	1	2	9	9	9	7
24	2	2	1	2	9	9	9	1
32	1	1	1	2	8	1	1	1
40	1	1	1	2	1	1	1	1
48	10	1	2	2	1	1	1	1

4 Conclusion

A novel (non-linear) template matching technique was presented. The proposed framework, based on the derivation of an MNT using interpolator coefficients, DITs and UITs, shows promising results. An efficient implementation strategy using the FFT and IFFT was highlighted. Future work will focus on the improvement of robustness and real-time implementation issues.

References

1. Theodoridis S. and Koutroumbas K. (1999), Pattern Recognition, Academic Press.
2. Ravichandran G. and Trivedi M.M. (1995), Circular-Mellin Features for Texture Segmentation, 2(12)1629-1641.
3. Hall E. (1979), Computer Image Processing and Recognition, Academic Press.
4. Uenohara M. and Kanade T. (1997) Use of the Fourier and Karhunen-Loeve Decomposition for Fast Matching with a Large Set of Templates, IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(8)891-899.
5. Ben-Arie J. and Rao K.R. (1993), A Novel Approach for Template Matching by Nonorthogonal Image Expansion, IEEE Transactions on Circuits and Systems for Video Technology, 3, 71-84.
6. Ben-Arie J. and Rao K.R. (1994), Nonorthogonal Image Expansion Related to Optimal Template Matching in Complex Images, CVGIP: Graphical Models and Image Processing, 56, 149-160.
7. De Figueiredo J.P. (1983), A Generalized Fock Space Framework for Nonlinear System and Signal Analysis, IEEE Transactions on Circuits and Systems, 30(9)637-647.
8. Van Wyk M.A. and Durrani T.S. (2000), A Framework for Multi-scale and hybrid RKHS-based Approximators, IEEE Transactions on Signal Processing, 48(12)3559-3568.
9. Máté L. (1990), Hilbert Space Methods in Science and Engineering, Adam Hilger, New York.
10. Wahba G. (2002), Soft and Hard Classification by Reproducing Kernel Hilbert Space Methods, Technical Report 1067, Department of Statistics, University of Wisconsin.
11. Aronszajn N. (1950), Theory of Reproducing Kernels, Transactions of the American Mathematical Society, (68)337-404.
12. Zyla L.V. and De Figueiredo J.P. (1983), Nonlinear System Identification Based on a Fock Space Framework, SIAM Journal of Control and Optimization, 21(6)931-939.
13. Luenberger D.G. (1969), Optimization by Vector Space Methods, John Wiley and Sons, New York.
14. Schölkopf B and Smola A.J. (2001), Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond (Adaptive Computation and Machine Learning), MIT Press.

Analyzing Weight Distribution of Feedforward Neural Networks and Efficient Weight Initialization*

Jinwook Go, Byungjoon Baek, and Chulhee Lee

Department of Electrical and Electronic Engineering, BERC, Yonsei University
134 Shinchon-Dong, Seodaemoon-Gu, Seoul, Korea
Tel: (82-2) 2123-2779, Fax: (82-2) 312-4584
chulhee@yonsei.ac.kr

Abstract. In this paper, we investigate and analyze the weight distribution of feedforward two-layer neural networks in order to understand and improve the time-consuming training process of neural networks. Generally, it takes a long time to train neural networks. However, when a new problem is presented, neural networks have to be trained again without any benefit from previous training. In order to address this problem, we view training process as finding a solution weight point in a weight space and analyze the distribution of solution weight points in the weight space. Then, we propose a weight initialization method that uses the information on the distribution of the solution weight points. Experimental results show that the proposed weight initialization method provides a better performance than the conventional method that uses a random generator in terms of convergence speed.

1 Introduction

Neural networks have been successfully applied in various fields such as pattern recognition, signal processing, and dynamic modeling. The increasing popularity of neural networks is partly due to their ability to learn and generalize. In particular, neural networks make no prior assumptions about the statistics of input data and can construct complex decision boundaries. The decision boundaries of a neural network are determined by a set of weights and the set of weights can be arranged as a vector, which can be represented as a weight point in the weight space that is defined by the weights. From this perspective, training process of neural networks may be viewed as finding a weight point that is optimal for a given problem (Fig. 1). We call such a weight point a solution weight point for a given problem. In practice, the solution weight point may not be unique. In other words, a weight point in the vicinity of the solution weight point may provide an equivalent performance and such weight points may form a region, which will be called a solution region in this paper (Fig. 1).

* This work was supported in part by Biometrics Engineering Research Center (BERC) at Yonsei University (KOSEF).

If the neural network is trained with a different initial weight point, the resulting weight point can be quite different, though the outputs of the neural network are similar. The behavior of neural networks that produce similar outputs with very different sets of weights leads us to suspect that there may exist several solution regions for a given problem. From this perspective, we explore the possibility to speed up the training process of neural networks by utilizing the information on the distribution of solution regions in the weight space, assuming that such information is available. In other words, by analyzing and accumulating the information on the distribution of solution regions in weight space for various problems, we may initialize neural networks more effectively for a new problem.

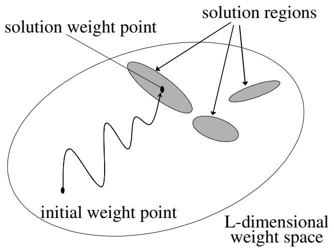


Fig. 1. Solution weight point and solution regions in the weight space.

Efficient weight initialization is one of the most important factors for fast convergence and generalization, and many authors have proposed various weight initialization methods. The simplest and most widely used weight initialization method is a random initialization assuming some probability distributions and some researchers proposed several modified methods to determine the best initialization interval [1-2]. Another initialization approach is to incorporate the known prior knowledge into weight initialization [3]. Generally, it has been shown that the weight initialization is a factor that may influence generalization results. Also, an analysis of the weight distribution may provide a way to reduce the complexity of neural networks [4]. However, the weight distribution in a trained neural network is not well understood, which is due to a strong problem dependency for the weight distribution [5]. In this paper, we analyze the distribution of solution regions for two-pattern class classification problems. From this analysis, we present a new insight into the weight distribution and propose a new weight initialization method that uses the information on the weight distribution.

2 Feedforward Neural Networks and Terminologies

We will briefly discuss the structure of feedforward neural networks that will be used in the experiments. A typical neural network has an input layer, a number of hidden layers, and an output layer. It may also include bias terms. Fig. 2 shows an example of two-layer feedforward neural networks with two outputs. In Fig. 2, let

$X_{in} = (x_1, x_2, \cdots, x_M)^T$ be the input vector and $Y = (y_1, y_2)^T$ the output vector. The bias vector is given by $B = (b_1, b_2)^T$. Without loss of generality, we can set $b_1 = b_2 = 1$. Then we may include the bias term in the input layer as follows:

$$X = (x_1, x_2, \cdots, x_M, 1)^T = (x_1, x_2, \cdots, x_M, x_{M+1})^T \text{ where } x_{M+1} = 1$$

If we assume that there are K neurons in the hidden layer, the weight matrices W_1 and W_2 for the two-pattern class neural network can be represented by

$$W_1 = \begin{bmatrix} w_{1,1}^{hi} & w_{1,2}^{hi} & \cdots & w_{1,M+1}^{hi} \\ w_{2,1}^{hi} & w_{2,2}^{hi} & \cdots & w_{2,M+1}^{hi} \\ \vdots & \vdots & \cdots & \vdots \\ w_{K,1}^{hi} & w_{K,2}^{hi} & \cdots & w_{K,M+1}^{hi} \end{bmatrix}, \quad W_2 = \begin{bmatrix} w_{1,1}^{oh} & w_{1,2}^{oh} & \cdots & w_{1,K+1}^{oh} \\ w_{2,1}^{oh} & w_{2,2}^{oh} & \cdots & w_{2,K+1}^{oh} \end{bmatrix}$$

where $w_{j,i}^{hi}$ is the weight between input neuron i and hidden neuron j and $w_{k,j}^{oh}$ is the weight between hidden neuron j and output neuron k. All the elements of W_1 and W_2 can be arranged as a *weight vector* W . In other words,

$$W = (w_{1,1}^{hi}, w_{1,2}^{hi}, ..., w_{K,M+1}^{hi}, w_{1,1}^{oh}, w_{1,2}^{oh}, ..., w_{2,K+1}^{oh})^T = (w_1, w_2, w_3, ..., w_L)^T$$

where L=((M+1)K+2(K+1)). Then, we may view the weight vector W as a *weight point* in an L-dimensional *weight space* that is defined by all the weights in the neural network. From this perspective, training a neural network may be viewed as finding a weight point that produces a desirable sequence of output vectors for a given sequence of input vectors. In this paper, such a weight point will be called a *solution weight point*. The solution weight point is located within the *solution region*, where the set of solution weight points is distributed densely in the weight space [6].

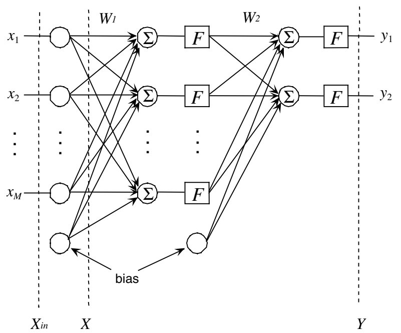


Fig. 2. An example of two-layer feedforward neural networks (two-pattern classes).

3 Weight Distribution in the Weight Space

Even for a simple neural network, the number of weights can be large. In order to make the computation manageable, we chose a neural network that has 2 input neurons, 3 hidden neurons, 2 output neurons, and 2 bias terms. The total number of weights of the neural network is 17 and the training process is equivalent to finding a solution weight point in a 17-dimensional weight space. The backpropagation algorithm is used for training and the decision rule is to select the class corresponding to the output neuron with the largest output [7].

In the following experiment, tests are conducted using generated data whose statistics are estimated from real remotely-sensed data collected from the field spectrometer system (FSS) assuming Gaussian distribution [8]. The 50 pairs of two classes were selected randomly from Table 1 and feature selection is applied to the original 60-dimensional data, producing a 2-dimensional input vector. Each input is preprocessed so that its mean value is zero and the inputs are scaled so that they are in the range of –2 to 2. For each two-pattern classification problem, the network was trained with 1000 different initial weight points until the difference between the classification error and the Bayes' error is smaller than 2%, resulting in 1000 solution weight points. The Bayes' error is estimated using the Gaussian ML classifier. In this paper, we will call the set of solution weight points for each problem a *solution set.*

Table 1. Description of the multi-temporal 15 classes.

Class #	Date	Species	No. of samples
1	770508	Winter Wheat	691
2	780515	Spring Wheat	474
3	771026	Winter Wheat	393
4	771018	Spring Wheat	313
5	780921	Winter Wheat	292
6	780816	Native Grass Pas	212
7	780726	Summer Fallow	200
8	780709	Summer Fallow	190
9	780921	Oats	182
10	780816	Oats	165
11	771018	Oats	161
12	770626	Grain Sorghum	157
13	780515	Summer Fallow	150
14	770920	Spring Wheat	129
15	780602	Barley	112

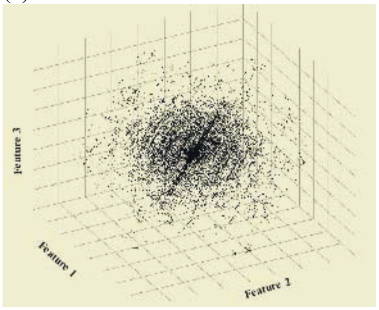
3.1 Eigenvalues and Determinant of Covariance Matrix of the Solution Set

In order to analyze the distribution of the solution sets, we investigate the eigenvalues and determinant of the covariance matrix estimated from the solution sets. Table 2 shows the largest ten eigenvalues (ordered by size) of the covariance matrix estimated from the 50 solution sets (1,000 solution weight points per set, a total of 50,000 solution weight points), as well as the proportions and accumulations of the eigenvalues. As can be seen in Table 2, the largest six eigenvalues account for more than 90% of the total energy. In particular, the largest three eigenvalues are dominant and the rest are small in value. It appears that most variations of solution weight points in weight space occur along a few eigenvectors corresponding to the largest eigenvalues and there is little variation along the other eigenvectors. This indicates that, assuming a normal distribution, the data will be distributed in the shape of an elongated hyperellipsoid with its center at the mean of the solution sets. The principal axes of this hyperellipsoid are given by the eigenvectors of the covariance matrix of the solution sets and the energies contained in these axes are proportional to the corresponding eigenvalues. Fig. 3(a) shows the distribution of the 50 solution sets projected onto the 3 dimensional space spanned by the three eigenvectors corresponding to the largest eigenvalues of the covariance matrix. It can be observed that solution weight points seem to form a single cluster with a dense spot in the center and are distributed closely around the center of the cluster. In addition, as there are little differences among the largest three eigenvalues in Table 2, it seems that the solution weight points are distributed in the shape of a sphere instead of an ellipsoid in the 3 dimensional space spanned by the three eigenvectors corresponding to the largest eigenvalues. Fig. 3(b) shows the histogram of the individual weight values of all the 50 solution sets and the histogram is compared with a normal distribution. The mean and the variance are 0.105 and 25.2, respectively. From Fig. 3(b), the weight values do not seem to be normally distributed, though quite often they are assumed to have a normal distribution [5].

Table 2. Eigenvalues of covariance matrix of 50 solution sets.

	1	2	3	4	5	6	7	8	9	10
Eigenvalue	114.8	113.8	108.2	17.4	17.1	16.2	12.3	12.3	11.6	2.9
Proportion (%)	26.9	26.7	25.3	4.0	4.0	3.8	2.9	2.9	2.7	0.7
Accumulation (%)	26.9	53.6	78.9	82.9	86.9	90.7	93.6	96.5	99.2	99.9

(a)



(b)

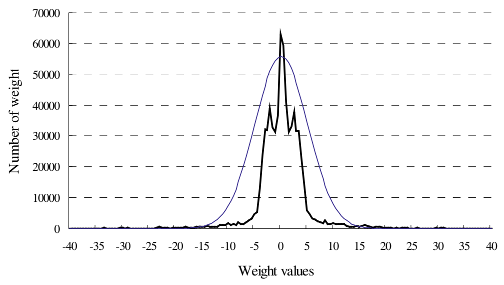


Fig. 3. (a) Three dimensional projection map of the 50 solution sets. (b) histogram of weights and the corresponding normal distribution for the 50 solution sets.

Fig. 4 shows the determinants of the covariance matrices for each of the 50 solution sets. The determinant of a covariance matrix is equal to the product of all eigenvalues and the determinant of a covariance matrix provides the information about the actual volume where the data are distributed. From Fig. 4, it can be said that there are significant differences among the solution sets, indicating that there are significant differences in the actual volumes where the solution sets are distributed.

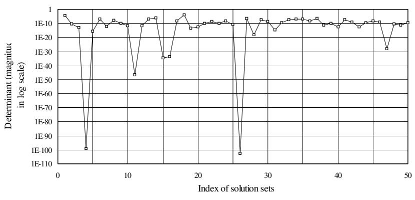


Fig. 4. Determinants of the 50 solution sets.

3.2 Distribution of Solution Region in the Weight Space

A solution set (the set of solution weight points for each problem) is not homogeneous in the sense that the weight points inside the volume containing the solution set do not provide the same classification accuracy. Thus, we need to find homogeneous subsets of a solution set in order to analyze the distribution of solution regions. In order to find homogeneous subsets inside a solution set, we applied clustering to each solution set using the Euclidean distance. It is noted that by rearranging the hidden neurons and corresponding weights, we can produce 6 different configurations of neural networks that produce the exactly same output. This indicates that there exist 6 equivalent weight points in the 17 dimensional weight space for the neural network (2 input neurons, 3 hidden neurons, 2 output neurons, and 2 bias terms). The number of equivalent weight points corresponds to the factorial of the number of hidden neurons. In the analysis of the weight distribution, these equivalent weight points should be treated as the same point, though their locations are different in the weight space. In order to assign equivalent weight points to the same point, we compared the 3 weight values between the bias term in the input layer and the hidden neurons, and rearranged the hidden neurons and corresponding weights such that $w_1 \geq w_2 \geq w_3$, where w_i is the weight value between the bias term in the input layer and the i-th hidden neuron. After this rearrangement of the hidden neurons and corresponding weights, clustering was applied. The resulting subsets are homogeneous in the sense that if we choose any weight point from the volume that contains a subset, the point provides the maximum classification accuracy without training. In this paper, we will call the homogeneous subset a *homogeneous solution region*. Clustering results show that there exist a number of homogeneous solution regions in a solution set and also there is a large variation among the numbers of homogeneous solution regions of the solution sets.

Since clustering was applied to just 1000 solution weight points, the resulting clusters may not be an accurate estimation of the distribution of solution regions. In the next step, we expand each homogeneous solution region to find the maximum volume such that any weight point within the volume provides the maximum classification accuracy. In other words, by expanding a homogeneous solution region, we try to find the maximum volume that contains the homogeneous solution region.

We propose the following expansion procedure for expanding a homogeneous solution region. First, we try to expand a homogeneous solution region along the eigenvector corresponding to the smallest eigenvalue of the covariance matrix estimated from a homogeneous solution region. After doubling the smallest eigenvalue, 1000 weight points are generated assuming the Gaussian distribution for the homogeneous solution region with the increased eigenvalue. Then, we compute the classification accuracy for the generated weight points. If the number of the generated weight points whose classification accuracy is within 2% of the maximum classification accuracy is greater than 995, which corresponds to 99.5% of the total generated points, we determine that the expansion is valid and the modified eigenvalue is retained. Otherwise, the original eigenvalue is retained. The same procedure is repeated for the rest of the eigenvalues in increasing order. Finally, 1000 weight points are generated with all the updated eigenvalues and only weight points whose classification accuracy is within 2% of the maximum classification accuracy are retained. From the new weight points, the mean vector and covariance matrix are estimated. This whole procedure is repeated until the eigenvalues cannot be increased any more.

After the expansion procedure, the volume of a homogeneous solution region increases substantially. Table 3 shows the largest seven eigenvalues of the homogeneous solution region after the expansion procedure. In this case, the volume of the homogeneous solution region increased about 10^{124} times. And most solution weight points are distributed in a few major components, indicating that there are high correlations between adjacent weight values. Table 3 also shows corresponding angles between the eigenvectors before expansion and the corresponding eigenvectors after expansion. The large angles indicate that the shape of the homogeneous solution region also changed substantially.

Table 3. Eigenvalues of the homogeneous solution region and angles between eigenvectors after expansion procedure.

	1	2	3	4	5	6
Eigenvalue	36.7	11.6	3.0	0.8	0.3	0.1
Proportion (%)	69.8	22.2	5.6	1.5	0.7	0.2
Accumulation (%)	69.8	92.0	97.6	99.1	99.8	99.9
Angles between eigenvectors (degree)	8	74	104	78	77	43

4 Weight Initialization Using the Distribution of Solution Regions

Now we explore the possibility to initialize feedforward neural networks with prior knowledge that was obtained by investigating the distribution of solution regions in the weight space. In other words, we wish to use the information on the weight distribution to provide a good initial weight point.

In the previous analysis, we obtained 2581 homogeneous solution regions for the 50 pattern classification problems. We computed the mean vector and the covariance matrix of each homogeneous solution region. From these statistics, we generated 300 weight points for each homogeneous solution region assuming the Gaussian distribution. Thus, the total number of solution weight points is 774,300, to which we applied

K-means clustering in order to divide the solution weight points into 100 clusters whose mean vectors may be used as initial weight points for a new problem.

In order to examine whether this information of weight distribution is useful for selecting a good initial weight point, we initialized the neural network with the cluster means for 780 new classification problems that were not used to obtain the 100 clusters. The new classification problems consist of 780 pairs of two classes from the combination of 40 classes. Fig. 5(a) shows the distribution of the 40 classes and Fig. 5(b) shows the distribution of the 15 classes shown in Table 1.

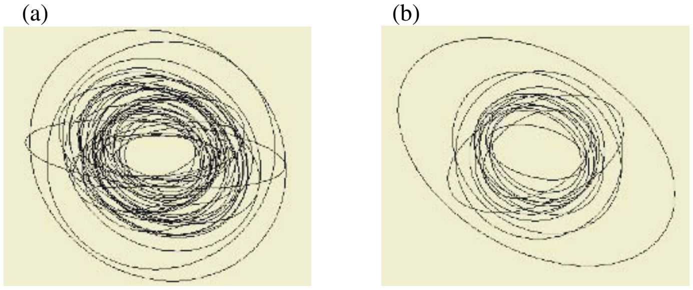


Fig. 5. (a) Distribution of 40 classes used for the 780 two-class problems, (b) distribution of 15 classes described in Table 1.

The mean vectors of the 100 clusters were used as initial weight points and then we selected the weight point that provides the best classification accuracy. The classification accuracy was estimated using 25 randomly selected samples from 500 training samples of each class, which correspond to 5% of the total training samples. Thus, the extra computation time is relatively minor. Random initialization assuming a uniform distribution over the interval from –0.5 to 0.5 was also tested to compare the performance of the proposed initialization method.

Table 4 shows the differences between the classification accuracies that were obtained using the mean vectors of the clusters as initial weights and the classification accuracies that were obtained using random initialization for the 780 new problems. The classification accuracies were obtained without any training. In 82.2% of the 780 problems, the proposed initialization method achieves higher classification accuracy than the random initialization method. It appears that the solution regions retain some useful information to solve new problems.

Fig. 6 shows the average classification accuracies of the 780 problems. As can be seen, the proposed method significantly outperforms the random weight initialization. After one iteration, the average classification accuracy obtained using the mean vectors of the clusters as initial weights was 81.58% for training data. On the other hand, the average classification accuracy obtained using randomly selected initial weights was 71.82%. The experiments show that the knowledge on the solution distribution can provide helpful information for a new problem. However, for a few problems, using the mean vectors of the clusters as initial weights was ineffective.

Table 4. The classification accuracy differences between the initial weights selected from cluster means and the random initial weights without any training.

Difference (D)	No. of problems	Proportion (%)	Accumulation (%)
D≥30%	24	3.08	3.08
30%>D≥20%	103	13.21	16.28
20%>D≥10%	227	29.10	45.38
10%>D>0%	311	39.87	82.18
0%>D≥-5%	92	11.79	97.05
-5%>D≥-10%	20	2.56	99.62
-10%>D≥-15%	3	0.38	100.00

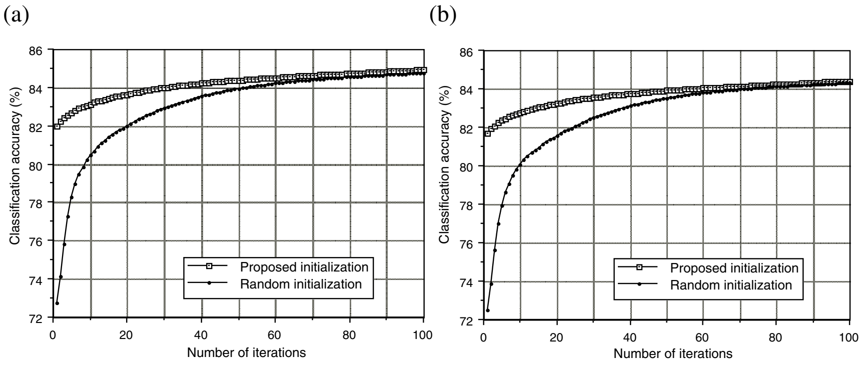


Fig. 6. Performance comparison between the proposed initialization using the information on the solution distribution and the random initialization (averages of the 780 experiments). (a) training data, (b) test data.

5 Conclusion

In this paper, by analyzing weight distribution of neural networks, we explored the possibility to speed up training process. First, we obtained 50,000 solution weight points for 50 two-pattern-class classification problems and found a number of homogeneous solution regions in the weight space by applying clustering to the solution weight points. Then, the homogeneous solution regions were expanded in order to obtain the maximum volume for each homogeneous solution region. By analyzing distribution of the homogeneous solution regions, we found that most weights are distributed along a few major directions, indicating that there are very high correlations between weight values of the homogeneous solution regions. Finally, we explored the possibility to use this prior information on the distribution of weights for efficient weight initialization. Experimental results indicate that the weight analysis may provide useful information on selecting initial weights. As the knowledge on the weight distribution accumulates, it would be possible to select good initial weights for a new problem.

References

1. L. F. A. Wessels and E. Barnard: Avoiding false local minima by proper initialization of connections. IEEE Trans. Neural Networks, Vol. 3. (1992) 899-905
2. G. P. Drago and S. Ridella: Statistically controlled activation weight initialization (SCAWI). IEEE Trans. Neural Networks, Vol. 3. (1992) 627-631
3. Z. Chen, T. Feng, and Z. Houkes: Incorporating a priori knowledge into initialized weights for neural classifier. Proc. Int. Joint Conf. Neural Networks, Vol. 2. (2000) 291-296
4. S. J. Nowlan and G. E. Hinton: Simplifying neural networks by soft weight-sharing. Neural Computation, Vol. 4, No. 4. (1992) 473-493
5. I. Bellido and E. Fiesler: Do backpropagation trained neural networks have normal weight distributions? Proc. Int. Conf. Artificial Neural Networks. (1993) 772-775
6. J. A. Richards: Remote Sensing Digital Image Analysis. Springer-Verlag. (1993)
7. R. P. Lippmann: An introduction to computing with neural nets. IEEE ASSP Magazine, Vol. 4. (1987) 4-22
8. L. L. Biehl and et. al.: A crops and soils data base for scene radiation research. Proc. Machine Process. of Remotely Sensed Data Symp., West Lafayette, Indiana. (1982) 169-177

One-Class Support Vector Machines with a Conformal Kernel. A Case Study in Handling Class Imbalance

Gilles Cohen[1,2], Mélanie Hilario[2], and Christian Pellegrini[2]

[1] Medical Informatics Service, University Hospital of Geneva, 1211 Geneva, Switzerland
Gilles.Cohen@sim.hcuge.ch

[2] Artificial Intelligence Laboratory, University of Geneva, 1211 Geneva, Switzerland
Melanie.Hilario@cui.unige.ch

Abstract. Class imbalance is a widespread problem in many classification tasks such as medical diagnosis and text categorization. To overcome this problem, we investigate one-class SVMs which can be trained to differentiate two classes on the basis of examples from a single class. We propose an improvement of one-class SVMs via a conformal kernel transformation as described in the context of binary SVM classifiers by [2, 3]. We tested this improved one-class SVM on a health care problem that involves discriminating 11% nosocomially infected patients from 89% non infected patients. The results obtained are encouraging: compared with three other SVM-based approaches to coping with class imbalance, one-class SVMs achieved the highest sensitivity recorded so far on the nosocomial infection dataset. However, the price to pay is a concomitant decrease specificity, and it is for domain experts to decide the proportion of false positive cases they are willing to accept in order to ensure treatment of all infected patients.

1 The Imbalanced Data Problem

Data imbalance is a crucial problem in applications where the goal is to maximize recognition of the minority class, as is typically the case in medical diagnosis. The issue of class imbalance has been actively investigated and remains widely open; it is handled in a number of ways [14], including: oversampling the minority class, building cost-sensitive classifiers [10] that assign higher cost to misclassifications of the minority class, stratified sampling on the training instances to balance the class distribution [15] and rule-based methods that attempt to learn high confidence rules for the minority class [1]. In this paper we investigate another way of biasing the inductive process to boost sensitivity (i.e., capacity to recognize positives). This approach, based on one-class support vector machines (SVMs) with a conformal kernel, is described in Section 2 and its application to nosocomial infection detection is discussed in Section 3. Experiments conducted to assess this approach as well as results are described in Section 4.

2 Using Conformal Kernels in One-Class SVMs

2.1 One-Class Classification

While the majority of classification problems consist in discriminating between two or more classes, some other problems are best formulated as *one-class* or *novelty detection* problems. In a probabilistic sense, one-class classification is equivalent to deciding whether an unknown case has been produced by the distribution underlying the training set of normal cases. In one-class classification the classifier is trained exclusively on cases from the majority class and never sees those from the minority class. It must estimate the boundary that separates two classes and minimize misclassification based only on data lying on one side of it.

The one-class approach is particularly attractive in situations where cases from one class are expensive or difficult to obtain for model construction (i.e. imbalanced datasets). The most straigthforward method for detecting novel or abnormal cases is to estimate the density of the training data and to set a threshold on this density [4, 17]. However, it is much simpler to model the support of a data distribution, i.e., to create a binary-valued function which is positive in those regions of input space containing most of the data and negative elsewhere; the following section describes this approach.

2.2 One-Class Support Vector Machines

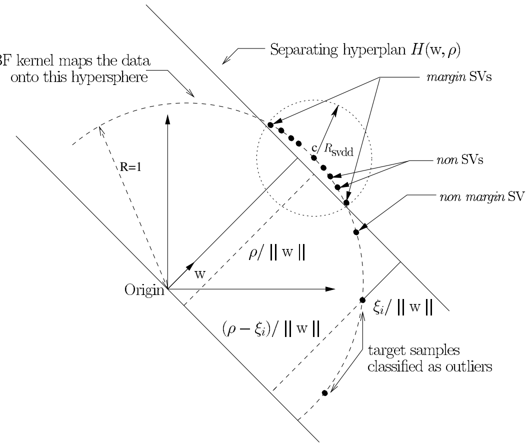
Support vector machines [18, 8] are learning machines based on the *Structural Risk Minimization principle* (SRM) from statistical learning theory. They were originally introduced to solve the two-class pattern recognition problem. An adaptation of the SVM methodology in order to handle classification problems using data from only one class has been proposed by [16]. This adapted method, termed one-class SVM, identifies "abnormal" cases amongst the known cases and assumes them to belong to the complement of the "normal" cases. Schölkopf et al. formulate the one-class SVM approach as follows:
Consider a training set $\mathcal{X} = \{\mathbf{x}_i\},\ i = 1, \ldots, n,\ \mathbf{x}_i \in \mathbb{R}^d$, and suppose its instances are distributed according to some unknown underlying probability distribution P. We want to know if a test example $\mathbf{x}$ is distributed according to P or not. This can be done by determining a region R of the input space X such that the probability that a test point drawn from P lies outside of R is bounded by some a priori specified value $\nu \in (0, 1)$. This problem is solved by estimating a decision function f which is positive on R and negative elsewhere.

$$f(\mathbf{x}) > 0 \text{ if } \mathbf{x} \in R \text{ and } f(\mathbf{x}) < 0 \text{ if } \mathbf{x} \in R \tag{1}$$

A non linear function $\Phi : \mathcal{X} \rightarrow \mathcal{F}$ maps vector $\mathbf{x}$ from the input vector space $\mathcal{X}$ endowed with an inner product to a Hilbert space $\mathcal{F}$ termed feature space. In this new space, the training vectors follow an underlying distribution P', and the problem is to determine a region R' of $\mathcal{F}$ that captures most of this probability mass distribution. In other words the region R' corresponds to the part of the feature space where most of the data vectors lie. To separate as many as possible of the mapped vectors from the origin in feature space $\mathcal{F}$ we construct a hyperplane $H(\mathbf{w}, \rho)$ in a feature space $\mathcal{F}$ defined by

$$H(\mathbf{w}, \rho) : \langle \mathbf{x}, \Phi(\mathbf{x}) \rangle - \rho \tag{2}$$

with $\mathbf{w}$ the weight vector and ρ the offset, as illustrated in Fig 1.



Separating hyperplan $H(\mathrm{w}, \rho)$
The RBF kernel maps the data onto this hypersphere
margin SVs
c
R_{svdd}
R=1
non SVs
non margin SV
$\rho / \|\mathrm{w}\|$
w
Origin
$\xi_i / \|\mathrm{w}\|$
$(\rho - \xi_i) / \|\mathrm{w}\|$
target samples classified as outliers

Fig. 1. Schematic 2D overview of a one-class SVM classifier with an RBF kernel. In the feature space, the vectors are located on a hypersphere. The hyperplane $H(\mathbf{w}, \rho)$ separates the training vectors from the rest of the surface of the hypersphere.

The maximum margin from the origin is found by solving the following quadratic optimization problem.

$$Minimize \quad \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle - \frac{1}{\nu n}\sum_{i=1}^{n} \xi_i \tag{3}$$

$$subject\ to \quad \langle \mathbf{w}, \Phi(\mathbf{w}) \rangle \geq \rho - \xi_i, \quad \forall_i\ \xi_i \geq 0 \tag{4}$$

where ξ_i are so-called slack variables that penalize the objective function but allow some of the points to be on the wrong side of the hyperplane, i.e. located between the origin and $H(\mathbf{w}, \rho)$ as depicted in Fig.1. $\nu \in (0, 1)$ is a parameter that controls the trade off between maximizing the distance from the origin and containing most of the data in the region created by the hyperplane. It is proved in [16] that ν is an upper bound on the fraction of outliers i.e. training errors, and also a lower bound on the fraction of support vectors.

Let $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ be n non negative Lagrange multipliers associated with the constraints, the solution to the problem is equivalent to the solution of the Wolfe dual [11] problem:

$$Maximize \quad \frac{1}{2}\alpha_i\alpha_j\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_i)\rangle \tag{5}$$

$$subject\ to \quad 0 \leq \alpha_i \leq \frac{1}{\nu n}, \quad \sum_{i=1}^{n}\alpha_i = 1 \tag{6}$$

the solution for $\mathbf{w}$ is $\sum_{i=1}^{n}\Phi(\mathbf{x}_i)$ where $0 \leq \alpha_i \leq \frac{1}{\nu n}$ and the corresponding decision function is :

$$f(\mathbf{x}_j) = sgn\left(\sum_{i=1}^{n}\alpha_i\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle - \rho\right) \tag{7}$$

All training data vectors $\mathbf{x}_i$ for which $f(\mathbf{x}_i) \leq 0$ are called support vectors (SVs); these are the only vectors for which $\alpha_i \neq 0$. SVs are divided in two sets : the *margin* SVs, for which $f(\mathbf{x}_i) = 0$, and the *non-margin* SVs, for which $f(\mathbf{x}_i) < 0$. Notice that in (5) only inner products between data are considered; for certain particular maps $\mathcal{F}$, there is no need to actually compute $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$; the inner product can be derived directly from $\mathbf{x}_i$ and $\mathbf{x}_j$ by means of the so-called "kernel trick". A kernel K is a symmetric function that fulfills Mercer's [18, 9] conditions. The main property of functions satisfying these conditions is that they implicitly define a mapping from $\mathcal{X}$ to a Hilbert space $\mathcal{F}$ such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle \tag{8}$$

and thus can be used in algorithms using inner products. Accordingly, the hyperplane (2) in feature space $\mathcal{F}$ becomes a non linear function in the input space $\mathcal{X}$.

$$f(\mathbf{x}) = sgn(\sum_{i=1}^{n}\alpha_i K(\mathbf{x}_i, \mathbf{x}) - \rho) \tag{9}$$

There are many admissible choices for the kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$. The most widely used in one-class SVMs is the Gaussian Radial Basis Function RBF kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2} \tag{10}$$

where σ is a parameter which controls the width of the kernel function around $\mathbf{x}_i$. Since $\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_i)\rangle = K(\mathbf{x}_i, \mathbf{x}_i) = \exp^0 = 1$ with an RBF kernel, the training data in $\mathcal{F}$ lie on a region on the surface of a hypersphere centered at the origin of $\mathcal{X}$ with radius 1 as depicted in Fig. 1. Finally one has the decision function of Eq. (9) with $\rho = \sum_{i=1}^{n}\alpha_i K(\mathbf{x}_i, \mathbf{x}_j)$ for any $\mathbf{x}_j$ such that α_i satisfies $0 < \alpha_j < \frac{1}{\nu n}$ which defines the contour of the region R in input space by cutting the hypersurface defined by the weighted addition of SVM kernels at a given altitude ρ.

2.3 Accuracy Improvement

The accuracy of the one-class classifier can be improved by enhancing the resolution in the support vector region boundaries. One way to reach this goal is via a *conformal transformation*[1] of the kernel. This approach has been described in the context of binary SVMs classifier by [2, 3], but the basic principle is also applicable to the one-class SVM. From a geometrical point of view the mapped data lie on a surface S in $\mathcal{F}$ with the same dimensionality as the input space $\mathcal{X}$ [6]. In the case of an RBF kernel function, the associated surface S in $\mathcal{F}$ can be considered as a Riemannian manifold [5] and a Riemannian metric thereby induced and expressed in the closed form in terms of the kernel [6, 2, 3]. A Riemannian metric, also called tensor, is a function which computes the intrinsic distance measured along the surface S itself between any two points lying on it. Its components can be viewed as multiplication factors which must be placed in front of the differential displacements dx_i in $\mathcal{X}$ to compute the distance ds of an element dz in $\mathcal{F}$ in a generalized Pythagorean theorem,

$$ds^2 = \sum_{i,j} g_{ij} dx_i dx_j \tag{11}$$

where g_{ij} is the induced metric, and the surface S is parametrized by the x_i. Let $\mathbf{x}$ be a point in $\mathcal{X}$ and $\mathbf{z}$ it corresponding mapping by Φ in $\mathcal{F}$. Letting $d\mathbf{x}$ represent a small but finite displacement, we have

$$\begin{aligned} ds^2 &= \|d\mathbf{z}\|^2 = \|\Phi(\mathbf{x}+d\mathbf{x}) - \Phi(\mathbf{x})\|^2 \\ &= K(\mathbf{x}+d\mathbf{x},\mathbf{x}+d\mathbf{x}) - 2K(\mathbf{x},\mathbf{x}+d\mathbf{x}) + K(\mathbf{x},\mathbf{x}) \\ &= \sum_{i,j} \left(\frac{\partial^2 K(\mathbf{x},\mathbf{y})}{\partial x_i \partial y_j} \right)_{\mathbf{y}=\mathbf{x}} dx_i dx_j \end{aligned}$$

From Eq. 11 we see that the Riemannian metric induced on S can be defined as

$$g_{ij} = \left(\frac{\partial^2 K(\mathbf{x},\mathbf{z})}{\partial x_i \partial y_j} \right)_{\mathbf{y}=\mathbf{x}} \tag{12}$$

Note how a local area around $\mathbf{x}$ in $\mathcal{X}$ is magnified in $\mathcal{F}$ under the mapping $\Phi(\mathbf{x})$. The principle of conformal mapping is to increase the metric $g_{ij}(\mathbf{x})$ around the boundary and to reduce it everywhere else. To do this the non linear mapping Φ is modified in such a way that $g_{ij}(\mathbf{x})$ is enlarged around the boundary. This can be done by introducing a conformal transformation of the kernel [2, 3],

$$\tilde{K}(\mathbf{x},\mathbf{y}) = c(\mathbf{x})c(\mathbf{y})K(\mathbf{x},\mathbf{y}) \tag{13}$$

where $c(\mathbf{x})$ is a defined positive function. The modified kernel satisfies the Mercer positivity condition [9]. From Eq. 12, we obtain the new Riemannian metric $\tilde{g}_{ij}$

$$\tilde{g}_{ij} = c(\mathbf{x})^2 g_{ij}(\mathbf{x}) + c_i(\mathbf{x})c_j(\mathbf{x}) + 2c_i(\mathbf{x})c(\mathbf{x})K_i(\mathbf{x},\mathbf{x}) \tag{14}$$

where $K_i(\mathbf{x},\mathbf{x}) = \partial K(\mathbf{x},\mathbf{y})/\partial x_i|_{\mathbf{x}=\mathbf{y}}$ and $c_i(\mathbf{x}) = \partial c(\mathbf{x})/\partial x_i$. For the Gaussian RBF kernel the last term is zero.

[1] A transformation that preserves the magnitude and orientation of the angle between any two curves intersecting at a given point is *conformal* at that point. A transformation is called conformal in a domain D if it is conformal at every point in D.

To expand the spatial resolution in the margin of a support vector $c(\mathbf{x})$ should be chosen in a way such that the metric $\tilde{g}_{ij}(\mathbf{x})$ has greater values around the decision boundary. However, in practice, we do not know where the boundary is, so an initial estimate is done by prior training of one-class SVMs.
A possible conformal transformal transformation $c(\mathbf{x})$ is

$$c(\mathbf{x}) = \sum_{i \in \hat{SV}} \hat{\alpha}_i e^{\frac{-\|\mathbf{x}-\hat{\mathbf{x}_i}\|^2}{2\tau^2}} \tag{15}$$

where $\hat{SV}$ is the set of margin support vectors, $\hat{\alpha_i}$ is a positive number representing the contribution of the ith support vector, $\mathbf{x}_i$ is the ith support vector and τ is a free parameter; $\hat{}$ refers to a one-class SVM previously trained on the same dataset.

3 Application to Nosocomial Infection Detection

We tested the performance of one-class SVMs with a conformal kernel on a medical problem, the detection of nosomial infections. A nosocomial infection (from the Greek word *nosokomeion* for hospital) is an infection that develops during hospitalization whereas it was not present nor incubating at the time of the admission. Usually, a disease is considered a nosocomial infection if it develops 48 hours after admission.

The University Hospital of Geneva (HUG) has been performing yearly prevalence studies to detect and monitor nosocomial infections since 1994 [13]. Their methodology is as follows: the investigators visit every ward of the HUG over a period of approximately three weeks. All patients hospitalized for 48 hours or more at the time of the study are included. Medical records, kardex, X-ray and microbiology reports are reviewed, and additional information is eventually obtained by interviewing nurses or physicians in charge. Collected variables include demographic characteristics, admission date, admission diagnosis, comorbidities, McCabe score, type of admission, provenance, hospitalization ward, functional status, previous surgery, previous intensive care unit (ICU) stay, exposure to antibiotics, antacid and immunosuppressive drugs and invasive devices, laboratory values, temperature, date and site of infection, fulfilled criteria for infection.

The resulting dataset consisted of 688 patient records and 83 variables. With the help of hospital experts on nosocomial infections, we filtered out spurious records as well as irrelevant and redundant variables, reducing the data to 683 cases and 49 variables. The major difficulty inherent in the data (as in many medical diagnostic applications) is the highly skewed class distribution. Out of 683 patients, only 75 (11% of the total) were infected and 608 were not. This application was thus an excellent testbed for assessing the efficacy of one-class SVMs with a conformal kernel in the presence of data imbalance.

4 Experimentation

4.1 Evaluation Strategy

The experimental goal was to assess the impact of a conformal kernel on the the ability of one-class SVMs to cope with imbalanced datasets. To train one-class SVM classifiers we used an RBF kernel (Eq. (10)) and experimented with different values for the parameters ν and σ. Generalization error was estimated using 5-fold cross-validation (10-fold cross-validation would have resulted in an extremely small number of infected test cases per fold). The complete dataset was randomly partitioned into five subsets. On each iteration, one subset (comprising 20% of the data samples) was held out as a test set and the remaining four (80% of the data) were concatenated into a training set. The training sets consisted only of non infected patients whereas the test sets contained both infected and non infected patients according to the original class distribution. Error rates estimated on the test sets were then averaged over the five iterations. The following strategy was followed for conformal mapping:

1. Train one-class SVM with the primary RBF kernel K to get the SVs. Then change the kernel K according to Eq. 13,15.
2. Train one-class SVM with the modified kernel $\tilde{K}$.
3. Apply the above two steps (1. and 2.) until the best performances is attained.

For Eq. 15 we took $\tau = \sigma/\sqrt{n}$ which is the optimal value reported in [3].

4.2 Results

Table 1 summarizes performance results for one-class SVMs. It shows the best results obtained by training classifiers using different parameter configurations on non infected cases only. The last three columns show results based on three performance metrics. Accuracy is the percentage of correctly classified cases, sensitivity is the number of true positives over all positive cases, while specificity is the number of true negatives over all negative cases. Clearly, for both RBF and conformal kernels, highest sensitivity is attained when σ is very small: the system

Table 1. Performance of one-class SVMs for different parameter settings using (1) an RBF Gaussian kernel and (2) a conformal kernel.

One-class SVM	ν	σ	Accuracy %	Sensitivity %	Specificity %
	0.05	$10^{-\bullet}$	74.6	92.6	43.73
	0.05	0.10	75.49	80.60	65.60
RBF kernel	0.2	$10^{-\bullet}$	75.69	79.28	68.27
	0.2	0.10	74.36	74.67	72.27
	0.05	$10^{-\bullet}$	75.6	93.4	43.15
	0.05	0.1	76.65	82.35	64.1
Conformal kernel	0.2	$10^{-\bullet}$	77.3	81.1	69.7
	0.2	0.06	76.25	79.1	69.2

Table 2. Best performance of four SVM-based approaches to class imbalance.

SVM Classifier	Accuracy	Sensitivity	Specificity
Binary with symm. margin	89.6%	50.6%	94.4%
Binary with asymm. margin	74.4%	92%	72.2%
One-class with RBF kernel	74.6%	92.6%	43.73%
One-class with conformal kernel	75.6%	93.4%	43.15%

puts a Gaussian of narrow width around each data point and hence most of the infected test cases are correctly recognized as abnormal. The price is that many non infected cases are equally labelled abnormal, thus yielding low specificity. Larger values of σ in the RBF kernel are required to achieve tight approximations for the region R (non infected patients). Therefore the kernel parameter σ is crucial in determining the balance between normality and abnormality as there is no explicit penalty for false positive in one-class classification, contrary to the two class formulation [7]. Since the goal of this study is to identify infected cases, the solution retained is that which achieves maximal sensitivity.

In a previous study on the same nosocomial dataset [7], we investigated a support vector algorithm in which asymmetrical margins are tuned to improve recognition of rare positive cases. Table 2 compares the best performance measures obtained in previous and the latest experiments. Classical binary SVMs with a symmetrical margin attain a baseline sensitivity 50.6%; with the use of asymmetrical margins, sensitivity jumps to 92%. This is further improved by one-class SVMs with an RBF kernel (92.6%) and with a conformal kernel (93.4%). Note however that this progress in sensitivity comes at the cost of a corresponding decrease in specificity.

5 Conclusion and Future Work

We proposed one-class SVMs with a conformal kernel as a novel way of handling class imbalance in classification tasks. We showed that this approach achieves higher sensitivity than all SVM models previously applied to this problem. However, the price paid in terms of loss in specificity is quite exhorbitant, and domain experts must decide if the high recognition rate is worth the cost of treating false positive cases. From this point of view, asymmetrical-margin SVMs might prove preferable in that they maintain a more reasonable sensitivity-specificity tradeoff. In the near future, we intend to prospectively validate the classification model obtained by performing in parallel a standard prevalence survey. Overall we feel that one-class SVMs with a conformal kernel are a promising approach to the detection of nosocomial infections and can become a reliable component of an infection control system.

Acknowledgements

The authors thank members of the University of Geneva Hospitals – Profs. A. Geissbuhler and D. Pittet for their support and Drs. H. Sax and S. Hugonnet for the nosocomial infection data.

References

1. K. Ali, S. Manganaris, and R. Srikant. Partial classification using association rules. In *Proc. 3rd International Conference on Knowledge Discovery in Databases and Data Mining*, 1997.
2. S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999.
3. S. Amari and S. Wu. An information-geometrical method for improving the performance of support vector machine classifiers. In *ICANN99*, pages 85–90, 1999.
4. C. Bishop. Novelty detection and neural network validation. *IEEE Proceedings on Vision, Image and Signal Processing*, 141(4):217–222, 1994.
5. W.M. Boothby. *An introduction to differential manifolds and Riemannian geometry.* Academic Press, Orlando, 1986.
6. C. Burges. Geometry and invariance in kernel based methods. In MIT Press, editor, *Adv. in kernel methods: Support vector learning*, 1999.
7. G. Cohen, M. Hilario, H. Sax, and S. Hugonnet. Asymmetrical margin approach to surveillance of nosocomial infections using support vector classification. In *Intelligent Data Analysis in Medicine and Pharmacology*, 2003.
8. C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20(3):273–297, September 1995.
9. N. Cristianini and Taylor J.S. *An Introduction to Support Vector Machines.* Cambridge University Press, 2000.
10. P. Domingos. A general method for making classifiers cost-sensitive. In *Proc. 5th International Conference on Knowledge Discovery and Data Mining*, pages 155–164, 1999.
11. R. Fletcher. *Practical Methods of Optimization.* John Wiley and Sons, 1987.
12. G. G. French, A. F. Cheng, S. L. Wong, and S. Donnan. Repeated prevalence surveys for monitoring effectiveness of hospital infection control. *Lancet*, 2:1021–23, 1983.
13. S. Harbarth, Ch. Ruef, P. Francioli, A. Widmer, D. Pittet, and Swiss-Noso Network. Nosocomial infections in Swiss university hospitals: a multi-centre survey and review of the published experience. *Schweiz Med Wochenschr*, 129:1521–28, 1999.
14. N. Japkowicz. The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5), 2002.
15. M. Kubat and S. Matwin. Addressing the curse of imbalanced data sets: One-sided sampling. In *Procsȯf the Fourteenth International Conference on Machine Learning*, pages 179–186, 1997.
16. B. Scholkopf, R.C. Williamson, A.J. Smola, J. Shawe-Taylor, and J.Platt. Estimating the support of a high-dimensional distribution. In *Neural Computation*, volume 13, pages 1443–1471. MIT Press, 1999.
17. L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady. Novelty detection for the identification of masses in mammograms. In *Proceedings of the 4th IEE International Conference on Artificial Neural Networks (ICANN'95)*, pages 442 – 447, 1995.
18. V. Vapnik. *Statistical Learning Theory.* Wiley, 1998.

Automatic Labeling of Sports Video Using Umpire Gesture Recognition

Graeme S. Chambers, Svetha Venkatesh, and Geoff A.W. West

Department of Computing, Curtin University of Technology, Perth, Western Australia
{chambegs,svetha,geoff}@cs.curtin.edu.au

Abstract. We present results on an extension to our approach for automatic sports video annotation. Sports video is augmented with accelerometer data from wrist bands worn by umpires in the game. We solve the problem of automatic segmentation and robust gesture classification using a hierarchical hidden Markov model in conjunction with a filler model. The hierarchical model allows us to consider gestures at different levels of abstraction and the filler model allows us to handle extraneous umpire movements. Results are presented for labeling video for a game of Cricket.

1 Introduction

Content characterisation and labeling of video sequences have become significant research areas with the goal of automatically describing arbitrary video. Extracting high level semantics from video data is a difficult problem. To begin annotation, there must be knowledge of the domain of video to be processed and some limitations imposed on the types of scenes that can be analysed. Rather than increasingly confining the domain of video to be processed, we consider the problem on a broader scope by introducing other sensors. If other sensor data is available, the difficulties associated with image processing can be avoided. In particular, we have accelerometer sensors in the form of wrist bands which can be worn by key actors in the video. In sports video, for example, umpires in the game can wear the sensors and have their movements recorded and analysed throughout the game.

Sports officials perform many gestures which are indicative of what is going on in the game. Their gestures can provide something meaningful about a player, a team, or the entire game. If the gestures of these officials are able to be recognised, meaningful information can be derived. We refer to a gesture as an intentional action whereby part of the body is moved in a predefined way to indicate a specific event. Detecting these events enables automatic generation of highlights and more importantly, rich, contextual labeling of video. To solve this problem we need to address the issues of segmenting continuous gesture data and performing robust gesture classification.

The novelty of the work presented here is the way in which we segment and classify candidate gestures from the continuous stream. We propose the use of the Hierarchical Hidden Markov Model (HHMM) in conjunction with a filler model for segmenting and classifying gestures at differing levels of detail. The HHMM allows us to consider gestures as sequences of sub-gestures, possibly reusing the sub-gesture parts for gestures.

The filler model allows us to potentially ignore unknown movements and automatically segment and classify gestures simultaneously from a continuous stream.

2 Background

The types of gestures we are interested in detecting are intentional movements by officials which indicate something about the game. In many sports, umpires move to an area where other officials can see them, perform the gesture that represents the event in the game, then return to officiating. Obviously these gestures can provide a first level for semantic labeling of the accompanying video. In the area of sports video, several attempts have been made at meaningful labeling, including specific sports [1, 2] and automatic generation of highlights [3]. These however, do not provide suitable reusable frameworks for recognising events in various types of sports. All assume specific knowledge of the domain and have heuristics for the sport being processed.

Gesture recognition from video sequences has also been limited in scope, primarily focused on individual gestures in constrained environments. Hand sign language recognition [4] and learning of T'ai Chi movements [5] are two such examples. Attempting to recognise real world gestures places much more demand on the image segmentation techniques and generally results in dramatically increased failure rates. Lighting conditions and occlusion are two significant challenges for creating robust image segmentation methods. Using sensors for gesture recognition has the advantage that movement information is provided directly.

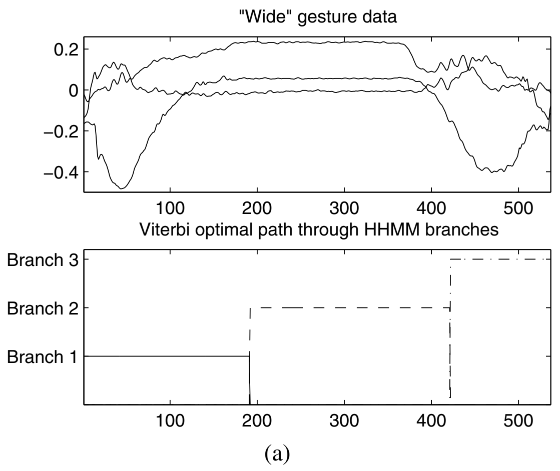
Gesture recognition using sensors has been studied for some time. Several systems using complex arrays of devices have been deployed for real world use [6] and others restricting themselves to constrained environments for human computer interaction [7], [8].

In a previous approach [9], we considered the problem of gesture recognition from a continuous stream, but used a standard hidden Markov model and considered any region of movement in the stream as a candidate gesture. The approach was unable to deal with unknown movements and was sensitive to the threshold for gesture duration. In the following, we describe how these problems were overcome.

3 Hierarchical Modeling of Gestures

Gestures can be considered to exist at multiple levels in a hierarchy, where simple movements are grouped into more complex movements and complex movements are grouped into ordered sequences. The advantage of hierarchical modeling is this temporal decomposition of gestures. The classification stage becomes more manageable for increasingly complex gestures as the dynamics of the gesture are explicitly encoded. Not only does grouping allow for segmenting a gesture into its subparts, hierarchical modeling allows new gestures to be learnt on-line by reusing subparts from already known gestures.

In previous work [10], we proposed an extension to the hidden Markov model for recognising hierarchical gestures. Our extension was applied to recognising Kung–Fu gestures where each individual move was considered a sub-gesture. The extension was capable of representing the hierarchy of gestures, however, higher levels in the model had to be hand crafted. Another recent extension to the hidden Markov model, the



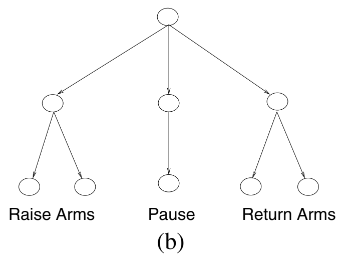


Fig. 1. "Wide" gesture data with corresponding viterbi path (a) and its HHMM representation (b).

HHMM, has been proposed [11]. The HHMM has a distinct advantage over our previous approach in that the model parameters can be estimated using a modified Expectation Maximisation algorithm. New gesture sequences can then be learnt on-line using the HHMM. For example, Figure 1 shows accelerometer data for the right arm and the corresponding optimal state path at the top level of the gesture HHMM. The movement corresponds to a "Wide" umpire gesture from the game of cricket whereby the umpire lifts both arms to the side until they are horizontal, holds that position for a short time so other officials can see the movement, then returns the arms back to the body. The HHMM allows us to explicitly model these three components of the gesture by considering the complete gesture as a sequence of three parts. The HHMM subtrees are trained independently using standard hidden Markov model techniques, then the parameters are merged into the HHMM structure. Instances of the complete gesture (combined subparts) are then used to train the higher levels of the HHMM. The Gaussian mixture components at the lowest level are not re-estimated.

Figure 1(b) shows the hierarchy used for the gesture, where the left-most branch is the raising of the arms, the centre branch is the pause, and the right-most branch is returning the arms to the body. The optimal (Viterbi) path through the top level in the HHMM illustrated in the lower portion of Figure 1(a) shows how the HHMM can automatically segment gestures at different levels. Time 1 through to 192 is recognised as the "raise arms" portion of the gesture, then from 193 to 422 is recognised as the "pause" portion of the gesture, then finally at time 423 to 537 is the "return arms" portion of the gesture.

4 Continuous Gesture Recognition

Any real application of gesture recognition has to be able to deal with segmenting gestures from a continuous stream, then classifying these candidate gestures to give an appropriate label. This process is similar to speech recognition systems whereby spoken words are detected and recognised by the system. An approach to detecting unknown words in speech is applied in our gesture recognition system. A "filler" (or "null") model is created, being essentially an average of all other models in the system [12]. In our case, the model is simply trained on all available gesture classes.

Let $P(X|M_i)$ be the likelihood of the observation sequence given the gesture models M_i. Let $P(X|F)$ be the likelihood of the observation sequence given the filler model F. Then the sequence X is classified as model M_i if

$$\begin{aligned} FR = \frac{P(X|M_i)}{P(X|F)} \geq A \quad \text{and} \\ P(X|M_i) > P(X|M_j)(i \neq j) \end{aligned} \tag{1}$$

where A is a suitably chosen threshold (1.2 in our case) and FR is the filler ratio. Equation 1 then provides a way of determining the significance of an observation sequence based on an appropriately chosen threshold.

4.1 Extraneous Movement

Sports officials are obviously going to perform movements which will not be modeled by the system. Bending down to tie a shoe lace, for example, gives very little information on what is going on in the game. These extra movements should therefore be identified and potentially ignored. It is quite possible that the likelihood of some unknown movement will approach the likelihood of known movements, thus there must be some kind of confidence on how well an observation matches the set of known gestures. Using the filler model allows us to compare different observations relative to each other and provides us with a measure of confidence on how well a model matches the observation sequence.

Potentially more than one model will exceed the threshold for the filler model, however in this work, we simply take the maximum. The difference between the most likely model and the second most likely model is not taken into account.

4.2 Segmentation

Following our previous approach [9], the gestures are first segmented by using the magnitude of accelerometer data (see Section 5 for details). A Gaussian distribution modeling the magnitude of gravity is used to detect periods of movement over a sliding window. The likelihood of the Gaussian is used to make a binary decision on whether the window contains movement. In some cases, when there is little acceleration, spurious responses to the Gaussian distribution can occur. For example, the sequence 1,1,0,1 (where 1 is movement and 0 is no movement), may result, however this is not consistent labeling since there is a large degree of overlap ($48/144$). Thus this sequence is replaced by the sequence 1,1,1,1. Similarly, if the sequence 0,0,1,0 occurs, it is replaced with the sequence 0,0,0,0. These two filters are ensuring that contiguous regions of data have consistent labeling over a sliding window. Figure 2 shows an example sequence of gestures with the corresponding movement decision below.

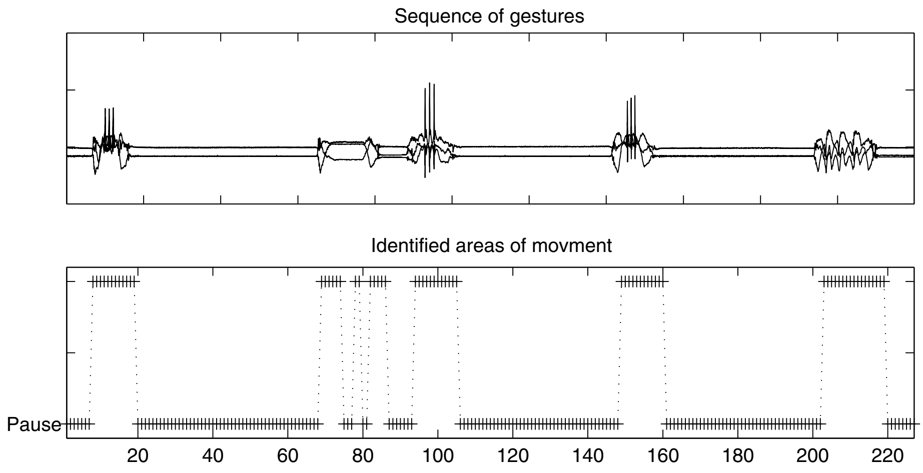


Fig. 2. Segmentation of gesture data using Gaussian of stationarity.

Once the regions of movement indicating candidate gestures are detected, the regions must be classified. Since many sport umpire gestures contain pauses such as the cricket example from Figure 1, there needs to be a method for grouping adjacent regions of identified movement including the pause periods. The problem is that a pause can be either a valid sub-gesture (as part of a gesture) or a pause between gestures. Our approach to overcoming this is by using a conservative estimate for the maximum length of a gesture (10sec) and grouping all detected gestures and pauses in the window. This window is then iteratively reduced in length removing the last region of movement at each step. In Figure 2, for example, the first gesture starting after time 50 would have four candidate regions for grouping (times 69 to 106, 69 to 86, 69 to 79, and 69 to 75). The algorithm proceeds as follows: for each period of movement ahead in time (up to 10sec) of the start of a candidate gesture, calculate the likelihood of each model for that region. After all candidate regions have been identified and their corresponding model likelihoods calculated, they are normalised by the filler ratio. The region and model corresponding to the maximum of the calculated ratios is considered the gesture for that region. Using a filler model to compare different length observations for accurately finding segmentation endpoints is novel and has worked well on our data and example domain. Without a filler model, different length observations can not be compared accross models as the HMM likelihood function is non-linear. The segmentation example referred to in Figure 2 is able to be correctly segmented with our approach.

5 Experimental Results

The architecture of the system is illustrated in Figure 3. As video data is recorded, the movement of key actors wearing the accelerometer wrist bands is also recorded. The accelerometer data is analysed then segmented and classified using the segmentation and classification algorithms described. If a gesture of interest was detected, the video

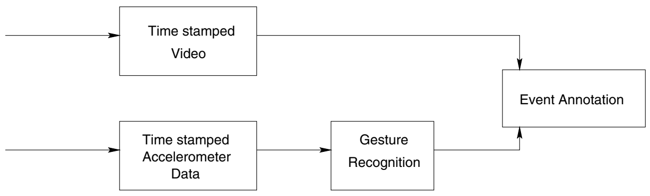


Fig. 3. System Overview.

Table 1. Results for the four controlled sequences.

	Known	Unknown	Recall
Known	40	0	100%
Unknown	2	7	77.8%
Precision	95%	100%	

at the time the gesture was performed is then annotated with the event indicated by the gesture.

Our accelerometers can measure acceleration in 3–D space. They are housed in a small wrist watch sized enclosure worn in the form of a wrist band. Obviously the recognition performance of the system could suffer if the band was worn in grossly different orientations on the wrist, thus we treat the band like a watch, where the face of the enclosure is in a similar direction each time the band is worn. The implementation measures acceleration of up to $\pm 2g$ at 150 samples/second. The feature set used is standard deviation and root mean square for each of the three directions of acceleration. Features are calculated using a window size of 48 samples ($\approx$ 300ms) with a 12 sample overlap. The window size for segmenting regions of movement is 144 samples with a 48 sample overlap.

The cricket umpire gestures we recognise are: *Dead Ball* – sway both arms in front of the body, *Four* – wave the right hand across the body, *Last Hour* – point to the watch on the raised left arm and tap it, *Leg-Bye* – tap the raised right knee, *No Ball* – extend right arm to the side, *One Short* – tap right shoulder with right arm, *Out* – raise arm in front of body with index finger extended, *Penalty Runs* – grasp left shoulder with right hand, *TV Replay* – Outline a rectangle with both hands, and *Wide* – extend both arms out to the sides of the body.

5.1 Controlled Sequences

To initially test our technique, four sequences containing each gesture and a number of unknown movements are recorded. In these sequences, the actor intentionally performs a gesture, either known or unknown, then pauses for some random period, then performs another gesture. In each of the four sequences, each of the 10 known gestures are performed. For the first sequence, one unknown gesture is performed, for the second sequence, two unknown gestures are performed, then for the third and fourth sequences, three unknown gestures are performed. The unknown gestures include the

Table 2. Results for three match segments.

	Known	Unknown	Recall
Known	12	0	100%
Unknown	16	51	76%
Precision	42%	100%	

actor scratching their head, picking up pens from a desk, and typing for a short duration on a computer keyboard. Some of the movements were intentionally performed with very little time between them to test the performance of the segmentation algorithm. Figure 2 shows a portion of one such sequence. Table 1 shows the classification results for the four sequences. The algorithm was able to correctly identify the start and end regions of all gestures in all sequences and classify all known gestures correctly. Two unknown movements were incorrectly classified as known movements.

5.2 Cricket

Two 15 minute portions of an Australia versus England test match and one 15 minute portion of an England versus Pakistan test match were recorded. In all three sequences, the actor mimicked the umpire and performed both known and unknown movements to better represent real world data. The unknown movements include actions such as walking, bending over to pick up something, and passing objects to players. The results are again shown as a confusion matrix in Table 2. When the gesture is known, the system performs well, classifying all gestures correctly. The table shows, however, that unknown gestures are frequently detected as known gestures. A large proportion of these unknown gestures being incorrectly classified as known gestures can be partially explained since the umpire consistently does an action which resembles one of the known gestures.

Typically a cricket umpire stands behind the wickets with their hands behind their back. When the ball is "in play", it is common for the umpire to take their hands from behind their back and start to walk away from the wickets to prevent interfering with the game play. The "Dead ball" gesture starts in a very similar manner and is frequently falsely detected as occurring in the sequences. Currently we are looking at reducing this by using more meaningful criteria than a simple threshold.

5.3 Video Labeling

Table 3 shows the indexing performance for one of the mimicked segments. The table lists the start (t_{1g}) and end (t_{2g}) times of the known gestures in the ground truth and the start (t_{1d}) and end (t_{2d}) times detected by the system. The difference in start time (δt_1) and gesture length ($\delta(t_2 - t_1)$) is also listed. Both the video frame rate and the size of the overlap in the sliding window for movement detection affect the accuracy of the system. The video is recorded at 25fps and the size of the appended data to the sliding window is 48 samples (0.32sec). The table shows that the first three gestures are detected consistently around 1.2 seconds before the gesture occurred. The final gesture

Table 3. Gesture Results for a match segment.

Gesture	$t_{\bullet g}$	$t_{\bullet g}$	$t_{\bullet d}$	$t_{\bullet d}$	$\delta t_{\bullet}$	$\delta(t_{\bullet} - t_{\bullet})$
Wide	90.8	95.6	89.6	94.22	1.2	0.04
Four	96.8	102.56	95.68	100.80	1.2	0.64
Four	319.4	323.36	318.08	323.52	1.3	0.52
Wide	722.8	727.76	717.44	726.08	5.3	3.68

however was detected 5.3 seconds early. The reason the start time of the final gesture was incorrect is that an unknown movement was performed just before the gesture and the segmentation algorithm grouped the adjacent regions together considering them as one complete gesture. This failure of the segmentation algorithm was the only time it incorrectly identified the regions corresponding to a complete gesture for all tested sequences.

The results show that the system performs well overall with the exception of handling unknown movements which have similarities to known movements. Detecting movement is robust, however utilizing the filler ratio requires further investigation for deciding when a known gesture occurs.

6 Conclusions

We have presented results on our approach toward automatic sports video annotation in which we solve the problem of automatic segmentation and robust gesture classification using a hierarchical hidden Markov model in conjunction with a filler model. The hierarchical model allows us to consider gestures at different levels of abstraction and the filler model allows us to handle extraneous umpire movements. We have used a variable sized window to group regions of movement to overcome the problem of recognising gestures which contain pauses as part of the gesture. Pauses between regions of movement need to be regarded (in some cases) as part of a gesture and in other cases, as gaps between gestures. The concept of a filler model from speech recognition is used in our system to aid in detecting unknown movements and classifying them accordingly. Further work will be in investigating the filler model ratio to provide more insight than the current approach which is simply a threshold.

References

1. Gong, Y., Sin, L.T., Chuan, C.: Automatic parsing of TV soccer programs. In: Proceedings of the IEEE International Conference on Multimedia Computing and Systems. (1995) 167–174
2. Zhou, W., Vellaikal, A., Kuo, C.: Rule–based video classification system for basketball video indexing. In: ACM Multimedia 2000, Los Angeles, USA (2000) 213–216
3. Pan, H., Beek, P.V., Sezan, M.I.: Detection of slow-motion replay segments in sports video for highlights generation. In: Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing. (2001) 1649–1652
4. Yang, M.H., Ahuja, N.: Extraction and Classification of Visual Motion Patterns for Hand Gesture Recognition. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Santa Barbara, California (1998) 892–897

5. Becker, D.A.: Sensie: A real–time recognition, feedback and training system for T'ai Chi gestures. Technical Report 426, M.I.T. Media Lab Perceptual Computing Group (1997)
6. VPL DataGlove: VPL DataGlove. VPL Research (2000)
7. Benbasat, A., Paradiso, J.: An inertial measurement framework for gesture recognition and applications. In Wachsmuth, I., Sowa, T., eds.: Gesture Workshop. Springer-Verlag (2002) 9–20
8. Hinckley, K., Pierce, J., Sinclair, M., Horvitz, E.: Sensing techniques for mobile interaction. Symposium on User Interface Software and Technology, CHI Letters **2** (2000) 91–100
9. Chambers, G.S., Venkatesh, S., West, G.A.W., Bui, H.H.: Segmentation of intentional human gestures for sports video annotation. In: International Conference on Multimedia Modelling. (2004) 124–129
10. Chambers, G.S., Venkatesh, S., West, G.A.W., Bui, H.H.: Hierarchical recognition of intentional human gestures for sports video annotation. In: International Conference on Pattern Recognition. (2002) 1082–1085
11. Fine, S., Singer, Y., Tishby, N.: The hierarchical hidden markov model: Analysis and applications. Machine Learning **32** (1998) 41–62
12. Williams, G., Renals, S.: Confidence measures for hubrid hmm/ann speech recognition. In: Proceedings of Eurospeech 1997, Rhodes (1997) 1955–1958

Comparison of Two Fast Nearest-Neighbour Search Methods in High-Dimensional Large-Sized Databases[⋆]

Javier Cano, Juan-Carlos Pérez-Cortés, and Ismael Salvador

Instituto Tecnológico de Informática, Universidad Politécnica de Valencia,
Camino de Vera, s/n 46071 Valencia, Spain
{jcano,jcperez,issalig}@iti.upv.es

Abstract. In this paper we show the results of a performance comparison between two Nearest Neighbour Search Methods: one, proposed by Arya & Mount, is based on a *kd*−tree data structure and a Branch and Bound approximate search algorithm [1], and the other is a search method based on dimensionality projections, presented by Nene & Nayar in [5]. A number of experiments have been carried out in order to find the best choice to work with high dimensional points and large data sets.

1 Introduction

Nearest Neighbour classifiers are widely used in the field of Pattern Recognition. In this and other areas, fast algorithms are needed that efficiently solve the *nearest neighbour search* problem, defined as follows: given a set P of points in a high dimensional space, construct a data structure that given any query point q finds the point in P closest to q. This definition involves the notion of a distance between two points. In our case the Euclidean distance has been used.

This and other problems of the same class (closest pair, diameter, minimum spanning tree, etc.) have been investigated in the field of computational geometry and many efficient solutions have been found that work reasonably well in low-dimensionality spaces. As the dimension scales up, however, many methods experiment serious performance degradation, with space and/or time requirements tending to grow very fast in most cases.

One way to alleviate this problem is to allow for approximate solutions rather than always forcing the search of the exact ones. In fact, some practical applications show no significant loss of precision when approximate nearest neighbours instead of the exact ones are used.

Several fast approximate search algorithms have been proposed in the literature. In this work we have focused on two of them. The first one is based on a well known data struture: the *kd*−tree, and an efficient approximate search algorithm that uses incremental distance calculations and a bound threshold in the search to save computing time. Previous work with this method, as in [3],

⋆ Work partially supported by Spanish CICYT under grant TIC 2003-08496-C04-02.

[2], shows that it is a valid option to cope with the problem of nearest neighbour in high-dimensional data and large-size databases.

The second solution achieves good average search times thanks to the simplicity of the method and the careful design of a precomputed data structure composed by a set of ordered lists of pointers. Then, when a search has to be performed, an optimized algorithm proceeds with an iterative list trimming process, which finally produces a short list of candidate points. A distance computation is then performed for each of this points and the closest one is the nearest neighbour of the query point [5].

In this work, a comparison of the two methods is carried out. A direct exhaustive search method has also been tested as a reference baseline providing us with an upper bound of the average time required to perform a search.

2 Corpora

Both real and synthetic databases have been used in the experiments presented. For the real data, the well-known OCR NIST databases have been chosen. Specifically, the upper-letter set of the NIST Special Database 3, which is composed of 44.951 images, has been used for training, and the NIST Special Database 7, composed of 11.941 images, has been used for test.

Each original 128x128 pixel binary image has been cropped and rescaled to a 9x12 pixel grey-level image. Then, a 108 component vector is built by simply concatenating the rows of an image. Finally, Principal Component Analysis (or Karhunen-Loeve Transform) is performed to reduce the dimensionality to 30.

Two different kinds of random distributions have been used to generate the synthetic data sets: normal, with $\mu = 0.0$ and $\sigma = 1.0$, and uniform, in the unit hypercube. In both cases, a set of 50.000 points has been generated for training and 10.000 points for test. The points, as before, are always represented by 30-dimensional vectors.

3 Search Methods

As discussed before, the problem of searching the nearest neighbour can be trivially solved by an exhaustive search in a simple list. However, the potentially huge size of the prototype set and the high dimensionality of data makes it extremely inefficient and, therefore, inappropiate for real tasks. Instead of this naive solution, several sophisticated methods have been developed in order to achieve approximate searches with very good average search times at the expense of a little increase of the error rate. On the next sections two of these methods are reviewed and some interesting propierties are remarked.

3.1 Kd-Tree + ANN

The $kd-$tree is a widely used data structure. It is a binary tree where each node represents a region in a $k-$dimensional space. Each internal node also defines a hyper-plane (a linear subspace of dimension $k - 1$) dividing the region into two disjoint sub-regions, each inherited by one of its sons. Most of the trees used in the context of our problem divide the regions accordingly to the points that lay in them. This way, the hierarchical partition of the space can either be carried out to the last consequences to obtain, in the leaves, regions with a single point in them, or can be halted in a previous level so as each leaf node holds b points in its region. This number of points, b, is commonly referred as the *bucket size.*

The search of the nearest neighbor of a test point in a $kd-$tree is performed starting from the root, which represents the whole space, and choosing at each node the sub-tree that represents the region of the space containing the test point. When a leaf is reached, an exhaustive search of the b prototypes residing in the associated region is performed. But the process is not complete at this point, since it is possible that among the regions defined by the initial partition, the one containing the test point does not contain the nearest prototype. If this happens, the algorithm backtracks as many times as necessary until it is sure to have checked all the regions that can hold a prototype nearer to the test point than the nearest one in the original region. The resulting procedure can be seen as a Branch-and-Bound algorithm.

As stated before, fast approximate nearest neighbors search algorithms allow for efficient classification without losing significant precision. The notion of approximate search can be viewed as a slightly modified search where instead of reporting a point p closest to q, the algorithm is allowed to report the first point found within a distance $(1 + \epsilon)$ times the distance from q to p.

An algorithm based on these concepts was proposed by Arya & Mount [1]. The algorithm, referred in the article as "Standard k-d Tree Search with Incremental Distance Calculation", works as follows. At each leaf node visited they compute the squared distance between the query point and the data point in the bucket and update the nearest neighbour if this is the closest point seen so far. At each internal node visited they first recursively search the subtree whose corresponding rectangle is closer to the query point. Later, they search the farther subtree if the distance betweeen the query point and the closest point visited so far exceeds the distance between the query point and the corresponding rectangle.

We have used it on conventional $kd-$trees in the experiments. We will refer to this algorithm as ANN (Approximate Nearest Neighbour).

3.2 Projection Based Search

This method is inspired on the original projection search paradigm [4]. However, when dealing with high dimensionalities, a naive version of the basic projection concept is not efficient enough in most cases.

The refined projection search method proposed by Nene & Nayar [5] can be summarized as follows: first, given a query point q, those prototypes in the first dimension which fall into the range $[q_0 - \epsilon, q_0 + \epsilon]$ are selected to constitute the first *candidate list.* Then, for the other dimensions, the bounds $[q_d - \epsilon, q_d + \epsilon]$ are computed and the *candidate list* is trimmed according to them. After the trimming process has been performed through all the dimensions, only a short list of prototypes which fall into the hyperrectangle defined by q and ϵ remains in the final *candidate list.* Finally, the distances from the query point to the prototypes included in the final *candidate list* are exhaustively computed, and the nearest prototype are returned.

A description of this method and some guidelines for an efficient implementation can be found in [5], where taking advantage of a precomputed data structure, a set of ordered lists, it is possible to perform a nearest neighbour search by only computing a small number of binary searches and integer comparisons.

The main problem with this method is its high sensivity to an inadequate estimation of ϵ. An excessively small value leads to an empty final candidate list. A large value of ϵ will produce a large candidate list and, consequently, a performance degradation.

Another important problem of the technique is related to the metric used in the candidate list trimming: the algorithm discards points that fall out of a threshold ϵ defined by an L_∞ distance, but we are searching for the L_2 nearest neighbour, therefore, if the threshold used is too small, the nearest neighbour can be trimmed and the final candidate list not be empty. Thus, when the Ecuclidean Distance is used as the metric, the method is in fact an approximate nearest neighbour search algorithm.

Moreover, the probability of finding other prototypes in an hyperrectangle that does not include the true nearest neighbour (the volume of the hyperrectanlge that is outside an inscribed hypersphere) grows with the space dimensionality.

A further degree of flexibility is possible if we allow the size of the intervals used to build and trim the lists to vary for the different coordinates, or even adaptively adjust for each component of each query point. This means that we can use a different ϵ_d for each axis, and change it, if desired, accordingly to the position of the query point in the space.

In order to evaluate the method, a baseline version has been implemented. We have assumed no “a priori” knowledge of the data distribution. Then, at the design phase the variance of every dimension d (σ_d^2) in the prototype set is computed and stored into a vector. Then, in the search phase, the interval size ϵ_d for each dimension is simply chosen as $\epsilon_d = C * \sigma_d^2$.

On the other end of the performance spectrum, an *optimistic, ideal version* of the algorithm has also been implemented in order to find the best search times that could be reached by the projections based search method. The idea consists on providing the algorithm with the ideal parameter setting for each single search. This has been accomplished by precomputing the difference from the query point to its true nearest neighbour in each coordinate. The ϵ_d associated to the corresponding projection is then set to that value (potentially different for each query point and each dimension). This finally produces, for each search, a candidate list with the minimum possible size including the *nearest neighbour.* No other combination of parameters can yield a faster search. Of course, in a real application setting, any conceivable (efficient) method to estimate the different ϵ_d will probably result in significantly longer running times.

4 Experiments

A first experiment was aimed at defining a lower bound in the average time required by the Nene & Nayar approximation. Making use of the precomputed ϵ_d for each test point, the same precision of an exhaustive search was achieved with a 27.58 times faster average search time.

The conditions of all the experiments were: AMD Athlon Processor at 1200Mhz, 640 Mb of main memory, Linux operating system kernel version 2.4.22 and gcc C compiler, version 3.3.2.

Table 1. Comparison of average execution times with real data (NIST SD3-SD7). An absolute ideal lower bound is shown for the projections (Nene & Nayar) approximation.

Search Method	**Avg. Time**	**Error Rate**
Exahustive	25.93 ms.	12.13%
***kd*-tree+ANN** (ϵ=0.0)	9.45 ms.	12.13%
Projections (ϵ_d precomputed)	0.94 ms.	12.13%
***kd*-tree+ANN** (ϵ=2.0)	0.45 ms.	12.32%

As can be seen in Table 1, the projections method does not improve on the speed up of the *kd*-tree+ANN if a little precision loss is allowed. Taking into account that precomputed ϵ_d (the absolute best possible value of that parameter for each dimension) have been used and that they are a key factor in the performance of the method, it is clear that unless we have a way to precisely estimate these parameters, the method will not work better than the *kd*-tree+ANN.

An experiment has been performed with this approximate method using the simple parameter specification technique described in the previous section and modifying the constant C, which took values in $\{0.5, 1.0, 1.1, 1.2, 1.5, 1.8\}$

In order to compare the performance of the two algorithms, the results of the *kd*-tree+ANN method, with ϵ taking values in $\{0, 1, 2, 8, 64, 2048\}$, are shown in Figure 1. It can be seen there that for all tested values, *kd*-tree+ANN always outperforms Projections. The best possible performance of the Nene & Nayar method achieved with the precomputed ϵ_d for each test point is also plotted in the graph as a small cross. Its position, very near the *kd*-tree+ANN curve suggests that it is very unlikely that a real implementation of the Nene & Nayar method can be superior in practice to a *kd*-tree for the task examined.

Another set of experiments have been performed to compare the algorithms in a more controlled, synthetic, setting. In this case, instead of the classification error in a Pattern Recognition task, the search performance has been directly measured, therefore no classification labels have been used for the data. The

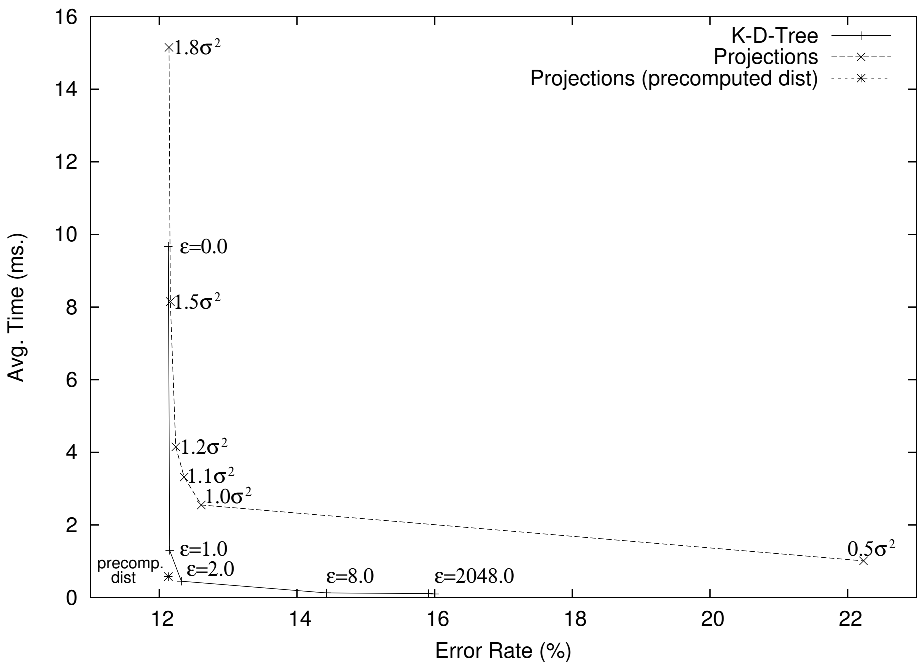


Fig. 1. *kd*-tree+ANN vs. Projections with real data (NIST SD3-SD7). A baseline version of the projections-based method is plotted for different values of the trimming distance, along with the best possible ideal instance of the method, shown with an asterisk in the graph. The ANN curve is also plotted for several values of ϵ.

measure of precision that has been used is the *average rank number*, computed as follows: for each query point a sequence of prototypes, sorted from smaller to larger distances to the point, has been built (using an exhaustive search); then, the sequence number in that sorted list of the point selected by the approximate algorithm as the nearest neighbour is the *rank number*. The *average rank number* for the 10.000 test points has been computed for each parameter setting of the search methods. As it was expected, the results (see Figures 2 & 3) show again better performance of the *kd*-tree+ANN method over the Projections approximation.

Again, the performance of the Nene & Nayar method with ideal parameters is only moderately better than the approximate *kd*-tree search, leaving little space for possible developments in the estimation of ϵ_d that improve the baseline implementation. However, adaptive methods to set the values of ϵ_d as a function of the region of the space where the query point is located can be used to try to get nearer the ideal performance. In the normally ditributed data-set, however, the room for improvement is a little larger than in the uniform distribution.

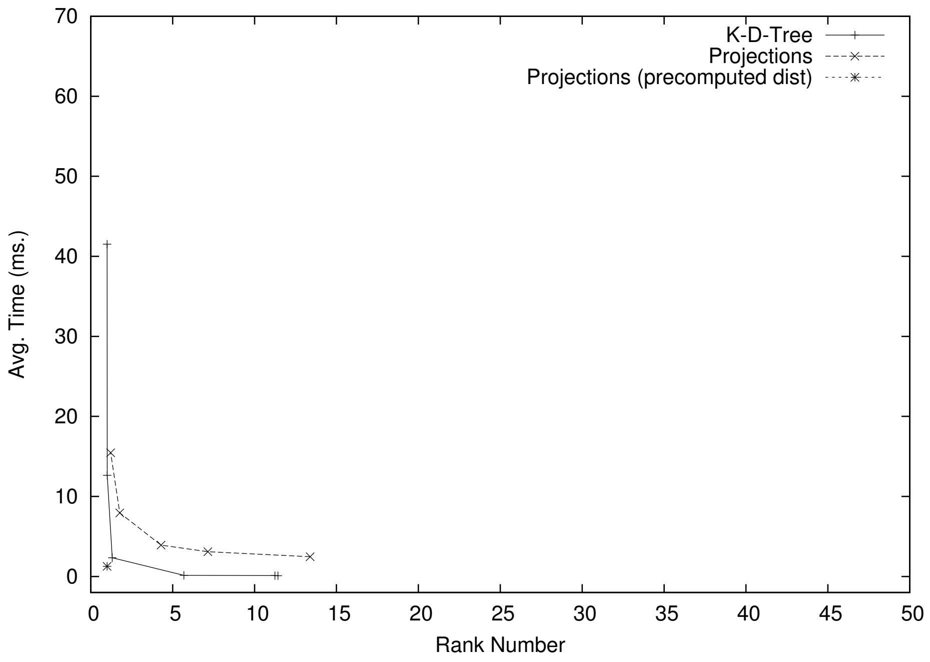


Fig. 2. *kd*-tree+ANN vs. Projections with uniformly distributed synthetic data; a *rank number* = 1.0 means the same result as an exact search. The curves are plotted for different values of ϵ. The ideal parameter setting for the projections method is shown as an asterisk.

5 Conclusions

In this work, two different methods aimed at solving the problem of efficient approximate search of Nearest Neighbours have been compared to find out what is the best option when dealing with high-dimensional and large-sized databases.

As it has been shown in the experiments on classification with real data, *kd*-tree+ANN average times for approximate search are not significantly improved by those obtained by the projection based approximation, even in the best possible (ideal) conditions. For instance, the lower bound found out (with precomputed values of the best possible ϵ_d) for the average time search in the Projection based approximation (0.95 ms.) is not better than the performance achieved with approximate search on the *kd*-tree+ANN, where at the expense of a little increment on the error rate (from 12.13% to 12.32%) an average search time of 0.45 ms. can be obtained.

The results with synthetic data, where the search precision has been directly measured instead of the classification error, confirm that the *kd*-tree+ANN method outperforms the Nene & Nayar technique with fixed values of ϵ_d, and that the ideal adaptive setting of these parameters leaves little space for potentially useful improvements in their estimation, although it cannot be ruled out

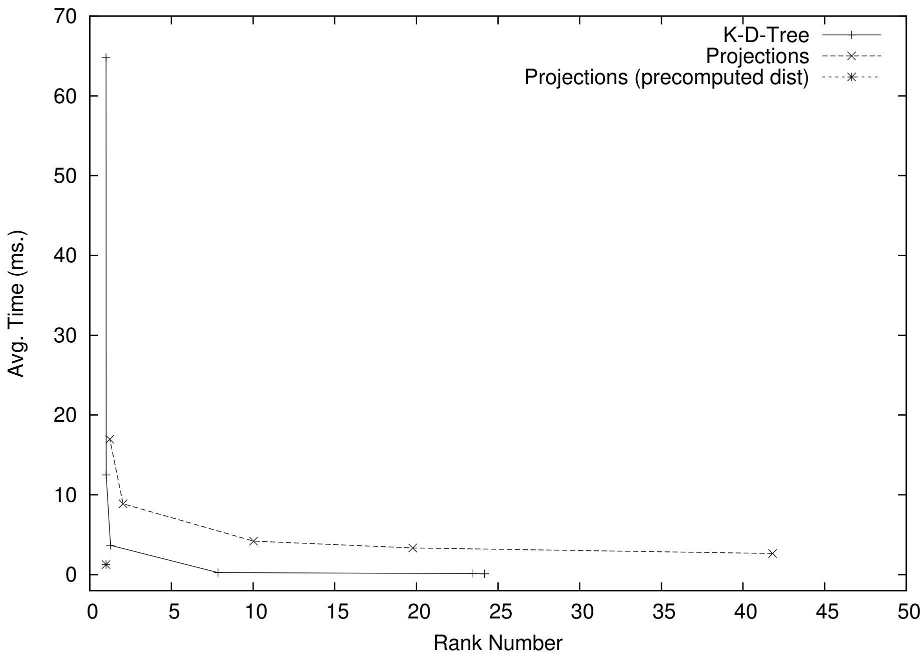


Fig. 3. *kd*-tree+ANN vs. Projections with normaly distributed synthetic data; a *rank number* = 1.0 means the same result as an exact search. The curves are plotted for different values of ϵ. The ideal parameter setting for the projections method is shown as an asterisk.

that new methods to find good values of ϵ_d, adapted to the query region could rival the performance of Approximate Nearest Neighbour search in *kd*-trees.

References

1. S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 45:891–923, 1998.
2. J. Cano and J.C. Perez-Cortes. Vehicle license plate segmentation in natural images. In *Proceedings of the 1st Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*, volume 2652 of *Lecture Notes in Computer Science*, pages 142–149, Puerto de Andratx (Mallorca, Spain), 2003.
3. J. Cano, J.C. Perez-Cortes, J. Arlandis, and R. Llobet. Training set expansion in handwritten character recognition. In *International Workshop on Statistical Pattern Recognition SPR-2002*, volume 2396 of *Lecture Notes in Computer Science*, pages 548–556, Windsor (Ontario, Canada), 2002.
4. J.H. Friedman, F. Baskett, and L.J. Shustek. An algorithm for finding nearest neighbors. *IEEE Transactions on Computers*, 25(10):1000–1006, 1975.
5. S. A. Nene and S. K. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEETPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 1997.

A Shallow Description Framework for Musical Style Recognition

Pedro J. Ponce de León, Carlos Pérez-Sancho, and José M. Iñesta

Departamento de Lenguajes y Sistemas Informáticos, Universidad de Alicante, Ap. 99, E-03080 Alicante, Spain
{pierre,cperez,inesta}@dlsi.ua.es

Abstract. In the field of computer music, pattern recognition algorithms are very relevant for music information retrieval (MIR). One challenging task within this area is the automatic recognition of musical style, that has a number of applications like indexing and selecting musical databases. In this paper, the classification of monophonic melodies of two different musical styles (jazz and classical) represented symbolically as MIDI files is studied, using different classification methods: Bayesian classifier and nearest neighbour classifier. From the music sequences, a number of melodic, harmonic, and rhythmic statistical descriptors are computed and used for style recognition. We present a performance analysis of such algorithms against different description models and parameters.

Keywords: music information retrieval, Bayesian classifier, nearest neighbours.

1 Introduction

The computer music research field is an emerging area for pattern recognition and machine learning techniques to be applied. The content-based organisation, indexing, and exploration of digital music databases (digital music libraries), where digitised (MP3), sequenced (MIDI) or structurally represented (XML) music can be found, is known as music information retrieval (MIR).

One of the problems to be solved in MIR is the modelization of the music style. The computer could be trained in the user musical taste in order to look for that kind of music over large musical databases. The same scheme is suitable to learn stylistic features of composers. Other applications of such a system is to be used in cooperation with automatic composition algorithms to guide this process according to a stylistic profile provided by the user.

A number of recent papers explore the capabilities of machine learning methods to recognise music style. Pampalk et al. [1] use self-organising maps (SOM) to pose the problem of organising music digital libraries according to sound features of musical themes, in such a way that similar themes are clustered, performing a content-based classification of the sounds. Whitman and Flake [2] present a system based on neural nets and support vector machines, able to classify an audio fragment into a given list of sources or artists. Also in [3], the authors describe a neural system to recognise music types from sound inputs. In the work by Thom [4] pitch histograms (measured in semitones relative to the tonal pitch and independent of the octave) are used to describe blues fragments of the saxophonist Charlie Parker. Also pitch histograms and SOM are used in [5] for musicological analysis of folk songs.

In a recent work, Cruz et al. [6] show the ability of grammatical inference methods for modelling musical style. A stochastic grammar for each musical style is inferred from examples, and those grammars are used to parse and classify new melodies. The authors also discuss about the encoding schemes that can be used to achieve the best recognition result. Other approaches like hidden Markov models [7] or multilayer feedforward neural networks [8] have been used to solve this problem.

2 Objectives

Our aim is to develop a system able to distinguish musical styles from a symbolic representation of melodies (digital scores) using shallow structural features, like melodic, harmonic, and rhythmic statistical descriptors. Our working hypothesis is that melodies from a same musical genre may share some common features that permits to assign a musical style to them. We have chosen two music styles, jazz and classical, as a workbench for our experiments. The initial results have been encouraging (see [9]) but now we want to explore the method performance for different classification algorithms, descriptor models, and parameter values.

First, our methodology will be presented, describing the musical data and the description models and classifiers we have used. Then, the classification results obtained with each classifier and an analysis of the recognition results against the different description parameters will be presented. Finally, conclusions and current and future lines of work are discussed.

3 Methodology

In this section we first present the musical sources from which the experimental framework has been established. Second, we will go into the details of the description models we have chosen to describe those musical data. Next we will discuss the free parameters space that sets up the whole experimental framework, and then the classifier implementation and tuning will be presented.

3.1 Musical Data

MIDI files from jazz and classical music, were collected. Classical melody samples were taken from works by Mozart, Bach, Schubert, Chopin, Grieg, Vivaldi, Schumann, Brahms, Beethoven, Dvorak, Haendel, Paganini and Mendehlson. Jazz music samples were standard tunes from a variety of authors like Charlie Parker, Duke Ellington, Bill Evans, Miles Davis, etc. The MIDI files are composed of several tracks, one of them being the *melody track* from which we actually extract our data[1]. The corpus is made up of a total of 110 MIDI files, 45 of them being classical music and 65 being jazz music. This is a somewhat heterogeneous corpus, not specifically created to fit our purposes but collected from different sources ranging from web sites to private collections.

The monophonic melodies consist of a sequence of paired events that can be either note onsets or note endings. The onset event encodes the note *pitch*, that can take a value from 0 to 127. Each onset event at time t has its corresponding ending event at time $t+d$, being d the note *duration*. Time intervals between an ending event and the next onset event are *silences*.

3.2 Description Model

Instead of using an explicit representation of the melodies, we have chosen a description model based on statistical descriptors that summarise the content of the melody in terms of pitches, note durations, silence durations, harmonicity, etc.

The datasets are vectors of musical descriptors computed from fixed length segments of the melodies found in the MIDI files (See section 3.4 for a discussion about how these segments are obtained). Each segment is labelled with the style of the melody it belongs to. We defined an initial set of descriptors based on three groups of features that assess the melodic, harmonic and rhythmic properties of a melody, respectively. Then, from this initial set of descriptors a selection procedure has been performed based on a per-feature separability test. This way, some reduced models have been constructed and their classification ability tested.

The features are computed using a time resolution of $Q = 48$ pulses per bar[2].

The initial model is made up of the following 22 descriptors:

- Overall descriptors:
  - *Number of notes* and *number of significant silences* (those larger than a sixteenth note) in the fragment.
- Pitch descriptors:
  - *Pitch range* (The difference in semitones between the highest and the lowest note in the melody), *average pitch*, and *standard deviation of pitch* (provide information about how the notes are distributed in the score).
- Note duration descriptors (these descriptors are measured in pulses):
  - *Minimum*, *maximum*, *average*, and *standard deviation* of note durations.
- Significant silence duration descriptors (in pulses):
  - *Minimum*, *maximum*, *average*, and *standard deviation*.
- Interval descriptors (distance in pitch between two consecutive notes):
  - *Minimum*, *maximum*, *average*, and *standard deviation*.

[1] Without loosing generality, all the melodies are written in the 4/4 meter. They are monophonic sequencies (at most one note is playing at any given time.)

[2] This is call quantisation. $Q = 48$ means that if a bar is composed of 4 times, each time can be divided, at most, into 12 pulses.

– Harmonic descriptors:
  - *Number of non diatonic notes.* An indication of frequent excursions outside the song key[3] or modulations.
  - *Average degree of non diatonic notes.* Describes the kind of excursions. It is a number between 0 and 4 that indexes the non diatonic notes of the diatonic scale of the tune key, that can be major or minor key[4]. It can take a fractional value.
  - *Standard deviation of degrees of non diatonic notes.* Indicates a higher variety in the non diatonic notes.

– Rhythmic descriptor: *number of syncopations*: notes that do not begin at the rhythm beats but in some places between them (usually in the middle) and that extend across beats. This is actually an estimation of the number of syncopations, but is enough for this task. Syncopations are supposed to appear more frequently in Jazz music than in classical music.

With this set of descriptors, we assume the following hypothesis: melodies of the same style are closer to each other in the description space than melodies from different styles. We will test the performance of different classifiers to verify this hypothesis.

This kind of statistical description of musical content is sometimes referred to as *shallow structure description* [10]. It is similar to histogram-based descriptions, like those found in [5], that try to model the distribution of musical events in a music fragment. Computing the minimum, maximum, mean and standard deviation from the distribution of musical features like pitches, durations, intervals and non-diatonic notes we reduce the number of features needed (each histogram may be made up of tens of features). Other authors have also used some of the descriptors presented here to classify music [11].

3.3 Feature Selection Procedure

The utilised features have been designed according to those used in musicological studies but there is no theoretical support for them. We have devised a selection procedure in order to keep those descriptors that actually contribute to make the classification. The method doesn't account for possible correlations between descriptors, but tests the separability provided by each descriptor independently, and uses this separability to obtain a descriptor ranking. For a detailed discussion on how descriptors are ranked and selected, see [9].

Four additional description models have been constructed with selected descriptors, as shown in table 1. Each model number denotes the number of descriptors included in that model.

We have chosen four reduced model sizes: 6, 7, 10 and 13 descriptors. The 7-descriptor model includes the best rated descriptors. The 6-descriptor model

[3] We used the *key* meta-event present in each MIDI file to compute the harmonic descriptors. The correctness of its value was verified for each file prior to the feature extraction process.

[4] Non diatonic degrees are: 0: ♭II, 1: ♭III (♮III for minor key), 2: ♭V, 3: ♭VI, 4: ♭VII.

Table 1. Description models. For each model the descriptors included are shown in the right column.

Model	Descriptors
6	Pitch range, max. interval, dev. note duration, max. note duration, dev. pitch, avg. note duration
7	+syncopation
10	+avg. pitch, dev. interval, number of notes
13	+number of silences, min. interval, num. non-diatonic notes
22	All the descriptors computed

excludes syncopation from the former model, to test the contribution of the rhythm descriptor on its own. The other two models include other average rated descriptors.

3.4 Free Parameter Space

Given a melody track, the statistical descriptors presented above are computed from fixed length segments of that melody. These segments are extracted defining a window of size ω. One segment is extracted and the window is shifted δ measures towards the end of the melody to obtain the next segment to be described. Given a melody with $m > 0$ measures, the number of segments s of size $\omega > 0$ obtained from that melody is

$$s = \begin{cases} 1 & \text{if } \omega \geq m \\ 1 + \left\lceil \frac{m-\omega}{\delta} \right\rceil & \text{otherwise} \end{cases} \tag{1}$$

showing that at least one segment is always extracted (ω and s are positive integers; m and δ may be positive fractional numbers).

Taking ω and δ as free parameters in our methodology, we have setup a framework where the style classification task is achieved, for different datasets of segments derived from the particular values for ω and δ. The goal is to investigate if there is an optimal combination of these parameters that gives the best segment classification results. The exploration space for this parameters would be referred as $\omega\delta$-space.

ω is the most important parameter in this framework, as it determines the amount of information available for the descriptor computations. A value around 1 would produce windows with a few notes inside, making statistical descriptors less reliable. A large value for ω would lead to merge the – probably different – principal parts of a melody into a single window and also produces datasets with too few samples for training the classifiers. The value of δ would affect primarily the number of samples in a dataset. A small δ value combined with somewhat large values for ω can produce datasets with a large number of samples. The details about the values we used for these parameters can be found in section 4.

3.5 Classifier Implementation and Tuning

Two different classifiers are used in this paper to automatic style identification. They are supervised methods: The Bayesian classifier and the nearest neighbour (NN) classifier [12].

For the Bayesian classifier, we assume that individual descriptor probability distributions for each style are normal, with means and variances estimated from the training data. This classifier computes the squared Mahalanobis distance from test samples to the mean vector of each style in order to obtain a classification criterion.

The NN classifier uses an Euclidean metrics to compute the distance between the test sample and the training samples. The style label of the nearest training sample is assigned to the test sample.

4 Experiments and Results

4.1 The $\omega\delta$-Space Experiment Framework

The melodic segment classification framework has been defined as follows:

$$\omega = 1, ..., 100 \tag{2}$$

and, for each ω

$$\delta = \begin{cases} 1, ..., \omega & \text{if } \ \omega \leq 50 \\ 1, ..., 20 & \text{otherwise} \end{cases} \tag{3}$$

The range for δ when $\omega > 50$ has been limited to 20 due to the very few number of samples obtained with larger δ values for this ω range. This setup let us with a total of 2275 points in the $\omega\delta$-space. We will denote a point in such a space as $\langle w, d\rangle$. A set of experiments have been done for each of these points. An experiment with each classifier (Bayesian and NN) has been prepared for each of the five description models discussed in section 3.3, in order to classify melodic segments. We therefore have 10 different experiments for each $\omega\delta$-point, denoted by $(\omega, \delta, \mu, \gamma)$ where $\mu \in \{6, 7, 10, 13, 22\}$ indicates the description model and $\gamma \in \{Bayesian, NN\}$ the classifier used in that experiment.

For obtaining reliable results a scheme based on *leave-k-out* has been carried out at the level of the source MIDI files for each of the $(\omega, \delta, \mu, \gamma)$ experiments. We want to end up with 10 sub-experiments, that is making $k \simeq 10\%$. The reason behind choosing to separate the files for testing and training rather than first extracting the segments from the files and then perform the leaving-k-out separation is that we want to minimise the probability of having identical segments in the test and training sets. Intuitively, compelling training samples to come from different sources (different MIDI files) than test samples would reduce such a probability to a minimum.

For each sub-experiment 5 jazz style files and 5 classical style files out of a total number of 110 files are kept for testing. Once they have been chosen, segments of ω measures are extracted from the melody tracks and test and training datasets containing μ-size descriptor vectors are constructed.

The segment classification sub-experiments are performed training the γ classifier with the corresponding training set. Classification tests are done with the trained classifiers and the success ratio is averaged over all the sub-experiments.

Thus, for each $(\omega, \delta, \mu, \gamma)$ experiment, we obtained an average classification success rate.

Summarising, 22750 experiments, each consisting of 10 sub-experiments, have been carried out. The maximum number of segments extracted is 8985 for the $\omega\delta$-point $\langle 3, 1\rangle$. The maximum is not located at $\langle 1, 1\rangle$ as expected, due to the fact that segments not containing at least two notes are discarded. The minimum number of segments extracted is 119 for $\langle 100, 20\rangle$, as expected. The average number of segments in the whole $\omega\delta$-space is 775. The average proportion of jazz segments is 36% of the total number of segments, with a standard deviation of about 4%. This is a consequence of the classical MIDI files having an average melody length greater than jazz files, although there are less classical files than jazz files.

4.2 Classification Results

In Fig. 1 the results in the $\omega\delta$-space for the Bayes classifier with the 10-descriptor model are displayed. Note that recognition percentages range from 76 to 91%, although the values below 80% concentrate in the low-left corner when the window width is very small. All the results with $\omega > 5$ are above 82%. The best results were obtained for $\omega \approx 70$ and $\delta \approx 15$ (more than 91%). Although this behaviour is not exactly the same for the other classifiers and models.

Figure 2 summarizes the behaviour of the Bayes and NN classifiers for the different values of ω, given a fixed value of $\delta = 1$. Note that the trend is to rise

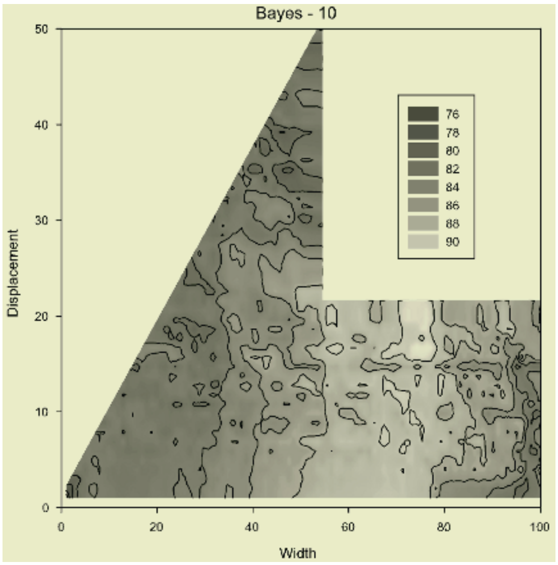


Fig. 1. Illustration of the recognition percentage in the $\omega\delta$ space. The best results (around 91.5 %) are found in the lighter area, with large widths and moderate displacements.

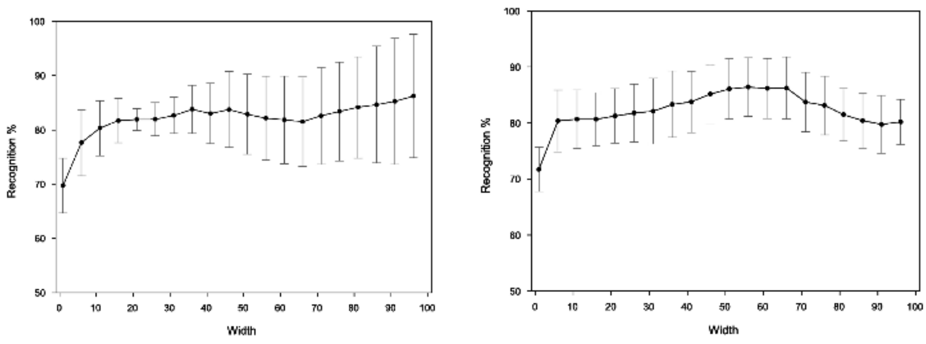


Fig. 2. Recognition results averaged for the different models against the window width, with a fixed $\delta = 1$. Bars indicate the deviation obtained for the different experiments. Only one point every five points is displayed for clarity. (left) Nearest neighbour; (right) Bayes.

rapidly for small values of ω and then to be more or less stable. The different experiments for NN have provided higher differences in recognition percentages than the Bayes classifier. Also, note that NN performs slightly better in average for large ω values. As for the entire $\omega\delta$-space, the NN classifier scored an 83.3% overall average success rate, while a 76.6% success rate was obtained for the Bayesian classifier.

5 Conclusions and Future Work

We have shown the ability of two classifiers to map symbolic representations of melodic segments into a set of musical styles using melodic, harmonic and rhythmic statistical descriptors. The experiments have been carried out over a large parameter space defined by the size of melodic segments extracted from melody tracks of MIDI files of both styles and the displacement between segments consecutively extracted from the same melodic source. A total of 227500 classification experiments have been performed.

Our main goal in this work has been to establish a framework for musical style recognition experiments, while concluding an answer for the following two questions:

1. Which classifier works better for this task?
2. Are there any optimal ω and δ values for style classification?

Answer to the first question is the NN classifier, as seen in previous section, with an 83.3% overall average succes rate, with a best success rate of 94% for model 10 and point $\langle 98, 1\rangle$. Answer to the second question is a somewhat disappointing one. The best results were obtained with very large segment sizes with the NN classifier, leading us to a new question: Would we achieve even more better results if we take as samples single descriptor vectors that represent the whole melodies? This issue must be investigated further with larger corpora.

New description models and different classifers can be easily incorporated to this framework, as well as different corpora and the exploration of different ranges for ω and δ. An extension to the framework is under development, where a voting scheme for segments will be used to obtain classification results for whole melodies. Connectionist approaches are also to be tested with the models and parameters presented here. Finally, different descriptor sets are currently under study in our search for a good statistical description for musical styles.

Acknowledgements

This paper has been funded by the Spanish CICyT project TIRIG; code TIC2003–08496–C04–04,

References

1. E. Pampalk, S. Dixon, and G. Widmer. Exploring music collections by browsing different views. In *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR'03)*, pages 201–208, Baltimore, USA, 2003.
2. B. Whitman, G. Flake, and S. Lawrence. Artist detection in music with minnowmatch. In *Proceedings of the 2001 IEEE Workshop on Neural Networks for Signal Processing*, pages 559–568. Falmouth, Massachusetts, September 10–12 2001.
3. H. Soltau, T. Schultz, M. Westphal, and A. Waibel. Recognition of music types. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-1998).* Seattle, Washington, May 1998.
4. B. Thom. Unsupervised learning and interactive jazz/blues improvisation. In *Proceedings of the AAAI2000*, pages 652–657, 2000.
5. P. Toiviainen and T. Eerola. Method for comparative analysis of folk music based on musical feature extraction and neural networks. In *III International Conference on Cognitive Musicology*, pages 41–45, Jyvskyl, Finland, 2001.
6. P. P. Cruz-Alcázar, E. Vidal, and J. C. Pérez-Cortes. Musical style identification using grammatical inference: The encoding problem. In *Proceedings of CIARP 2003*, pages 375–382, La Habana, Cuba, 2003.
7. W. Chai and B. Vercoe. Folk music classification using hidden markov models. In *Proc. of the Int. Conf. on Artificial Intelligence*, Las Vegas, USA, 2001.
8. G. Buzzanca. A supervised learning approach to musical style recognition. In *Music and Artificial Intelligence. Additional Proceedings of the Second International Conference, ICMAI 2002*, Edinburgh, Scotland, 2002.
9. P. J. Ponce de León and J. M. Iñesta. Feature-driven recognition of music styles. In *1st Iberian Conference on Pattern Recognition and Image Analysis. Lecture Notes in Computer Science, 2652*, pages 773–781, Majorca, Spain, 2003.
10. Jeremy Pickens. A survey of feature selection techniques for music information retrieval. Technical report, Center for Intelligent Information Retrieval, Departament of Computer Science, University of Massachussetts, 2001.
11. S. G. Blackburn. *Content Based Retrieval and Navigation of Music Using Melodic Pitch Contours.* PhD thesis, Department of Electronics and Computer Science, University of Southampton, UK, 2000.
12. Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition).* Wiley-Interscience, 2000.

A Nearest Neighbor Method Using Bisectors

Mineichi Kudo, Hideyuki Imai, Akira Tanaka, and Tetsuya Murai

Division of Systems and Information Engineering
Graduate School of Engineering
Hokkaido University
Kita-13, Nishi-8, Kita-ku, Sapporo 060-8628, Japan
{mine,imai,takira,murahiko}@main.eng.hokudai.ac.jp
http://ips9.main.eng.hokudai.ac.jp

Abstract. A novel algorithm for finding the nearest neighbor was proposed. According to the development of modern technology, the demand is increasing in large-scale datasets with a large number of samples and a large number of features. However, almost all sophisticated algorithms proposed so far are effective only in a small number of features, say, up to 10. This is because in a high-dimensional space many pairs of samples share a same distance. Then the naive algorithm outperforms the others. In this study, we considered to utilize a sequential information of distances obtained by the examined training samples. Indeed, a combinatorial information of examined samples was used as bisectors between possible pairs of them. With this algorithm, a query is processed in $O(\alpha\beta nd)$ for n samples in a d-dimensional space and for $\alpha, \beta < 1$, in expense of a preprocessing time and space in $O(n^{\cdot})$. We examined the performance of the algorithm.

1 Introduction

The k nearest neighbor (k-NN) method (Cover and Hart, 1967) is very popular in pattern recognition. This method is effective both for estimation of densities and for classification. However, the method requires a large amount of computation to calculate the distances of a query sample to all training samples. To reduce the amount of computation, many methods have been proposed (Hart, 1968; Gates, 1972; Dasarathy, 1994; Chang and Wu, 1993; Fukunaga and Narenda, 1975), especially for classification (Hart, 1968; Gates, 1972; Dasarathy, 1994).

There are two approaches to reduce the computational cost: one group of methods aims to reduce the size of the training sample set to be referred in query stage and another group aims to reduce the number of distance calculations in query stage by adopting an efficient search procedure. In this paper, we focus on the latter approach.

A trial to find a $(1+\epsilon)$ approximate nearest neighbor, Arya et *al.* (1998) proposed an algorithm which runs in $O(c \log n)$ for answering a query, where c grows exponentially in d and polynomially in $1/\epsilon$. In addition, Kleinberg (1997) proposed two algorithms for the same goal; one runs in $O((d \log^2 d)(d + \log n))$ in query time and another is said to run linear in d asymptotically. Both are able to use to find the nearest neighbor with a sufficiently small value of ϵ. However, in even these algorithms, it is still unclear if they really runs faster than the naive algorithm in very high-dimensional cases. This is because we have to set ϵ close to zero in the case, and the complexity analysis hides the influence by $1/\epsilon$. So, we propose an algorithm which runs really faster than the naive algorithm when the distance is measured by Euclidean distance.

2 Key Ideas

The key idea to do search efficiently is to use ultimately a sequential information obtained so far. After we examined some points (samples) in distance calculation between them and a query point (sample), we have much information more than the sum of individual distance evidences. In many algorithms, only the most critical evidence, the distance to the current candidate of the nearest neighbor, is used. However, we may use the second and the third candidates as well. In fact, it is possible to consider a sequence of these points examined up to the current stage. We will use one of such information. In this study, we focus on the nearest neighbor, 1-NN, instead of k-NN, but the idea is easily extended to k-NN cases.

Let us assume that we have examined two points x_1 and x_2 and have calculated the distances $d(x_1, q)$ and $d(x_2, q)$, and assume that $d(x_1, q) < d(x_2, q)$. What is the information at this time ? Most often used information is that the true nearest neighbor of q has a distance less than $d(x_1, q)$, thus we do not have to check samples with distance larger than this value. However, information like "x_1 is closer to q than x_2" may give us a possibility such as any sample to which x_2 is closer than x_1 can omit from the further search. If so, for three points examined so far, we have three same kinds of information, and $\binom{4}{2} = 6$ for four points, and so on. Of course, this is not always true. However, under some condition, we can make such a decision.

Under which condition can we use such decisions ? Let us explain it in a two-dimensional case (Fig.1). In Fig.1, x_5 is the closest point to a query point q, that is, x_5 is the answer, but has not examined. Four points $(x_1 - x_4)$ are already examined, and six perpendicular bisectors between them are drawn. Here, bisector b_{13} should be out of consideration. This is because this bisector separates the answer x_5 and q into different regions. With five bisectors, we can conclude that the closest point to q is in the hatched region. In the following, we analyze about which bisectors we can use for narrowing the search area.

3 Effective Bisectors

It is easy to determine which bisectors are effective and which are not. Here a bisector is said to be "effective" when the bisector dose not separate the answer

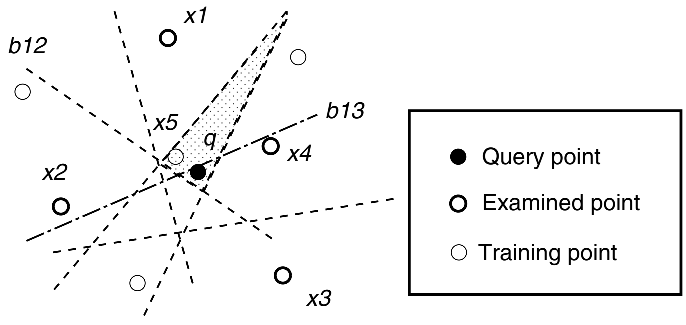


Fig. 1. Narrowing by bisectors. Four of eight samples are already examined and five effective bisectors between these four points are used for narrowing the search area.

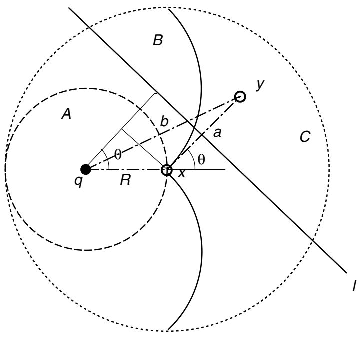


Fig. 2. An effective bisector. Among distinct regions A, B and C, point y having an effective bisector must exist in only C.

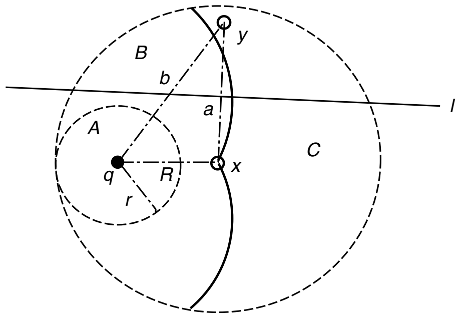


Fig. 3. Extended effective region (C). Here, a solution is assumed in the small ball centered at q.

point and the query point into different regions separated by the bisector. An illustrative examples is shown in Fig. 2.

In this figure, two points x and y are already examined (x is closer to q) and we know distances $R = d(x, q)$ and $b = d(y, q)$. Let us assume that the distance between x and y, $a = d(x, y)$, is also known. It is enough to search region A for finding a better candidate than x. So, the effective bisectors should not intersect A. Then, the necessary and sufficient condition for the bisector between x and y to be effective is

$$b^2 \geq R^2 + 2Ra.$$

(Proof)
Let us assume that the perpendicular bisector l of x and y does not pass though the ball with radius R centered at q. Then following has to be satisfied

$$\begin{aligned} a/2 + R\cos\theta &\geq R \\ a &\geq 2R(1 - \cos\theta). \end{aligned} \tag{1}$$

This inequality shows the condition that angle θ has to satisfied. To translate this angle condition to the distance condition related to $b = d(q, y)$, the following is derived:

$$\begin{aligned} b^2 &= (R + a\cos\theta)^2 + a^2\sin^2\theta \\ &= R^2 + a^2 + 2Ra\cos\theta \\ &= R^2 + 2Ra(a/2R + \cos\theta) \\ &\geq R^2 + 2Ra \end{aligned}$$

The last inequality is from (1). Q.E.D.

The boundary equation (1) with equality is called "Cardioid" in geometry. It is noted that region A in Fig. 2 can be ignored because we impose that x is closer than y. If not, exchange x and y. So the probability that y occurs in region C is $P(C)/P(B + C)$, where $P(X)$ is the probability of region X. In a uniform distribution, the probability $P(C)$ is about 0.39 for $d = 2$ and increases to 0.5 as d increases. However, $P(B)$ grows with dimensionality up to $1 - P(C)$. As a result, we may assume that the probability that a new point (y in this example) generates an effective bisector incorporated with x is 1/2 regardless of dimensionality.

It should be noted that we cannot narrow the search region directly into region A, because knowing whether a sample exists in A or not is knowing the distance between that point and the query point. On the contrary, with a bisector associated with x and y, it is enough to check to which of x and y that sample is closer. This is easily done if we know the distance between any pair of training points. As a result, with preprocessing to calculate the distances of every pair, we can do this efficiently.

In this study, we extend the condition for effective bisectors to another in the situation in which we want to find a sample close to q within distance r. Such a query is interesting itself and is able to extend to find the nearest neighbor by repeating this query with several values of $r = r_1, r_2, \ldots, r_t (r_1 < r_2 < \cdots < r_t = \infty)$. As will be described later, this brings a benefit in computation cost. It is also possible to set the value of r to the distance R of the current candidate of the nearest neighbor of q in the middle of search. Then, it is guaranteed to find the nearest neighbor.

In this problem setting, for a solution within r, the condition under which a bisector $b(x, y)$ is effective in the same setting above is given by

$$b^2 \geq R^2 + 2ra. \tag{2}$$

This is easy to confirm (Fig. 3). For example, for $r = 0$, every bisector is effective. According to a smaller value of r, a larger effective region C is obtained.

3.1 Algorithm

The algorithm is shown in Fig. 4.

```
Step 0:  (Preprocessing)
         $n$ : The number of training samples
         $q$ : The query sample
         $r$ : The search threshold (user-specific)
         $t$ : The possible number of $r$ (user-specific)
         $s$ : The number of bisectors (user-specific)
         $\theta[i] \leftarrow \infty$ : The termination threshold on $x_i$
         $x_c$ : The current nearest neighbor sample
         $R \leftarrow \infty$ : The current minimum distance to $q$
         **For**  every pair of $x_i$ and $x_j$
                  Store the distance $d(x_i, x_j)$ in array $d[i, j]$
         **For**  $i = 1$ **to** $n$
                  $\theta[i] \leftarrow d(x_i, x^i_{\bullet-NN})/2$
                  where, $x^i_{\bullet-NN}$ is the nearest neighbor of $x_i$

                  Procedure **main**
Step 1   Repeat the following with $r = r_{\bullet}, r_{\bullet}, \ldots, r_t = \infty$
         $r^* \leftarrow r$
         **For** $i = 1$ **to** $n$
Step 2            **if BallTest** is "not passed" **then** continue with $i \leftarrow i + 1$
Step 3            **if BisectorTest** is "not passed" **then** continue with $i \leftarrow i + 1$
Step 4            Calculate distance $d(q, x_i)$
Step 5            Do **CreateBisector**
Step 6            If $d(q, x_i) < R$, update $R \leftarrow d(q, x_i); c \leftarrow i; r \leftarrow \min(r, R)$
Step 7            **if TerminationTest** is valid **then** goto Step 8
Step 8   **if** $R < r^*$ **then** output $x_c$, **else** go to Step 1

         Procedure **BallTest**
         **if** $d[c, i] < R$ return "passed", otherwise return "not passed"

         Procedure **BisectorTest**
         **For** the latest $s$ bisectors $b(x_j, x_k)$ $(d(x_j, q) < d(x_k, q))$
                  **if** $d[i, j] > d[i, k]$ **then** return "not passed"
         return "passed"

         Procedure **CreateBisector**
         $x \leftarrow x_c; y \leftarrow x_i$
         **if** $d(x, q) > d(y, q)$ **then** exchange $x \leftrightarrow y$
         $R \leftarrow d(q, x); a \leftarrow d[i, c]; b \leftarrow d(q, y)$
         **if** $b^{\bullet} \geq R^{\bullet} + 2ra$ **then** register $b(x, y)$

         Procedure **TerminationTest**
         **if** $d(q, x_i) < \theta[i]$ **then** return "valid" else "not valid"
```

Fig. 4. Algorithm.

The termination condition (Kudo et *el.*, 2003) is also incorporated in Step 7. With the termination condition, we may find the solution in the middle of search and terminate the procedure, but this condition does not work well in high-dimensional cases.

In this algorithm, 1) a ball test is carried out in $O(1)$ (Step 2), 2) a bisector test is in $O(2s)$ (Step 3), 3) a distance calculation in $O(d)$ (Step 4), and 4) a construction of a new bisector in $O(1)$. Let us assume that a query sample passes Step 2 in probability α and Step 3 in probability β. Then the expected cost is $O\left(n\left((1-\alpha)+2s\alpha(1-\beta)+\alpha\beta(d+1)\right)\right) \simeq O(\alpha\beta nd)$.

4 Effectiveness of Bisectors

We conducted a simple experiment to measure to what degree the proposed algorithm is efficient. The samples are generated according to a multivariate Gaussian with a unit covariance and zero mean. The number of samples was varied as $n = 1000, 5000, 10000, 20000$ and the dimensionality was varied as $d = 2, 5, 10, 20, 30, 50$. The computational costs can be estimated by how often a query sample passes Step 2 (ball test), Step 3 (bisector test) and Step 4 (distance calculations). So, we counted the number of samples passed these steps. The results are shown in Figs. 5 and 6.

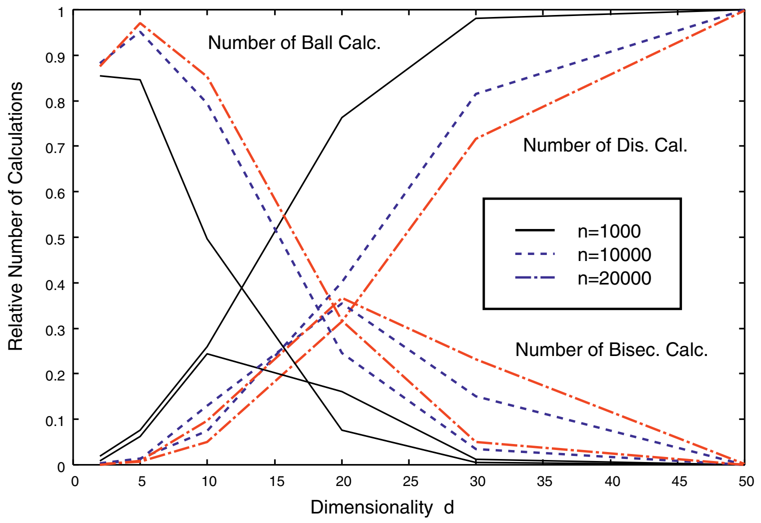


Fig. 5. Relative number of calculations as letting that of the naive algorithm be one.

In Fig. 5, the summand of three curves becomes almost one. If the amount is less than one, it means that the termination condition happened. From this figure, we can see

1. The ball test works well in low-dimensional cases, but decreases exponentially in dimensionality.
2. The bisector test works even up to a moderate size of dimensionality. The effectiveness decreases also as dimensionality increases, but still useful to reduce the number of distance calculations. It looks too low in low-dimensional cases, but this is because the ball test is carried out before the bisector test.

From Fig. 6, it can be read that the relative distance calculation decreases as n increases, which means that the order is less than $O(nd)$.

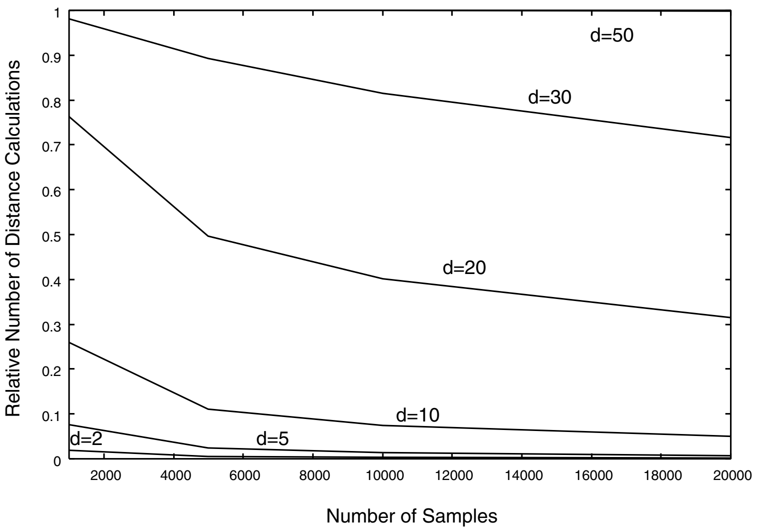


Fig. 6. Relative number of distance calculations as the number of samples increases.

Table 1. Search time (Pentium4, 2GHz, 2GB Memory, 256KB cache, Linux). Preprocessing time is not included.

Dataset	C	d	$n(n_{tr} : n_{te})$	Time(ms)		
				Naive	Proposed	Ann
sonar	2	60	103 (8:2)	0.008	0.007	0.01
mfeat	10	187	2000 (8:2)	0.475	0.585	0.454
musk	2	166	476 (8:2)	0.047	0.028	0.037
Jap. Char	30	196	6000 (1:1)	11.7	7.07	8.42

Table 2. Probabilities of passing several steps.

Dataset	α	β	$\alpha\beta$
sonar	0.63	0.89	0.56
mfeat	0.84	0.99	0.83
musk	0.31	0.93	0.29
Jap. Char	0.45	0.88	0.40

5 Experiments

We used four real-world datasets of 'sonar', 'mfeat', 'musk' and 'Japanese Characters.' The first three are taken from machine learning databases in UCI (UCI Repository of Machine Learning Databases, 1991). The sample was divided to the training and testing sample sets by $n = n_{tr} + n_{te}$ as shown in Table 1. For comparison, ANN (Arya et *al.* (1998)) was carried out with $\epsilon = 0$. The parameter used in the proposed method, $t = 3$ and $r_i (i = 1, 3)$ were determined experimentally. In addition, s was set to $d/2$. The results are shown in Tables 1 and 2. From Table 2, we can see that the probability that training samples pass the ball test, α, is lesser than as expected, but the probability that training samples pass the bisector test, β, is higher than as expected. The highness of $1 - \alpha$ can be explained as in these practical problems data form some clusters from their natures. On the other hand, the lowness of $1 - \beta$ shows that many training samples share almost a same distance to a query point. This is just the problem that we mentioned first, so, unfortunately, we cannot recognize the effectiveness of our way using bisectors. To make clear the reason, we examined further. Then it turned out that the number of found effective bisectors was very small, a few percent of the total number of training samples, in contrast to our analysis (1/2), but once such a bisector was found, then it worked well. More precisely, only a few bisectors (less than one percent of total number of training samples, for example, 0.2% in Japanese Character Database) were contributed the value of β. Therefore, the probability that effective bisectors were found was not so high, but the found effective bisectors worked very good.

6 Discussion

The question to be answered is that this is faster than the previous many algorithms. It requires a preprocessing time of $O(n^2)$ and the same amount of space. It is obvious that the algorithm needs $O(\alpha\beta nd)$ in query time that is less than that of the naive algorithm that needs $O(nd)$, although there is no progress in the definition of large Oh. So, the question is reduced to that α (the probability that training samples pass the ball test) and β (the probability that training samples pass the bisector test) is really less than one even in high dimensional cases. The answer was partly yes from the presented experiments.

We can know about the time complexity as follows. Let p be the probability that an effective bisector is found and q be the probability of the area for further search reduced by the bisector. Then, the time complexity $T(n)$ for examining n training samples in d dimension can be written as $T(n) = d + 2 + pT(qn) + (1 - p)T(n - 1)$. Then, we can see that $T(n) \leq O(nd)$. The equality holds for $qn = n - 1$. Therefore, usually the complexity is better than that of the naive one. To have more precise complexity, we have to take into consideration the influence by successive bisectors.

One more practical issue is the effectiveness of cache. Recent CPUs are equipped with some amount of memory cache, so that the memory access to a sequential data is processed very fast. On the contrary, a random access as the proposed method does, is very slow compared with such a sequential memory access. This make less the advantage of the proposed algorithm. However, for very large datasets, every data should be kept in a disk not in a memory, then the advantage would be alive.

7 Conclusion

A novel nearest neighbor algorithm was proposed. It utilized a sequence of information obtained so far. The experimental results seem to support its efficiency even in high dimensional cases in which the previous algorithms are not better than the naive algorithm. The main drawback of $O(n^2)$ space would be reduced by developing the data structure.

References

1. T.M. Cover and P.E. Hart, Nearest neighbor pattern classification, IEEE Trans. Information Theory, vol.IT-13, no.1, pp.21-27, Jan. 1967.
2. P.E. Hart, The condensed nearest neighbor rule, IEEE Trans. Information Theory, vol.IT-14, no.3, pp.515-516, May 1968.
3. G.W. Gates, The reduced nearest neighbor rule, IEEE Trans. Information Theory, vol.IT-18, no.3, pp.431-433, May 1972.
4. B.V. Dasarathy, Minimal consistent set(MCS) identification for optimal neighbor decision systems design, IEEE Trans. Systems, Man, & Cybernetics, vol.SMC-24, no.3, pp.511-517, March 1994.
5. K. Fukunaga and P.M. Narendra, A branch-and-bound algorithm for computing k-nearest neighbors, IEEE Trans. Computers, vol.C-24, no.7, pp.750-753, July 1975.
6. C.-C. Chang and T.-C. Wu, A hashing-oriented nearest neighbor searching scheme. Pattern Recognition Letters, vol.14, pp.625–630, August 1993.
7. K. Fukunaga ,Nonparametric Density Estimation, in Introduction to Statistical Pattern Recognition, pp.268–287, Academic Press, 1990.
8. P. M. Murphy and D. W. Aha, *UCI Repository of Machine Learning Databases [Machine-Readable Data Repository].* University of California, Department of Information and Computer Science, Irvine, California, 1991.
9. S. Maneewongvatana and D. M. Mount, The Analysis of a Probabilistic Approach to Nearest Neighbor Searching. *In Proc. 7th Workshop on Algorithms and Data Structures (WADS 2001),*2001, pp. 276-286.
10. S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, An optimal algorithm for approximate nearest neighbor searching, *Journal of the ACM,*45 (1998), pp. 891-923.
11. J. Kleinberg, Two algorithms for nearest-neighbor search in high dimensions. *In Proc. 29th ACM Symposium on Theory of Computing,*1997, pp. 599-608.
12. M. Kudo, N. Masuyama and M. Shimbo, Simple termination conditions for k-nearest neighbor method. *Pattern Recognition Letters*, **24**(2003), 1213–1223.

Distance Based Strategy for Supervised Document Image Classification

Fabien Carmagnac[1,2], Pierre Héroux[2], and Éric Trupin[2]

[1] A2IA SA
40 bis rue Fabert
75 007 Paris cedex - France
Fabien.Carmagnac@a2ia.com
http://www.a2ia.com
[2] Laboratoire PSI
CNRS FRE 2645 - Université de Rouen
76 821 Mont-Saint-Aignan cedex - France
http://www.univ-rouen.fr/psi

Abstract. This paper deals with supervised document image classification. An original distance based strategy allows automatic feature selection. The computation of a distance between an image to be classified and a class representative (point of view) allows to estimate a membership function for all classes. The choice of the best point of view performs the feature selection. This idea is used by an algorithm which iteratively filters the list of candidate classes. The training phase is performed by computing the distances between every class. Each iteration of the classification algorithm computes the distance d between the image to be classified and the chosen representative. The classes whose distance with this point of view differs from d are deleted in the list of candidate classes. This strategy is implemented as a module of A2IA FieldReader to identify the class of the processed document. Experimental results are presented and compared with results given by a knn classifier.

1 Introduction

Since a few years, handwritten recognition systems have improved (automatic zip code, form field or cheque amount reader) leading to commercial applications. The good reading rates of such systems allow to process less constrained documents (eg. order forms, invoices). To validate the reading results on such semi-structured documents, it is necessary to associate a reading model to each document class. This reading model contains information about the fields of the document: position, nature (letters, digits, consistency rules, meaning).

Some systems process only a unique kind of document. Others receive an heterogeneous stream of documents. In that case the system has to identify the document class (and therefore the corresponding reading model) of a given document, which means a document class identification module precedes the reading module. In many systems, the classes to discriminate are known before the classification module conception [1] [2]. During the desing it is possible to determine the most discriminating features by studying the images representing the classes. On the other hand, systems are now conceived without knowing the number or the nature of the document classes. Then the classification module has to automatically determine the best features set for the given set of document classes during a learning step.

This paper presents a document classification module, which automatically performs feature selection on any set of document classes. This module is designed to be added to an existing system that reads handwritten fields on documents from a unique class. This system needs to know the reading model. By associating a reading model to each document class, the document classification module allows the system to process an heterogeneous stream of documents.

The system is intended to be sold to final users such as government services or banks, which are not specialists in document analysis nor classification specialists. Therefore the module determines reflexively the most discriminating set of features given a specific problem. If the available features set is not able to efficiently separate the images of a particular project, new features can be easily inserted inside the module.

Our strategy simultaneously performs the feature selection for a given problem and the document classification. It is based on the computation of distance between documents. In section 2, we remind the context in which the classification module is inserted and we expose the constraints attached to a commercial use. Section 3 introduces the main definitions. It details the notion of document class, presents the structure of the features used to extract feature sets, and finally introduces the concept of distance between document images according to to a feature and its metric. Section 4 details the principles of the classification process which is presented in section 5 and 6. Section 5 presents the different steps of the supervised learning phase. Section 6 describes the processes applied to document image to determine its class. Finally, experimental results are presented in section 7. These are compared with results given by a *knn* classifier.

2 Context and Constraints

We remind that the module presented in this article is added to an existing system that reads handwritten fields of forms from a unique class. The purpose of the module is to identify the class of the document so that the system can process documents from multiple classes. Once the document class is determined, the document is processed as done before by using the associated reading model. This reading model includes rules (location, nature, syntax, meaning, consistency) which allow to improve the automatic reading.

The use of the classification module as a part of a commercial product implies to respect some constraints. It must be able to discriminate the class of a document among a hundred classes, and the processing time has to be lower than a limit fixed by the final user (1 or 2 seconds per document with a 1GHz PC). This implies that the processing time is a sub linear function of the classes number. The time needed to classify a document among 100 classes must not be 10 times greater than the time needed to classify a document among 10 classes. On the other hand, final users are not specialists in document analysis. They do not know how to determine the best parameters for the classification. The classification module must be able to automatically adapt its parameters to be as robust as possible on any document set, by evaluating its discrimination performance.

3 Definitions

This section defines the most important terms used in the description of the classification module principle. More accurately, the concepts of document classes, feature, and distance between documents are introduced and detailed.

As mentioned above, we define a document image class as a set of document images on which the same reading model is applied. In our case, the reading model is defined by the final user. A document class is then defined because of the subsequent processing. Two documents are in the same class if the fields to be read are at the same location even if the images look structurally very different. On the other hand, two documents which images are very similar can be in two different classes if, for example, two fields have their location inverted. For a given problem, two documents from the same class can be in two different classes in an other problem, if their reading model becomes different.

Notation 1 *Document image and document images set*
The document image classes set is called $ClassSet$.
The document images set is called $ImageSet$. The set of images belonging to class C is called $ImageSet_C$.

The features used can be numerical, syntactical and/or structural. It is possible to associate different metrics to each feature. Each metric allows to compute a distance between two document images.

Definition 1. *Feature and Metric*
The feature space associated to the feature F is called $Space_F$. A metric associated to F is called M_F.

$$F : ImageSet \rightarrow Space_F$$
$$M_F : Space_F \times Space_F \rightarrow [0, 1]$$

The features set is called $FeatureSet$. The metric set is called $MetricSet$.

The metric associated to each feature can be computed with Euclidean, Hamming, Max, Min [3], edition distance, graph distance.

Definition 2. *Distance between document images*
The distance between document images I_1 and I_2 according the feature F is defined by:

$$D_F : ImageSet \times ImageSet \rightarrow [0, 1]$$
$$D_F (I_1, I_2) = M_F(F(I_1), F(I_2))$$

A distance computed between two documents from the same class is called an intra-class distance. Otherwise, it is called an inter-class distance. For each metric M_F and for each class C, an image is chosen as representative. This choice is detailed in section 5.

Notation 2 *The image which represents the class C according to M_F is called $I^*_{M_F,C}$.*

The distance between an image I and a class C according to M_F is defined as the distance between I and $I^*_{M_F,C}$.

Definition 3.

$$\begin{aligned} D_{(M_F,C)} &: ImageSet \to [0,1] \\ D_{(M_F,C)}(I) &= D_{(M_F,C)}\left(I^*_{M_F,C}, I\right) \\ &= M_F(F(I^*_{M_F,C}), F(I)) \end{aligned}$$

Notation 3 *intra-class and inter-class distances*
The set of inter-class distances between $I^*_{M_F,C}$ and images from C' is called $DX_{M_F,C}(C')$.
The set of intra-class distances $DX_{M_F,C}(C)$ is called $DI_{M_F,C}$.

4 Classification Principle

In classical approaches, a document image is represented by a point in a feature space. Then, classes correspond to clusters. The classification process consists in finding frontiers between these clusters. Our approach differs from commonly used classifiers. The feature space is projected in a one dimensional distance space. A point in this space represents the distance according a metric between a point of the feature space and a point of view. The representative $I^*_{M_F,C}$ defines a point of view. Thanks to the metric M_F, it is possible to compute the distances between $I^*_{M_F,C}$ and each other class.

Let $\tilde{I}$ be the image to classify and $\tilde{C}$ be the class of $\tilde{I}$. The computation of $D_{(M_F,C)}(\tilde{I})$ gives the position of $\tilde{I}$ with respect to C. If we know the relative position of every class C' and even if only one distance is computed, we can estimate the value of the membership function for each class. Indeed, if the distance between C and C' differs from the distance between C and $\tilde{I}$, the probability that $\tilde{I}$ is a member of C' is low.

More formally, for each class C', the set of inter-class distances $DX_{M_F,C}(C')$ between $I^*_{M_F,C}$ the representative of C according M_F and images from the C' is computed. $D_{(M_F,C)}(I)$ is the distance between the representative of C et $\tilde{I}$. Then, $\tilde{I}$ might be a member of every class C'' where $D_{(M_F,C)}(I)$ is close to $DX_{M_F,C}(C'')$.

This is the main idea of our approach. In classical methods, $\tilde{I}$ is considered as a member of the nearest class in the feature space. Our approach computes the distance d between $\tilde{I}$ and a point of view. $\tilde{I}$ is then considered as a member of a class whose distance to the point of view is close to d.

The classification module has $Card(MetricSet) \times Card(ClassSet)$ couples (M_F, C). Each of these can be a point of view. Then the training step of our module consists in computing the distances between the document images from the different classes.

The classification of an unknown document consists in selecting a sequence of couples (M_F, C) and exploiting the different classification hypothesis to identify the real class.

5 Learning

The learning phase is performed for each feature F and each corresponding metric M_F. First, a document image from $ImageSet_C$ is chosen for each couple (M_F, C) to represent the C class according M_F. This is performed by computing the intra-class distances $D_{M_F}(I, J), \forall (I, J) \in ImageSet_C \times ImageSet_C$. We deduce $I^*_{M_F,C}$ from the intra-class distances. Different choice might be implemented. The representative can be the gravity center, the image which minimize the greatest intra-class-distance or the image which minimizes the standard deviation of the intra-class distances. This choice may be the subject of future study. Our choice is the image which minimizes the standard deviation.

The second step of the learning phase consists in giving to each class the knowledge of the distance which separates it from other classes according to each metric. To do so, we could evaluate the distances between every document image from a C class and every document image from C' classes. But to reduce the training time, only the distances between the class representative of C and every document image from the other classes are computed. Theses distances allow to build the sets $DX_{M_F,C}(C')$ and $DI_{M_F,C}$ of inter and intra-class distances (cf. notation 3).

Each class has now the knowledge of the distance between its representative and every image from the other classes according every M_F.

Then, a membership function can be defined from the computed learning data. Indeed, according to each feature, each class has the knowledge of the distance with the other classes.

So if a document image is represented by a point in the feature space, and if the projection of this point in the distance space is located in the interval $[\min(DX_{M_F,C}(C')), \max(DX_{M_F,C}(C'))]$, then the image might be a member of the class C'. On the other hand, the more the point is far of this interval, the lower is the value of the membership function.

6 Classification

Classification is performed by iteratively filtering the list of candidate classes.

First, for each iteration, a (M_F, C) couple is selected. The module selects the couple for which the ranges of the $DX_{F,C}(C')$ are the most different for every C' in the candidate classes. The most informative couple is the one which maximizes the distances between intervals and which minimizes their intersection. These

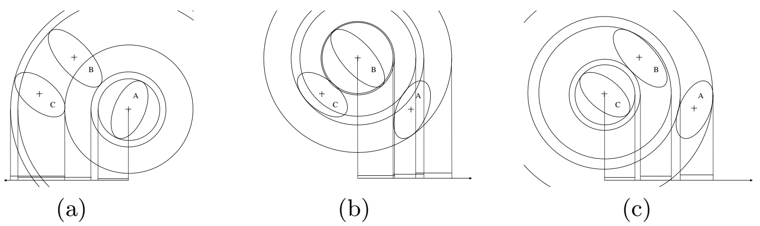


Fig. 1. Choice of the point of view

intervals in the distance space correspond to hyper-rings in the feature space (see Fig. 1(a), 1(b) and 1(c)). In other words, the couple (M_F, C) which gives the best point of view, if it is the one with the lowest intersection between hyper-rings. Then, an image represented by a point in the distance space, will belong to a low number of intervals.

Figures 1(a), 1(b) and 1(c) presents a problem with 3 classes and 1 feature space. It illustrates the discriminating power of different points of view. Indeed, if the selected point of view is the center of the class A, there is an intersection between the ring including the class B and the ring including the class C. This means that the corresponding intervals in the distance space overlap. The interval overlap is lower if the selected point of view is the center of class B. There is no more overlap if the center of class C is chosen. Then each point of the feature space belongs to one ring at most. This choice can be made in different feature spaces and with different metrics.

The distance $D_{(M_F,C)}(\tilde{I})$ between $\tilde{I}$ and the selected point of view $I^*_{M_F,C}$ is then computed with the metric of the selected feature F. This distance computation corresponds to draw a circle in $Space_F$ centered on the representative image of the selected class C. All classes which intersect the circle are kept in the candidate class list because the membership function is equal to 1. Classes whose membership function value is lower than a threshold are removed from the list.

This process is repeated until one class or less remains in the candidate list. The selection of the best point of view correspond to choose the feature space and the metric with the best discriminating power for the candidate classes. The iterative process corresponds to determine a path in a dynamically built decision tree [4]. If no class remains, $\tilde{I}$ is rejected.

The choice of the best point of view does not take into account the eliminated classes. However, their representative are still potential points of view.

```
Candidates ← $ClassSet$
while Card(Candidates) > 1
 choose best couple(F,C)
 compute d= $D_{(F,C)}(I)$
 remove far classes in Candidates according to $DX_{F,C}$
endWhile
if Card(Candidates) > 0
 then C=Candidates[0]
 else reject I
```

7 Experimental Results

The classification module contains 6 features (graphical [2] [5] and structural [1] [6]). For the presented test, the representative of each couple (M_F, C) is chosen as the image which minimizes the intra-class distance standard-deviation.

The tests have been performed on different bases containing images representing differents classes of cheque-deposits, forms and air flight coupons. Experimental results are given in Table 1.

As mentionned in section 2, the classes are defined by the position and the nature of the handwritten fields to be read. The preprocessing (eg. binarization) is an other source of variability. On the other hand, users only provide few images for learning (5 to 10 images per class). Fig. 2 shows examples of images from two different bases.

The results presented in table 1 have be compared with the ones given by a $k$nn classifier. In a first experiment, tests have been performed with a classifier

Table 1. Results

	#Images	#Classes	#Iterations	Good classification rate	Reject rate	Confusion rate
A	25125	2	1	100%	0%	0%
B	123	8	2 to 4	96%	4%	0%
C	65689	6	2 to 3	92%	4%	4%
D	8770	6	2 to 3	95%	4%	1%
C+D	10000	12	2 to 4	89%	1%	9%

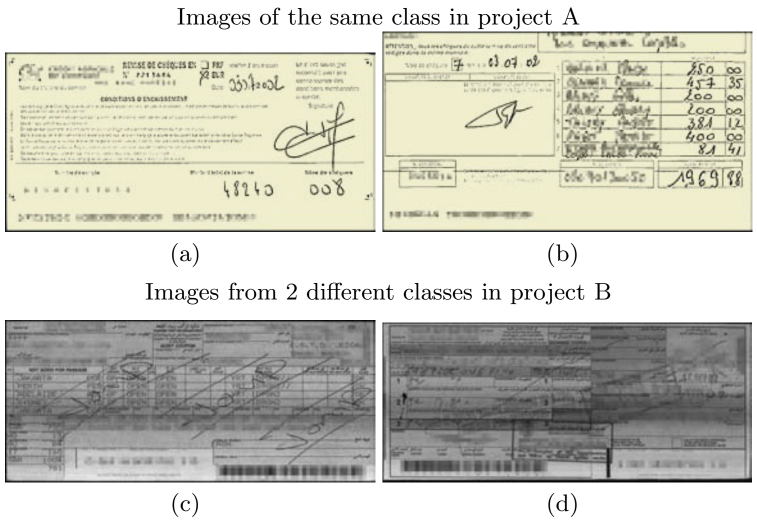

Images of the same class in project A
(a)
(b)
Images from 2 different classes in project B
(c)
(d)

Fig. 2. Example of images

for each of the 6 features. Several values of k have been tested (1, 3, 5 and 10) but it seems that it has a low influence on the results. The results also showed that some features are well suited for a problem and have a low discriminating power on an other. Indeed, the best configuration gave a classification rate of 96% on a problem, but the same feature gave classification rate lower than 60% on an other base.

In a second experiment, all the features have been grouped in a single feature vector. The results were lower than 40% for every base, the discriminating power of well suited features being disturbed by others. A feature selection should have been performed for each base.

This comparison validate the dynamic feature selection of our approach. Moreover, the $k$nn classifier needs to compute the distance with every point of the training set whereas our approach only compute a distance per iteration.

8 Conclusion

This article presents an original classification strategy which is applied to document image classification. This strategy is implanted as a module of a complete solution which aims at automatically reading handwritten fields on document images from multiple classes.

The strategy presents some advantages. First, the classification process is a sub-linear function of the number of classes. We do not have to compute the distance with every class representative. This allow to use it in an industrial context. Moreover, the module is easily configurable. By setting the number or the proportion of candidate classes to eliminate, a maximum number of iteration can be set. Lastly, the module architecture allows to add features very easily. In our system, a feature is only defined by its features space and its distance computation. Indeed, it is only required to write a function that places an image in the applied feature space and a distance function returning a real in $[0, 1]$. So for a particular difficult project with specific constraints, we can easily add features dedicated to the discrimination of the not easily separable classes.

At each iteration of the classification process, the best feature vector with the best point of view is dynamically determined.

The main advantage to work in distance spaces, is that it allows to use a large variety of feature. The processed feature vectors can be very different. There is no restriction concerning its size and nature. Numeric, syntactic and structural features can be mixed in the feature set and even in the same feature vector if a distance can be computed between two vectors.

The learning phase can be incremental. If a new class is added, the previous computed distance are kept, only the distances with the new documents have to be evaluated. New features can be added in the same way.

It seems that this strategy can be applied to a large scope of classification problems with supervised learning.

The strategy presented in this paper is a very preliminary work. The first results seem to be promising, but they could be improved by further works on many aspects (choice of the class representative, dynamic determination of the best feature vector for the best point of view).

Another future work would be to improve the used feature set. For example, the better feature vector could be determined to limit the computation complexity. The feature vectors could be combined to build new ones, for example by using genetic algorithms. We can imagine that the built feature vector would use features exploiting different zones of the document image. The genetic algorithms would then select the features associated to the most discriminating zones.

References

1. Héroux, P., Diana, S., Ribert, A., Trupin, E.: Classification methods study for automatic form class identification. In: 14th IAPR International Conference on Pattern Recognition ICPR'98, Brisbane, Australie, International Association on Pattern Recognition (1998) 926–928
2. Clavier, E.: Stratégies de tri: un système de tri des formulaires. PhD thesis, Université de Caen (2000)
3. Ribert, A.: Structuration évolutive de données: Application à la construction de classifieurs distribués. PhD thesis, Université de Rouen (1998)
4. Vannoorenberghe, P., Denoeux, T.: Handling uncertain labels in multiclass problems using belief decision trees. In: IPMU'2002. (2002)
5. Unser, M., Aldroubi, A., Gerfen, C.R.: A multiresolution image registration procedure using spline pyramids. In: Wavelet Applications in Signal and Image Processing. Volume 2034., SPIE (1993) 160–170
6. Azokly, A.S.: Approche uniforme pour la reconnaissance de la structure physique des documents composites baése sur l'analyse des espaces blancs. PhD thesis, Université de Fribourg (1995)

Identification of Humans Using Robust Biometric Features

Byungjun Son, Jung-Ho Ahn, Ji-hyun Park, and Yillbyung Lee

Division of Computer and Information Engineering, Yonsei University
134 Shinchon-dong, Seodaemoon-gu, Seoul 120-749, Korea
{sonjun,jhpark,yblee}@csai.yonsei.ac.kr
{jungho}@cs.yonsei.ac.kr

Abstract. The size of the feature set is normally large in a recognition system using biometric data, such as Iris, face, fingerprints etc. As dimensionality reduction is an important problem in pattern recognition, it is necessary to reduce the dimensionality of the feature space for efficient biometric identification. In this paper, we present one of the major discriminative learning methods, namely, Direct Linear Discriminant Analysis (DLDA). Also, we specifically apply the multiresolution decomposition of 2-D discrete wavelet transform to extract the robust feature set of low dimensionality from the acquired biometric data and to decrease the complexity of computation when using DLDA. This method of features extraction is well suited to describe the shape of the biometric data while allowing the algorithm to be translation and rotation invariant. The Support Vector Machines (SVM) approach for comparing the similarity between the similar and different biometric data can be assessed to have the feature's discriminative power. In the experiments, we have showed that that the proposed method for human iris and face gave a efficient way of representing iris and face patterns.

1 Introduction

The identification of humans for e.g. financial transactions, access control, or computer access has almost always been conducted by ID numbers, such as a PIN or a password. The main problem with those numbers is that, let aside the fact that they sometimes can be cracked quite easily, they can be stolen and used by and unauthorized person without detection. Biometric identification systems use personal features of the user itself to check the identity. If, for example, biometric features stored on a chip card are stolen, they cannot be used, because the imposter's biometric features do not match the features stored on the card. For this reason, the interest in biometric systems has risen very much lately. Many systems arise using eye, face, fingerprint, or voice features. All of them have different advantages and disadvantages. But they have the common disadvantage that the size of the feature set is normally large.

We discuss feature extraction strategies for a class of iris features with very high dimension. In a recognition system using the biometric features, one may try to use large feature set to enhance the recognition performance. However, the increase in the number of the biometric features has caused other problems. For example, the recognizer using higher dimension feature set requires more parameters to characterize the classifier and requires more storage. Thus, it will increase the complexity of computation and make its real-time implementation more difficult and costly. A larger amount of data is also needed for training. To avoid these problems, a number of dimensionality reduction algorithms have already been proposed to obtain compact feature set.

The feature extraction process needs to be effective so that salient features that can differentiate between various classes can be extracted [1]. Several methods like Principal Component Analysis (PCA) [2] [3], transform the input data so that the features are well separated and classification become easier. Linear transformations are extensively used because they are easy to compute and analytically tractable. Typically, these transformations involve projecting features from a high dimensional space to a lower dimensional space where they are well separated. Linear Discriminant Analysis (LDA) is one such discriminative technique based on Fisher's linear discriminant [4]. Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) are two major methods used to extract new features [4].

In this paper, we deal with basic issues for iris and face recognition. Also, we use the wavelet and direct discriminant analysis [5] for high-dimensional data set of iris and face. Most of works on personal identification and verification by iris patterns have been done in 1990s, and recent noticeable studies among them include those of [6], [7] and [8]. Daugman and Wildes implemented a whole system for personal identification or verifications including the configuration of image acquisition device, whereas the Boles system only focused on the iris representation and matching algorithm without an image acquisition module.

This paper is organized as follows. Section 2 briefly describes the image preprocessing which mainly involves the quality check of an image and iris localization. In section 3, we overview a multilevel two-dimensional 2D discrete wavelet transform (DWT) to obtain the feature vector of an iris and face image with lower dimensionality and more robust features. Also, we describe the DLDA scheme to linearly transform the subimages of biometric data obtained by wavelet transform to new feature space with higher separability and lower dimensionality. The same operations of DWT and DLDA are performed in training as well as testing phases. Section 4 describes feature matching approach based on SVM. Experimental results and analysis will be stated in section 5, and finally the conclusions are given in section 6.

2 Image Preprocessing

The images acquired from an image acquisition device always contain not only the appropriate images but also some inappropriate ones. Therefore, we need to check the quality of eye image to to determine whether the given images are appropriate for the subsequent processing or not and then to select the proper

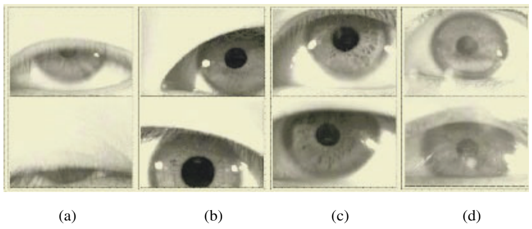


Fig. 1. Examples of images with bad quality: (a)the images with the blink (b)the images whose the pupil part is not located in the middle (c)the images obscured by eyelids or the shadow of the eyelids (d)the images with severe noises.

ones among them in real time. Some images ascertained as inappropriate ones are excluded from the next processing.

The images excluded from the subsequent processing include as follows; the images with the blink (Fig. 1(a)), the images whose the pupil part is not located in the middle and some parts of the iris area disappear (Fig. 1(b)), the images obscured by eyelids or the shadow of the eyelids (Fig. 1(c)), and the images with severe noises like Fig. 1(d). Fig. 1 shows the examples of images with bad quality.

An iris area can be localized from the eye image passed in the quality check step by separating the part of an image between the inner boundary and outer boundary. Fig. 2. shows the results of finding the inner boundary, the outer boundary and the collarette boundary in the eye image and the image of iris area, where is used in feature extraction, localized by using these boundaries.

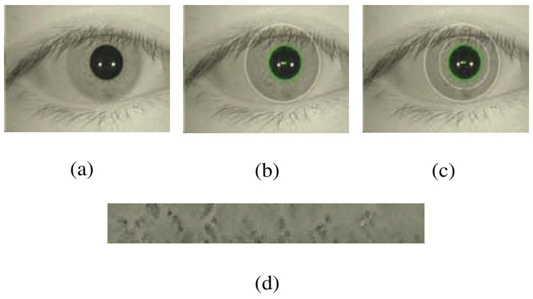


Fig. 2. (a)Original eye image (b)Image of the inner boundary and outer boundary (c)Image of the collarette boundary (d)localized iris image.

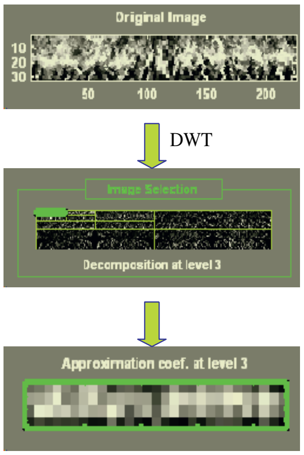
3 Feature Extraction

Most applications emphasize finding a feature set that produces efficient and implementable results. If the dimension of features defining a problem is too high, we must select a robust set of features from an initial set to provide appropriate representation. We also must design an appropriate classifier to the selected features set. We have chosen the DWT and DLDA approach to obtain a robust and lower dimensional set of features with high discriminating power.

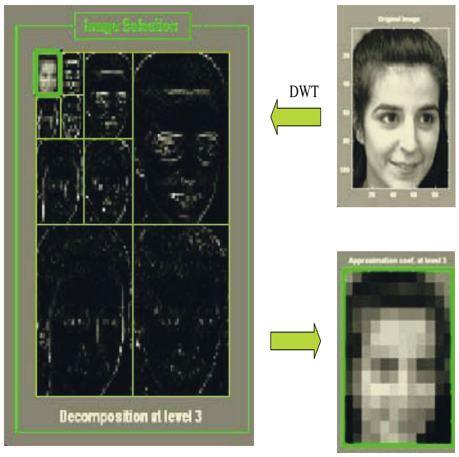
3.1 Wavelet Transform

The hierarchical wavelet functions and its associated scaling functions are to decompose the original signal or image into different subbands. The decomposition process is recursively applied to the subbands to generate the next level of the hierarchy. The traditional pyramid-structured wavelet transform decomposes a signal into a set of frequency channels that have narrower bandwidths in the lower frequency region. The DWT was applied for texture classification and image compression because of its powerful capability for multiresolution decomposition analysis. The wavelet decomposition technique can be used to extract the intrinsic features for the recognition of persons by their biometric data. We employ the multilevel 2D Daubechies wavelet transform to extract the iris features. Using the wavelet transform, we decompose the image data into four subimages via the high-pass and low-pass filtering with respect to the column vectors and the row vectors of array pixels.

Figure 3 shows the process of pyramid-structured wavelet decomposition.



(a) Iris



(b) Face

Fig. 3. Example of a 3-level wavelet transform of the iris and face images.

In this paper, we always select low frequency subimage for further decomposition. The three-level lowest frequency subimage is extracted as the feature vector. Generally, low frequency components represent the basic figure of an image, which is less sensitive to varying images. These components are the most informative subimages gearing with the highest discriminating power. The level of low frequency subimage chosen to extract the feature vector depends on size of the image. If the size is smaller then our localized iris image and ORL face, the one or two-level lowest frequency subimage might be have higher discriminating power.

3.2 Direct Linear Discriminant Analysis

After extraction of the iris and face feature vector by wavelet transform, the original iris vector x of 7,200 dimensions is transformed to the feature vector y of 116 dimensions. Also ORL face vector of 10,304 dimensions is reduced to 168 dimensions. To further reduce the feature dimensionality and enhance the class discrimination, we apply DLDA to convert the feature vector y into a new discriminant vector z with lower dimensions then the feature vector y.

Existing LDA methods first use PCA to project the data into lower dimensions, and then use LDA to project the data into an even lower dimension. The PCA step, however, can remove those components that are useful for discrimination. The key idea of DLDA method is to discard the null space of between-class scatter S_b - which contains no useful information - rather than discarding the null space of S_w , which contains the most discriminative information [5]. Each scatter is given as follows:

$$S_b = \sum_{i=1}^{J} n_i(\mu_i - \mu)(\mu_i - \mu)^T \qquad (n \times n)$$

$$S_w = \sum_{i=1}^{J} \sum_{x \in C_i} (x - \mu_i)(x - \mu_i)^T \qquad (n \times n)$$

where n_i is the number of class i feature vectors, μ_i is the mean of class i, μ is the global mean, and J is the number of classes.

The DLDA method is outlined below. We do not need to worry about the computational difficulty that both scatter matrices are too big to be held in memory because the dimensionality of input data is properly reduced by wavelet transform.

First, we diagonalize the S_b matrix by finding a matrix V such that

$$V^T S_b V = D$$

where the columns of V are the eigenvectors of S_b and D is a diagonal matrix that contains the eigenvalues of S_b in decreasing order. It is necessary to discard eigenvalues with 0 value and their eigenvectors, as projection directions with a total scatter of 0 do not carry any discriminative power at all [5].

Let Y be the first m columns of V (an $n \times m$ matrix, n being the feature space dimensionality),

$$Y^T S_b Y = D_b \qquad (m \times m)$$

where D_b contains the m non-zero eigenvalues of S_b in decreasing order and the columns of Y contain the corresponding eigenvectors.

The next step is to let $Z = YD^{1/2}$ such that $Z^T S_b Z = I$. Then we diagonalize the matrix $Z^T S_w Z$ such that

$$U^T(Z^T S_w Z)U = D_w \tag{1}$$

where $U^T U = I$. D_w may contain zeros in its diagonal. We can sort the diagonal elements of D_w and discard some eigenvalues in the high end, together with the corresponding eigenvectors.

We compute the LDA matrix as

$$A = U^T Z^T \tag{2}$$

Note that A diagonalizes the numerator and denominator in Fisher's criterion.

Finally, we compute the transformation matrix(2) that takes an $n \times 1$ feature vector and transforms it to an $m \times 1$ feature vector.

$$x_{reduced} = D_b^{-1/2} A x \tag{3}$$

4 SVM-Based Pattern Matching

We only give here a brief presentation of the basic concepts needed. The reader is referred to [10] for a list of applications of SVMs. SVMs are based on structural risk minimization, which is the expectation of the test error for the trained machine. This risk is represented as $R(\alpha)$, α being the parameters of the trained machine. Let β be the number of training patterns and $0 \leq \gamma \leq 1$. Then, with probability $1 - \gamma$ the following bound on the expected risk holds:

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\log(2\beta/h) + 1) - \log(\gamma/4)}{\beta}} \tag{4}$$

$R_{emp}(\alpha)$ being the empirical risk, which is the mean error on the training set, and γ is the VC dimension. SVMs try to minimize the second term of (4), for a fixed empirical risk.

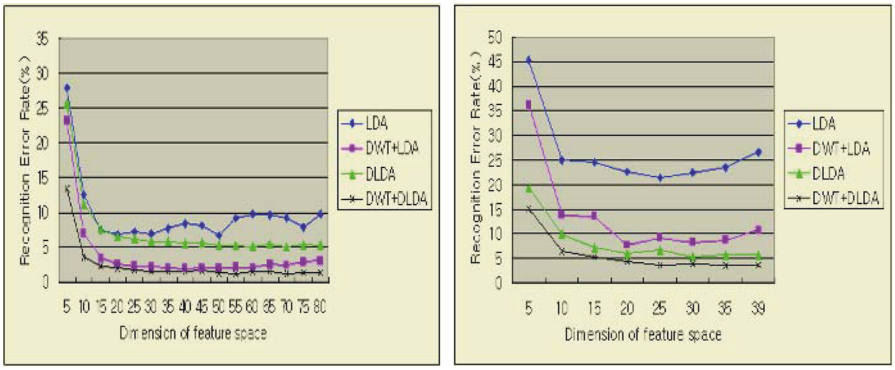
For the linearly separable case, SVM provides the optimal hyperplane that separates the training patterns. The optimal hyperplane maximizes the sum of the distances to the closest positive and negative training patterns. This sum is called *margin*. In order to weight the cost of missclassification an additional parameter is introduced. For the non-linear case, the training patterns are mapped onto a high-dimensional space using a kernel function. In this space the decision boundary is linear. The most commonly used kernel functions are polynominals, gaussian, sigmoidal functions.

5 Experimental Results

Eye images were acquired through CCD camera with LED (Light-Emitting Diode) lamp around lens under indoor light. The size of eye images is pixels with 256 grey intensity values, and the size of normalized iris images is 32×225 pixels. Our data set consists of 1200 eye data acquired from 120 individuals (left and right eye). The images of the left and right eye were treated differently from the same person, because they have different patterns. Both males and females were among the data. The ages of all ranges from the early twenties to mid-thirties. In case of individuals with glasses, images are captured both removing their glasses and having their glasses; however, contact lenses remained place.

We also used face images from Olivetti-Oracle Research Lab(ORL) [11]. The ORL data set consists of 400 frontal faces: 10 tightly cropped images of 40 subjects with variations in poses, illuminations, facial expressions and accessories. The size of each image is 92×112 pixels, with 256 grey levels per pixel.

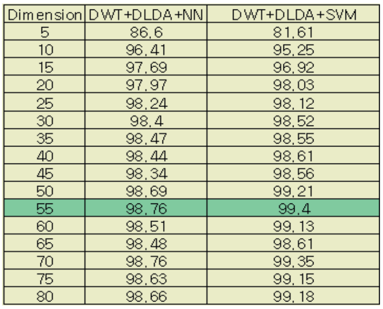
We randomly choose five images per person for training, the other five for testing. To reduce variation, each experiment is repeated at least 20 times. We applied LDA, DWT +LDA, DLDA, and DWT +DLDA to a training set. Also, we evaluated the recognition performances using nearest neighbor(NN) and SVM.



(a) Iris Results (b) Face Results

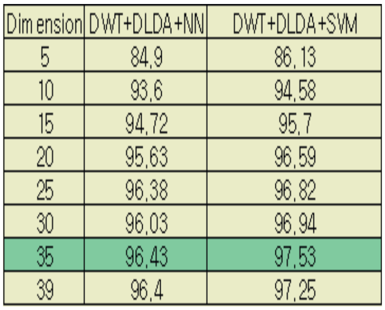
Fig. 4. Recognition Error Rate vs. Dimension of feature space.

Figure 4 shows the result of recognition error rate vs. dimension of feature space. The smallest recognition error rate of the LDA approach about the iris and face data is 6.7% and 21.42% with feature vector consisted of 20 and 25 components, respectively. The DLDA approach about the iris and face achieves 5.11% and 5.35 recognition error rate with feature vector consisted of 70 and 30 components, respectively. The DWT +LDA and DWT +DLDA approach about the iris achieve 1.96% and 1.24% with 40 and 55 features, respectively. About face achieve DWT +LDA and DWT +DLDA approach 7.75% and 3.57% with 20 and 35 features, respectively. the From Fig. 4, We can also see the DWT +DLDA method achieves the smallest recognition error rate. The recognition error rates of these DLDA methods are almost fixed after the number of features of iris and face reaches around 20. In addition, recognition rate of the DWT +DLDA approach about iris and face is 98.24% and 94.72% when the number



Dimension	DWT+DLDA+NN	DWT+DLDA+SVM
5	86.6	81.61
10	96.41	95.25
15	97.69	96.92
20	97.97	98.03
25	98.24	98.12
30	98.4	98.52
35	98.47	98.55
40	98.44	98.61
45	98.34	98.56
50	98.69	99.21
55	98.76	99.4
60	98.51	99.13
65	98.48	98.61
70	98.76	99.35
75	98.63	99.15
80	98.66	99.18

(a) Iris Results



Dimension	DWT+DLDA+NN	DWT+DLDA+SVM
5	84.9	86.13
10	93.6	94.58
15	94.72	95.7
20	95.63	96.59
25	96.38	96.82
30	96.03	96.94
35	96.43	97.53
39	96.4	97.25

(b) Face Results

Fig. 5. DWT+DLDA+NN vs. DWT+DLDA+SVM.

of features is 25 and 15, respectively. It is higher than the best performance of the other methods and has lower dimension than others. This shows that the DWT+DLDA approach can achieve better performance although it uses smaller number of basis vectors than the others.

As compared in Fig. 5, we find that DWT +DLDA +SVM for gaussian kernel outperforms DWT +DLDA +NN. Such observations are almost consistent for different numbers of feature set. The best recognition rate of DWT+DLDA +SVM approach about the iris and face is 99.4% and 97.53 when the number of features is 55 and 35, respectively.

6 Conclusion

In this paper, we have presented effective methods for the recognition of persons by their biometric features. We specifically uses the multiresolution decomposition of 2-D discrete wavelet transform for extracting the robust feature set of low dimensionality. In addition, the DLDA method is used to obtain the feature set with higher discriminative power and lower dimensionality. These methods of feature extraction well suit with iris and face recognition system while allowing the algorithm to be translation and rotation invariant.

We showed that the DWT+DLDA method outperformed the LDA, DWT +LDA, and DLDA in terms of classification rate. For the complex data consisting of many classes in the problem of iris and face recognition, the DWT+DLDA method can be used for an alternative of LDA. Support vector machines also has the advantage of efficient testing, and good performance compared to other linear classifiers. For future works, it is necessary to conduct experiments on a large number of data so as to verify the efficiency and robustness of our approach. Other techniques for feature extraction and pattern matching can be handled from this point of view so as to propose the efficient methods for a reliable human recognition system.

Acknowledgements

This work was supported in part by Brain Neuroinformatics Program sponsored by KMST.

References

1. H. Watanabe et al: " Discriminative Metric Design for Robust Pattern Recognition", IEEE Trans. On Signal Processing, vol. 45, PP. 2655-2662, 1997.
2. Ian T. Jollife:"Principal Component Analysis", Springer Verilag, New York, 1986.
3. Richard O. Duda, Peter E. Hart, David G. Stork:"Pattern Classification and Scene Analysis", Wiley Interscience, 2000.
4. W. L. Poston and Marchette, D. J.: "Recursive Dimensionality Reduction Using Fisher's Linear Discriminant," Pattern Recognition, 31(7):881–888, 1988.
5. Jie Yang, Hua Yu: "A Direct LDA Algorithm for High-Dimensional Data - with Application to Face Recognition," Pattern Recognition, 34(10):2067–2070, 2001.
6. John G. Daugman:"High confidence visual recognition of persons by a test of statistical indenpendence", IEEE Trans. On Pattern Analysis and Machine Intelligence, 15(11):1148–1161, January 1993.
7. R. P. Wiles: "Iris Recognition : An Emerging Biometric Technology," Proc. of the IEEE, 85(9):1348–1363 , 1997.
8. W. W. Boles, B. Boashash: "A Human Identification Technique Using Images of the Iris and Wavelet Transform," IEEETrans. of Signal Processing, 46(4):1185–1188, 1998.
9. Dimitrios Ioammou, Walter Huda, Andrew F. Laine: "Circle recognition through a 2D Hough Transform and radius histogramming," Image and Vision Computing, 7:15–26, 1999.
10. A. Tefas , C. Kotropoulos , and I. Pitas , "Using support vector machines to enhance the performance of elastic graph matching for frontal face authentication," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 7, 2001.
11. AT&T Laboratories Cambridge. The ORL Database of Faces. `http://www.cam-orl.co.uk/facedatabase.html`

Facial Expression Recognition Based on the Two-Dimensional Structure of Affect

Young-suk Shin

Department of Information and telecommunication Engineering, Chosun University, #375 Seosuk-dong, Dong-gu, Gwangju, 501-759, Korea
ysshin@mail.chosun.ac.kr

Abstract. We present an expression recognition system based on the two-dimensional structure of affect. The system is capable of identifying the various emotions using automated feature extraction. A method for extracting information about facial expressions from images is presented in three steps. In the first step, Gabor wavelet representation is constructed to provide edge extraction of major face components using the average value of the image's 2-D Gabor wavelet coefficient histogram. In the second step, sparse features of facial expression image are extracted using fuzzy C-means clustering(FCM) algorithm on neutral faces. In the third step, features of facial expressions are extracted using the Dynamic Linking Model(DLM) on expression images. The result of facial expression recognition is compared with dimensional values of internal states derived from semantic ratings of words related to emotion by experimental subjects. The two-dimensional structure of affect recognizes not only six facial expressions related to six basic emotions (happiness, sadness, surprise, angry, fear, disgust), but also expressions of various internal states.

1 Introduction

Face is an important social stimulus in human interactions. Specially, facial expression plays a major role in human communication. If a computer can understand emotions from human's facial expressions, it is possible to help humans in various situations dynamically.

Currently, most facial expression recognition systems use the six principle emotions of Ekman [1]. Ekman considers six basic emotions: happiness, surprise, fear, anger, disgust, sadness; and categorizes facial expressions with these six basic emotions. Facial action coding system is an analysis of facial muscle actions which make facial expressions based on the six basic emotions [2]. Most research on facial expression recognition includes studies using the basic emotions of Ekman[3, 4, 5, 6, 7], therefore these studies have limitations for recognition of natural facial expressions which consist of several other emotions and many combinations of emotions. Here we describe research extended on the dimension model of internal states for recognizing not only facial expressions of basic emotions but also expressions of various emotions.

Previous work on facial expression processing includes studies using representation based on optical flow from image sequences [3, 8, 9], principle components analysis of single image [9,10], physically based models [11], and wavelets transformation[12]. These methods are similar in that they first extract some features from the images, then these features are used as inputs into a classification system.

The first stage detects the edges of major face components, using the average value of the image's 2-D Gabor wavelet coefficient histogram on all the images. Since Gabor vectors with neighboring pixels are highly correlated and redundant, it is sufficient to use sparse pixels on a face. The second stage, FCM clustering algorithm is used to select sparse pixels from edges of major facial components extracted previously from a neutral face of each expressor. The third stage is an application of the Dynamic Link Architecture [13] and is used here to detect sparse local features on expression images from preselected points in the neutral face. Lastly, from a multilayer perceptron we recognize the facial expressions based on the two-dimensional structure of affect.

2 Database

The images used in this study were obtained from the Korean facial expression database for mapping of facial expressions into internal states [14]. This database consists of 500 facial expression images of males and females under well controlled lighting condition. Expressions were divided into two dimensions(pleasure-displeasure and arousal-sleep dimension) according to the study of internal states through the semantic analysis of words related with emotion by Kim et al. [15] using expressive 83 words.

These posed pictures may differ from the words of emotion between the posed expression and named expression since expressors depend strongly on the personal subjectivity. We thus picked up 44 internal state expressions being a high agreement of posed expression and named expression; and for our experiment used 11 expressions in a set of 44 internal state expressions from each of 6 person. The 11 expressions are happiness, surprise, sadness, disgust, fear, satisfaction, comfort, distress, tiredness, worry(including neutral face). A few of these are shown in Fig. 1. Our paper shows recognition of facial expressions using 2 dimension. The result of the dimension analysis of 44 emotion words related to internal emotion states is shown Fig. 2.

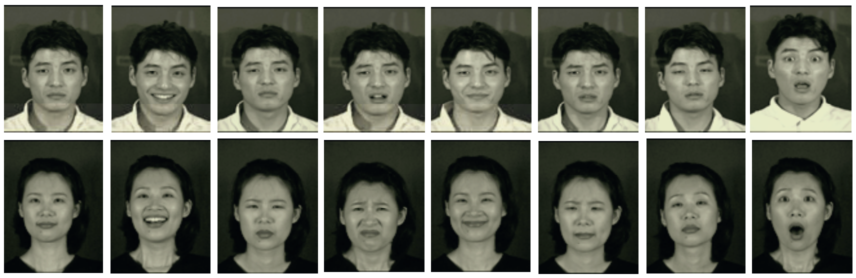


Fig. 1. Examples from the facial expression database

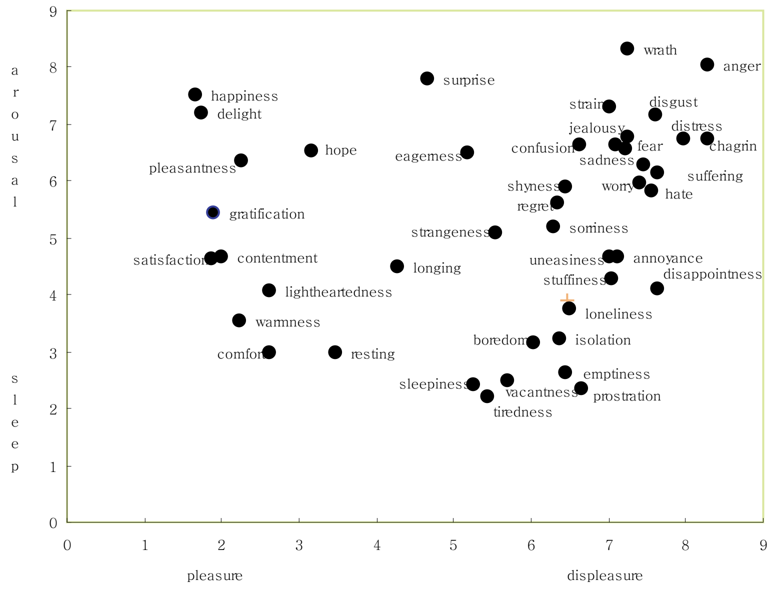


Fig. 2. The two-dimensional structure of affect

3 Automatic Feature Extraction

To extract information of facial expression, we use 287 images of 6 person, each image using 640 by 480 pixels included face images almost in the frontal pose. Original images have been rescaled and cropped such that the eyes are roughly at the same position with a distance of 60 pixels in the final image. In this section, we present the process of an automatic feature extraction. The process consists of three steps.

3.1 Preprocessing with Gabor Wavelets

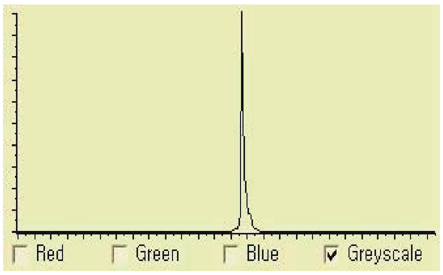
For edges of major facial components, an average value of the image's 2-D Gabor wavelet coefficient histogram is used. Each image was convolved with both even and odd Gabor kernels in an image $I(\vec{x})$ around a given pixel $\vec{x}=(x,y)$. The general form of two dimensional Gabor wavelets function is given by Daugman [16].

$$\psi_{\vec{k}}(\vec{x}) = \frac{k^2}{\sigma^2}\exp(-\frac{k^2x^2}{2\sigma^2})[\exp(i\vec{k}\cdot\vec{x}) - \exp(-\frac{\sigma^2}{2})] \tag{1}$$

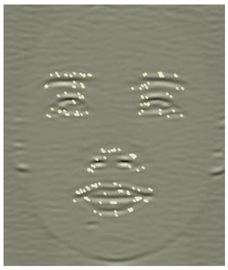
The wave vector $\vec{k}$ of length $k \equiv \left\|\vec{k}\right\|$ defines the spatial wavelength and at the same time controls the width of the Gaussian window. The parameter σ denotes the width of the Gaussian window relative to the wavelength corresponding to k. To detect features of major face components, we use a specific frequency band, a wave number, k=0.78, and 5 distinct orientations in 22.5 ° steps between 0 and π, and chose σ = π. The complex valued $\psi_{\bar{k}}$ applied to each image combines an even and odd part. We use only the magnitudes because they represent local information of an image in a smoothly varying way. Let G be the set of Gabor functions to be applied to *I*. G is G_1, G_2. The computation proceeds as follows:

$$\omega_1 = \sum\sum G_1\, I, \quad \omega_2 = \sum\sum G_2\, I$$
$$\varpi = \sqrt{(\omega_1^2 \ + \ \omega_2^2\,)}. \tag{2}$$

Fig. 3(a) shows the result of the 2-D Gabor coefficients histogram using the magnitudes of Gabor coefficients from an expression image. This means these coefficients completely capture local facial feature points in special frequency and special orientation. Thus, we applied the average value of 2-D Gabor coefficient histogram to extract local facial feature points. The average value of Gabor coefficients histogram is controlled by optional value ±α since experimental images may be a noise. Fig. 3(b) shows the resulting image which applied an optional value to an average value of the Gabor coefficients histogram.



(a)



(b)

Fig. 3. (a) 2-D Gabor coefficient histogram. (b) Extracted edges of major face components

3.2 Sparse Feature Points Extraction Using FCM Clustering Algorithm on Neutral Face

Extracted feature points are similar to edges of major facial components. Since Gabor vectors with neighboring pixels are highly correlated and redundant, it is sufficient to use sparse pixels on a face. We thus pick out sparse feature points based on the FCM clustering algorithm in edges extracted from the 2-D Gabor wavelet coefficient histogram. FCM algorithm applies to neutral facial images that are used as a template to extract sparse feature points from edges of major facial components on expression images.

Fuzzy C-means clustering [17] is a data clustering algorithm in which each data point belongs to a cluster to a degree specified by a membership grade. The potentiality of fuzzy clustering algorithms can be demonstrated by their application in clustering tasks which involve a large number of feature vectors of high dimension and a large number of clusters. Such an application is the codebook design required in image compression based on vector quantity, regardless of their initialization [18].

FCM consider the set X formed by N feature vectors from an N dimensional Euclidean space, $\mathbf{x}_i \in \Re^n \ \forall i = 1,2,...,N,\ X = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_N}\}$. The clustering is based on the assignment of the feature vector $\mathbf{x}_i \in X$ into c clusters, which are represented by $\mathbf{c}_i \in \Re^n,\ \ \forall i = 1,2,...,c$. The degree of the assignment of the feature vector $\mathbf{x}_i \in X$ into various clusters is measured by the membership function $u_{ij} \in [0,1]$, which satisfy the properties

$$\sum_{i=1}^{c} u_{ij} = 1, \forall_j = 1,...,N\,. \tag{3}$$

Fuzzy C-means algorithm is developed by solving the minimization problem. The cost function for FCM is

$$J(U, c_1, ..., c_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{j}^{N} u_{ij}^m d_{ij}^2. \tag{4}$$

c_i is the cluster center of fuzzy group i; $d_{ij} = \| \mathbf{c}_i - \mathbf{x}_j \|$ is the Euclidean distance between ith cluster center and jth data point ; and $m \in [1 < m\ ,\infty]$ is a weighting exponent. The necessary conditions for Equation (4) to reach a minimum are

$$\mathrm{c_i} = \sum_{j=1}^{N} u_{ij}^m X_j \Big/ \sum_{j=1}^{N} u_{ij}^m \tag{5}$$

$$u_{ij} = 1 \Big/ \sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}}\right)^{2/(m-1)}. \tag{6}$$

We determined sparse feature points using the following steps: **Step1**. Initialize the membership matrix U with random values between 0 and 1 such that the constraints in Equation (3) are satisfied. **Step2.** Calculate c fuzzy cluster centers ($c_i,\ i = 1,2,....,c$) using Equation (5). **Step3.** Compute the cost function according to Equation (4), and stop if either it is below a certain tolerance value or its improvement over previous iteration is below a certain threshold. **Step4.** Compute a new ***U*** using Equation (6), then go to Step2. Fig. 4(a) shows a result that extracted sparse pixel points by FCM algorithm: c=60, m=2. The number of clusters is decided in the range that can reflect the same topological relationship as major face components in human vision.

3.3 Sparse Feature Points Extraction Using DLM on Expression Image

After extracting the sparse feature points on neutral faces, which are used as a template to extract sparse feature points from edges on the expression images extracted previously since each neutral face plays a standard role to decide the degree of expression change against an expression image.

To match point to point feature points on an expression face against each feature point on a neutral face, it consists of two different domains, which are called the neutral domain (N) and the expression domain (E). The expression domain contain the jets of the Gabor transformation we defined in Equation (1) and Equation (2). The Gabor jet refers to the set of Gabor magnitudes obtained by sampling the image at the point $\vec{x}_i$ with sampling functions of all sizes (frequencies) and orientations, and refers to $\vec{J}_i^E$. The entire wavelet family for our study consists of two frequency bands, the wave number $k = \|\vec{k}\| = (\pi/4, \pi/8)$ using inverse pixels and seven different orientations from 0° to 180°, differing in 30° steps.

The linking procedure is performed under the constraint that the matching points found in the expression face have approximately the same topological relations as the preselected points in the neutral image. From the assumption, before starting a dynamic linking we apply FCM algorithm to edges of major face components on the expression images in the expression domain with the same cluster number applied for clustering edges of neutral facial image. A coarse-to-fine matching process performed to locate facial features on an expression image as follows.

1. A matching point should be chosen in the neutral face and then computed in the Euclidean distance between the preselected point in neutral face and the centroid point of each cluster clustered in the expression image in $\vec{\Delta}_{ij}^{NE} = \vec{x}_i^N - \vec{x}_j^E$. Thus, we expect to speed up mapping feature points between neutral face and expression face.
2. After selecting a cluster with the centroid point of a minimum Euclidean distance in an expression image, again we compute the Euclidean distance between all the points in the cluster selected in an expression image and the preselected point in neutral face. Here, we choose the candidate points in the cluster selected in an expression image with the minimum Euclidean distance or more the minimum Euclidean distance calculated in the step 1.
3. Among the candidate points in the cluster selected in an expression image should be chosen as a final point. We calculate a distance value between an average(ε) of the Gabor transformation's jets of all the points(n) in the cluster including the candidate points in an expression image, $\varepsilon = \frac{1}{n}\sum_{i=1}^{n} \vec{\mathbf{J}}_i^E$, and the Gabor transformation's jet of each candidate point in an expression image, $\vec{\mathbf{J}}_i^E$. Thus, a candidate point with a minimum distance value on the expression image is chosen as a final corresponding feature point against a preselected point in the neutral face. From the steps 2 and 3, we can reflect the local topology of non-rigid distortion of the face caused from changing expression. For the resulting the algorithm mentioned above see Fig. 4(b).

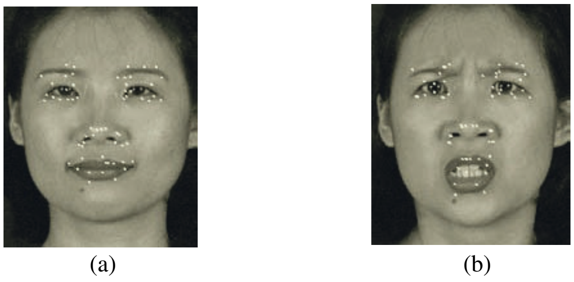


Fig. 4. (a) Sparse pixel points extracted with FCM algorithm on neutral face. (b) Sparse pixel points extracted with DLM on expression image

4 Recognition and Discussion

The system for facial expression recognition uses a three-layer neural network. The first layer is the distance values from each feature point on a neutral face to each feature point on an expression face which are normalized by size from 0 to 1. The second layer is 240 hidden units and the third layer is two output nodes to recognize the two dimensions: pleasure-displeasure and arousal-sleep.

Training applies error back propagation algorithm which is well known to the pattern recognition field. The activation function of hidden units uses the sigmoid function. 250 images for training and 37 images excluded from the training set for testing are used. The first test verifies with the 250 images trained already. Recognition result produced by 250 images trained previously showed 100% recognition rates. The rating result of facial expressions derived from the semantic rating of emotion words by subjects is compared with experimental results of a neural network (NN). The similarity of recognition result between human and NN is computed in Equation (7). The dimension values of human and NN in each two dimension are given as vectors of $\vec{H}$ and $\vec{N}$.

$$S(\vec{H},\vec{N}) = \frac{\vec{H}\cdot\vec{N}}{\|\vec{H}\|\|\vec{N}\|}\min\left(\frac{\|\vec{H}\|}{\|\vec{N}\|},\frac{\|\vec{N}\|}{\|\vec{H}\|}\right) \tag{7}$$

Table 1 describes a degree of similarity of expression recognition between human and NN on the two-dimensional structure of affect. In Table 1, the result of expression recognition of NN is matched to the most nearest emotion word in 44 emotion words related to internal emotion states. The result of expression recognition of NN looks very similar to the result of expression recognition of human. The displeasure emotions of high level arousal dimension seems like fairly generalized and stabled emotions in expression space. Such expressions are surprise, disgust, fear, distress, and worry. It seems that the quantity of physical changes can be detected easily. High level arousal dimension in the pleasure dimension also seems like a important component to discriminate the expression images. For instance, satisfaction, tiredness, and comfort show low level arousal dimension (sleep dimension) in the pleasure dimension, while happiness shows high level arousal dimension in the pleasure dimension.

Table 1. The result data of expression recognition between human and NN(Pleasure-Displeasure: P – D, Arousal-Sleep: A – S)

Emotion words	Human(Mean)		Neural Network		Similarity	Recognition on Neural Network
	P – D	A – S	P – D	A – S		
happiness	1.65	7.53	3.88	3.44	0.54	lightheartedness
			4.92	4.6	0.71	boredom
			2.86	5.86	0.82	pleasantness
			1.31	5.69	0.75	gratification
			4.43	4.8	0.73	Longing
satisfaction	1.85	4.65	1.49	6.07	0.79	pleasantness
			2.14	4.96	0.92	contentment
			6.32	5.9	0.52	shyness
comfort	2.61	2.98	5.0	5.7	0.52	strangeness
			3.65	3.64	0.77	lightheartedness
sadness	7.22	6.57	7.07	5.23	0.89	shyness
			3.7	6.37	0.72	hope
			6.62	7.12	0.91	surprise
tiredness	5.44	2.2	7.94	6.29	0.56	strain
			4.06	4.05	0.90	sleepiness
			4.39	4.28	0.89	longing
			4.8	5.09	0.76	strangeness
			6.39	5.65	0.65	uneasiness
worry	7.4	5.96	6.89	6.09	0.97	confusion
			7.39	6.84	0.94	strain
surprise	4.65	7.8	4.55	8.29	0.95	surprise
			4.61	7.67	0.98	surprise
			4.65	5.60	0.79	hope
disgust	7.93	6.74	6.35	3.42	0.68	isolation
			7.33	6.14	0.91	hate
			7.68	6.03	0.98	distress
			6.05	6.72	0.86	surprise
Fear	7.25	6.77	6.75	4.49	0.80	sadness
			6.43	5.21	0.83	stuffiness
			6.68	7.97	0.94	disgust
			7.30	7.96	0.91	chagrin
			5.91	4.17	0.72	isolation
distress	7.46	6.29	7.48	7.16	0.94	disgust
			4.28	5.81	0.72	hope
			4.77	4.97	0.70	boredom
			5.60	4.11	0.71	boredom
			5.81	5.05	0.79	strangeness

This study is a new approach of human's emotion processing, it is interesting to note in this context that machine vision may represent various emotions similar to human with the combination of each dimension in the internal emotion states. To future study we are planning to recognize the expressions with person independent and a wider range of emotions in much larger database than present system.

References

1. Ekman, P.: Universal and cultural difference in facial expressions of emotions. In: J. K. Cole(Ed.), Nebraska symposium on motivation, Lincoln: University of Nebraska Press, (1972) 207-283
2. Ekman, P., Friesen, W.V.: Facial action coding system. Consulting Psychologists, Palo Alto, CA., (1977)
3. Mase, K.: Recognition of facial expression from optical flow. IEICE Transactions, E 74, **10** (1991) 3473-3483
4. Essa, I., Pentland, A.:Coding, analysis, interpretation, and recognition of facial expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence, **19** (1997) 757-763
5. Lien, J.: Automatic recognition of facial expressions using hidden Markov models and estimation of expression intensity. Ph.D. Thesis, Carnegie Mellon University, (1998)
6. Oliver, N. Pentland, A., Berard, F.: LAFTER:a real-time face and lips tracker with facial expression recognition. Pattern Recognition **33** (2000) 1369-1382
7. Cohen, I., Sebe, N., Garg, A., Chen, L. S., Huang, T. S.: Facial expression recognition from video sequence:temporal and static modeling. Computer Vision and Image Understanding , In Press (2003)
8. Yacoob, Y., Davis, L.S.: Recognizing human facial expression from long image sequences using optical flow. IEEE Trans. Pattern Anal. Machine Intell. 18(6) (1996) 636-642
9. Bartlett, M., Viola, P., Sejnowski, T., Larsen, J., Hager, J., Ekman, P.: Classfying Facial Action. In: Advances in Neural Information Processing Systems 8. D. Touretzky et al. editors, MIT Press, Cambridge, MA (1996)
10. Padgett, C., Cottrell, G.: Identifying emotion in static face images. In Proceeding of the 2nd Joint Symposium on Neural Computation, 5 (1995) 91-101
11. Essa, I. Pentland, A. : Facial Expression Recognition using Visually Extracted Facial Action Parameters. Proceedings of the International Workshop on Automatic Face and Gesture Recognition (1995) 35-40
12. Lyons, M., Akamatsu, S.:Coding facial expressions with Gabor wavelets. Proceeding of the Third International Conference on Automatic Face and Gesture Recognition, (1998) 200-205
13. von der Malsburg, C.: Nervous structure with dynamical links. Ber. Bunsenges. Phy.Chem, **89** (1985) 703-710
14. Bahn, S., Hahn, J. and Chung, C.: Facial expression database for mapping facial expression onto internal state. '97 Emotion Conference of Korea, (1997) 215-219
15. Kim, Y., Kim, J., O, S., O, K., Chung, C.: The study of dimension of internal states through word analysis about emotion. Korean Journal of the Science of Emotion and Sensibility, **1** (1998) 145-152
16. Daugman, J: Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. Journal of the Optical Society of America **2** (1985) 1160-1169
17. Bezdek, J.C.: Fuzzy mathematics in pattern classification. Ph.D. thesis, Applied Math. Center, Cornell University, Ithaca (1973)
18. Karayiannis, N.B., Pai, P.-I.: Fuzzy vector quantization algorithms and their application in image compression. IEEE Transactions on Image Processing, (1995)

Clustering Using WCC Models

Miguel Adán[1], Antonio Adán[2], and Andrés S. Vázquez[2]

[1] Departamento de Matemática Aplicada, UCLM
13071 Ciudad Real, Spain
{Miguel.Adan@uclm.es}

[2] Departamento de Ingeniería Eléctrica, Electrónica y Automática, UCLM
13071 Ciudad Real, Spain
{Antonio.Adan,Andres.Vazquez}@uclm.es

Abstract. In this paper we show how Weighted Cone-Curvature *(WCC)* Models are suitable to carry out clustering tasks. *CC* is a new feature extracted from mesh models that gives an extended geometrical surroundings knowledge for every node of the mesh. *WCC* concept reduces the dimensionality of the object model without loss of information. A similarity measure based on the *WCC* feature has been defined and implemented to compare 3D objects using their models. Thus a similarity matrix based on *WCC* corresponding to an object database is the input of a fuzzy c-means algorithm to carry out an optimal partition of it. This algorithm divides the object database into disjoints clusters, objects in the same cluster being somehow more similar than objects in different clusters. The method has been experimentally tested in our lab under real conditions and the main results are shown in this work.

1 Introduction

Clustering is a well known topic in the image processing field. Roughly speaking, the clustering's goal is to achieve the best partition over a set of objects stored in a database in terms of similarity. For partitioning we need to extract or define features of each object in such a way as to be well characterized. Depending on how that information was dealt with, several strategies of clustering can be found in the literature.

Methods based on perceptual/functional organizations aim to make hierarchical procedures where the step from one level to another must be controlled in some way. A perceptual organization mechanism is developed by Sengupta and Boyer in [1, 2] where surfaces belonging to the same object are identified. A graph is constructed, each surface corresponding to a node in this graph. Finally a partitioning scheme is carried out by comparing graphs corresponding to the objects. Selinger et al. argue that a single level of perceptual grouping is inadequate for recognition and use four levels of perceptual grouping [3]. In [4] a generic clustering scheme combining structural and functional approaches is presented. In this, a mapping of functionality to the primitive shape parts is the base for classifying objects.

Similarity-base clustering is a simple technique that uses a similarity measure to guarantee if two objects are similar enough to put them in the same cluster. The similarity measure is usually defined through features of the object. In this sense, a good similarity measure is essential to carry out further clustering tasks. Lately, several works can be mentioned in this area. Yeung and Wang [5] introduce a similarity measure based on feature weight learning which is a reduction of the uncertainty existing in the clustering process. Thus clustering performance is improved. A stochastic clustering algorithm over silhouettes is accomplished in [6] where, in order to obtain a silhouettes database, a large number of views of each object is processed. Then, a dissimilarity matrix is obtained and a clustering algorithm is run over it. Cyr and Kimia [7] measure the similarity between two views of the 3D object by a metric that measures the distance between their corresponding 2D projected shapes. Ohbuchi et al. [8] present a version of the shape functions proposed by Osada [9] for 3D polygonal mesh models that allow them to make an efficient shape similarity search.

Lately we have developed a new strategy for 3D objects recognition using a flexible similarity measure based on spherical mesh models called Cone Curvature models [10, 11]. The difference between other strategies and ours is that we are able to compare two objects taking any part of information of the mesh model. In this sense, it can be said that our method is ‘flexible’ to experimental specifications. Consequently, an adaptable (or flexible) similarity measure contrary to previous fixed similarity measures is defined in our case. Secondly, we have carried out a reduction of the dimensionality of the object representation. So, unlike other techniques, no redundant information but a synthetic characterization is used. This reduction notoriously simplifies the volume of data handled in the model without loss of information and alleviates the computational cost of algorithms based on the model.

Now we present the applicability of such a method for clustering where our similarity matrix is the input of a fuzzy c-means algorithm. To show this we have structured the paper as follows. In Section 2, Weighted Cone Curvature model is briefly described defining Cone-Curvature and Weighted Cone Curvature as features of the object. Section 3 is devoted to defining a similarity measure and presenting the clustering algorithm. Clustering experimentation is dealt with throughout Section 4 presenting the results of a set of the tests.

2 WCC Models

2.1 Cone-Curvature Feature

Our solid representation model is defined on a mesh of *h* nodes from the tessellation of the unit sphere. Let T_I be this initial spherical mesh and T_M the mesh fitted to the object . For building T_M, T_I is deformed until it fits into the normalized surface of the object. In this process, mesh regularizing/smoothing tasks are also included. Then several geometric features are extracted from T_M and finally, mapped again into T_I.

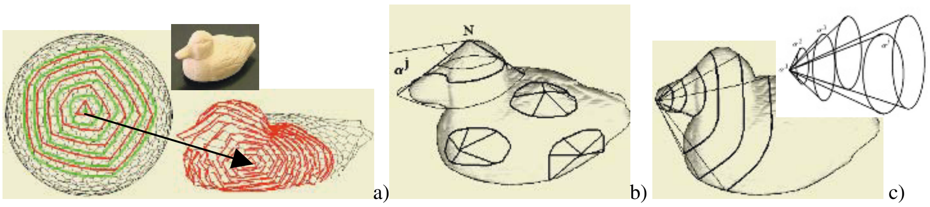


Fig. 1. a) MW drawn over T_I and a detail of WFs over T_M. b) Definition of CCs. b) Visualization of the CCs vector for a node N.

On the initial tessellation T_I, a topological structure called *Modeling Wave* (MW) [12] organizes the nodes of T_I in disjointed subsets following a new relationship. In this sense a three-neighbour relationship is just a kind of local topology. Each subset contains a group of nodes spatially disposed over the sphere as a closed quasi-circle, resulting in subsets that look like concentric rings on the sphere. Since this organization resembles the shape of a wave, this has been called *Modeling Wave (MW)*. Consequently each of the disjointed subsets is known as *Wave Front (WF)* and the first WF is called *Focus*. Of course, MW structure remains after the modeling process has finished. In other words, the WF structures remain in T_M (see Figure 1 a)).

From the previous definition it can be deduced that any node of T_I may be *Focus* and, therefore, it can generate its MW. Therefore *h* different MWs can be generated.
Although several kinds of features have been mapped into T_I in previous works [12] in this case Cone-Curvature (CC) is defined as a new and intuitive feature based on the MW structure taking into account the location of the WFs inside the model T_M. Its formal definition is as follows:

Let N be *Initial Focus* on T_M. We call *j*th Cone Curvature α^j of N, the angle of the cone with vertex N whose surface inscribes the *j*th *Wave Front* of the *Modeling Wave* associated to N.

The range of CC values is [-π/2, π/2], being the sign assigned taking into account the relative location of O, C^j, and N, where O is the origin of the coordinate system fixed to T_M and C^j is the barycentre of the *j*th WF. Negative values are for concave zones, values next to zero correspond to flat areas and positive values correspond to convex zones. Figure 1 b) illustrates this definition.

Note that a set of values $\{\alpha^1, \alpha^2, \alpha^3, \ldots \alpha^q\}$ gives an extended curvature information around N until the *q*th WF, where the word 'curvature' has a non-local meaning. So for each node N a set of *q* values could be used for exploring its surroundings (see Figure 1 c)).

On the other hand, it can be said that a vector $C=\{c^1, c^2, c^3 \ldots c^q\}$ where $c^j : T_I \rightarrow [-\pi/2, \pi/2] \quad j=1,\ldots q$, *q* being the number of *Wave Front* considered, is established for all the nodes of T_I. The whole Cone Curvature information is stored in a CC-matrix Θ, of *h*x*q* dimension. Note that Θ is invariant, unless row permutations, to changes in the pose of the object.

2.2 Weighted Cone Curvature

Now we show a study for the dimensionality reduction of the CCs by defining a new feature called Weighted Cone-Curvature (WCC). To do that a principal component analysis must be carried out over the CC vectors.

With the purpose of showing the degree of correlation existing between different curvature orders, correlation values for CCs from 2nd to 18th order are plotted in Figure 2 a) (notice that 1st order has no meaning). In this Figure, the correlation values are coded in grey levels (corresponding 0 to black and 1 to white). Each plotted value is computed as the average of the correlation values obtained for all the objects in the handled database (70 objects). It can be seen that, in general, there are very high correlation values for near orders of CC. On the contrary, small correlation values are found between lower orders of CC (2nd and 3rd) and all the others. It is also noticeable from the same Figure the high correlation existing between the upper orders of CC.

The meaning of *h* and *q* dimensions of the *O* CC-matrix could be explained as follows. The choice of a particular row N is equivalent to selecting the *Focus* of the MW whereas *q* provides CCs values corresponding to a specific depth; the higher order the more depth.

Our purpose is to obtain a single value for each row of the *O* matrix from the analysis of the principal components performed on all the rows, so that each row is reduced to a single representative value (see Principal Component Analysis or Karhunen-Loeve transform in [13]). Then, by means of the adequate linear combination, for each node N a single variable c_w will fuse the *q* values provided by its CCs. Therefore, every node will have just a variable c_w associated that is called Weighted Cone Curvature (WCC). Consequently, the *O* matrix is reduced to a vector *C* of *hx1* dimension.

The variable Weighted Cone Curvature can be defined as a weighted combination of the different CCs given by the expression:

$$c_w = \sum_j v^j \cdot c^j \ . \tag{1}$$

The coefficients v^j of the linear combination are the coordinates of the eigenvector associated to the highest eigenvalue of the covariance matrix for the set of indexes *j* considered. These coefficients have been empirically determined by evaluating the principal component analysis over the models of our object database in our lab.

Note that different combinations of CC orders with respect to different criteria could be chosen. So, local, half, global, contiguous or discontiguous CCs could be chosen. That is why labeled our method as 'flexible'. Because we are interested in using the CCs for partial views we have considered the CC of orders *j=4,5,...9*. This means to consider only the CC information near to the *focus* but filtering also the first two orders (which are sensitive to noise).

The study of the principal components has been carried out over a database with 70 objects. Figure 2 b) shows a plot of the highest eigenvectors. Each line corresponds to the highest eigenvector obtained for a given object where the X axis represents the index and Y axis represents the eigenvector values. Very similar eigenvec-

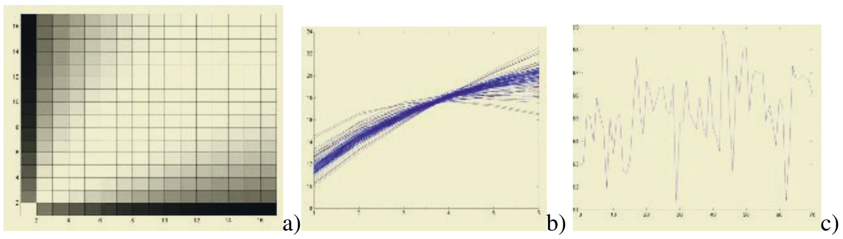


Fig. 2. a) Illustration of the correlation between different CC values from 2nd to 18th orders, in the object database. b) Plots of the highest eigenvectors. c) Percentage of the total variance that corresponds to c_w.

tors v^j can be appreciated for most of the objects showing higher dispersion for the greater indexes.

Finally, we consider the weighted array $\{v^j\}$ by computing the eigenvectors average obtained above. To evaluate the goodness of the dimensionality reduction an analysis of the percentage of the total variance that corresponds to the new variable is required. If a high percentage is achieved the new variable will satisfactorily explain the initial variables and the dimensionality reduction makes sense. In our study this percentage has been 95,34%. which means that the dimensionality reduction does not provoke any significant loss of information. Figure 2 c) shows, for all objects, the results of the percentage of the total variance that corresponds to the new variable c_w.

With these results it can be concluded that the dimensional reduction of the CC vector to a single variable c_w is very appropriate because it adequately summarizes the information provided by the entire vector. Each node of T_I will have a numeric value associated, called Weighted Cone Curvature, and the complete model can be characterized by the WCC vector C of h components.

3 Clustering Based on Similarity Matrix

In this section, a similarity measure for 3D shapes based-on WCC feature is firstly defined. After that the procedure of clustering is dealt with.

Keeping the WCC concept in mind, a distance d between two models T^i and T^j is defined as:

$$d(T^i,T^j) = \left|C^i - C^j\right| = \sqrt{\sum_{k=1}^{h}(C^i(k) - C^j(k))^2} \tag{2}$$

where C^i and C^j are sorted distributions of WCC vectors for both models.

Once the distance d has been defined and considering a model database where D is the maximum distance, a binary relationship through a *similarity function s* is established as follows.

$$s: X \times X \to [0,1] \qquad s(T^i,T^j) = s_{ij} = 1 - d(T^i,T^j)/D \tag{3}$$

X being the model database. Thus we can define a Similarity Matrix $S= (s_{ij})$ which stores the whole similarity information for a database.

Roughly speaking, clustering procedures yield a data description in terms of clusters or groups of data points that possess strong similarities [14]. In our case, we have an object database *H* that we want to divide in groups of objects. We may consider *H* as a set of *n* unlabelled samples $\mathbf{x_1}$, $\mathbf{x_2}$, …$\mathbf{x_n}$ where $\mathbf{x_i}$ is a feature vector of the *i*th object. For us, every object is characterized by the vector of similarity values with respect to the rest of the objects so $\mathbf{x_i}=(s_{i1},\ldots s_{in})$. Note that $\mathbf{x_i}$ corresponds to *i*th row into the *Similarity Matrix S*. Therefore the clustering procedure would be to divide *H* into *p* disjoint subsets H_1, H_2,…H_p, samples in the same cluster being somehow more similar than samples in different clusters.

Once we fix the clustering and similarity measurement problems, we are obliged to evaluate the partitioning. Then, the problem is one of finding the partition that extremizes a criterion function. For that we have used the fuzzy c-means function, which is a variation of sum-of-squared-error criterion. Although complete information about this can be found in [15], we will make a brief reference to this criterion function.

For a given integer $p>1$ and real $m>1$, the criterion function to be minimized is defined as follows:

$$J=\sum_{i=1}^{p}\sum_{k=1}^{n}(u_{ik})^m\left|x_k-m_i\right| \tag{4}$$

where $\mathbf{m_i}$ is the mean vector in H_i, $|\mathbf{x}|$ denotes the Euclidean norm of $\mathbf{x}$ and u_{ik} is the *i*th membership function on the *k*th sample $\mathbf{x_k}$ to the cluster H_i. The vectors $\mathbf{m_1}, \ldots, \mathbf{m_p}$ can be interpreted as prototypes of cluster (called *cluster centers*). So, high memberships occur for samples close to the corresponding cluster centers. The number *m* is called the *exponent weight* and is used to control the contribution of the samples to *J* depending on their memberships values. Once chosen *p* and *m*, an iterative procedure recomputes $\mathbf{m_i}$ and u_{ik} until a local minimum of *J* is achieved.

4 Experimental Tests

Taking our similarity measure as the basis of the clustering procedure, we have evaluated it by carrying out several tests for different sets of objects. This experimentation has been accomplished with 41 free form objects sensed in the lab where different people participated in their election. Curiously, there were different opinions for establishing the clusters a priori. Obviously the choosing of natural groups was not completely clear for us. Finally we grouped the objects in sub-sets that we have labeled as: *cubes (1-6)*, *prisms (7-9)*, *round shapes (10-13)*, *cars (14-17)*, *polyhedral (18-23)*, *free shapes (24-29)*, *cone shapes (30-36)* and *cylinders (37-41)* (Figure 3). The goal of our experimentation is to know what the performance of our method in real environments is and what concordance exists between the clusters computed with the clusters established a priori.

In order to graphically illustrate the Similarity Matrix *S* we have presented it as a two dimension grey map where the grey level goes from 0, plotted as white, to 1, plotted as black. This election, that seems illogical, has been adopted in order to achieve better visualization. Note that reflexive (black diagonal) and symmetrical properties can be clearly seen on it. Looking at the grey levels for the *i*th row we can check the feature vector (called $\mathbf{x_i}$ in section 3) which is the similarity vector. Making a visual analysis of *S* we can sometimes appreciate dark grouping zones for a set of rows (or columns indistinctly). These groups might be thought of as candidate clusters.

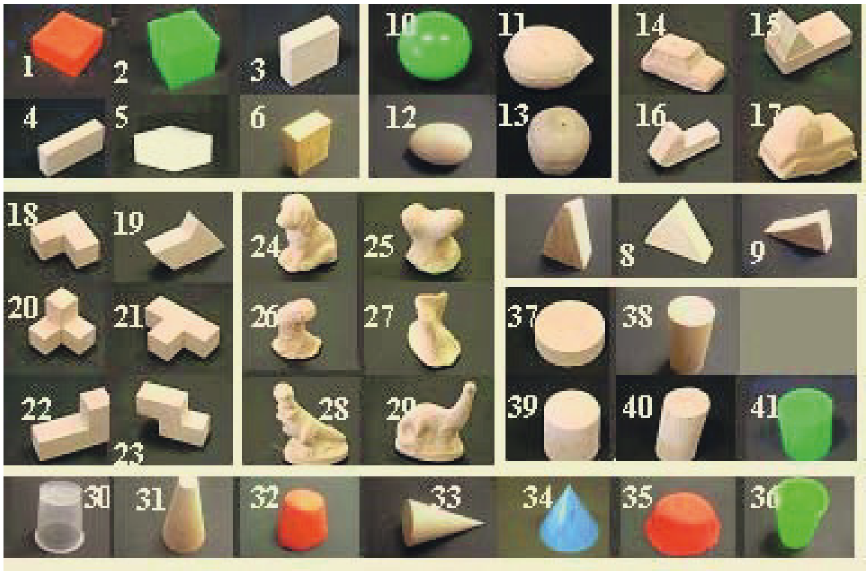


Fig. 3. Objects.

When a set *H* of *n* objects (samples) $\mathbf{x_1}, \mathbf{x_2}, \ldots \mathbf{x_n}$ is considered, each object $\mathbf{x_i}$ is inside a *n*-dimensional space ($\mathbf{x_i}$ has *n* components) and therefore the resulting cluster is in a *n*-dimensional space. In order to graphically show the clusters we have included two-dimensional projections of that space (corresponding to first and last dimensions).

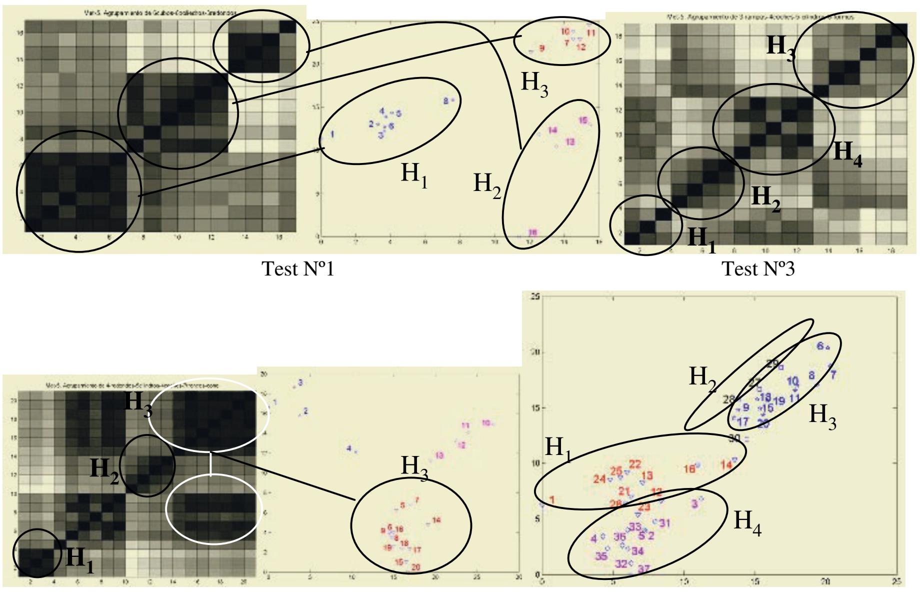
Table 1 summarizes the results obtained for twelve tests. For each test, a set of object *H* belonging to different groups are chosen. Then the clustering algorithm is run and clusters $H_1,\ldots,H_p$ are computed. The number of cluster *p* is set taking into account the mean values of the membership functions of the objects in their respective clusters (denoted as *u* in the table). In the first column the name of the groups set a priori appears and last rows shows the cases where an object is put in an unexpected cluster.

There are cases where clustering results coincide quite well with the natural groups (test nº: 1, 3, 7, 10). For example in Test nº1, the three clusters correspond to *cubes,* *round shapes* and *polyhedral.* This grouping can be seen clearly in the *Similarity Matrix* (Figure 4 above) because three disjointed dark zones appear. On the right, four groups are also evident in the *Similarity Matrix* corresponding to Test nº3.

On the other hand, when the number of cluster is forced to be less than the number of initial groups, cases exist where two groups are put in the same cluster (test nº: 2, 4, 5, 6, 8, 9, 11 and 12). Nevertheless, a coherent clustering is made in such cases. Test nº 2 is a good example confirming that. In this case *H* is formed by *cars*, *polyhedral*, *cones* and *cylinders*. When the clustering procedure is executed for *p=3*, *cones* and *cylinders* are put in the same cluster. Figure 4 below shows the nearness of such groups. Test nº 4 is a more complex case where this coherence can also be seen.

Table 1. Results for tests.

Test	*1*	2	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
p	3	3	4	4	3	2	3	3	3	3	3	5
u	0.93	0.92	0.72	0.81	0.90	0.86	0.88	0.88	0.82	0.85	0.84	0.76
n	16	20	18	37	37	18	19	19	23	11	22	41
Cubes	H_1			H1	H1	H1	H1				H1	H1
Prisms			H1	H1	H1			H1		H1	H1	H1
Round shapes	H2	H1		H2	H2			H2		H2	H2	H2
Cars		H2	H2						H1	H3	H3	H3
Polyhedrals	H3			H3	H3		H2	H3	H1			H3
Free shapes			H3	H3	H3			H3	H2			H4
Cones		H3		H4	H1	H2	H3		H3			H5
Cilinders		H3	H4	H4	H1	H2					H1	H5
Unexpected cases	$19\rightarrow H_1$		$24\rightarrow H_2$	$37\rightarrow H_1$ $19\rightarrow H_1$	$19\rightarrow H_1$	$37\rightarrow H_1$	$19\rightarrow H_1$	$19\rightarrow H_1$	$24\rightarrow H_1$			$19\rightarrow H_1$ $37\rightarrow H_1$ $24\rightarrow H_3$ $17\rightarrow H_1$



Test Nº1 Test Nº3

Test Nº 2 Test Nº4.

Fig. 4. Clustering results.

In conclusion, it can be said that the method has worked in a real environment which implies working with noise and error sources. None of the experiments has given unexpected clustering; on the contrary the clustering algorithm has given results according to the human grouping established by us. Nevertheless, we have started several initiatives to improve the method concerning to increase the database and deal with the problem of the validity of the clusters.

References

1. Sengupta K., Boyer K.L.: Creating Random Structural Descriptions of CAD Models and Determining Object Classes. Proc. IEEE Workshop on CAD-based Vision. Pp 38-45. 1994.
2. Sengupta K. and Boyer K.L.: Modelbase Partitioning Using Property Matrix Spectra. Computer Vision and Image Understanding Vol 70, No 2, pp 177-196, 1998.
3. Selinger A. and Nelson R.: A perceptual Gruping Hierarchy for Appearance-Based 3D Object Recognition. Computer Vision and Image Understanding, Vol76, No1 pp 83-92, 1999.
4. Froimovich G., Rivlin E., Shimshoni I.: Object Classification by Functional Parts. Fisrt Symposium on 3D Data, Proccesing, Viasualization and Trnasmision. Padova, pp 648-655. 2002.
5. Yeung D.S., Wang X. Z.: Improving Performance of Similarity-Based Clustering by Feature Weight Learning. IEEE Trans. On Pattern Analysis and Machine Intelligence. Vol 24, No 4, pp 556-561.
6. Gdalyahu Y. and Weinshall D.: Flexible Syntactic Matching of Curves and Its Application to Automatic Hierarchical Classification of Silhouettes. IEEE Trans. On Pattern Analysis and Machine Intelligence. Vol 21 No 12, pp 1312-1328. 1999.
7. Cyr C. Kimia M, B. B. : 3D Object recognition using shape similarity-based aspect graph. Int. Conference on Computer Vision. Vancouver. pp 254-261. 2001
8. Ohbuchi R., Minamitani T., Takei T.: Shape-Similarity search of 3D Models by Using Enhanced Shape Functions. Proc of the Theory and Practice of Computer Graphics (TPCG'03). 2003.
9. Osada R., Funkhouser T., Chazelle D., Dobkin D. Matching 3D Models with Shapes Distributions. Proc. Int. Conf. On Shape Modeling and Applications, pp 154-166. Genova. 2001.
10. Adán M., Adán A., Solids Characterization Using Modeling Wave Structures. Lecture Notes in Computer Science, LNCS 2652, pp 1-10. 2003.
11. Adán M., Adán A., Cerrada C., Merchán P., Salamanca S.: Weighted Cone-Curvature: Applications to 3D Similarity Shapes. The Fourth International Conference on 3-D Digital Imaging and Modeling (3D DIM). Banff. Pp 458-465. 2003.
12. Adán A., Cerrada C., and Feliú V.: Modeling Wave Set: Definition and Application of a new Topological Organization for 3D Object Modeling. Computer Vision and Image Understanding, 79, pp. 281-307. 2000
13. Jolliffe, I. T. , *Principle Components Analysis*, Springer-Verlag, New York, 1986.
14. Duda R. O. and Hart P. E.: Pattern Classification and Scene Análisis. Jonh Wiley & Sons . 1973.
15. Bezdek J. C and Pal S. K.: Fuzzy Models For Pattern Recognition. IEEE Press. 1992.

Algorithms for Constructing Min-Max Partitions of the Parameter Space for MDL Inference

Adriana Vasilache, Ioan Tăbuş, and Jorma Rissanen

Tampere University of Technology, Tampere, Finland

Abstract. In this paper we present several algorithms for the construction of min-max optimal partitions of the parameter space. Two interpretations of the problem lead to two families of practical algorithms that are tested and compared.

1 Introduction

The problem of partitioning the parameter space of a given class of models arises as a necessary step in statistical inference based on the minimum description length (MDL) principle. This is because the model cost is linked to the code-length needed to represent a quantized version of the parameters, either in implicit or explicit manner. The exact construction of the partition is not always needed, and establishing bounds for the average or min-max distortions may suffice, for instance when one wants to find only the structure of the model by the MDL principle. However, when the optimal MDL model including the parameter values themselves are needed, for instance when the models are used for classification or prediction, it is necessary to have an algorithm to give the actual quantization of the parameters.

For a class of models defined by a parametric probability distribution $f = f(y^n;\theta)$, the additional increase in code-length incurred by quantizing the parameter vector θ to the center of a partition cell θ^i can be measured by the Kullback-Leibler (K-L) distance between the un-quantized model $f = f(y^n;\theta)$ and the quantized one $f_j = f(y^n;\theta^j)$. It is therefore of interest to design partitions of the parameter space such that either the average or the min-max loss is minimized. A related problem was discussed in [1], where a preliminary step in the construction of the Kolmogorov structure function was that of obtaining a partition of the parameter space by minimizing the min-max K-L distance between the models $f_i = f(y^n;\theta^i)$ and $f_j = f(y^n;\theta^j)$, defined by the centers of two adjacent partition cells, $\theta^i = \theta(i)$ and $\theta^j = \theta(j)$.

In this paper we propose several algorithmic approaches to the construction of min-max optimal partitions. To ensure the practicality of the algorithms we decide to allocate a model $f = f(y^n;\theta)$ to the center $f_j = f(y^n;\theta^j)$ by the nearest neighbor rule; i.e. the smallest Euclidean norm, $||\theta - \theta^j||^2$. With this rule, the parameter space is partitioned into polygonal cells $V(\theta^i)$; i.e. the Voronoi cells corresponding to the centers θ^j. As in universal coding, the relevant criteria are of a min-max nature, since usually there is no apriori distribution on the parameter space for defining meaningful averages. If we allow $\{V(\theta^i)\}$ to be a covering instead of a partition, the optimal min-max problem for the K-L distance $D(\cdot\|\cdot)$

$$\min_{\{\theta^i\}_{i=1}^N} \max_i \max_{\theta \in V(\theta^i)} D(f(y^n;\theta)||f(y^n;\theta^i)) \tag{1}$$

has a simple solution if one can find the set of centers $\{\theta^i\}_{i=1}^N$ such that

$$\max_{\theta \in V(\theta^i)} D(f(y^n;\theta)||f(y^n;\theta^i)) = d' \tag{2}$$

is constant. In the same way the slightly different min-max problem

$$\min_{\{\theta^i\}_{i=1}^N} \max_i \max_{j:(i,j)\text{are neighbors}} D(f(y^n;\theta^j)||f(y^n;\theta^i)) \tag{3}$$

has a simple solution when

$$D(f(y^n;\theta^j)||f(y^n;\theta^i)) = d' \tag{4}$$

is constant for all neighbors i and j.

By the Taylor expansion the K-L distance between two adjacent models can be approximated as [2]

$$D(f_i||f_j) = \frac{n}{2}(\theta^j - \theta^i)^T J(\tilde{\theta})(\theta^j - \theta^i), \tag{5}$$

where $\tilde{\theta}$ is a point between θ^i and θ^j and $J(\tilde{\theta})$ is the Fisher information matrix [3] calculated at $\tilde{\theta}$.

Assuming further that the information matrix varies smoothly so that it can be calculated at θ^i instead of $\tilde{\theta}$, the goal of optimization expressed by Eq. (4) can be redefined as one of aiming at the partition of the parameter space such that the weighted distance

$$d_F(\theta^i, \theta^j) = (\theta^j - \theta^i)^T J(\theta^i)(\theta^j - \theta^i) \tag{6}$$

between each two adjacent centers θ^i and θ^j is the same.

In the quantization literature weighted metrics like $d(x,y) = (x-y)^T B_y (x-y)$, where B_y is a positive definite matrix depending on y, have been studied in view of creating a partition of the space such that the average weighted metric is minimized [4]. Also in the clustering literature the discriminative clustering using a mutual information criterion was shown to be asymptotically equivalent to vector quantization using Fisher weighted distance [5].

We give first the basic definitions and notations, after which we propose several methods for obtaining the desired partition. The methods will be applied to a multinomial source. Also, theoretical considerations concerning the significance of having the neighbors on the hyper-ellipsoids are presented.

2 Definitions and Notations

We consider the parameter space to be finite. There are then, say N, models in this space, whose neighborhood relation is determined by the Euclidean metric (to simplify the calculations). The distance between two neighboring models is in turn evaluated by the Fisher weighted distance (6). The matrix $J(\theta^i)$ is the parametric Fisher information matrix. If $J(\theta^i)$ is decomposed into its the singular values

$$J(\theta^i) = V(\theta^i)\Lambda(\theta^i)V^T(\theta^i), \tag{7}$$

the Fisher weighted distance is in fact an Euclidean weighted distance in a rotated space, the rotation matrix being $V(\theta^i)$ and the weights being given by the singular values of $J(\theta^i)$.

2.1 Neighborhood Relationship

Gabriel neighbors. Given a set of points, two points are Gabriel neighbors if they share a facet of their Voronoi cells and if the line passing through the two points intersects that facet. In order to check if two points are Gabriel neighbors their midpoint is considered. If the midpoint has as a nearest neighbor none of the two points in discussion, then they are not Gabriel neighbors. If the midpoint has as a nearest neighbor one of the two points then one more check should be performed to establish that the two points are Gabriel neighbors. Two more points, equally distanced from the midpoint and situated on the line bisecting the line passing through the two initial points should be considered (see Fig. 1). If either one of the two initial points is their nearest neighbor then the initial points are Gabriel neighbors. This procedure implies the use of one parameter defining the distance of the secondary check points from the midpoint.

Voronoi neighbors. Two centers are Voronoi neighbors if they share a facet of their partition cells. To check if two centers are Voronoi neighbors one should check first if they are Gabriel neighbors. If they are not and if they have one common Gabriel neighbor, it should be subsequently checked if the points of the line bisecting the line connecting the two centers are ever quantized to one of the two centers.

3 Optimal Partition

From a practical point of view, Eq. (2) is interpreted as the realization of a partition having the vertices of each partition cell on an ellipse defined by Eq. (8) and Eq. (4) is interpreted as the realization of a partition having the neighbors of each center on an ellipse defined by

$$(\theta^j - \theta^i)^T J(\theta^i)(\theta^j - \theta^i) = \frac{d'}{n} = d. \tag{8}$$

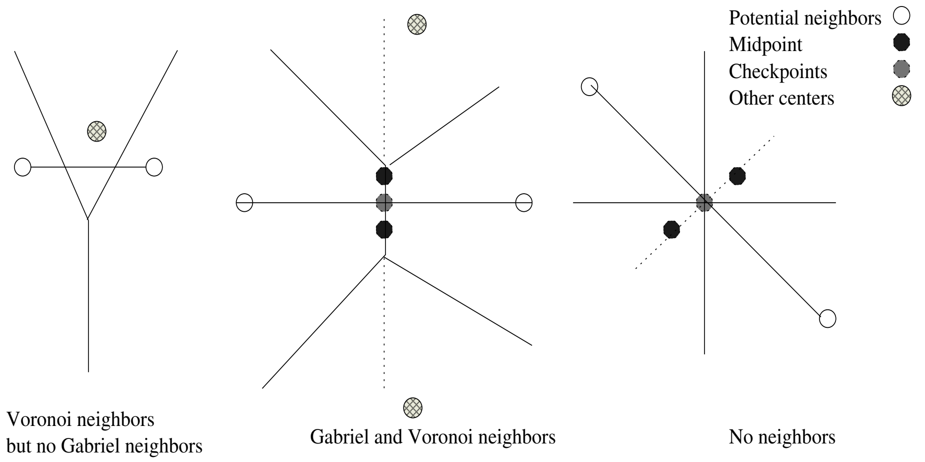

Potential neighbors ○
Midpoint ●
Checkpoints ●
Other centers ▨
Voronoi neighbors but no Gabriel neighbors
Gabriel and Voronoi neighbors
No neighbors

Fig. 1. Neighborhood relationship definition.

Consider the general problem of making a partition in which certain points defining the partition should be positioned on ellipses whose shape and size vary in space. A practical measure to be minimized is the average distance between each of such points and the ellipses on which they should be. We denote this distance to the ellipses by (DE).

3.1 Enforcing the Neighbors of Each Center on an Ellipse (NE)

The practical criterion to be used for the set of points $P = \{\theta^i\}_{i=1}^N$ is mathematically expressed as

$$D_1(P) = \sum_{i=1}^{N} \sum_{j=1}^{\ell_i} \|\theta^i - \mathrm{Pr}_j(\theta^i)\|^2, \tag{9}$$

where $\mathrm{Pr}_j(\theta^i)$ is the projection of the point θ^i on the ellipse centered in θ^j, and ℓ_i is the number of neighbors of the center θ^i. The ellipse centered in θ^j is given by Eq. (8). The measure used to evaluate the performance of the algorithm will be an average of the estimates of d, obtained by fitting an ellipse to the Gabriel neighbors of each point in the set that generates the partition. The orientation of the ellipse at each point is given by the information matrix calculated at that point. This is virtually equivalent to calculating the average Fisher weighted distance for the neighbors of each point.

Uniform partitioning (UP). Before trying more complex algorithms a reasonable starting point is the evaluation of a uniform partitioning of the parameter space. We consider a lattice Z_n of partitions with different scaling factors. The set of points P in the parameter space is then defined as

$$P = \{s \cdot v | s \in \mathbb{R}^+, v \in \mathbb{Z}^n\}, \tag{10}$$

where s is the scaling factor. The optimization of this uniform partition is performed on the scaling factor with respect to DE for a given d.

Centroid assignment (CA). Consider a given center (point in the parameter space) and its Gabriel neighbors. In conformity with our goal this center should lie on all the ellipses centered at its neighbors. In one iteration each codevector is assigned to the centroid of its projections on the ellipses corresponding to its Gabriel neighbors, which then minimizes the DE. Note that in this case we are using the Euclidean distance. All the points are updated in each iteration step.

A major problem consists of the points near the boundary of the domain since they do not have neighbors toward the border, therefore they can only be pushed inside the domain. If it happens that after an iteration a point gets out of the domain, it will be eliminated. This means that some of the initial points get eliminated since no procedure exists with which extra points could be inserted.

Grid neighborhood structure (GNS). The partition centers are let free to move, but the neighborhood relation between them is predefined by the original rectangular grid. Only one point is considered in each iteration step, and its neighbors given by the grid are updated by moving them toward the ellipse of the center. In geometrical interpretation this is similar to the preservation of the structure in the self-organizing maps [6], but here the goal of the optimization is completely different. The algorithm is parameterized by a factor indicating how far each neighbor is moved toward its projection on the ellipse. Compared to the centroid assignment method a supplementary feature is the possibility to add centers in the domain by allowing a larger initial set of centers in the grid. Out of this set of centers only those that are inside the admissible domain are considered for the calculation of the information matrix and the corresponding ellipse, but their neighbors are moved when called for by the algorithm, even if they lie outside the domain.

3.2 Enforcing the Vertices of Each Cell on an Ellipse (VE)

The practical criterion to be optimized in this case is

$$D_2(P) = \frac{1}{\sum_{i=1}^{N} nv_i} \sum_{i=1}^{N} \sum_{j=1}^{nv_i} \|v_i^j - \mathrm{Pr}_i(v_i^j)\|^2, \tag{11}$$

where nv_i is the number of vertices of the partition cell of the center i and $v_i^j, j = \overline{1, nv_i}$ are the vertices of the partition cell for the center θ^i. The measure used to evaluate the performance of the algorithm will be an average of the estimates of d, obtained by fitting an ellipse to the partition cell vertices of each center.

Uniform partition of space. A uniform partition of the parameter space can be used as a reference and/or as initialization for further algorithms. A scaled Z_n lattice can be used, the optimization being this time the average distance between the partition set vertices and their projections on the ellipses on which they should lie.

Centroid assignment. In this case the centroid assignment algorithm is related to the minimization of the average distance of the partition set vertices to the ellipses on which they should lie. Therefore one vertex is replaced at each iteration by the average of its projections on the ellipses on which it should lie. All the vertices are updated at each iteration, and the new centers are updated to be the centroids of the polyhedral volume defined by a set of vertices. The delimitation of a partition set is considered to be realized by the same vertices even if the vertex positions change from an iteration to another.

Minimization of the maximum Fisher (MMF) weighted distance. Another method of putting the partition set vertices on the ellipses is by selecting the center of the partition as the point that minimizes the maximum Fisher weighted distance between the center and the vertices. The consistency of this approach with the declared goal will be proved in the next section. The minimization of the maximum Fisher weighted distance is done numerically. There are two possible approaches for updating the set of centers, first by updating all the centers at one iteration (MMF1), and secondly by updating only one center at one iteration (MMF2). The second approach implies supplementary calculations of the vertices of the partition sets.

4 Optimal Location of Points on an Ellipse

We have presented two approaches with goal to equalize the distance between pairs of adjacent models. In this section we check if the methods might have any other significance as well. More precisely we show that, under certain conditions when some neighboring points in the parameter space are on the ellipse whose center defines a model, the center minimizes the maximum K-L distance to the neighboring points as well.

The following theorems are given for the 2-dimensional case, but the generalization to higher dimensions is straightforward. The proofs are left out due to lack of space.

Theorem 1. *If more than three points $\{x_i\}$ are located on a circle $C(R, c)$, but not on the same semicircle, then the center of the circle c is the point for which $\max_i d(c, x_i)$ is minimized, where $d(c, x_i)$ is the Euclidean distance between x_i and c.*

If we consider the ellipse given by the equation $\frac{\lambda_1}{d} y_1^2 + \frac{\lambda_2}{d} y_2^2 = 1$, then the transformation given by $T = \begin{pmatrix} \sqrt{\frac{\lambda_1}{\lambda_2}} & 0 \\ 0 & \sqrt{\frac{\lambda_2}{\lambda_1}} \end{pmatrix}$ with $\lambda_1 \neq 0, \lambda_2 \neq 0$, transforms the ellipse into a circle. The transform is invertible.

Theorem 2. *If more than three points $\{x_i\}$ are located on an ellipse $E(a,b,c)$, but not on the same semi-ellipse, then the center of the ellipse c is the point for which $\max_i d_{a,b}(c,x_i)$ is minimized, where $d_{a,b}(c,x_i)$ is the weighted Euclidean distance between x_i and c, having the weights a and b on the first and the second dimensions respectively. Here a and b are the semi-axes of the ellipse.*

For the points in the parameter space excluding the border of the domain, the neighbors as well as the vertices of the corresponding partition sets, are around the point in the sense of the Theorem 2. Therefore, Theorem 2 insures the equivalence of the minimization of the maximum Fisher weighted distance from a point c to a set of points $\{x_i\}$ and the placement of the points on the ellipse centered in c.

5 Results

We consider a ternary independent source having parameters $\theta_0, \theta_1, \theta_2 = 1-\theta_0-\theta_1$ where the independent parameters are grouped in the vector $\theta = [\theta_0 \ \ \theta_1]'$. The information matrix for the probability density function $P(x;\theta) = \theta_0^{1_{x,0}} \theta_1^{1_{x,1}} (1-\theta_0-\theta_1)^{1_{x,2}}$ is given by

$$J_n(\theta) = nJ(\theta) = n \begin{bmatrix} \frac{1}{\theta_0} + \frac{1}{1-\theta_0-\theta_1} & \frac{1}{1-\theta_0-\theta_1} \\ \frac{1}{1-\theta_0-\theta_1} & \frac{1}{\theta_1} + \frac{1}{1-\theta_0-\theta_1} \end{bmatrix}, \tag{12}$$

and the matrix of interest is $J(\theta) = \lim \frac{1}{n} J_n(\theta)$.

5.1 Neighbors on Ellipses

First, the uniform partition has been tested with differently scaled versions of the Z_2 lattice. For a given target d, the number of points in the admissible domain results by setting the scale to the value minimizing the average distance from the centers to the ellipses centered in their neighbors. The results are presented in Fig. 3, the average d being generally larger than the target d. On the same plot the results obtained with the grid of the neighborhood structure are also presented. Starting with the optimal uniform partitions the GNS leads to the average d closer to the target values. The GNS algorithm is used without adding any supplementary centers. This feature is useful when initialized with a general partition or, at least with a non-optimal uniform one, which shows that the GNS method finds only a local optimum. Figure 2 exemplifies the resulting partition centers after CA together with the corresponding ellipses for a target d of 0.03. The effect of evading the borders can be observed along the axes. Due to this effect, it is not fair to have the graphics for the CA method on the same figure as for the UP and GNS methods.

5.2 Partition Set Vertices on Ellipses

Since the distance to ellipses in this case does not vary smoothly as a function of scale for a uniform partition the results for the UP method are only suitable

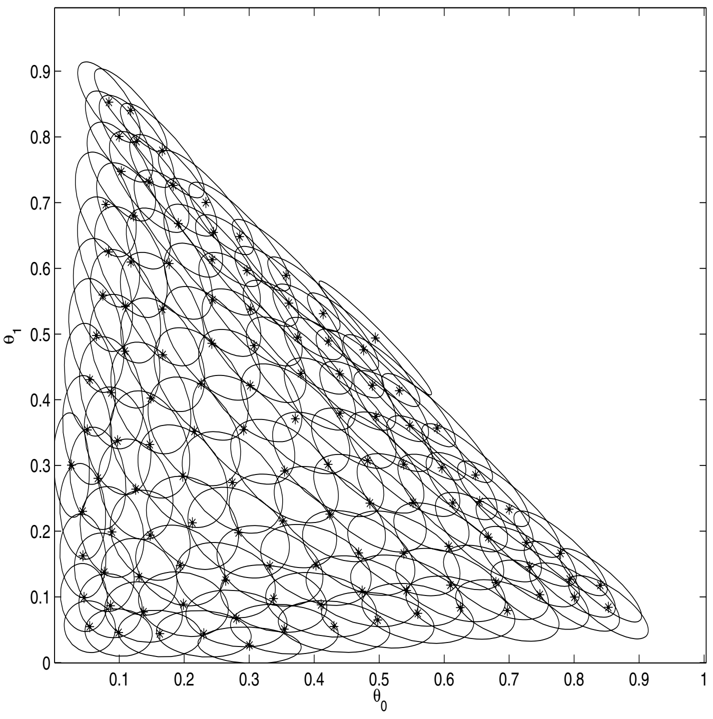


Fig. 2. Centroid assignment method for target $d = 0.03$

to be used as starting points for subsequent algorithms. The centroid approach gives better results than the uniform partition but, like in the NE case, it can only be used for a part of the domain since the centers tend to move away from the borders. The MMF method has the best results (see Tab. 1) and, generally, the update of one center per iteration gives better results. In all the methods the vertices at infinity have been neglected.

Table 1. Comparison of methods for vertices on ellipses. N is the number of points in the parameter space, d_{UP}, $d_{MMF\bullet}$ and $d_{MMF\bullet}$ are the average d for the corresponding methods.

N	d_{UP}	$d_{MMF\bullet}$	$d_{MMF\bullet}$
300	0.00582	**0.00481**	0.00484
528	0.00374	0.00308	**0.00303**
780	0.00293	0.00241	**0.00236**
1225	0.00205	0.00169	**0.00166**

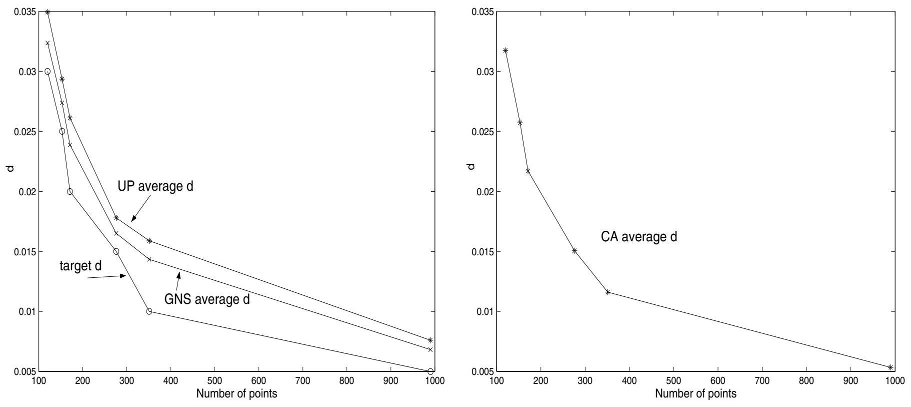


Fig. 3. Method comparison for neighbors on ellipses case.

6 Conclusion

We have presented several methods for obtaining optimal partitions in the min-max sense. We compared them within each of two families, and found GNS to be the best in NE family and MMF in the VE family. The CA method gives promising results, but the resulting partition does not cover the entire space. The proposed methods are exemplified for a 2-dimensional case, but they can also be applied to higher dimensional spaces.

References

1. Rissanen, J., Tăbuş, I.: Kolmogorov's structure function in MDL theory and lossy data compression. In: Advances in Minimum Description Length: Theory and Applications. The MIT Press Cambridge (2004)
2. Rissanen, J.: Lectures on statistical modeling theory. Lecture notes (2003) Tampere University of Technology.
3. Kay, S.M.: Fundamentals of Statistical Signal Processing: Estimation Theory. Prentice Hall International, Inc. (1993)
4. Li, J., Chaddha, N., Gray, R.M.: Asymptotic performance of vector quantizers with a perceptual distortion measure. IEEE Trans. on Information Theory **45** (1999) 1082–1091
5. Sinkkonen, J., Kaski, S., Nikkilä, J.: Discriminative clustering: optimal contigency tables by learning metrics. In Elomaa, T., Mannila, H., Toivonen, H., eds.: Proceedings of the 13th European Conference on Machine Learning. Lecture Notes in Artificial Intelligence (2002) 418–430
6. Kohonen, T.: Self organization and associative memory. 3 edn. Springer, Berlin (1989)

Preclustering of Electrocardiographic Signals Using Left-to-Right Hidden Markov Models

Pau Micó[1], David Cuesta[1], and Daniel Novák[2]

• Department of Systems Informatics and Computers,Polytechnic School of Alcoi, Plaza Ferràndiz i Carbonell 2, 03801 Alcoi, Spain
{pabmitor,dcuesta}@disca.upv.es

• Department of Cybernetics, Czech Technical University in Prague, Czech Republic
xnovakd1@lab.felk.cvut.cz

Abstract. Holter signals are ambulatory long-term electrocardiographic (ECG) registers used to detect heart diseases which are difficult to find in normal ECGs. These signals normally include several channels and its duration is up to 48 hours. The principal problem for the cardiologists consists of the manual inspection of the whole holter ECG to find all those beats whose morphology differ from the normal synus rhythm. The later analisys of these arrhythmia beats yields a diagnostic from the pacient's heart condition. The Hidden Markov Models (HMM) can be used in ECG diagnosis avoiding the manual inspection. In this paper we improve the performance of the HMM clustering method introducing a preclustering stage in order to diminish the number of elements to be finally processed and reducing the global computational cost. An experimental comparative study is carried out, utilizing records form the MIT-BIH Arrhythmia database. Finally some results are presented in order to validate the procedure.

1 Introduction

HMMs [11], [3], [8] is a useful tool for biomedical signal analysis. The proposal would be to use them for clustering the beats contained in a Holter ECG signal in order to facilitate the cardiologist work. The problem lies within the great quantity of elements to be initially processed. Taking into account that the most of the beats will be grouped into the same cluster, it would be a good idea to reduce the number of redundant elements in a preclustering stage [2], [13]. After the preclustering, the more complex clustering algorithm (based on trained HMMs) will be applied.

In this paper we present a new method for HMMs quickly initialization (referred as left-to-right HMM). In addition, and improving the method proposed in [9], we are going to use left-to-right HMMs to drastically reduce in a preclustering stage the number of representative Holter beats.

2 Method

Actually, the real goal of any Holter computer-aided process is to finally separate heart beats into different groups. The fact of classifying objects within these groups is known as clustering process. The accurate object feature extraction is needed in order to group these objects by means of dissimilitude evaluation. If the selected features does not represent the intrinsic quality of each object, the final results derived from clustering process will not become acceptable. We can approximate a biomedical signal using polygonal lines (segments) defined by two different features [6]: the amplitude of the sample and the duration of the segment. Through this beat approximation it is possible to initialize a left-to-right HMM to firstly cluster all the beats. In this case, the dissimilitude among objects is given by the HMM probability measure. Hence, the initialized model is applied to each beat and generates an output probability measure. This probability measure is used in order to calculate the dissimilitude estimator among beats. One beat will be clustered in certain group depending on its dissimilitude related to the cluster. Otherwise, if the dissimilitude is too high, beats should be unclassified. So, a general threshold is set and if the processed beat does not reach it, this beat will be directly labelled as unclassified. Threshold can be defined in two ways: fixing the probability or the dissimilitude level. Once all the beats have been (*pre*)clustered by the first model, the left-to-right HMM creation process will start again taking as the input a randomly selected unclassified beat. The loop will repeat until there are left no more unclassified beats. The whole process followed for HMM initialization and preclustering stages is shown in *Figure 1.* The input signal consists of a preprocessed ECG composed by a certain amount of segmented and normalized beats [2]. The algorithm output are the initialized HMMs and a lower number of beats to be processed in the later clustering stage. All these stages will be developed in the sections below.

2.1 ECG Normalisation and Segmentation

In most of the cases a 24 hour Holter is composed by more than $110,000$ beats [7]. The signal preprocessing plays a very important role for further analysis as classification or clustering. In this stage we solve the three main problems which normally arise in signal processing tasks: *(i) characteristic point detection* as a tool for signal preprocessing, *(ii) baseline removal* and *(iii) signal denoising.* As a result of the preprocessing, we will obtain (from the Holter) a clearly segmented set of beats. All the mentioned method has been developed under wavelet framework [9], [1] and [4].

2.2 HMM Initialization

A complete specification of an HMM [11] requires specification of two model parameters (the number of states, N; and the number of mixtures, M), and specification of the three probability measures (the state transition probability

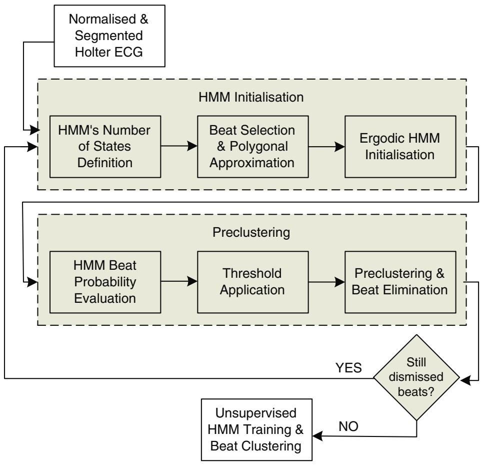

Normalised & Segmented Holter ECG
HMM Initialisation
HMM's Number of States Definition
Beat Selection & Polygonal Approximation
Ergodic HMM Initialisation
Preclustering
HMM Beat Probability Evaluation
Threshold Application
Preclustering & Beat Elimination
Still dismissed beats?
YES
NO
Unsupervised HMM Training & Beat Clustering

Fig. 1. Stages of the process.

distribution, A; the emission probability, b; and the initial state distribution, π). We will refer to all HMM parameters using the set:

$$\lambda = \{A, B, \pi\} \tag{1}$$

This type of statistical model has shown its capacities for certain types of biological signal modelling problem [6], [12] or [10]. In our case, to model an ECG throughout a left-to-right HMM (assuming that $M = 1$), we will have to define the appropriate number of states in order to evaluate the three probability measures (*Equation 1*).

HMM's Number of States. The definition of the number of states of the model has serious implications in the way how the beats are approximated, and will be given by the morphology of the beat [6] that determines the minimum number of polygonal lines needed to approximate the beat. In this case, we just need 12 segments to approximate quite well the normal beat morphology [6] (*Figure 2*). This fact results in a left-to-right $12 - state$ HMM definition (*Section 3*).

Beat Selection and Polygonal Approximation. The HMM's number of states N, is closely related to the number of polygonal lines used for beat segmentation. In fact, the preclustering model briefly described in *Section 2* is based on evaluating these polygonal lines passing through the model states (dissimilarity measure). The result of the polygonal approximation (θ_k) of the k^{th} beat from a sequence of segmented beats, consists of a set of points $\in \Re^2$ composed by two features: the duration (t_{ki}), and the amplitude of the line segment (h_{ki}). Once defined the number of states of the model (N), we describe beat observation as follows:

$$\theta_k = \Big[(t_{k1}, h_{k1}), (t_{k2}, h_{k2}), \ldots, (t_{kN}, h_{kN})\Big] \in \Re^2 \tag{2}$$

In this way, the question is how to keep every beat feature by just selecting the N points $\in \Re^2$ in *Eq. 2* needed for the N-state HMM. To solve the problem we have used a simplified and faster version of the algorithm proposed in [6]. This new algorithm evaluates an error calculating the difference between the ordinate component from the point of the original record to its ordinate projection over the polygonal line used to approximate the interval processed, in order to quickly select the N points.

Left-to-Right HMM Initialization. The left-to-right model assumes that each state is assigned to each polygonal line used for the ECG approximation. This topology reflects the sequential activity of the cardiac conduction system. Using the hidden Markov modelling approach, one observation (one of the lines from the beat polygonal approximation defined by the couple t_{ki}, h_{ki}) is generated by just one state transition, so there is a one-to-one registration between the observation sequence and the undelying state sequence. In order to initialize a left-to-right HMM we will choose randomly any beat (its 12-segment polygonal approximation) from the input ECG, resulting as follows:

- The *state transition probability distribution* A, will take the sub-diagonal form taking into account the left-to-right model definition.
- The *initial state distribution* π, will be centered on the initial state (giving the total likelihood to the state number one).
- The *observation symbol probability distribution* b, will depend on the selected beat's approximation. As this density estimation can be approximated with a Gaussian mixture (in this paper we assume that all the components have d-variate Gaussian distributions) [5], the mean value for each state will be set using the values of the 12 polygonal approximation selected points, that have been obtained from the concrete beat input observation sequence θ_k (*Eq. 2*). The variance for each state is calculated as a percentage of the mean (in our case, the 10%).

On the other hand, the HMM lacks adaptibility if the inicialization is done as described above. In fact, if a concrete beat changes in some way (because baseline time shift, noise, etc.) but still belonging to a concrete morphology group, the HMM defined above will not be able to follow the variation and, consequently will classify the object into the wrong cluster. Using HMMs in this way help us to discard a big amount of similar (regular) beats although without ensuring that all of them with the same morphology are going to be clustered in the same group. This is the reason because preclustering models are only used to improve the unsupervised trained HMM stage and not for the clustering itself. A further development using unsupervised HMM training techniques (runned over the results from the preclustering stage) will provide larger HMMs (in terms of number of states) that will be able to adapt themselves to a concrete morphology without taking care of the different segment durations.

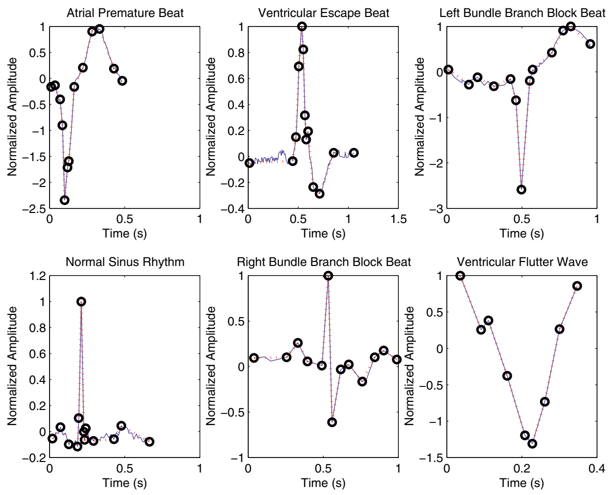


Fig. 2. 12-segment beat polygonal approximation determining a 12-state HMM. Different morphologies selected from initially unclassified beats are used to generate left-to-right HMMs to improve the preclustering stage.

2.3 Preclustering

Due to the great amount of beats contained in a Holter ECG, it is convenient to diminish (without loosing relevant information) the number of beats, in order to further alleviate computational burden.

The mathematical definition of clustering is as follows. Let $\mathcal{X}$ be our data set composed of vectors x_i, $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$. We define as an *m-clustering* of $\mathcal{X}$, the partition of $\mathcal{X}$ into m sets (clusters), $\mathcal{C}_1, \ldots, \mathcal{C}_m$, so that the following three conditions are met:

- $\mathcal{C}_i \neq \emptyset, \quad i = 1, 2, \ldots, m$
- $\bigcup_{i=1}^{m} \mathcal{C}_i = \mathcal{X}$
- $\mathcal{C}_i \bigcap \mathcal{C}_j = \emptyset, \quad i \neq j, j = 1, \ldots, m$

As we explain below, the developed preclustering algorithm is based on this definition and on the recurrent loop to identify and initializate the m left-to-right HMMs for preclustering operations.

Beat Probability Evaluation. If we project the input beat θ_k to the model hmm_j we obtain the probability that the beat is generated by the model, $Pr(\theta_k \mid hmm_j)$. Thus, when calculating this parameter for every beat from the input beat sequence of length T and using every model obtained while repeating the loop from the preclustering model initialization stage (*Figure 1*), we will get the $P_{M \times T}$ beat probability matrix.The MIT-BIH Arrhythmia database [7] was used for results evaluation.

Threshold Application. Depending on the beat probability matrix $P_{M \times T}$, it is possible to fix a threshold level in order to label (as non-classsified, we will use the symbol @) all those objects whose dissimilitude measure do not exceed the threshold. Thus, with an accurate threshold definition, we can easily simplify the preclustering stage from all those beats that have not been clearly evaluated by the models.

Preclustering and Beat Elimination. There are two parameters to apply at this stage: the threshold value and the dissimilitude distance. In the latter case we have used a very simple dissimilitude evaluation algorithm, since this distance is calculated for each beat as the maximum value from the corresponding column of the probability matrix, P. Thus, if the beat's dissimilitude exceeds the threshold, this beat will be clustered within the model that provides the maximum value (*Eq. 3*), otherwise it will be labelled as unclassified (@).

$$c_j \mid \theta_k = argmax_{1 \le j \le M} \left(P_{j,k}\right), \qquad \forall \theta_k \in \Theta, \forall c_i \in C \tag{3}$$

When all the beats have been (*pre*)clustered by the first model, the left-to-right HMM creation process will start again taking as the input a randomly selected unclassified beat. The loop will repeat until there are left no more unclassified beats. This process will generate so many left-to-right HMMs as necessary for the classification of the complete set of beats from the input ECG. Once every beat has been clustered, we have proceeded to the analisys of the cluster purity. The cluster purity is the percent of beats with the same morphology that have been grouped in that cluster. A high purity means that nearly every beat in the cluster has a similar morphology. So, and depending on the analysed cluster's purity we will be able to clean the cluster, deleting all those beats with a similar dissimilitude and keeping all those whose morphology maybe does not match exactly with the eliminated ones.

3 Results

Several input sets of segmented beats, from MIT-BIH database, have been prepared for the tests. These sets have been used both for HMMs evaluation and beat preclustering stage. A total of 18.700 beats have been processed in a series of ten different experiments, in each one of them threshold value has been changed. The results obtained from two of these test series are presented on *Tables 1-2*. Taking into account that every beat used in our tests has been previously labelled from MIT-BIH database, information and percents displayed within table cells follows the next format:

– The 1^{st} value from $cell_{i,j}$ gives us the number of beats labelled as i-morphology which have been classified to cluster j.
– The 2^{nd} value (among parenthesis,()) gives us the percentage of beats labelled as i-morphology in cluster j from the total of beats that presents this concrete morphology.

– The 3^{rd} value (bracketed,[]) gives us the percentage of beats labelled as *i*-morphology in cluster j from the total of beats that have been classified in that cluster. We have referred to this parameter as cluster purity.

Beat labels used in tests have the following meaning:

– θ_A: atrial premature beat.
– θ_E: ventricular escape beat.
– θ_L: left bundle branch block beat.
– θ_N: normal sinus rhythm beat.
– θ_R: right bundle branch block beat.
– θ_T: ventricular flutter wave.

Table 1. Thresholded test to classify six different beat morphologies (θ_i) within seven possible clusters (C_j), where cluster $C_\bullet$ is used for threshold-dismissed beats.

	θ_A	θ_E	θ_L	θ_N	θ_R	θ_T
$C_\bullet$	82(78.8)[97.6]	0(0)[0]	0(0)[0]	2(0.5)[2.4]	0(0)[0]	0(0)[0]
$C_\bullet$	0(0)[0]	97(97)[52.7]	0(0)[0]	87(23.7)[47.2]	0(0)[0]	0(0)[0]
$C_\bullet$	0(0)[0]	0(0)[0]	29(35.4)[16]	74(20.2)[40.8]	78(79.6)[43]	0(0)[0]
$C_\bullet$	0(0)[0]	1(1)[0.5]	0(0)[0]	201(54.9)[99.5]	0(0)[0]	0(0)[0]
$C_\bullet$	0(0)[0]	0(0)[0]	0(0)[0]	2(0.7)[12.5]	14(14.2)[87.5]	0(0)[0]
$C_\bullet$	0(0)[0]	0(0)[0]	0(0)[0]	0(0)[0]	0(0)[0]	27(10.5)[100]
$C_\bullet$	22(21.2)[7.2]	2(2)[0.6]	53(64.6)[16.8]	0(0)[0]	6(6.2)[1.9]	231(89.5)[73.5]

As it is shown in *Table 1* there are some unclassified beat in almost every morphology. Nevertheless, the purity percentage for clusters C_1, C_4, C_5 and C_6 is big enough to consider them as a good approximation in order to delete redundant beats and start the unsupervised HMM training stage. However it can be clearly observed how, ventricular escape beats (labelled as $beats_E$) are the best grouped in the same cluster (97%) in spite of its dirtiness in terms of purity (52.7%) (*Figure 3*).

Table 2. Non-thresholded test to classify the ECG signal composed by two different beat morphologies (θ_i) within three possible clusters (C_j).

	θ_R	θ_N
$C_\bullet$	74(4)[77]	22(2.4)[23]
$C_\bullet$	70(3.8)[8]	[810][88.3][92]
$C_\bullet$	1671(92)[95.2]	85(9.2)[4.8]

In *Table 2*, there are only needed three from the eight HMM initially prepared for clustering the whole set of beats. In addition, the high purity percentage presented in clusters C_2 and C_3 reveals us that they fit almost perfectly with the preclustering models obtained from θ_N and θ_R. Cluster C_1 has been used to group all those beats whose morphology have not been clearly identified (only the 4% of the total amount from θ_R and the 2.4% from θ_N) (*Figure 4*).

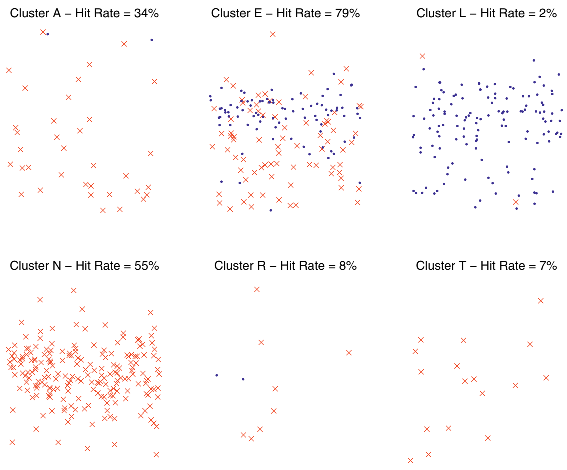


Fig. 3. Clusters obtained from results in *Table 1*. Rigth clustered beats are presented with vanes and badly ones with spots. Hit rate gives us the percent of beats classified within the right cluster.

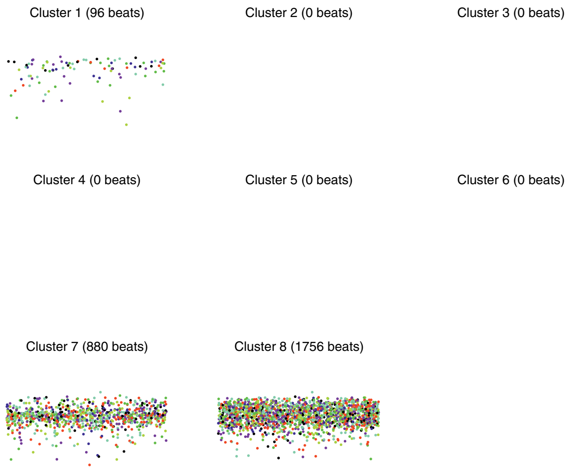


Fig. 4. Clusters obtained from results in *Table 2*. Consequently, it is possible to eliminate at least the 96.4% of the beats, what is a significant reduction of the input beat sequence size.

4 Discussion and Conclusion

We can conclude that it is very important to note that if more strict (in terms of probability) threshold is applied, the clusters are less contaminated. This is our final goal because of two reasons: first of all, to eliminate as much similar beats as possible in order to reduce the size of the input set of beats and alleviating, in this way, the computational burden; and secondly, to facilitate the HMM unsupervised training using this preclustering results. Although clustering with a very demanding threshold value is feasible, this fact presents the non-desirable effect of the non-classified objects increase, minimizing the preclustering stage performance. In spite of good results, it is very important to note the relevance of a correct model initialization, since the beat dissimilitude measure will depend on how the model fits with that beat. A badly initialized model will provide a low beat probability, and this fact will result in the labeling as non-classified or, what is worse, in the wrong clustering. Future work will be focused on improving each stage of the process by using dynamic programming techniques as Dynamic Time Warping (DTW).

References

1. Burrus, S.: Introduction to Wavelets and Wavelet Transforms. Prentice Hall, (1997)
2. Cuesta, D.: Estudio de métodos para procesamiento y agrupación de senales electrocardiográficas. Ph.D.Thesis, Valencia (2001)
3. Cuesta, D., Micó, P., and Novák, D.: Clustering electrocardiograph signals using Hidden Markov Models. European Medical and Biological Engineering Conference, Vienna (2002)
4. Daubechies, I.: Ten lectures on wavelets. (1992)
5. Duda, R., Hart, P. and Stork, D.: Pattern Classification. John Wiley & Sons, (2001)
6. Koski, A.: Modelling ECG signals with hidden Markov models. Artificial Intelligence in Medicine, Vol. 8, (1996)
7. Mark,R., and Moody G.: MIT-BIH arrhythmia data base directory. Massachusetts Institute of Technology - Beth Israel Deaconess Medical Center, (1998)
8. Novák, D., Cuesta, D., Micó, P., and Lhotská, L.: Number of arrhythmia beats in Holter ECG: how many clusters? 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Cancún (2003)
9. Novák, D.: Electrocardiogram Signal Processing using Hidden Markov Models. Ph.D.Thesis, Prague (2004)
10. Obermaier, B.: Hidden Markov Models for online classification of single trial EEG data. Pattern Recognition Letters, Vol. 22,(2001)
11. Rabiner, R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE, vol. 77, (1989)
12. Rezek, I. and Roberts, S.J.: Learning Ensemble Hidden Markov Models for Biosignal Analysis. 14th International Conference on Digital Signal Processing,(2002)
13. Vidal, E., and Marzal, A.: A Review and New Approaches for Automatic Segmentation of Speech Signals. Signal Processing V: Theories and Applications. Elsevier Science Publishers B. V, (1990)

Graph-Based Clustering of Random Point Set

Atsushi Imiya[1] and Ken Tatara[2]

[1] Institute of Media and Information Technology, Chiba University,
Yayoi-cho 1-33, Inage-ku, 263-8522, Chiba Japan
[2] School of Science and Technology, Chiba University,
Yayoi-cho 1-33, Inage-ku, 263-8522, Chiba Japan

Abstract. In this paper, we define clusters and the boundary curves of clusters in a random point set using the Delaunay triangulation and the principal curve analysis. The principal curve analysis is a generalization of principal axis analysis, which is a standard method for data analysis in pattern recognition.

1 Introduction

In this paper, we develop a graph-based algorithm for clustering of point sets and learning of the boundary of random point sets. The boundary of a random point set is extracted by the principal curve analysis. The principal curve analysis is a generalization of principal axis analysis, which is a standard method for data analysis in pattern recognition.

For the vector space method of data mining, each datum is expressed as a point in the higher dimensional Euclidean space. Symbolic expressions of these point sets are required for the visual interface for the data mining systems. Furthermore, these data are sometimes transformed as a point distribution in lower dimensional vector spaces, usually tow or three dimensional spaces, for the visualisation of data distribution on CRT. Therefore, the extraction of the symbolic features of random point sets in two and three dimensional is a basic process for the visual interpretation of random point sets for the visualisation of the data space.

Computational geometry provides combinatorial methods for the recovery of boundary curves as polygonal curves. These algorithms are based on Voronoi tessellation, Delaunay triangulation, Gabriel graphs, crust, α-shape, and β-skeleton [1–3]. The reconstructed curves by these methods are piecewise linear. Furthermore, the solutions are sensitive against noise and outlayers, since these methods construct polygons and polyhedrons using all sample points. Furthermore, the algorithm extracts a boundary curves.

In this paper, we introduce method for the estimation of the boundary of a random point set which permits the extraction of boundary curves of clusters in a random point set.

2 Mathematical Preliminary

Delaunay Triangulation. Setting $\{\boldsymbol{p}_i\}_{i=1}^n$ to be a point set in $\mathbf{R}^n$, the region

$$\mathbf{V}_i = \{\boldsymbol{x} | |\boldsymbol{x} - \boldsymbol{p_i}| \leq |\boldsymbol{x} - \boldsymbol{p}_j|, i \neq j\} \tag{1}$$

is called Voronoi region with respect to the generator $\boldsymbol{p}_i$. The hyperplane

$$F_i = \{\boldsymbol{x} | |\boldsymbol{x} - \boldsymbol{p_i}| = |\boldsymbol{x} - \boldsymbol{p}_j|\} \tag{2}$$

is the Voronoi face. Setting $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ to be the generators of Voronoi regions sharing a face, a geometric graph which connect all pairs of generators in the face-sharing region is called Delaunay triangulation. The Voronoi tessellation and the Delaunay triangulation are dual figures each other.

Mathematical Morphology. Setting $\mathbf{A}$ to be a finite closed set in the n-dimensional Euclidean space $\mathbf{R}^n$, the Minkowski addition and subtraction of sets are defined as

$$\mathbf{A} \oplus \mathbf{B} = \bigcup_{\boldsymbol{x} \in \mathbf{B}, \boldsymbol{y} \in \mathbf{B}} (\boldsymbol{x} + \boldsymbol{y}), \ \mathbf{A} \ominus \mathbf{B} = \overline{\mathbf{A} \oplus \overline{\mathbf{B}}}. \tag{3}$$

The inner and outer boundary of point set $\mathbf{A}$ with respect to radius λ are defined as

$$\Delta_\lambda^+ \mathbf{A} = (\mathbf{A} \oplus \lambda \boldsymbol{B}) \setminus \mathbf{A}, \ \Delta_\lambda^- \mathbf{A} = \mathbf{A} \setminus (\mathbf{A} \ominus \lambda \boldsymbol{B}) \tag{4}$$

for the unit n-sphere such that $\boldsymbol{B} = \{\boldsymbol{x} | \boldsymbol{x} \leq 1\}$, where $\lambda \mathbf{B} = \{\lambda \boldsymbol{x} | \boldsymbol{x} \in \mathbf{B}\}$ for $\lambda > 0$. We call $\mathbf{A}_\lambda = \Delta_\lambda^+ \mathbf{A} \bigcup \Delta_\lambda^- \mathbf{A}$ the boundary belt of $\mathbf{A}$ with respect to λ. Geometrically, we have the relation

$$\lim_{\lambda \to +0} \mathbf{A}_\lambda = \partial \mathbf{A}, \tag{5}$$

where $\partial \mathbf{A}$ is the boundary curve of set $\mathbf{A}$.

Principal Curve Analysis. Let $\mathbf{X}$ be a mean-zero point distribution in $\mathbf{R}^n$. The major principal component $\boldsymbol{w}$ maximizes the criterion

$$J(\boldsymbol{w}) = E_{\boldsymbol{x} \in \mathbf{X}} |\boldsymbol{x}^\top \boldsymbol{w}|^2 \tag{6}$$

with respect to $|\boldsymbol{w}| = 1$, where $E_{\boldsymbol{x} \in \mathbf{X}}$ expresses the expectation over set $\mathbf{X}$. Line $\boldsymbol{x} = t\boldsymbol{w}$ is a one-dimensional linear subspace which approximates $\mathbf{X}$. A maximization criterion

$$J(\boldsymbol{P}) = E_{\boldsymbol{x} \in \mathbf{X}} |\boldsymbol{P}\boldsymbol{x}|^2 \tag{7}$$

with respect to $rank\boldsymbol{P} = 1$, determines a one dimensional linear subspace which approximates $\mathbf{X}$. If $\mathbf{X}$ is not a mean-zero point distribution in $\mathbf{R}^2$ and the centroid of $\mathbf{X}$ is not predetermined, the maximization criterion

$$J(\boldsymbol{P}, \boldsymbol{g}) = E_{\boldsymbol{x} \in \mathbf{X}} |\boldsymbol{P}(\boldsymbol{x} - \boldsymbol{g})|^2 \tag{8}$$

with respect to $rank\boldsymbol{P} = 1$, determines a one-dimensional linear manifold which approximates point distribution $\mathbf{X}$.

For the partition of $\mathbf{X}$ into $\{\mathbf{X}_i\}_{i=1}^N$ such that $\mathbf{X} = \cup_{i=1}^N \mathbf{X}_i$, vectors $\boldsymbol{g}_i$ and $\boldsymbol{w}_i$ which maximize the criterion

$$J(\boldsymbol{w}_1, \cdots, \boldsymbol{w}_N, \boldsymbol{g}_1, \cdots, \boldsymbol{g}_N) = \sum_{i=1}^{N} E_{\boldsymbol{x} \in \mathbf{X}_i} |(\boldsymbol{x} - \boldsymbol{g}_i)^\top \boldsymbol{w}_i|^2 \tag{9}$$

determine a polygonal curve [4],

$$\boldsymbol{l} = \boldsymbol{g}_i + t\boldsymbol{w}_i. \tag{10}$$

Furthermore, for an appropriate partition of $\mathbf{X}$ into $\{\mathbf{X}\}_{i=1}^N$, such that $\mathbf{X} = \cup_{i=1}^N \mathbf{X}_i$, vector $\boldsymbol{g}_i$ and orthogonal projector $\boldsymbol{P}_i$, which maximize the criterion

$$J(\boldsymbol{P}_1, \cdots, \boldsymbol{P}_N, \boldsymbol{g}_1, \cdots, \boldsymbol{g}_N) = \sum_{i=1}^{N} E_{\boldsymbol{x} \in \mathbf{X}_i} |\boldsymbol{P}_i(\boldsymbol{x} - \boldsymbol{g}_i)|^2 \tag{11}$$

with respect to $rank\boldsymbol{P}_i = 1$, determine a piecewise linear curve,

$$\mathbf{C}_i = \{\boldsymbol{x} + \boldsymbol{g}_i | \boldsymbol{P}_i \boldsymbol{x} = \boldsymbol{x}\}. \tag{12}$$

This piecewise linear is called the principal curve [4]. Figure 1 shows the principal components and principal curves on a plane.

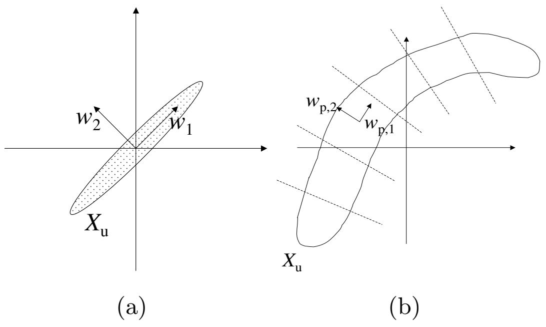


Fig. 1. Principal components (a) and Principal curve (b).

3 Graph-Based Clustering

Using Delaunay triangulation of a random point set, we develop an algorithm for the separation of clusters of point set. We assume that in each subset, the distances between two points connected by a Delaunay edge is shorter than a constant. We call this constant the resolution of point sets. We adopt the median of Delaunay edges as the estimation of the resolution of the point sets.

Definition 1 *For a point $\boldsymbol{p}$ in a random point set $\mathbf{V}$, we call*

$$\boldsymbol{p}_\delta = \{\boldsymbol{x} | \, |\boldsymbol{p} - \boldsymbol{x}| < \delta, \boldsymbol{p} \in \mathbf{V}, \forall \boldsymbol{x} \in \mathbf{R}^2\}$$

the effective region of point $\boldsymbol{p}$ with respect to radius δ.

As the union of the effective region of each point, we define the effective region of a random point set.

Definition 2 *For a random point set $\mathbf{V}$, we call*

$$\overline{\mathbf{V}} = \bigcup_{\boldsymbol{p} \in \mathbf{V}} \boldsymbol{p}_\delta$$

the effective region of point set $\mathbf{V}$ with respect to radius δ.

If points in $\mathbf{V}$ are sampled from a connected region in $\mathbf{R}^n$, $\overline{\boldsymbol{V}}$ becomes a connected region in $\mathbf{R}^n$, selecting an appropriate δ. Therefore, we introduce a method for the selection of a suitable radius for the estimation of the connected region from a random point set. Using this estimated connected region, we develop an algorithm for the construction of the boundary of a random point set.

Setting E to be the set of edges of the Delaunay triangulation D constructed from the points in random point set $\mathbf{V}$, we set

$$\delta = \text{median}_{e \in E} |e|, \tag{13}$$

if points distribute uniformly in a region. Then, we define the boundary set as

$$\mathbf{V}_\gamma = \overline{\mathbf{V}}_\gamma \bigcap \mathbf{V}, \;\; \overline{\mathbf{V}}_\gamma = \overline{\mathbf{V}} \setminus \{\overline{\mathbf{V}} \ominus \gamma \boldsymbol{D}(\delta)\}, \tag{14}$$

where $\gamma > 1$ is a constant and $\boldsymbol{D} = \{\boldsymbol{x} | |\boldsymbol{x}| \leq \delta\}$ is the set of all points in the circle with radius δ. We call $\mathbf{V}_\gamma$ the γ-boundary of random point set $\mathbf{V}$.

Next, we construct the new Delaunay triangulation D' for points in

$$\mathbf{V}' = \mathbf{V} \oplus \delta\{\{\boldsymbol{d}_i\}_{i=1}^n\}, \tag{15}$$

where $\boldsymbol{d}_i$ is an appropriate vector such that $|\boldsymbol{d}_i| = 1$. We call the point set $\mathbf{V}'$ the effective set of $\mathbf{V}$.

After Delaunay triangulation D' of the new point set, there exist three types of edges as shown in Figure 2 (e), edges connect points in $\mathbf{V}$, edges connect points in $\mathbf{V}'$, and edges connect points in $\mathbf{V}$ and $\mathbf{V}'$. As shown in (a), in the neighborhood of points in each cluster of $\mathbf{V}$, there exist points in $\mathbf{V}'$, since points in $\mathbf{V}' \setminus \mathbf{V}$ lie in the region $D(\delta) \oplus \boldsymbol{p}_i$ for a point $\boldsymbol{p}_i \in \boldsymbol{V}$. This geometrical property leads to the conclusion that in Delaunay triangulation D', edges which connect points in $\mathbf{V}$ is the bridges which connect clusters. Therefore, we have the following clustering algorithm using Delaunay triangulation.

1. Construct Delaunay triangulation D from random point set $\mathbf{V}$.
2. Assign label $\mathbf{1}$ to points in the random point set $\mathbf{V}$.

3. For the collection of all edges E of D, detect the median length, and set it as δ.
4. Construct $\mathbf{V}' = \mathbf{V} \oplus \delta\{\{\boldsymbol{d}_i\}_{i=1}^n\}$, for appropriate unit vectors $\{\boldsymbol{d}_i\}_{i=1}^n$
5. Assign label **2** to point in $\mathbf{V}'$.
6. Construct Delaunay D' from $\mathbf{V}'$.
7. Eliminate edges connect whose both vertices are labelled as **1**.

Figure 2 shows a process for the clustering. Delaunay triangulation (b) of the original point set (a) determines the effective region (c). After Delaunay triangulation (e) of point in the effective set (d), the algorithm extract bridges (f). By eliminating bridges (g) and (h), the algorithm yields clusters.

4 Detection of Cluster Boundary

After extracting clusters from random point set, we extract the boundary of each cluster using the principal curve analysis. As the boundary of each cluster, we extract the principal curve from the γ-boundary.

Definition 3 *The principal boundary of a random point set is the principal manifold of the point in the γ-boundary of a random point set.*

We also call this principal manifold extracted from random point set $\mathbf{V}$ the γ-curve of $\mathbf{V}$.

4.1 Principal Curves Detection

Set $\mathbf{D}$ and $\mathbf{S}$ to be a random point set and the vertices of polygonal curve, respectively, and the distance between point $\boldsymbol{x} \in \mathbf{S}$ and $\boldsymbol{y} \in \mathbf{D}$ is defined as $d(\boldsymbol{x}, \mathbf{D}) = \min_{\boldsymbol{y} \in \mathbf{D}} d(\boldsymbol{x}, \boldsymbol{y})$ for the Euclidean distance in a plane.

The initial shapes $\mathbf{S}$ and $\mathbf{C}$ are a line segment whose direction is equivalent to the major component $\boldsymbol{w}_1$ of a random point set and a regular triangle whose vertices are determined from the principal components $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$. For a sequence of vertices $\langle \boldsymbol{v}_1, \boldsymbol{v}_2, \cdots \boldsymbol{v}_n \rangle$ of a polygonal curve, we define the tessellation as

$$\begin{aligned} V_\alpha &= \{\boldsymbol{x} | d(\boldsymbol{x}, \boldsymbol{v}_\alpha) < d(\boldsymbol{x}, \boldsymbol{v}_i), d(\boldsymbol{x}, \boldsymbol{v}_\alpha) < d(\boldsymbol{x}, \boldsymbol{e}_{ij}), \alpha \neq i\}, \\ E_{\alpha\alpha+1} &= \{\boldsymbol{x} | d(\boldsymbol{x}, \boldsymbol{e}_{\alpha\alpha+1}) < d(\boldsymbol{x}, \boldsymbol{v}_i), d(\boldsymbol{x}, \boldsymbol{e}_{\alpha\alpha+1}) < d(\boldsymbol{x}, \boldsymbol{e}_{ii+1}), \alpha \neq i\}, \end{aligned}$$

where $\boldsymbol{e}_{ii+1}$ is the edge which connects $\boldsymbol{v}_i$ and $\boldsymbol{v}_{i+1}$. The minimization criterion of reference [5] is expressed as

$$I = \sum_{\boldsymbol{v}_k \in \mathbf{C}} F(\boldsymbol{v}_k, \mathbf{D}) + \lambda \sum_{\boldsymbol{v}_k \in \mathbf{P}} \sum_{i=-1}^{1} \frac{\boldsymbol{v}_{i-1i}^\top \boldsymbol{v}_{ii+1}}{|\boldsymbol{v}_{i-1i}||\boldsymbol{v}_{ii+1}|} \tag{16}$$

for

$$F(\boldsymbol{v}_k, \mathbf{D}) = \sum_{\boldsymbol{x} \in E_{k-1k}} d(\boldsymbol{x}, \boldsymbol{v}_k) + \sum_{\boldsymbol{x} \in V_k} d(\boldsymbol{x}, \boldsymbol{v}_k) + \sum_{\boldsymbol{x} \in E_{kk+1}} d(\boldsymbol{x}, \boldsymbol{v}_k).$$

Using this criterion, we obtain an algorithm for the detection of the principal curve [5] where I_K is the value of I with K vertices.

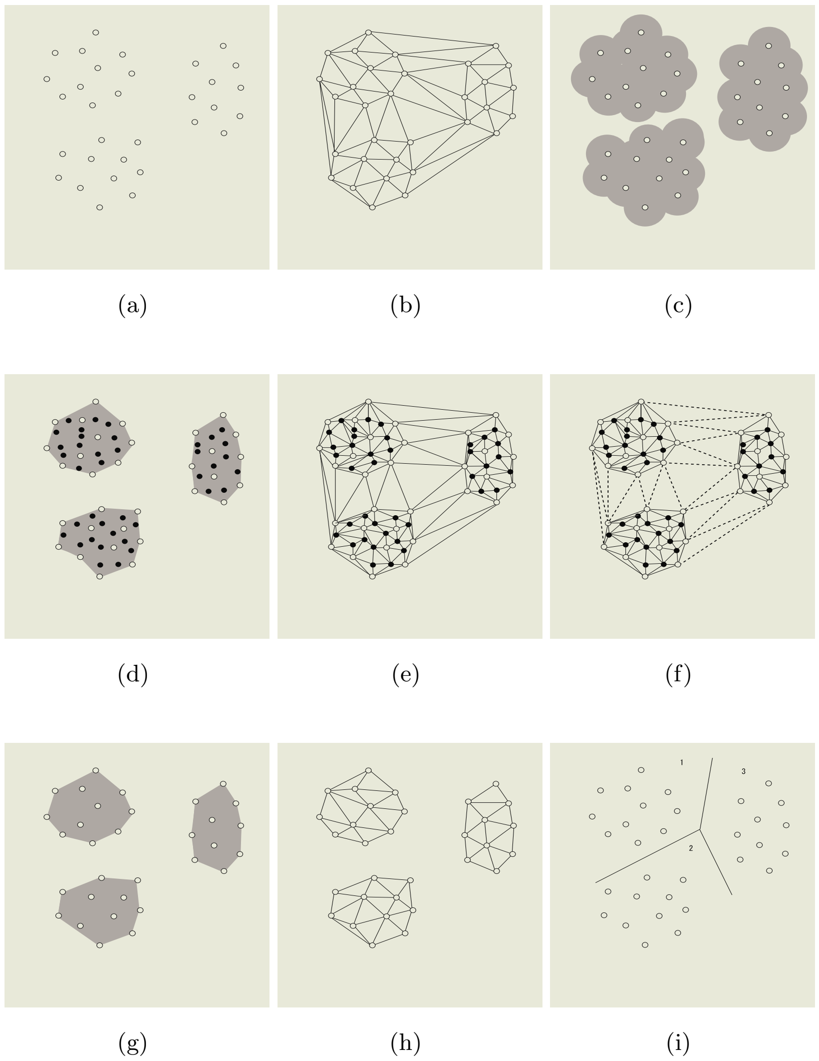


Fig. 2. Clustering algorithm: Delaunay triangulation (b) of the original point set (a) determines the effective region (c). After Delaunay triangulation (e) of point in the effective set (d), the algorithm extract bridges (f). By eliminating bridges (g) and (h), the algorithm yields clusters.

1. Set the vertices of the initial curve as $\mathbf{S}$.
2. Move all vertices $\boldsymbol{v}_i$ $i = 1, 2, \cdots, K$, to minimize I_K.
3. Generate the new vertex $\boldsymbol{v}_{K+1}$ on the curve $\mathbf{S}$.
4. If $|I_K - I_{K-1}| \leq \varepsilon$ for a positive constant ε, then stop, else set $\mathbf{S} := \mathbf{S} \cup \{\boldsymbol{v}_{K+1}\}$ and go to 2.

This incremental algorithm preserves the topology of the initial curve, since the algorithm generates new vertices on the curve.

4.2 Boundary Curve Detection

It is possible to detect boundary of random point set if we can extract the boundary belt of random point set. In this section, we develop an algorithm for the extraction of the boundary belt.

Using definitions in the previous sections, we have the following algorithm for the construction of the principal boundary of a random point set.

1. Construct Delaunay triangulation D from random point set $\mathbf{V}$.
2. For the collection of all edges E of D, detect the median length, and set it as δ.
3. Compute the effective region of random point set $\mathbf{V}$.
4. Compute γ-boundary of random point set $\mathbf{V}$.
5. Compute γ-curve of random point set $\mathbf{V}$.

The construction of the Delaunay triangulation using all points in $\mathbf{V}$ is in practice an time-consuming process for a large number of points even if we use an optimal algorithm. Furthermore, we only need the lengths of the Delaunay triangles for the construction of the effective region of the neighborhood of a random point set. Therefore, we replace steps 1 and 2 of the algorithm to the following random sampling process.

1. Select a finite closed subset $\mathbf{S}$ of $\mathbf{R}^2$.
2. Compute Delaunay triangulation for points in $\mathbf{S}\bigcap\mathbf{V}$.
3. Compute the median of length of edges of Delaunay triangles with respect to subset $\mathbf{S}$.
4. Repeat steps 1 to 3 until the predetermined number of times.
5. Select the maximum length.

In Figure 3, we show, the sequence of boundary curves of a sequence of random point sets which changes number of clusters. The clusters are generated using geometric property of the Delaunay edges.

Figure 4, we also show an example of clustering. 4299 points in (a) are separated to three clusters which are encircled by curves in (b).

In these examples, the density of point distribution are 0.3/unit length^2. Therefore, the average distance between points are 3 to 4 units. Furthermore, the median of the edges of the first Delaunay triangulation are 4 to 5 units. These results allow that the median of the lengths the edges of the first Delaunay triangulation is a reasonable estimation of the resolution of the random point sets.

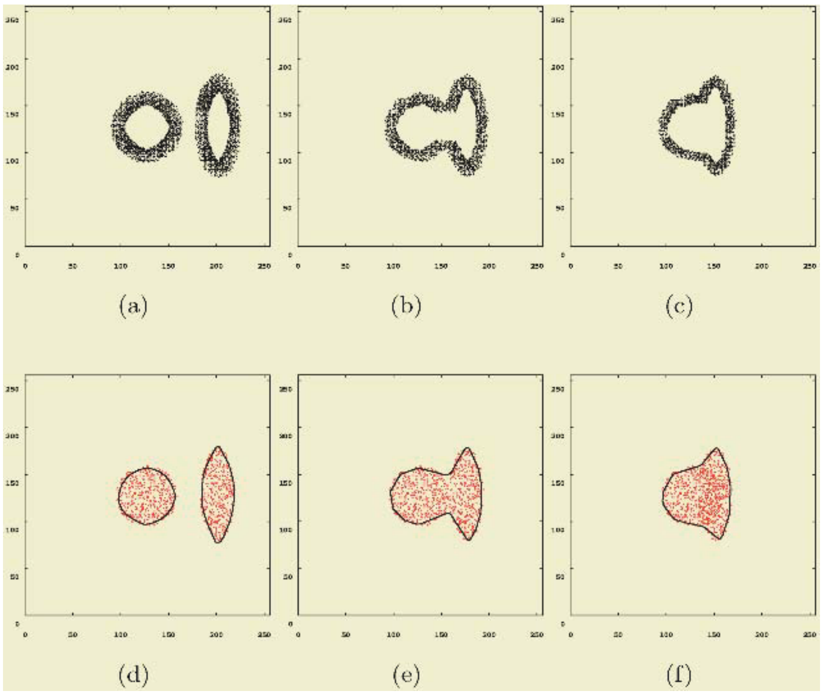


Fig. 3. The sequence of boundary belts (a), (b), and (c) and curves (d), (e), (f) of a sequence of random point sets which changes number of clusters. The clusters are generated using geometric property of the Delaunay edges.

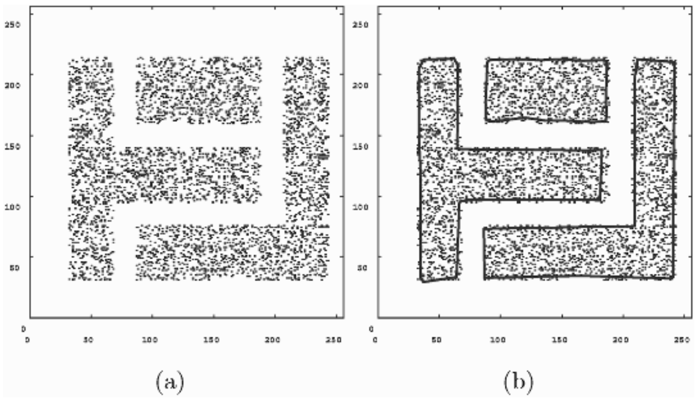


Fig. 4. An example of clustering. 4299 points in (a) are separated to three clusters which are encircled by curves in (b).

5 Conclusions

In this paper, we have defined clusters and the boundary curves of clusters in a random point set using the Delaunay triangulation and the principal curve analysis. Once the polygonal boundary of a random point set is estimated, it is possible to compute the linear skeleton of the polygonal boundary [10, 11]. We adopt the linear skeleton of the polygonal boundary as the skeleton of a random point set.

References

1. Amenta, N., Bern, M., and Eppstein, D., The crust and the β-skeleton:Combinatorial curve reconstruction, Graphical Models and Image Processing, **60**, 125-135, 1998.
2. Attali, D. and Montanvert, A., Computing and simplifying 2D and 3D continuous skeletons, CVIU, **67**, 261-273, 1997.
3. Edelsbrunner, H., Shape reconstruction with Delaunay complex, Lecture Notes in Computer Science, **1380**, 119-132, 1998.
4. Hasite, T., Stuetzle, T., Principal curves, J. Am. Statistical Assoc., **84**, 502-516, 1989.
5. Kégl, B., Krzyzak, A., Linder, T., Zeger, K., Learning and design of principal curves, IEEE PAMI, **22**, 281-297, 2000.
6. Oja, E., Principal components, minor components, and linear neural networks, Neural Networks, **5**, 927-935, 1992.
7. Imiya, A., Ootani, H., Kawamoto, K., Linear manifolds analysis: Theory and algorithm, Neurocomputing, **57**, 171-187, 2004.
8. Imiya, A., Kawamoto, K., Learning dimensionality and orientations of 3D objects, Pattern Recognition Letters, **22**, 75-83, 2001.
9. Silverman, B. W., Some aspects of the spline smoothing approach to non-parametric regression curve fitting, J. R. Statist. Soc, B. **47**, 1-52, 1985.
10. Bookstein, F. L., The line-skeleton, CVGIP, **11**, 1233-137, 1979
11. Rosenfeld, A., Axial representations of shapes, CVGIP, **33**, 156-173, 1986.

Cluster Validity and Stability of Clustering Algorithms*

Jian Yu, Houkuan Huang, and Shengfeng Tian

Dept. of Computer Science, Beijing Jiaotong University
Beijing 100044, P.R.China
jianyu@center.njtu.edu.cn

Abstract. For many clustering algorithms, it is very important to determine an appropriate number of clusters, which is called cluster validity problem. In this paper, we offer a new approach to tackle this issue. The main point is that the better outputs of clustering algorithm, the more stable. Therefore, we establish the relation between cluster validity and stability of clustering algorithms, and propose that the conditional number of Hessian matrix of the objective function with respect to outputs of the clustering algorithm can be used as cluster validity cluster index. Based on such idea, we study the traditional fuzzy c-means algorithms. Comparison experiments suggest that such a novel cluster validity index is valid for evaluating the performance of the fuzzy c-means algorithms.

1 Introduction

Cluster analysis plays an important role in pattern recognition fields. However, the outputs of clustering algorithms are sensitive to parameters of the clustering algorithm. Sometimes, the same algorithm can lead to totally different outputs with respect to different parameters. A good clustering algorithm could produce undesirable results if parameters are chosen improperly. In the literature, many researches have been done on how to choose the optimal parameters in the clustering algorithms, particularly on how to choose the optimal number of clusters, for example, [1-4]. In general, selection of appropriate number of clusters and evaluation of outputs of clustering algorithms are called cluster validity problem.

In the literature, how to choose the optimal number of clusters depends on specific clustering algorithm. Many results on this issue are relevant to *c*-means or fuzzy c-means, for example, [1-4]. As for cluster validity for the FCM, one common approach is to design a cluster validity index to evaluate the performance of clustering algorithms. Frequently, there are two ways to design cluster validity index. One is based on the concept of fuzzy partition, the main assumption is that the performance of the FCM is better when its outputs are closer to crisp partition, for example, partition coefficient [2], partition entropy [5], uniform data functional [6], proportion

* This work was partially supported by the Key Scientific Research Project of MoE, China under Grant No.02031 and the National Natural Science Foundation of China under Grant No.60303014.

exponent [7], nonfuzziness index [8], *etc*. However, as noted in [9], they lack of direct connection to the geometrical property of data set. Taking into account geometrical property results in another way to design cluster validity index, for instance, Xie-Beni index [9], Gunderson's separation coefficient [10], *etc*. In fact, the above cluster validity indices have the similar drawback, i.e., all of them do not pay enough attention to the property of the FCM itself.

From a point of algorithmic view, it is necessary to study the properties of clustering algorithms in order to determine number of clusters and evaluate clustering results.

Speaking roughly, given that data set truly follows the assumption of clustering algorithm, the probable outputs of a clustering algorithm should be the optimal clustering results. Obviously, it is a reasonable assumption. Otherwise, it has little chance to obtain the optimal clustering results no matter what cluster validity index is used. Therefore, we need to measure the probability of occurrence with respect to different outputs produced by clustering algorithm. It is easy to conjecture that clustering results with high stability are outputted with large probability. Therefore, we need to obtain and study the stability criterion of clustering algorithm. In the following, we study cluster validity for the FCM according to the above idea.

The reminder of this paper is organized as follows: In Section 2, the FCM and relevant cluster validity indices are related. In Section 3, a new cluster validity index, stable index, is defined and analyzed. In Section 4, numerical experiments are carried out to make a comparison between Xie-Beni index and our cluster index, and experimental results are analyzed. In the final, we draw conclusion and make a discussion.

2 The FCM Algorithm and Related Cluster Validity Indices

Let $X=\{x_1,x_2,\ldots,x_n\}$ be a s-dimensional data set, $u=\{u_{ik}\}$ is partition matrix, $v=\{v_1,v_2,\ldots,v_c\}$ is clustering prototype, the objective function is defined as $J_m(u,v,X)=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m \ \|x_k-v_i\|^2$. The aim of the FCM algorithm is to obtain the partition matrix $u=\{u_{ik}\}_{c\times n}$ and clustering prototype $v=\{v_1,v_2,\ldots,v_c\}$corresponding to the minimum of the objective function J_m, where $\forall k\in\{k\,|\,1\le k\le n\}$, $\forall i\in\{i\,|\,1\le i\le c\}$, the membership u_{ik} represents the degree that x_k belongs to the clustering center v_i, and $u=[u_{ik}]_{c\times n}\in M_{fcn}=\{u=[u_{ik}]_{c\times n}\ \big|\ \sum_{i=1}^{c}u_{ik}=1, u_{ik}\ \ge 0, 0<\sum_{i=1}^{c}u_{ik} < n\}$.

By Lagrange multiplier's approach, we obtain the necessary conditions for the minimum of $J_m(u,v,X)$ as:

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m} \tag{1}$$

$$u_{ik} = \left(\sum_{j=1}^{c} \left(\|x_k - v_i\|^2 \|x_k - v_j\|^{-2} \right)^{\frac{1}{m-1}} \right)^{-1} \tag{2}$$

Consequently, the procedure of the FCM is described as follows:

Step 1. Fix the number of clusters, the weighting exponent, the iteration limit T and the tolerance ε , and set $J_m(u, v, X) = \infty$; initialize the partition matrix;

Step 2. Update the cluster center v_i ($1 \le i \le c$) by (1) ;

Step 3. Update the membership function u_{ik} ($1 \le i \le c$, $1 \le k \le n$) by (2);

Step 4. Repeat Step 2 and Step 3 until the decreasing value of $J_m(u, v, X)$ between two successive iterations is less than ε or the iterations reach T.

The objective function of the FCM can be reduced to (3) by (2), which is obtained by Bezdek in [11] as follows: $J_m(X, v) = \sum_{k=1}^{n} \left(\sum_{i=1}^{c} \|x_k - v_i\|^{\frac{-2}{m-1}} \right)^{1-m}$ (3)

It has been proved that the above algorithm converges to local minimum or saddle point of the objective function of the FCM when m>1. Let $\bar{x} = n^{-1} \sum_{k=1}^{n} x_k$, when m approaches infinite, the only solution of the FCM is $\bar{x}$ according to [1]. It is easily proved that $\bar{x}$ is a fixed point of the FCM algorithm.

As the clustering results of the FCM are greatly influenced by the weighting exponent, number of clusters, etc, it is a key issue for users to properly evaluate the clustering results of the FCM algorithm. In the literature, many methods are proposed to tackle this issue. One of the most used methods is to design an appropriate cluster validity index; Halkidi *et al* presented a well-written review of cluster validity indices in [12]. As noted in [1], many cluster validity indices like $V_{pc}(u)$ or V_{XB} have a monotone tendency with number of clusters increasing. Hence, it is always supposed that the optimal number of clusters has an upper bound $c_{\max} \le \sqrt{n}$, more details can be seen in [13]. In [1], Pal and Bezdek evaluated that the partition coefficient and entropy index, Xie-Beni Criterion, extended Xie-Beni Criterion, and the Fukuyama-Sugeno Index [14] by numerical experiments and limit analysis. They experimentally discovered that Xie-Beni Criterion provided the best response over a wide arrange of choices for number of clusters c and weighting exponent m, and the Fukuyama-Sugeno Index is not robust to both high and low values of weighting exponent m and its performance may be not stable as cluster validity index. Therefore, we use Xie-Beni index as benchmark of cluster validity index for the FCM algorithm in the following. As a matter of fact, we have another theoretical explanation of choosing a Xie-Beni index as benchmark of cluster validity index, more details will be given in Section 3.

3 A Novel Cluster Validity Index-Stability Index

As noted above, many cluster validity indices for the FCM algorithm have been proposed in the literature. However, all of them do not pay enough attention to the properties of the FCM algorithm. In this paper, we propose a novel cluster validity index based on the properties of the FCM algorithm itself. It is a reasonable assumption that the probable clustering output is the optimal clustering result of clustering algorithms if the data set complies with its clustering hypothesis. Obviously, the more stable the clustering result is, the more probable it is outputted. Therefore, we need to obtain stability criterion of clustering algorithm. Transparently, the stability criterion of a clustering algorithm depends on its optimality test. Fortunately, the optimality test of the FCM algorithm has been given in [15] or [16] as:

$$F(v) = \min_{u \in M_{fcn}} J_m(u, v, X) = \sum_{k=1}^{n} \left(\sum_{i=1}^{c} \|x_k - v_i\|^{\frac{-2}{m-1}} \right)^{1-m}$$

$$\frac{\partial^2 F(v)}{\partial v_i \partial v_j} = \frac{4m}{m-1} \sum_{k=1}^{n} u_{ik}^{\frac{m+1}{2}} u_{jk}^{\frac{m+1}{2}} \frac{(x_k - v_j)}{\|x_k - v_j\|} \frac{(x_k - v_i)^T}{\|x_k - v_i\|}$$

$$+ 2\delta_{ij} \left[\left(\sum_{k=1}^{n} u_{ik}^m \right) \times I_{s \times s} - \frac{2m}{m-1} \sum_{k=1}^{n} u_{ik}^m \frac{(x_k - v_i)(x_k - v_i)^T}{\|x_k - v_i\|^2} \right], \quad \forall 1 \le i, j \le c$$

Hessian matrix H_v of $F(v)$ can be represented by $H_v = \left(\partial^2 F(v) / \partial v_i \partial v_j\right)$. It is well known that H_v can judge whether the clustering result is stable or not, i.e., if the clustering result is a local minimum of the objective function. However, how to measure the probability the clustering result v is outputted? It is easy to conjecture that the stable degree of a clustering result is proportional to the probability the clustering result v is outputted. Therefore, we need to define an index to measure the stability of a clustering result. Since the conditional number of Hessian matrix H_v reflects the stability of Hessian matrix H_v, it can be reasonably used as an index to show the stability of a clustering result. In order to clearly visualize the experimental results in this paper, we use an ad hoc definition of the conditional number of Hessian matrix as follows: $cond(H_v) = \lambda_{\min}(H_v) / \lambda_{\max}(H_v)$, where $\lambda_{\max}(H_v)$, $\lambda_{\min}(H_v)$ are the maximum and the minimum eigenvalues of H_v, respectively.

If $1 \ge cond(H_v) > 0$, the clustering result of FCM is stable; if $cond(H_v) < 0$ or $cond(H_v) > 1$, the clustering result of FCM is unstable. Therefore, we call $cond(H_v)$ stability index. Obviously, if $1 \ge cond(H_v) > 0$, the larger $cond(H_v)$, the more stable the clustering result, therefore the more probable it is outputted. In other words, $cond(H_v)$ can measure the probability of a clustering result outputted by its corresponding algorithm. According to the above analysis, $cond(H_v)$ can be used as cluster validity index to choose the optimal number of clusters.

According to [17], undesirable solutions of clustering algorithm can be defined as follows: if $v = (v_1, v_2, \cdots, v_c)$ is an output of clustering algorithm and $\exists\ 1 \le i \ne j \le c$ such that $v_i = v_j$, then it is called undesirable solutions of clustering algorithm. Noticing that Xie-Beni Criterion (in this paper, Xie-Beni Criterion and Xie-Beni index are interchangeable) is defined as:

$$V_{XB} = \left(\sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2\right) \Big/ \left(n \times \min_{1 \le i \ne j \le c} \|v_i - v_j\|^2\right)$$

So, we can use Xie_Beni index to determine whether or not outputs of the FCM are undesirable solutions. But others cluster indices can not well conduct this task. For example, Fukuyama-Sugeno Index [14], partition coefficient [2], partition entropy [5], uniform data functional [6], proportion exponent [7], nonfuzziness index [8], Gunderson's separation coefficient [10]. Noticing the value of Xie_Beni index is infinite when outputs of the FCM are undesirable solutions, we use the following form instead of Xie-Beni index, which is convenient for calculation and visualization in the computer and denoted by V_{XB}^{-1}:

$$V_{XB}^{-1} = \frac{\min_{1 \le i \ne j \le c} \|v_i - v_j\|^2}{\sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2}.$$

In Section 4, we verify the above conclusions by numerical experiments.

4 Numerical Experiments and Analysis

In this section, we verify the above conclusions by numerical experiments. In the following, we set the same initial partition matrix, $\varepsilon = 10^{-8}$ and the maximum iteration T=200 for the FCM algorithm, and run the FCM algorithm with different weighting exponent m and number of clusters c, then calculate V_{XB}^{-1} and $cond(H_v)$ according to the clustering outputs.

The datasets used in this section are described as follows:

IRIS data: The Iris data set has 150 data points. It is divided into three groups and two of them are overlapping. Each group has 50 data points. Each point has four attributes. More details about the IRIS data are available in Anderson [18].

Cube_6. This data set is drawn as in Fig.1 (a), and consists of 6 clusters. Each cluster consists of 8 points located at 8 corners of a cube. More details about Cube6 can be seen in [16].

Data 3: Data are composed of 4 clusters as shown in Fig.1 (b). The cluster centers are as follows: μ_1 = [-4, 4]; μ_2 = [5, 5]; μ_3 = [14, 5]; μ_4 = [20, -3]. Each cluster includes 100 points and the points in the ith cluster are independently drawn from the normal distribution $N(\mu_i, I_2)$.

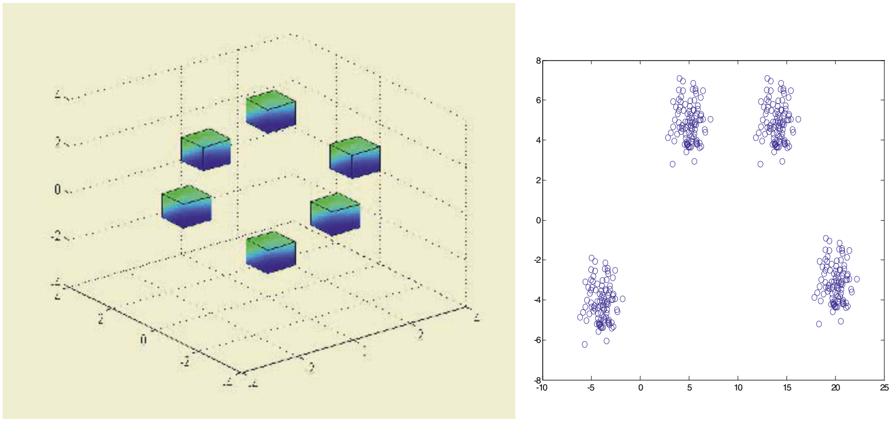

(a) Cube 6 (b) Data 3

Fig. 1. (a) Cube 6, (b) Data 3

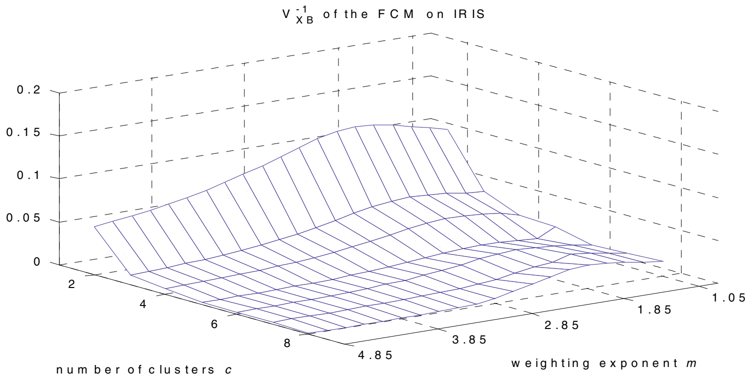


Fig. 2. V_{XB}^{-1} with varying c and m for IRIS

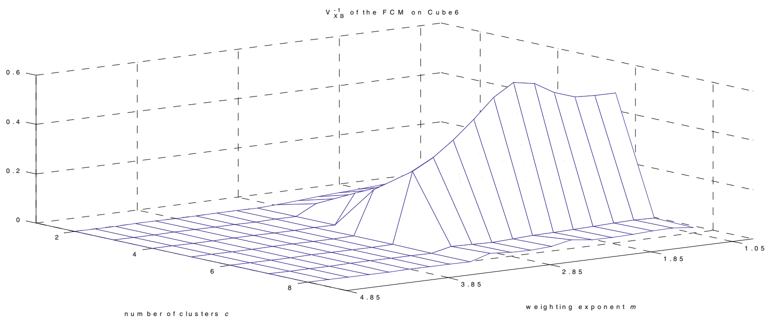


Fig. 3. V_{XB}^{-1} with varying c and m for Cube6

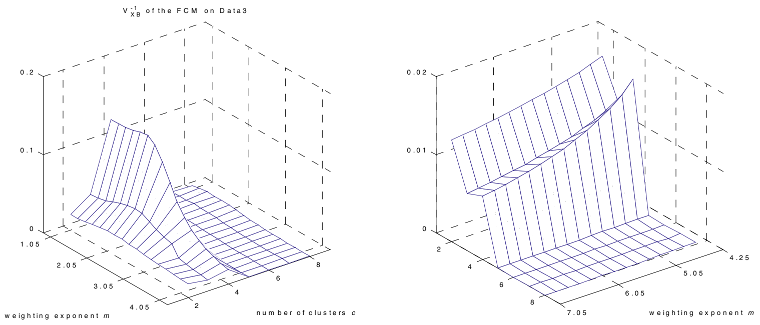


Fig. 4. V_{XB}^{-1} with varying c and m for Data 3. In the left, the weighting exponent varies from 1.05 to 4.05, V_{XB}^{-1} shows that the optimal number of clusters is 4, which is consistent with the real substructure of Data 3. However, in the right, the weighting exponent varies from 4.25 to 7.05, V_{XB}^{-1} shows that the optimal number of clusters becomes 2. According to [19], we know the FCM algorithm works well for m>1, at least in theory. It easily demonstrates that the performance of V_{XB}^{-1} sometimes heavily depends on the weighting exponent m in the FCM algorithm.

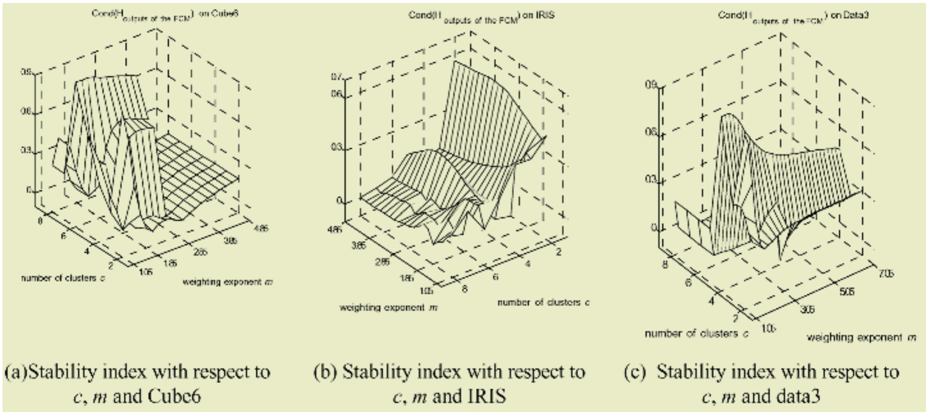


Fig. 5. Stability index with respect to c and m, and datasets

When m>1, Fig.5 demonstrates that the outputs of the FCM algorithm are local minimum of (3) with probability close to 1, and the performance of $cond(H_v)$ as cluster validity index is the same as Xie-Beni index. We also know that weighting exponent m plays a key role in the FCM algorithm. In [19], it is proved that if $\lambda_{\max}\left(F_{U^*_{data}}\right) < 0.5$ and $m \geq \left(1 - 2\lambda_{\max}\left(F_{U^*_{data}}\right)\right)^{-1}$, then U^*_{data} is a stable fixed point of the FCM, and if $\lambda_{\max}\left(F_{U^*_{data}}\right) \geq 0.5$, then U^*_{data} is not a stable fixed point of the FCM, where $F_{U^*_{data}} = \left(f_{kr}^{U^*_{data}}\right)_{n\times n}$, $f_{kr}^{U^*_{data}} = \frac{1}{n}\left(\frac{x_k - \bar{x}}{\|x_k - \bar{x}\|}\right)^T\left(\frac{x_r - \bar{x}}{\|x_r - \bar{x}\|}\right)$, $U^*_{data} = \left[\kappa_{ij}\right]_{n\times n}$, $\kappa_{ij} = c^{-1}$.

It is easy to calculate that $\lambda_{\max}(F_{IRIS})$ =0.8079, $\lambda_{\max}(F_{Cube6})$ =0.3333 and $\lambda_{\max}(F_{Data3})$=0.7036. Therefore, we know that *m*<3 should hold in order to keep the FCM algorithm work well on Cube 6 according to [19]. Since $\lambda_{\max}(F_{IRIS})$=0.8079>0.5 and $\lambda_{\max}(F_{Data3})$=0.7036>0.5, any value of *m*>1 is theoretically appropriate for the FCM algorithm.

Fig. 2, 4 verify that the FCM algorithm may not outcome undesirable solutions when $\lambda_{\max}\left(F_{U^*_{data}}\right) \geq 0.5$. However, Fig.3 tells us that the FCM algorithm indeed outcomes undesirable solutions when $\lambda_{\max}\left(F_{U^*_{data}}\right) < 0.5$. Simultaneously, Fig.5 empirically proves that the outputs of the FCM algorithm are local minima, which is consistent with our intuition. As for IRIS and Cube6, the performances of $cond(H_v)$ and Xie-Beni index are the same with respect to a wide range of the weighting exponent *m*. As for data3, Fig.4 shows that Xie-Beni index is sensitive to high values of the weighting exponent *m*. However, Fig.5 shows that the performance of $cond(H_v)$ is still satisfactory with respect to a wide range of the weighting exponent *m*. Such facts suggest that $cond(H_v)$is more robust than Xie-Beni Criterion with respect to *m* as cluster validity index.

5 Conclusions and Discussions

In this paper, we propose a novel cluster validity index for the FCM algorithm, the stability index, based on the optimality test. The major contribution of this paper is that our approach is totally out of mathematical analysis of the FCM algorithm, while other previous methods out of geometrical or psychological tuition. The theoretical analysis and experimental results suggest that the stability index is valid for the FCM algorithm as a cluster validity index. Moreover, the stability index also can be used as the optimality test of the FCM algorithm.

References

1. Pal, N.R. and Bezdek, J.C. On cluster validity for the fuzzy c-means model, IEEE Trans. Fuzzy Systems, 3(3):370-379, June,1995
2. Bezdek, J.C. Pattern recognition with fuzzy objective function algorithms, Plenum Press, New York, 1981
3. Catherine A., Sugar and Gareth M.James, Finding the number of clusters in a dataset: an information-theoretic approach, Journal of the American Statistical Association, 98(463):750-763, Sept.2003

4. Tibshirani R., Walther G. & Hastie T., Estimating the number of clusters in a data set via the gap statistic, J.R.Statist.Soc.B, 63,Part 2, pp.411-423, 2001
5. Bezdek, J.C. Cluster validity with fuzzy sets, J.Cybernt., 3(3): 58-72,1974
6. Windham, M.P., Cluster validity for the fuzzy c-means clustering algorithm, IEEE Trans. PAMI, vol. PAMI-4, no.4, 357-363, July, 1982.
7. Windham, M.P., Cluster validity for fuzzy clustering algorithms, Fuzzy Sets Systems, vol.5: 177-185,1981
8. Backer, E., Jain, A.K. A Cluster performance measure based on fuzzy set decomposition, IEEE Trans. PAMI, vol.PAMI-3,no.1, Jan. 1981.
9. Xie, X.L.; Beni, G., A validity measure for fuzzy clustering, IEEE Trans. PAMI, 13 (8): 841 –847, Aug. 1991
10. Gunderson, R. Applications of fuzzy ISODATA algorithms to startracker printing systems, in Proc. 7th Triannual World IFAC Congr. 1978, pp.1319-1323
11. Bezdek, J.C. A physical interpretation of Fuzzy ISODATA, IEEE Trans. SMC, SMC-6: 387-390,1976
12. Halkidi, M., Batistakis, Y., Vazirgiannis, M. Cluster algorithms and validity measures, Proceedings of Thirteenth International Conference on Scientific and Statistical Database Management, 3 -22, 2001.
13. Yu Jian, Cheng Qiansheng, The upper bound of the optimal number of clusters in fuzzy clustering, Science in China, series F, 44(2):119-125, 2001
14. Fukuyanma, Y.and Sugeno, M. A new method of choosing the number of clusters for the fuzzy c-means method, in Proc. 5th Fuzzy Syst. Symp., 247-250(in Japanese). 1989.
15. Wei, W. and Mendel, J.M. Optimality tests for the fuzzy c-means algorithm, Pattern Recognition, vol. 27(11): 1567-1573, 1994.
16. Yu Jian, Huang Houkuan, Tian Shengfeng, An Efficient Optimality Test for the Fuzzy c-Means Algorithm, Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, vol. 1, 98-103, 2002
17. Yu Jian, General c-means clustering model and its applications, CVPR2003, v.2, 122-127, June. 2003
18. Anderson, E. The IRISes of the Gaspe Peninsula, Bull. Amer. IRIS Soc., vol.59, pp.2-5, 1935.
19. Yu Jian, Cheng Qiansheng, Huang Houkuan, Analysis of the weighting exponent in the FCM, IEEE Transactions on Systems, Man and Cybernetics-part B: Cybernetics, 34(1):634-639, Feb.2004

Bagging Classification Models with Reduced Bootstrap

Rafael Pino-Mejías[1,2], María-Dolores Cubiles-de-la-Vega[2], Manuel López-Coello[3], Esther-Lydia Silva-Ramírez[3], and María-Dolores Jiménez-Gamero[2]

[1] Centro Andaluz de Prospectiva, Avda. Reina Mercedes, s/n, 41012 Sevilla, Spain
[2] Departamento de Estadística e Investigación Operativa, Facultad de Matemáticas
Universidad de Sevilla, Avda. Reina Mercedes, s/n, 41012 Sevilla, Spain
{rafaelp,cubiles,dolores}@us.es
[3] Departamento de Lenguajes y Sistemas Informáticos, E. Superior de Ingeniería
Universidad de Cádiz, C/ Chile 1, 11002 Cádiz, Spain
{manuel.coello,esther.silva}@uca.es

Abstract. Bagging is an ensemble method proposed to improve the predictive performance of learning algorithms, being specially effective when applied to unstable predictors. It is based on the aggregation of a certain number of prediction models, each one generated from a bootstrap sample of the available training set. We introduce an alternative method for bagging classification models, motivated by the reduced bootstrap methodology, where the generated bootstrap samples are forced to have a number of distinct original observations between two values k_1 and k_2. Five choices for k_1 and k_2 are considered, and the five resulting models are empirically studied and compared with bagging on three real data sets, employing classification trees and neural networks as the base learners. This comparison reveals for this reduced bagging technique a trend to diminish the mean and the variance of the error rate.

1 Introduction

Bagging (Bootstrap Aggregating) is a method proposed by Breiman [1] to improve the performance of prediction models. Given a model, bagging draws B independent bootstrap samples from the available training set, fits a particular model to each bootstrap sample, and finally it aggregates the B models by computing the mean (regression) or majority voting (classification). Bagging is a very effective procedure when applied to unstable learning algorithms such as classification and regression trees and neural networks. The empirical success of the first published works has been confirmed by theoretical results as we can see in [2], where bagging is shown to smooth hard decision problems, yielding smaller variance and mean squared error (MSE). These results have been derived for classification and regression trees, but the variance and MSE reduction effect of bagging is not necessarily true for other models, as it is shown in [3] for U-statistics.

Bagging averages models constructed over nearby empirical distributions corresponding to replacement samples from the training set. However, if we consider other classes of neighborhoods of the empirical distribution of the original sample, or if we vary the method to carry out the aggregating process, a more general bagging is defined. The use of robust location measures, as the median, is an example of the second approach. For the first approach, we could draw samples with or without replacement, and sample sizes not necessarily equal to the training set size would also be considered, as is the case for Subbagging (Subsample Aggregating) in [2].

If we maintain the replacement sampling process, a generalization is motivated by the following reasoning of [4]: bootstrap samples are simple random samples of size n selected with replacement from the original n sized sample, so not all bootstrap samples are equally informative, due to the randomness associated to the number of distinct original observations in the bootstrap sample. The variability of this number is neither necessary nor desirable, having negative effects on the performance of the bootstrap technique in certain applications. For example, the bootstrap does not provide a consistent estimator for the variance of the median, but an alternative bootstrap resampling scheme which solves that inconsistency is presented in [5]. We propose to consider this alternative bootstrap procedure, namely the reduced bootstrap, as the sampling algorithm for bagging. In section 2 we present this new method, while section 3 is devoted to some empirical comparisons with the usual bagging procedure, resuming the main conclusions and the future work in section 4.

2 Reduced Bootstrap

We consider a classification problem where a training set $\mathbf{D}=\{U_i=(X_i,Y_i),\ i=1,\ldots,n\}$ is available. X_i is a realization of a multidimensional predictor variable and Y_i contains the label of the class of the case i, for example an element of $\{1,2,\ldots,K\}$ for a K-class problem. Given a classification model g, depending on a set of parameters to be optimized, bagging was defined in [1] as follows:

Definition 1. Algorithm Bagging

```
Fix B
 For $b$=1,2,…,B
  Draw a bootstrap sample, i.e., a simple random sample with replacement
   $\mathbf{D}^* =(U_1^*,\ldots,U_n^*)$ taken from $\mathbf{D}$.
  Fit $g$ to $\mathbf{D}^*$, obtaining $g_b$.
 Next b.
```

The aggregate model g_{agg} is defined by voting of the B computed models:

$$g_{agg}(x) = \arg\max_j f_j(x) \tag{1}$$

$$f_j(x) = \#\{g_b(x) = j\} \tag{2}$$

We must note that the bootstrap procedure inside bagging is really what Efron called in [6] the bootstrap Method II, used to approximate a theoretical distribution by Monte Carlo simulation. However, this simulation process is affected by a series of errors and variabilities, as is formalized in [7]. For this reason, several alternative techniques have been proposed, as those recorded by [4], [8], [9].

In [7] we defined a variation of Efron´s method II based on the outlier bootstrap sample concept, namely OBS, that is based on only considering those bootstrap samples having a number of distinct original observations d_n greater or equal to some value computed from the distribution of such random variable d_n. Several empirical studies carried out in [7] showed closer estimations of the parameters under study and a reduction of the standard deviations of such estimations. These results were theoretically confirmed in [10].

In this paper we consider a generalization of the OBS method, that consists of drawing bootstrap samples verifying $k_1 \le d_n \le k_2$ for some $1 \le k_1 \le k_2 \le n$. We will name RB (Reduced Bootstrap) to this method. This way, only αn^n bootstrap samples are considered, where $\alpha = P[k_1 \le d_n \le k_2]$. The use of RB inside a bagging procedure lets us to define Bagging with Reduced Bootstrap. We will name Rbagging the resulting procedure.

Definition 2. Algorithm Rbagging.

```
Fix B, $k_1$, $k_2$
 For $b$=1,2,…,B
  Draw a reduced bootstrap sample, i.e., a bootstrap sample D*= $(U_1^*,\dots,U_n^*)$
    with $k_1 \le d_n^* \le k_2$, taken from D
  Fit $g$ to D*, obtaining $g_b$
Next $b$.
```

The resulting aggregated model is also computed as in (1) and (2). To obtain a bootstrap sample **D*** with $k_1 \le d_n^* \le k_2$, we propose the next algorithm.

Definition 3. Algorithm Reduced Bootstrap Sampling.

1. Select a random sample of size k_2 without replacement from $\{1,\dots,n\}$, say I_1
2. Select a random sample of size k_1 without replacement from I_1, say I_2
3. Draw a random sample of size $n-k_1$ with replacement from I_1 say I_3
4. Let $L=(l_1,\dots,l_n)$ be a vector whose components are obtained by randomly permuting the string formed by concatenating I_2 and I_3
5. The sample obtained taking the elements of **D** indexed by $(l_1,\dots,l_n)$ is a bootstrap sample **D*** $=(U_1^*,\dots,U_n^*)$, with $k_1 \le d_n^* \le k_2$.

In [5], six choices for k_1 and k_2 are proposed in a study about the consistent estimation of the variance of the sample median, including the usual bagging as a particular case. Because of its good performance, we have used these selections to study the performance of Rbagging. The six resulting methods are presented in table 1, being identified by RB1,…, RB6, where p=1-1/e, q=1-p. Note that RB1 is the original method II presented by Efron, while RB2 and RB6 are particular cases of the OBS method.

Table 1. The six selections for k_1 and k_2

Method	k_1	k_2
RB1	1	n
RB2	$[np\text{-}(npq)^{1/2}]+1$	n
RB3	$[np\text{-}(npq)^{1/2}]+1$	$[np+(npq)^{1/2}]$
RB4	$[np]+1$	$[np]+1$
RB5	$[np+(npq)^{1/2}]+1$	$[np+(npq)^{1/2}]+1$
RB6	$[np+(npq)^{1/2}]+1$	n

3 Numerical Results

We have made an empirical comparison of the six considered methods over three real data sets. Two unstable classification models, classification trees and neural networks, are used as the base algorithm. R system [11] has been the selected computational tool for our study, whereas the tree and nnet libraries have provided us with the implementation of classification trees and multiplayer perceptrons, respectively. Tree library is based on the CART methodology [12] proposed by Breiman. Nnet library fits single-hidden-layer neural networks by a quasi-Newton method (also known as a variable metric algorithm), specifically that published simultaneously in 1970 by Broyden, Fletcher, Goldfarb and Shanno. We have used the logistic activation function in the hidden layer and the identity function as the activation function for the output layer, selecting the hidden layer size by cross validation.

3.1 Fragile X Syndrome Data

Fragile X is the most common inherited cause of mental impairment and the most common known cause of autism. In 1991, the gene (called FMR1) that causes Fragile X was discovered. In individuals with Fragile X, a defect in FMR1 (a "full mutation") shuts the gene down. Symptoms of fragile X include: mental impairment, ranging from learning disabilities to mental retardation, attention deficit and hyperactivity, anxiety and unstable mood, autistic-like behaviors, long face, large ears, flat feet, and hyperextensible joints, especially fingers. A DNA based test to diagnose Fragile X was developed in 1992. This test is quite accurate, and it can detect both carriers and fully-affected individuals. However it can be very expensive, and for this reason, an automatic classification model would be acknowledged, motivating a study conducted in Andalusia, Spain, where 100 FMR1 mutated children and 72 children with Fragile X symptoms but not mutated were selected, being the last 72 the control cases. From the 61 recorded variables, we selected those variables retained by a stepwise logistic regression analysis performed with SPSS v11.0, reducing to 9 the number of predictor variables.

We randomly divided the data set into training (80%) and test (20%) sets, and we applied the six bagging procedures with B=100 to the classification tree and mutilayer perceptron with 12 hidden nodes, computing the error rate (percent of incorrectly classified cases) for both data sets for each method. The whole procedure were independently repeated 50 times. Table 2 shows the mean and standard deviation of the 50 test error rates for each bagging procedure, where "raw" denotes no bagging.

Table 2. Fragile X Syndrome data. Mean and standard deviation of the 50 test error rates for each procedure

	Classification trees		Multilayer perceptron	
Method	Mean	S.D.	Mean	S.D.
Raw	5.652	5.111	5.403	4.592
RB1	5.977	5.043	5.607	4.919
RB2	5.225	4.584	5.285	4.675
RB3	5.799	5.524	5.388	4.571
RB4	5.225	4.828	5.669	4.958
RB5	5.448	4.629	5.375	5.203
RB6	5.577	4.388	5.329	4.549

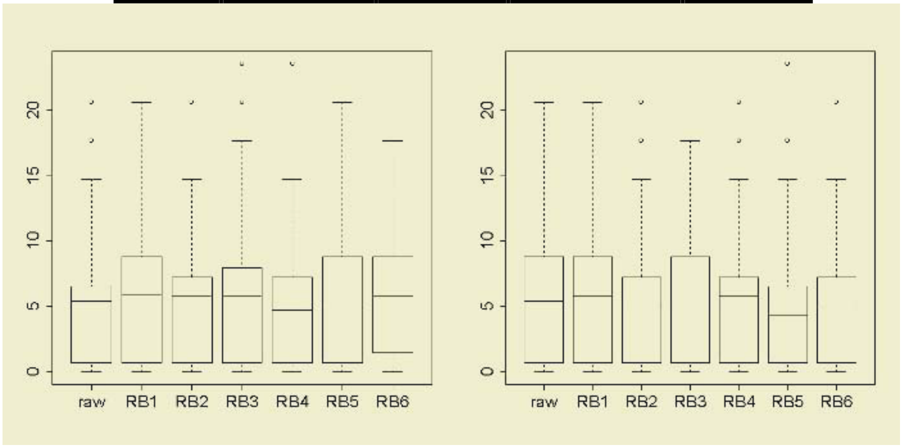


Fig. 1. Distribution of the 50 test mean error rates for the *raw* and *bagged* classification trees (left) and multilayer perceptrons with 12 hidden nodes (right) for the fragile X syndrome data

We can see in table 2 and figure 1 that for both models the mean error rate is increased when bagging is applied. However, for classification trees reduced bagging 2, 4, 5 and 6 yield a lower mean error rate, accompanied by a lower variance of the error rate. A similar comparative performance is observed for the multiplayer perceptron, with the exception of an increase in the variability of RB5 (motivated by two outliers), though RB3 also offers a reduction in the mean and standard deviation of the test error rate. We must note that RB2, a reduced bagging based on OBS bootstrap, produces the minimum mean error rate and a clear reduction of the variability, for both classification models.

3.2 Forensic Glass Data

The forensic glass dataset has 214 points from six classes with nine measurements, and provides a fairly stiff test for classification methods. As in 3.1, we randomly divided the data set into training (80%) and test (20%) sets, applying the six bagging procedures with B=100 to the classification tree and mutilayer perceptron with 15 hidden nodes, also computing the error rate (percent of incorrectly classified cases) for both data sets for each method. The whole procedure were independently repeated 50 times, and the main results are exhibited in table 3 and figure 2.

Table 3. Forensic glass data. Mean and standard deviation of the 50 test error rates for each procedure

	Classification trees		Multilayer perceptron	
Method	Mean	S.D.	Mean	S.D.
Raw	32.662	3.211	50.761	13.141
RB1	23.441	2.334	39.627	10.892
RB2	23.239	2.441	39.243	24.924
RB3	23.621	2.178	39.426	10.010
RB4	23.622	2.156	39.042	9.326
RB5	23.337	2.561	40.572	9.964
RB6	23.620	2.357	39.624	9.153

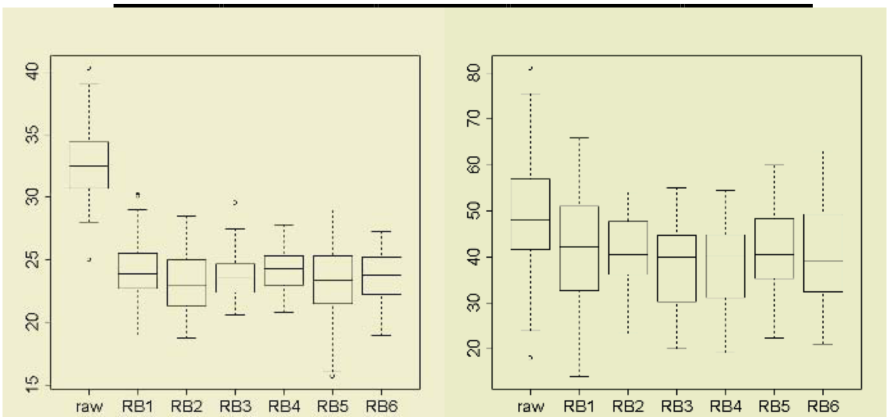


Fig. 2. Distribution of the 50 test mean error rates for the *raw* and *bagged* classification trees (left) and multilayer perceptrons with 15 hidden nodes (right) for the forensic glass dataset

Figure 2 (left) shows the box-and-whisker plots for the classification tree, where a clear reduction in the mean error rate is observed for all the six bagging procedures. However, a slight additional reduction with RB2 and RB5 is observed, though last method has a greater variability. Figure 2 (right) contains a similar representation for the multiplayer perceptron. The bests results are also achieved by RB2, with a great reduction in the mean percent error rate and in its variability.

3.3 South Africa Heart Disease Data

This dataset, utilized in [13], contains 463 cases selected from a larger retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa. The target is the absence/presence of a coronary heart disease, existing nine predictor variables. A similar study as in 3.1 and 3.2 were conducted: we randomly divided the data set into training (80%) and test (20%) sets, applying the six bagging procedures with B=100 to the classification tree and mutilayer perceptron with 12 hidden nodes, also computing the error rate (percent of incorrectly classified cases) for both data sets for each method. The whole procedure were also independently repeated 50 times. Table 4 shows the mean and standard deviation of the 50 test error rates for each bagging procedure, and the whole distributions are plotted in the figure 3.

Table 4. South Africa heart disease data. Mean and standard deviation of the 50 test error rates for each procedure

	Classification trees		Multilayer perceptron	
Method	Mean	S.D.	Mean	S.D.
Raw	33.217	4.398	34.434	4.783
RB1	31.173	4.350	34.134	5.042
RB2	30.693	4.572	34.326	4.143
RB3	30.608	4.355	34.413	4.378
RB4	30.630	4.295	34.413	4.433
RB5	31.021	4.082	34.086	4.313
RB6	30.562	4.082	34.108	4.358

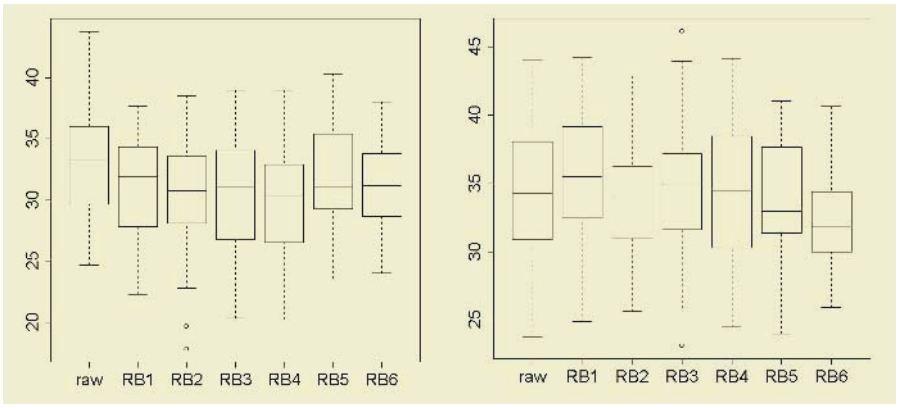


Fig. 3. Distribution of the 50 test mean error rates for the *raw* and *bagged* classification trees (left) and multilayer perceptrons with 12 hidden nodes (right) for the South Africa heart disease dataset

We see that the five reduced bagging procedures yield a mean percent error rate lower than the raw and bagged classification trees, standing out the reduced bagging 6, which also provides the minimum standard deviation, as it is confirmed by the figure 3 (left). For the multiplayer perceptron the bagging procedure is not so clearly improved, but the procedures rbagging 5 and 6 provide the lowest mean values, accompanied by a standard deviation lower than that achieved by the usual bagging procedure. This better performance, particularly for rbagging 6, is more clearly illustrated in the figure 3 (right).

4 Concluding Remarks

The alternative bagging methodology based on reduced bootstrap sampling shows good and hopeful results. It has outperformed the usual bagging in our empirical study over real data sets, at least one rbagging method which offers a lower mean and variance of the test error rate is found for each data set.

However, a further study may be realized following some guidelines, for example: the theoretical study of the properties of rbagging, the development of criteria to select the parameters k_1 and k_2, a comparison with other ensemble methods, to analyze the effect of rbagging over other learning algorithms, and the application to prediction problems.

References

1. Breiman, L.: Bagging Predictors. Mach. Learn. 24 (1996) 123–140
2. Bühlman, P., Yu, B.: Analyzing Bagging. Ann. Stat. 30 (4) (2002) 927-961
3. Buja, A., Stuetzle, W.: The effect of bagging on variance, bias, and mean squared error. Preprint, AT&T Labs-Research (2000)
4. Rao, C.R., Pathak, P.K., Koltchinskii, V.I.: Bootstrap by sequential resampling. J. Statist. Plan. Infer. 64 (1997) 257-281
5. Jiménez-Gamero, M.D., Muñoz-García, J., Pino-Mejías, R.: Reduced bootstrap for the median. Stat. Sinica (2004) (in press)
6. Efron, B.: Bootstrap methods: another look at the jackknife. Ann. Stat. 7 (1979) 1-26
7. Muñoz-García, J., Pino-Mejías, R., Muñoz-Pichardo, J.M., Cubiles-de-la-Vega, M.D.: Identification of outlier bootstrap samples. J. Appl. Stat. 24 (3) (1997) 333-342
8. Hall, P.: Antithetic resampling for the bootstrap. Biometrika (1989) 713-724
9. Johns, M.V.: Importance sampling for bootstrap confidence intervals. J. Am. Stat. Assoc. 83 (1988) 709-714
10. Jiménez-Gamero, M.D., Muñoz-García, J, Muñoz-Reyes, A., Pino-Mejías, R.: On Efron´s method II with identification of outlier bootstrap samples. Computation. Stat. 13 (1998) 301-318
11. Ihaka, R. & Gentleman, R.: R: A Language for Data Analysis and Graphics. J. Comput. Graph. Stat. 5 (1996) 299-314
12. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth (1984)
13. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer (2001)

Multiple Classifier Prediction Improvements against Imbalanced Datasets through Added Synthetic Examples

Herna L. Viktor and Hongyu Guo

School of Information Technology and Engineering, University of Ottawa
800 King Edward Road, Ottawa, Ontario, Canada, K1N 6N5
{hlviktor,hguo028}@site.uottawa.ca

Abstract. Ensembles of classifiers have successfully been used to improve the overall predictive accuracy in many domains. In particular, the use of boosting which focuses on hard to learn examples, have application for difficult to learn problems. In a two-class imbalanced data set, the number of examples of the majority class is much higher than that of the minority class. This implies that, during training, the predictive accuracy against the minority class of a traditional boosting ensemble may be poor. This paper introduces an approach to address this shortcoming, through the generation of synthesis examples which are added to the original training set. In this way, the ensemble is able to focus not only on hard examples, but also on rare examples. The experimental results, when applying our Databoost-IM algorithm to eleven datasets, indicate that it surpasses a benchmarking individual classifier as well as a popular boosting method, when evaluated in terms of the *overall accuracy*, the *G-mean* and the *F-measures.*

1 Introduction

Over the past few years, ensembles have emerged as a promising technique with the ability to improve the performance of weak classification algorithms [1, 2]. Ensembles of classifiers consist of a set of individually trained classifiers whose predictions are combined to classify new instances [1, 2]. In particular, boosting is an ensemble method where the performance of weak classifiers is improved by focusing on hard examples which are difficult to classify. Boosting produces a series of classifiers and the outputs of these classifiers are combined using weighted voting in the final prediction of the model [3]. In each step of the series, the training examples are re-weighted and selected based on the performance of earlier classifiers in the training series. This produces a set of "easy" examples with low weights and a set of hard ones with high weights. During each of the iterations, boosting concentrates on classifying the hard examples correctly. Recent studies have indicated that boosting algorithm is applicable to a broad spectrum of problems with great success [3, 4].

The class imbalance problem corresponds to domains for which one class is represented by a large number of examples while the other is represented by only a few [5]. Many real world applications involve learning from imbalanced sets, such as fraud detection, telecommunications management, oil spill detection and text classification [6]. When learning from imbalanced data sets, machine learning algorithms tend to produce high predictive accuracy over the majority class, but poor predictive accuracy over the minority class [7]. There have been several proposals for coping with imbalanced data sets [5], including under-sampled examples of the majority class and/or over-sampling of the minority class [6, 7, 8] and weighing examples in an effort to bias the learning toward the minority class [7]. Research into the use of boosting includes [9], which evaluated boosting algorithms to classify rare classes; and [10] which combined boosting and synthetic data to improve the prediction of the minority class.

In this paper, we discuss the DataBoost-IM approach, which combines data generation and boosting procedures to improve the predictive accuracies of both the majority and minority classes, without forgoing one of the two classes. The aim of our approach is therefore to ensure that the resultant predictive accuracies of both classes are high. Our approach differs from prior work in the following ways. Firstly, we separately identify hard examples from, and generate synthetic examples for, the minority as well as the majority classes. Secondly, we generate synthetic examples with *bias* information toward the hard examples on which the next component classifier in the boosting procedures needs to focus. That is, we provide additional knowledge for the majority as well as the minority classes and thus prevent boosting over-emphasizing the hard examples. Thirdly, the class frequencies of the new training set are rebalanced to alleviate the learning algorithm's bias toward the majority class. Fourthly, the total weights of the different classes in the new training set are rebalanced to force the boosting algorithm to focus on not only the hard examples, but also the minority class examples. In this way, we focus on improving the predictions of both the minority and majority classes.

This paper is organized as follows. The performance measures used to evaluate the performance of our approach is introduced in Section 2. Section 3 describes the DataBoost-IM algorithm. This is followed, in Section 4, with an evaluation of the DataBoost-IM algorithm against eleven data sets from the UCI data set repository [11]. Finally, Section 5 concludes the paper.

2 Performance Measures

Traditionally, the performance of a classifier is evaluated by considering the overall accuracy against test cases [12]. However, when learning from imbalanced data sets, this measure is often not sufficient [12]. Following [7, 8, 10, 13], we employ the *overall accuracy*, *G-Mean* [8] and *F-Measures* [14] metrics to evaluate our DataBoost-IM method. The confusion matrix, as shown in Table 1, represents the typical metrics for evaluating the performance of machine learning algorithms on skew class problems.

Table 1. Confusion Matrix

	Predicted Negative	Predicted Positive
Actual Negative	*TN* (the number of True Negatives)	*FP*(the number of False Positives)
Actual Positive	*FN* (the number of False Negatives)	*TP*(the number of True Positives)

In Table 1, the *Precision* and *Recall* are calculated as *TP / (TP + FP)* and *TP / (TP + FN)*. The *F-measure* is defined as

$$((1+\beta^2)\times Recall \times Precision) / (\beta^2 \times Recall + Precision) \tag{1}$$

where β corresponds to the relative importance of *precision* versus the *recall* and it is usually set to 1. The *F-measure* incorporates the *recall* and *precision* into a single number. It follows that the *F-measure* is high when both the *recall* and *precision* are high [9]. This implies that the *F-measure* is able to measure the "goodness" of a learning algorithm on the current class of interest. Note that we also use this measure for the majority class, since we are interested in measuring the performance of both classes. Another criteria used to evaluate a classifier's performance on skew data is the *G-mean* [8, 10, 13]. The *G-mean* is defined as

$$\sqrt{PositiveAcuracy \times NegativeAcuracy} \tag{2}$$

where *Positive Accuracy* and *Negative Accuracy* are calculated as *TP / (FN+TP)* and *TN / (TN+FP)*. This measure relates to a point on the *ROC* curve and the idea is to maximize the accuracy on each of the two classes while keeping these accuracies balanced [8]. For instance, a high *Positive accurac*y by a low *Negative accuracy* will result in poor *G-mean* [8].

3 DataBoost-IM Algorithm

The DataBoost-IM approach extends our earlier DataBoost algorithm which was successfully used to produce highly accuracy classifiers in balanced domains containing hard to learn examples [15]. In this section, we describe a variation, the DataBoost-IM algorithm, applied to imbalanced data sets. This approach extends the original DataBoost algorithm as follows. Firstly, we *separately* identify hard examples from and generate synthetic examples for different classes. Secondly, the class distribution and the total weights of different classes are *rebalanced* to alleviate the learning algorithms' bias toward the majority class.

Recall that boosting involves the creation of a series of classifiers which aims to correctly classify hard to learn examples, through focusing on these hard examples during training. Following this mechanism, the DataBoost-IM algorithm, as shown in Figure 1, consists of the following three stages. Firstly, each example of the original training set is assigned an equal weight. The original training set is used to train the first classifier of the DataBoost-IM ensembles. Secondly, the hard examples (so-called seed examples) are identified and for each of these seed examples, a set of synthetic examples is generated. During the third of the algorithm, the synthetic examples are added to the original training set and the class distribution and the total weights of different classes are rebalanced. The second and third stages of the DataBoost-IM algorithm are re-executed until reaching a user-specified number of iterations or the current component classifier's error rate is worse than a threshold value. Following the AdaBoost ensemble method, this threshold is set to 0.5 [1, 2].

The seed selection, data generation and re-balancing process of the DataBoost-IM algorithm are described next.

```
Input:      Sequence of $m$ examples $\langle (x_1, y_1),...,(x_m, y_m)\rangle$ with labels $y_i \in Y = \{1,...,k\}$
            Weak learning algorithm WeakLearn
            Integer $T$ specifying number of iterations
Initialize  $D_j(i) = 1/m$ for all $i$.
Do for      $t = 1, 2, ..., T$
    1. Identify hard examples from the original data set for different classes.
    2. Generate synthetic data to balance the training knowledge of different classes
    3. Add synthetic data to the original training set to form a new training data set
    4. Update and balance the total weights of the different classes in the new training data set
    5. Call WeakLearn, providing it with the new training set with synthetic data and rebalanced weights
    6. Get back a hypothesis $h_t : X \rightarrow Y$.
    7. Calculate error of $h_t : \varepsilon_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$ If $\varepsilon_t > 1/2$, set $T = t-1$ and abort loop.
    8. Set $\beta_t = \varepsilon_t / (1' - \varepsilon_t)$.
    9. Update distribution $D_t : D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t \leftarrow if h_t(x_i) = y_i \\ 1 \leftarrow otherwise \end{cases}$, where $Z_t$ is a normalization constant (chosen so that $D_{t+1}$ will be a distribution).
Output      The final hypothesis: $h_{fin}(x) = \arg\max_{y \in Y} \sum_{t:h_t(x)=y} \log \frac{1}{\beta_t}$
```

Fig. 1. Pseudo-code of the DataBoost-IM algorithm

3.1 Identify Seed Examples

The aim of the seed selection process is to identify hard examples for both the majority and minority classes. These examples, are used as input for the data generation process as discussed in Section 3.2.

The seed examples are selected as follows. Firstly, the examples in the training set (E_{train}) are sorted descending, based on their weights. The original training set E_{train} contains N_{maj} examples from the majority class and N_{min} examples from the minority class. The number of examples that is considered to be hard (denoted by N_s) is calculated as (E_{train} x Err), where Err is the error rate of the currently trained classifier. Next, the set E_s, which contains the N_s examples with the highest weights in E_{train}, is created. The set E_s consists of two subset of examples E_{smin} and $E_{smaj,}$ i.e. examples from the minority and majority classes, respectively. Here, E_{smin} and E_{smaj} contain N_{smin} and N_{smaj} examples, where $N_{smin} < N_{min}$ and $N_{smaj} < N_{maj}$. We select a number of seed examples of the majority class in E_{smaj} by calculating M_L, which is equal to $min(N_{maj} / N_{min}, N_{smaj})$. Correspondingly, a subset M_S of the minority class examples in $E_{smin,}$ is selected as seeds, where M_S is calculated as $min((N_{maj}$ x $M_L) / N_{min}, N_{smin})$. The final sets of seed examples are placed in sets E_{maj} and E_{min}.

3.2 Generate Synthetic Data and Balance Class Frequencies

The aim of the data generation process is to generate additional synthesis instances to *add to* the original training set E_{train}. The data generation process extends our earlier work, as presented in [15, 16, 17], by generating data for the majority and minority classes separately. That is, the data generation process generates two sets of data. Firstly, a total of M_L sets of new majority class examples, based on each seed instance in $E_{maj,}$ are generated. For each attribute included in the synthetic example, a new value is generated based on the following constraints [15, 16, 17].

For ***Nominal*** attribute, the data generation produces a total of N_{maj} attribute values for each seed in E_{maj}. The values are chosen to reflect the distribution of values contained in the original training attribute with respect to the particular class. This is achieved by considering, for each class, the number of occurrences of different attribute values in the original data set.

For ***Continuous*** attribute, the data generation produces a total of N_{maj} attribute values. The values are chosen by considering the range *[min, max]* of the original attribute values with respect to the seed class. Also, the distribution of original attribute values, in terms of the deviation and the mean, is used during data generation.

Similarly, M_s different sets of new minority class examples, each based on a seed instance in E_{min}, are constructed. These sets of instances are added to the original training set. Interested readers are referred to [15, 16, 17] for a description of the data generation process and its evolution.

3.3 Balance the Training Weights of Separate Classes

In the final step prior to re-training, the total weights of the examples in the different classes are rebalanced. By rebalancing the total weights of the different classes, boosting is forced to focus on hard as well as rare examples.

Recall that the data generation process generates sets of synthetic examples based on seed examples E_{maj} and E_{min} corresponding to the majority and minority classes. Before the generated data are added to the original data set, each of the synthetic examples is assigned an initial weight. The initial weight of each example is calculated by dividing the weight of the seed example by the number of instances generated from it. In this way, the very high weights associated with the hard examples are balanced out. Rebalancing ensures that the boosting algorithm focuses on hard as well as minority class examples.

When the new training set is formed, the total weights of the majority class examples (denoted by W_{maj}) and the minority class examples (denote by W_{min}) in the new training data are rebalanced as follows. If $W_{maj} > W_{min,}$ the weight of each instance in the *minority* class is multiplied by $W_{maj} / W_{min,}$ Otherwise, the weight of each instance in the *majority* class is multiplied by $W_{min} / W_{maj.}$ In this way, the total weight of the majority and minority classes will be balanced. Note that, prior to training, the weights of the new training set will be renormalized, following the AdaBoost method, so that their sum equals 1 [1, 2, 3].

4 Experiments

This section describes the results of evaluating the performance of the DataBoost-IM algorithm against imbalanced data sets, in comparison with the AdaBoost benchmarking boosting algorithm [1, 2] and as well as the C4.5 decision tree, which has become a de facto standard against which new algorithms are being judged [18]. The C4.5 algorithm is also used as base classifier.

Table 2. Summary of the data sets used in this paper. Shown are the number of examples in the data set; the number of minority class; the number of majority class; the class distribution; the number of continous, and the number of discrete input features

Data set	Case	Minority Class	Majority Class	Class Distribution	Feature Continuous	Feature Discrete
CREDIT-G	1000	300	700	0.30:0.70	7	13
BREAST-CANCER	286	85	201	0.30:0.70	0	9
PHONEME	5484	1586	3818	0.29:0.71	5	0
VEHICLE	846	199	647	0.23:0.77	18	0
HEPATITIS	155	32	123	0.20:0.80	6	13
SEGMENT	2310	330	1980	0.14:0.86	19	0
GLASS	214	29	185	0.13:0.87	9	0
SATIMAGE	6435	626	5809	0.09:0.91	33	0
VOWEL	990	90	900	0.09:0.91	10	3
SICK	3772	231	3541	0.06:0.94	6	23
PRIMARY-TUMOR	339	14	325	0.04:0.96	0	17

To evaluate the performance of the DataBoost-IM method, we obtained eleven data sets from the UCI data repository [11]. These data sets were carefully selected to ensure that they (a) are based on real-world problems, (b) varied in feature characteristics, and (c) vary extensively in size and class distribution. Table 2 gives the characteristics of the data sets used for the experiments. Shown are the number of cases, the number of the majority and minority classes, the class distribution, and the type of the features. For the glass, vowel, vehicle, satimage and primary-tumor data sets, we increased the degree of skew by converting all but the smallest class into a single class. For the sick data sets, we deleted the 'TBG' attribute due to the high occurrence of missing values.

4.1 Methodology and Experimental Results

We implemented the experiments using Weka [19], a Java-based knowledge learning and analysis environment developed at the University of Waikato in New Zealand. Results for the data sets, as shown in Table 2, are averaged over five standard 10-fold cross validation experiments. For each 10-fold cross validation the data set was first partitioned into 10 equal sized sets and each set was then in turn used as the test set while the classifier trains on the other nine sets. A stratified sampling technique was applied here to ensure that each of the sets had the same proportion of different classes. For each fold an ensemble of *ten* component classifiers was created. In the experiments, the C4.5 decision trees were pruned [18].

Table 3. Test set G-mean, F-measure of minority class, F-measure of majority class, overall accuracy , true positive rate of minority class, and true positive rate of majority class for the data sets using (1) C4.5, (2) AdaBoost ensembles, and (3) DataBoost-IM ensembles

Data Set Name	Methods	G-Mean	F-measure of min. class	F-measure of maj. class	Overall Accuracy	TP rate of min. class	TP rate of Maj. Class
Credit-g	C4.5	56.72	43.88	80.53	71.10	37.66	85.42
	AdaBoost	58.91	45.74	78.69	69.40	43.00	80.71
	DataBoost-IM	61.96	49.64	80.22	71.60	46.66	82.28
Breast-cancer	C4.5	50.84	39.31	84.39	75.17	27.05	95.52
	AdaBoost	58.12	44.06	74.93	65.38	45.88	73.63
	DataBoost-IM	60.04	46.51	77.00	67.83	47.05	76.61
Phoneme	C4.5	84.22	77.23	90.07	86.17	79.88	88.78
	AdaBoost	86.74	81.86	92.60	89.48	80.83	93.08
	DataBoost-IM	88.40	83.83	93.29	90.52	83.73	93.34
Vehicle	C4.5	92.50	87.90	96.19	94.20	89.44	95.67
	AdaBoost	95.58	92.57	97.67	96.45	93.96	97.21
	DataBoost-IM	95.77	93.70	98.06	97.04	93.46	98.14
Hepatitis	C4.5	57.91	42.10	86.95	78.70	37.50	89.43
	AdaBoost	66.86	52.45	88.35	81.29	50.00	89.43
	DataBoost-IM	76.25	62.68	89.71	83.87	65.62	88.61
Segment	C4.5	92.28	87.23	97.87	96.36	86.96	97.92
	AdaBoost	95.98	93.59	98.94	98.18	93.03	99.04
	DataBoost-IM	97.35	95.59	99.26	98.74	95.45	99.29
Glass	C4.5	85.91	78.57	96.77	94.39	75.86	97.29
	AdaBoost	89.48	81.35	97.01	94.85	82.75	96.75
	DataBoost-IM	92.34	89.28	98.38	97.19	86.20	98.91
Satimage	C4.5	72.70	56.44	95.41	91.70	55.27	95.62
	AdaBoost	77.01	66.72	96.76	94.11	60.70	97.71
	DataBoost-IM	80.42	68.86	96.76	94.14	66.61	97.10
Vowel	C4.5	95.81	93.78	99.38	98.88	92.22	99.55
	AdaBoost	97.69	97.17	99.72	99.49	95.55	99.88
	DataBoost-IM	99.38	98.88	99.88	99.79	98.88	99.88
Sick	C4.5	93.03	89.13	99.30	98.70	87.01	99.46
	AdaBoost	94.23	91.15	99.43	98.93	89.17	99.57
	DataBoost-IM	95.96	91.84	99.46	98.99	92.64	99.40
Primary-tumor	C4.5	0.00	0.00	97.89	95.87	0.00	100.0
	AdaBoost	37.50	19.04	97.41	94.98	14.28	98.46
	DataBoost-IM	52.62	28.57	96.92	94.10	28.57	96.92

The experimental results for all eleven data sets described in Table 2 are presented in Table 3. For each data set, we present the results achieved when using the C4.5, AdaBoostM1 and DataBoost-IM methods. Also, for each algorithm, the table presents the results in terms of the *G-mean*, *overall accuracy rates, TP rate* of the majority class and the minority class, *F-measure* of the majority class and the minority class.

One conclusion drawn from the experimental results is that the DataBoost-IM method consistently achieves higher performance on the minority class as well as the majority class in comparison with the C4.5 and AdaBoost algorithms. However, this improvement varies with different data sets. Consider the *G-mean* scores, which reflect the classifier's predictive accuracies against the majority and the minority class, as well as the degree of balance between classes. The results show that, for the *G-mean* score, the DataBoost-IM method surpasses the performance of the C4.5 and the AdaBoost algorithms in all cases. The experimental results also show that in many cases, the improvements are large. For example, in the Hepatitis data set, the AdaBoost algorithm improves on the C4.5 method's *G-mean*, i.e. 57.91% compared to 66.86%, and the DataBoost-IM algorithm significantly improved on the AdaBoost algorithm's performance, with a *G-mean* of 76.25%. In the case of the Primary-

Tumor data sets, the *G-mean* for the C4.5 method is 0%, since the C4.5 classifier achieved a 100% accuracy rate for the majority class but misclassified all minority examples. In this case, the AdaBoost algorithm produced a *G-mean* of 37.50%, whilst the DataBoost-IM ensemble achieved a *G-mean* of 52.62%. This improvement was credited with the achievement of the minority class's *TP Rate* of 28.57% by the DataBoost-IM compared to 0% by the C4.5 classifier and 14.28% by the AdaBoostM1 method. When considering the *F-measures* a similar conclusion holds. Table 3 moreover shows that, when considering the overall accuracy, the DataBoost-IM method consistently produces highly accurate ensembles.

In conclusion, the results, as shown in Table 3, indicate that the DataBoost-IM approach described here extends the predictive capabilities of the boosting ensemble and the component classifier when evaluated in terms of *overall accuracy*, *G-mean* and *F-measures.* This is achieved through integrating the data generation approach, in which class frequencies and training weights on different classes are rebalanced, into the boosting procedures.

5 Conclusion

This paper introduced a technique to create an ensemble of highly accuracy classifiers, when learning from imbalanced data sets. The DataBoost-IM algorithm extends the standard boosting approach by generating additional synthetic examples for both the majority and the minority classes, balancing the class distribution and rebalancing the training weights on different classes. In this way, boosting focuses not only on hard examples, but also on rare minority class examples. The DataBoost-IM algorithm was illustrated by means of eleven UCI data sets with various features, degrees of imbalance and sizes. The results obtained indicate that the DataBoost-IM approach increases the performance power of boosting algorithms when applied to imbalanced data sets. In particular, the DataBoost-IM algorithm achieved better predictions, in terms of the *G-mean* and *F-measures metrics*, against both the minority and majority classes, when compared with the C4.5 and AdaBoostM1 algorithms. Importantly, our method does not sacrifice one class for the other, but produce high predictive accuracy against both the majority and the minority class.

It follows that our approach should address two issues, namely finding optimal way to re-balance the learning bias toward majority class and investigating the performance against noisy data**.** Also, other weight-assignment methods will be further investigated. Future work will also include studying the voting mechanism of the boosting algorithm using different metrics such as the *ROC* curve and to provide a sound theoretical justification of the Databoost-IM parameter selection process.

References

1. Y. Freund and R. Schapire (1996): Experiments with a new boosting algorithm. the Proceedings of the Thirteenth International Conference on Machine Learning, Bari, Italy, 148-156.
2. L. Breiman (1996): Bias, Variance and Arcing Classifiers. Technical report 460, University of California: Berkeley, Berkeley, CA: USA.

3. H. Schwenk and Y. Bengio (1997): AdaBoosting Neural Networks: Application to On-line Character Recognition, International Conference on Artificial Neural Networks (ICANN'97), Springer-Verlag, 969-972.
4. T.G. Dietterich (2000): An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine Learning 40, 139-157.
5. N. Japkowicz (2000): Learning from imbalanced data sets: A comparison of various strategies, Learning from imbalanced data sets: The AAAI Workshop 10-15. Menlo Park, CA: AAAI Press. Technical Report WS-00-05.
6. N. Chawla, K. Bowyer, L. Hall and W. Kegelmeyer (2002): SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321-357.
7. M.A. Maloof (2003): Learning when Data Sets are Imbalanced and when Costs are Unequal and unknown, ICML-2003 Workshop on Learning from Imbalanced Data Sets II.
8. M. Kubat and S. Matwin (1997): Addressing the curse of imbalanced training sets: One-sided selection. Proceedings of the Fourteenth International Conference on Machine Learning San Francisco, CA, Morgan Kaufmann, 179-186.
9. M. Joshi, V. Kumar and R. Agarwal (2001): Evaluating boosting algorithms to classify rare classes: comparison and improvements. Technical Report RC-22147, IBM Research Division
10. N. Chawla, A. Lazarevic, L. Hall and K. Bowyer (2003): SMOTEBoost: improving prediction of the minority class in boosting. 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia , 107-119.
11. C.L. Blake and C. J. Merz (1998): UCI Repository of Machine Learning Databases [http://www.ics.uci.edu/~mlearn/MLRepository.html], Department of Information and Computer Science, University of California, Irvine, CA.
12. F. Provost and T. Fawcett (1997): Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In proceedings of the Third international conference on Knowledge discovery and data mining, Menlo park, CS. AAAI Press, 43-48.
13. M. Kubat, R. Holte and S. Matwin (1998): Machine Learning for the Detection of Oil Spills in Satellite Radar Images. Machine Learning, 30, 195–215.
14. C. J. van Rijsbergen (1979): Information Retrieval. Butterworths, London.
15. H Guo and HL Viktor (2004): Boosting with data generation: Improving the Classification of Hard to Learn Examples, the 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE), Ottawa, Canada, May 17-20, 2004.
16. HL Viktor (1999), The CILT multi-agent learning system, South African Computer Journal (SACJ), 24, 171-181.
17. HL Viktor and I Skrypnik (2001) : Improving the Competency of Ensembles of Classifiers through Data Generation, ICANNGA'2001, Prague: Czech Republic, April 21-25, 59-62.
18. JR Quinlan, C4.5 (1994): Programs for Machine Learning, Morgan Kaufmann, California: USA.
19. I. Witten, E.Frank (2000): Data Mining: Practical Machine Learning tools and Techniques with Java Implementations, Chapter 8, Morgan Kaufmann Publishers.

Automatic Design of Cascaded Classifiers

Etienne Grossmann*

Center for Visualization and Virtual Environments, University of Kentucky

Abstract. Cascades of boosted classifiers have become increasingly popular in machine vision and have generated a lot of recent research. Most of it has focused on modifying the underlying Adaboost method and far less attention has been given to the problem of dimensioning the cascade, i.e. determining the number and the characteristics of the boosted classifiers. To a large extent, the designer of a cascade must set the parameters in the cascade using ad-hoc methods.
We propose to automatically build a cascade of classifiers, given just a family of weak classifiers a desired performance level and little more. First, a boosted classifier with the desired performance is built using any boosting method. This classifier is then "sliced" using dynamic programming into a cascade of classifiers in a nearly computation-cost-optimal fashion.

1 Introduction

Boosting, [5, 17] combines the output of many weak classifiers - ones that perform slightly better than guessing [11]. By increasing the number of weak classifiers, arbitrarily small error rates can be reached on a training set. For example, one can achieve very good face recognition by combining hundreds of very simple classifiers [21]. However, the computational complexity increases linearly with the number of weak classifiers. If the target application has few true positives, as is the case in face detection, running such a big boosted detector would be an overkill, since, as shown by Viola and Jones [21], most true negatives can be rejected easily, using a small boosted detector. Their approach was thus to build a cascade of increasingly discriminative detectors, resulting in a face detector with performance similar or better to that of previous approaches [16, 18] while being computationally much less demanding.

Since then, cascades of classifiers have become increasingly popular [9, 10, 6]. Moreover, the methodology in [21] is not specific to face detection and was effectively applied to other cases [3]. The designer of a cascade does not need an advanced task-specific classifiers, although doing so can improve the overall performance of the system.

Many variants of [21] of have been developed. In most, the research effort focuses on employing better boosting methods. Viola and Jones [20] adapt Adaboost to the case of different misclassification costs for false positive and false

* This work was funded by the Kentucky Office of New Economy.

Algorithm 1 Cascaded classifier.

```
Given: Weak classifiers $h_{\bullet}\left(\right),\ldots,h_T\left(\right)$ and weights provided by a boosted classifier
    $H_B\left(X\right)$, cascade schedule $1 = T_{\bullet} < T_{\bullet} < \ldots < T_L = T + 1$ and thresholds
    $\theta_{\bullet},\ldots,\theta_L$.
Input: Vector $X$
    Initialization: Set $l = 1, \eta = 0$
    While $l \leq L$
        1. Set $\eta = \eta + \alpha_{T_{l-1}} h_{T_{l-1}}\left(X\right) + \ldots + \alpha_{T_l-\bullet} h_{T_l-\bullet}\left(X\right)$
        2. If $\eta < \theta_l$, classify $X$ as a negative, i.e. $H(X) = -1$. Exit.
        3. Set $l = l + 1$.
    Classify $H\left(X\right) = \text{sign}\left(\eta\right)$.
```

negatives. Li et al [8] modify Adaboost to obtain better performance using less weak classifiers, resulting in better performance on the training set and lower run-time complexity, but increased training time. Wu et al [6] and McCane and Novins [12] reduce drastically the training time[1] at the cost of slightly reduced performance. Little science, in contrast, has been applied to the characteristics of the boosted classifiers in a principled manner : empirical guidelines are given in [21] and by Lienhart et al [10] to dimension the intermediate classifiers of the cascade. McCane and Novins [12] propose a method to do this automatically and we will discuss their method further in this article. The present article proposes another such method.

We argue that choosing the characteristics of the classifiers in the cascade - whether it be number of weak classifiers or error rates - is the fundamental problem. After all, the main difference between a cascade and a single big boosted algorithm is precisely this architecture, which allows to reject true negatives early and thus reduce the computation load.

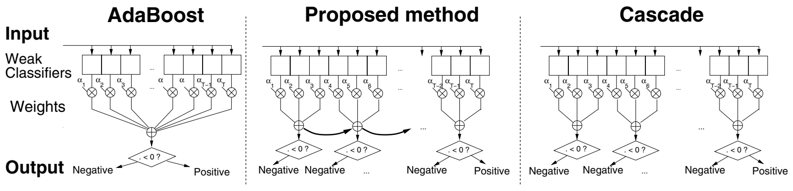
In the present work, we propose to build a cascade from a single big boosted classifier. The -very simple- idea is to compute a subset of weak classifiers of the big boosted classifier and test whether the input is positive or negative. If it is positive, compute another subset of weak classifiers and test again. We continue this way until the input is rejected or the complete boosted classifier has been computed. The resulting classifier algorithm is shown in Algorithm 1.

Some properties of the proposed method are easily seen. Most importantly, the set of positive inputs of the proposed classifier $H\left(X\right)$ is included in that of the boosted classifier $H_B\left(X\right)$ on which it is based. The true positive ratio of $H\left(X\right)$ is thus smaller than that of $H_B\left(X\right)$ and its true negatives is higher. The point in "ROC space" corresponding to $H\left(X\right)$ is thus below and on the left of the point corresponding to $H_B\left(X\right)$.

This algorithm shows a clear resemblance both with Adaboost, for the choice of weights and with the cascaded classifiers cited above. Figure 1 compares graphically the three architectures considered so far. In Adaboost (a), all weak classi-

• Viola and Jones mention weeks of training time.

fiers are computed at once, whereas in the proposed method and in the cascade of [21], only a subset of the weak classifiers is computed between each test. The difference between the proposed method (b) and [21] (c) is that in the former, the output the previous classifiers is kept to contribute to the input of the next. As a result, the output of the last classifier in (b) is exactly the same (assuming the weights and weak classifiers are the same, which is the case) as that of the boosted algorithm (a).


AdaBoost
Proposed method
Cascade
Input
Weak Classifiers
Weights
Output
α_1 α_2 α_3 α α_{T-1} α_T
$\cdot < 0?$
Negative
Positive
Negative Negative ... Negative Positive
Negative Negative ... Negative Positive

(a) (b) (c)

Fig. 1. Comparison of classifier architectures. The difference between adaboost (a) and (b,c) is that in the later, negative classification can be achieved after just a few weak classifier evaluations. The difference between the proposed method (b) and the cascade of [21] (c) is that the output of previous stages of decision is taken into account at each decision (bold arrows in (b)).

Going back to Algorithm 1, one sees that the output of $H(X)$ is defined by the weak classifiers $h_1(X), \ldots, h_T(X)$ and weights $\alpha_1, \ldots, \alpha_T$ that define the boosted classifier $H_B(X)$, but also by the schedule $T_1, \ldots, T_L$ and thresholds $\theta_1, \ldots, \theta_L$ of the tests. The choice of the T_l and θ_l is determinant to obtain a computationally efficient classifier and at first sight, this it may appear that setting these parameters is as hard as setting the parameters for a cascade of classifiers. However we will see that, using dynamic programming, it is possible to set these parameters in a way that:

a) Preserves the output of the boosted classifier on any given data set, e.g. on the training set of the boosted classifier.

b) Nearly optimal in terms of the computational cost of the classifier, amongst all cascades that verify **a)**.

Another benefit of the proposed method is that it is easier to choose the true positive and negative rates of the detector. Indeed, in a boosted classifier, there is a unique threshold that determines the point on the ROC curve on which the classifier lies. It is sufficient to set this threshold prior to building the cascade to guarantee that the cascade will achieve this point of the ROC curve (for the given dataset). Since the computation cost of building the cascade is negligible with respect to that of boosting, it is easy to build classifiers for any point on the ROC curve achievable by the original boosted algorithm. In contrast, previous cascading methods would either require to train a new cascade for each desired

Algorithm 2 Cascade building procedure.

Assume given

- A family of weak classifiers $h(X)$ a dataset $X_{\bullet}, \ldots, X_N$ and $y_{\bullet}, \ldots y_N$.
- Target false positive and true positive ratios for the classifier.
- An estimate of the proportion $P_{\bullet}$ of positives in the targeted real-world input.
- An estimate of the computational cost of the weak classifiers and of performing a test on the targeted computer architecture.

The cascade is obtained by:

Boost the weak classifiers and build its ROC until the targeted performance or better [14] is attained.
Build a computationally-optimal cascade with the architecture of Alg. 1 that has exactly the same output as the boosted classifier on the training data.

ROC point, or use an ad-hoc method to adjust the thresholds of the boosted classifiers.

Also, the proposed cascading method is not limited to a particular boosting method. It is possible to use a boosted classifier specialized for a given cost metric [4, 20, 13] or aimed at being computationally more efficient [8]. In the present work, we will use the most studied Adaboost method [17, 11].

The proposed procedure for building a cascade of classifiers is described in Algorithm 2.

Note that we assume the proportion of true positives P_0 in the targeted real-world application is approximately known. This proportion is usually not known precisely, but one knows that positives are a small minority - without this assumption, the whole cascade architecture stops making sense. We will see below that the exact value of P_0 does not influence greatly the resulting cascade.

Also, it is assumed that the computational cost of the weak classifiers is known, as well as the computational cost of performing the test in Algorithm 1. These quantities can be determined experimentally by benchmarking the targeted computer system. We do not assume that the cost of a test is negligible, as the cost of the weak classifiers may itself be very low.

We model explicitly the computational complexity of the classifier and determine a nearly optimal cascade amongst all cascades derived from a given boosted classifier. Although there exists work that considers the cost of evaluating a classifying tree [19], we are only aware of McCane and Novins who adopt [12] a similar approach to build a computationally near-optimal cascade of detectors. However, [12] assumes an approximate model for the computational cost of a boosted classifier as a function of its false positive rate. Also, they need to consider every possible (plausible) cascade length separately and determine, by numerical optimization, sub-optimal parameters for each. In contrast, we obtain the nearly-optimal sequence using very few assumptions and with very little computation, by using dynamic programming.

2 Computationally Optimal Cascade Design

We assume having a boosted classifier that achieves desired true positive and true negative rates on a given dataset. This can be done boosting a family of weak classifiers. until the ROC curve of the classifier passes above the desired true positive and negative ratios.

At this point, we have a boosted classifier $H_B(X)$ defined by weak classifiers $h_1(X)$,..., $h_T(X)$ and weights $\alpha_1, \ldots, \alpha_T$:

$$H_B(X) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \alpha_t h_t(X) > 0 \\ -1 & \text{otherwise.} \end{cases}$$

Additionally, we are given a dataset $X_1, \ldots, X_N$, with known classes $y_1, \ldots, y_N$ ($y_n \in \{-1, 1\}$). Define I^+ (resp. I^-) the set of indices n such that $H_B(X_n) = 1$ (resp. -1).

2.1 Choice of Thresholds $\theta_1, \ldots, \theta_L$

Consider an algorithm as in Alg. 1, characterized by the test schedule $T_1, \ldots, T_L$ and thresholds $\theta_1, \ldots, \theta_L$. Define the intermediate real-valued classifiers:

$$G_t(X) = \sum_{s=1}^{t} \alpha_s h_s(X) \tag{1}$$

Define thresholds θ'_t for each $1 \le t \le T$ so that $G_t(X_n) > \theta'_t$ for all the positive examples $n \in I^+$:

$$\theta'_t > \min\left\{G_t(X_n) \mid n \in I^+\right\}. \tag{2}$$

In practice, we may usually set θ'_t to the midpoint between the value above and the smallest $G_t(X_n)$ that is greater than Eq. (2) and $n \in I^-$.

We will set

$$\theta_l = \theta'_{T_l}. \tag{3}$$

It is clear that, with this choice, and independently of the $T_1, \ldots, T_L$, the cascade defined in Algorithm 1 has the *same output* as the original boosted algorithm on the training data.

2.2 Computational Cost Model

The proportion of true positives and false positives that are accepted at level[2] t is estimated by:

$$\begin{aligned} p_t &= \#\{n \mid y_n = 1 \,\text{and}\, G_t(X_n) > \theta'_t\} / \#\{n \mid y_n = 1\} \\ r_t &= \#\{n \mid y_n = -1 \,\text{and}\, G_t(X_n) > \theta'_t\} / \#\{n \mid y_n = -1\}. \end{aligned}$$

[2] We will call levels $1 \le t \le T$ the indices of the weak classifiers that constitute the boosted classifier.

where # denotes the cardinal. Assuming that the proportion of positives in the real-world input is P_0, the proportion of real-world inputs that are accepted at level t is approximately:

$$q_t = P_0 p_t + (1 - P_0)\, r_t. \tag{4}$$

We assume in the following that q is decreasing in t. Although this is not necessarily true in the training data, the true ratio can be expected to be monotonously decreasing.

Now, considering any cascade of the type of Algorithm 1, the expected computation cost for a real-world input is approximately:

$$C = (\mathcal{A}_1 + B) + (\mathcal{A}_2 + B)\, q_{T_1} + \ldots + (\mathcal{A}_L + B)\, q_{T_L}, \tag{5}$$

where

$$\mathcal{A}_l = \sum_{T_{l-1} \le t < T_l} A_t,$$

A_t is the computational cost of the t^{th} weak classifier and B is the cost of performing a test on the targeted computer.

2.3 Optimal Test Schedule $T_1, \ldots, T_L$

We now show that the optimal computational cost is obtained efficiently using dynamic programming [2], because the cost C_t of the optimal cascade starting at level $t \in \{1, \ldots, T\}$ is can be defined recursively from the costs $C_{s>t}$.

The optimal cascade starting at t is necessarily one of the following: (1) the trivial cascade consisting in computing all the remaining classifiers and testing the result; (2) the cascade consisting in computing classifiers $t, \ldots, T-1$ performing the test at $T-1$ and following the optimal sequence from $T-1$ to T; ... ; $(T-t+1)$ computing the t^{th} weak classifier, testing and following the optimal sequence from $t+1$ to T. The costs of each possibility is:

$$\begin{array}{ll} C_{t,T} & = q_t\,(A_t + \ldots + A_{T-1} + A_T) + 0 \\ C_{t,T-1} & = q_t\,(A_t + \ldots + A_{T-1}) + C_T \\ \vdots & \quad \vdots \\ C_{t,t} & = q_t A_t + C_{t+1} \end{array} \tag{6}$$

The optimal cascade cost and optimal "jump" at t are thus:

$$C_t = \min\{C_{t,s} \mid t < s \le T\}, \; S_t = 1 + \arg_s \min\{C_{t,s} \mid t < s \le T\}. \tag{7}$$

From there, the optimal cascade sequence is:

$$T_0 = 1, T_1 = S_1, T_2 = S_{T_1}, \ldots, T_t = S_{T_{t-1}}, \ldots, T_L = T + 1. \tag{8}$$

Note that it is not necessary to specify the length of the optimal sequence. Also, the resulting cascade does not depend on the proportion of true and false positives in the training data, but on the expected proportion P_0 of positives in the real-world data.

The dependence on P_0 is itself quite mild : the terms in Eq. (6), can be developed as affine expressions of P_0 with coefficients depending on p_t, r_t, A_t and B in which the constant term is proportional to the false positive rate r_t. Near the end of any cascade, the cost is thus, nearly linear in P_0 and the optimal cascade does not depend on P_0. Even earlier in the cascade, when r_t is greater, we have found that the optimal sequence does not vary greatly with P_0.

Finally, on should note that the obtained scheduling is optimal only if the series q_t defined in Eq. (4) is decreasing, which is not necessarily the case for a given dataset, although this is the expected behaviour of the boosted classifier. We found that in practice, these series are not monotonous, and that some smoothing could be applied[3]. However, since the general tendency of the unsmoothed series is monotonous, we consider that the cost of the resulting sequence is near the true minimum.

3 Experimentation

This section, presents experimental validation for the theory developed above. For this purpose, we use a face classifier still in development, which uses the same Haar filters as [21]. The training dataset consists in 2000 face images taken from the BioID database [7], from dataset C of the CMU database [15] and from images gathered on the internet. All face and non-face images were scaled to 18-by-31 pixels and were normalized in variance and range. Images from the BioID database were cropped repeatedly at slightly different positions and scales. A validation dataset of 5000 other images is built in the same way. On our computer system, it was found that the cost A of weak classifiers is approximately 40 times that of performing a test and this value is used in the sequel.

Comparison of performance of boosted classifier and cascades. In this experiment, a boosted classifier[4] consiting of $T = 100$ weak classifiers is built using the training dataset and its ROC is computed using the validation dataset. Only points on the upper convex hull of the ROC curve are kept. For each non-trivial vertex of the ROC, the corresponding cascaded classifier is built, using the training dataset and assuming $P_0 = 0.001$. The cascades have from 29 to 37 levels (average: 34.1).

Figure 2 (left) shows part of the ROC curve of the boosted classifier (topmost curve) the curve linking linking the performances of the cascaded classifiers obtained from the boosted classifier with the thresholds of the ROC vertices. The lower dotted ROC is that of the boosted classifier truncated at the 34th weak classifier, having thus similar computation cost as the cascades. This experiment shows the performance of the cascade is very close to that of the underlying boosted classifier, while decreasing the average computation cost very much.

• We do not use any in the experimental section, unless otherwise specified.

• We use an unpublished boosting method inspired of [1] and [20] adapted to unequal importances of false positives and false negatives.

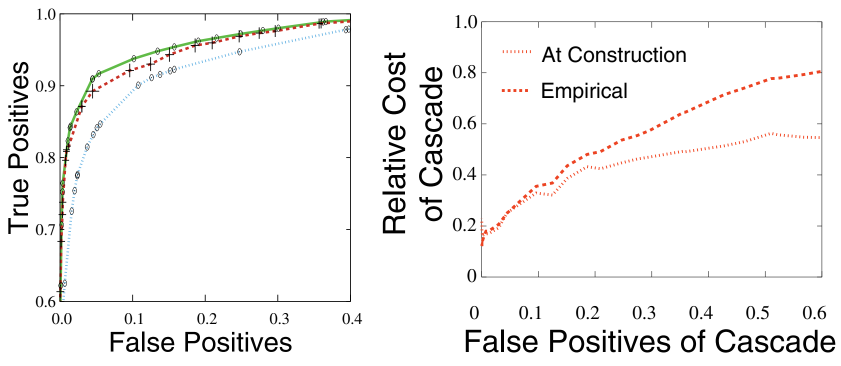


Fig. 2. Left: Detail of ROC of boosted (plain line), cascaded classifiers (dashed) and boosted classifier with same cost as cascades (dotted). **Right:** Relative computational cost of cascade w.r.t. original classifier v.s. false positive rates.

Figure 2 (right) shows the computational cost of the cascade, assuming that $P_0 = 0.001$ divided by that of the original boosted classifier. The dotted curve is the expected value determined while constructing the cascade, while the the dashed curve above is the value observed on the validation dataset.

Since P_0 is very low, it is natural to set the classifier with a very low false positive rate, to avoid being overwhelmed by false positives. Figure 2 (right) shows that in this condition, the empirical and theoretical compuation costs are very similar and that they are at their lowest.

Architecture of cascade with varying P_0. In this experiment, the cascade is built for $P_0 \in \left\{1/10^{1+i/2} \mid 0 \leq i \leq 6\right\}$ with a fixed boosted classifier ($T = 100$) and input data. This experiment showed that the schedule is the same for nearly all cascades, with those with higher levels of P_0 skipping some tests. This confirms our claim that the value of P_0 is not determinant in the resulting cascade.

4 Discussion and Conclusions

We have proposed a method to automatically design a cascade classifier with a desired performance, targeted for a given type of input and a given computer architecture. This methodology applies to any underlying boosting method, whether the best-known algorithm of Schapire and Singer [17] or, as was the case in the experimental section, a boosting method targeted at a given region of the ROC space. Since the resulting cascade is so closely related to its underlying boosted classifier, we can expect that the theoretical properties of this cascade will be easier to study than that of ad-hoc cascades and that this would be a step towards being able to set the generalization bounds of the cascade, rather than its performance on a given dataset.

Future research is required to find ways of setting the thresholds of the cascade levels so as to follow the ROC of the original classifier better still than was shown in the experimental section. Another direction of research is to better estimate the proportion q_l of examples that reach a level l of the cascade.

References

1. J. A. Aslam. Improving algorithms for boosting. In *Annual Conf. on Computational Learning Theory*, pages 200–207. Morgan Kaufmann, San Francisco, 2000.
2. R. Bellman. *Dynamic programming.* Princeton University Press, 1957.
3. O. Carmichael and M. Hebert. Object recognition by a cascade of edge probes. In *BMVC*, pages 103–112, 2002.
4. W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan. AdaCost: misclassification cost-sensitive boosting. In *Intl. Conf. on Machine Learning*, pages 97–105, 1999.
5. Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *proc. International Conference on Machine Learning*, pages 148–156, 1996.
6. M. D. Mullin J. Wu, J. M. Rehg. Learning a rare event detection cascade by direct feature selection. In *NIPS*, 2003.
7. O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz. Robust face detection using the hausdorff distance. In *Proc. Intl. Conf. on Audio- and Video-based Biometric Person Authentication*, pages 90–95, 2001.
8. S. Li, Z. Zhang, L. Zhu, H.-Y. Shum, and H. Zhang. Floatboost learning for classification. In *NIPS*, 2003.
9. S.Z. Li, L. Zhu, Z.Q. Zhang, A. Blake, H.J. Zhang, and H. Shum. Statistical learning of multi-view face detection. In *ECCV*, 2002.
10. R. Lienhart, A. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM Pattern Recognition Symposium*, 2003.
11. S. Mannor and R. Meir. Geometric bounds for generalization in boosting. In *Proc. Conference on Computational Learning Theory*, 2001.
12. B. McCane and K. Novins. On training cascade face detectors. In *Image and Vision Computing New Zealand*, 2003.
13. S. Merler, C. Furlanello, B. Larcher, and A. Sboner. Automatic model selection in cost-sensitive boosting. *Information Fusion*, 4(1):3–10, 2003.
14. F. J. Provost and T. Fawcett. Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions. In *International Conference on Knowledge Discovery and Data Mining*, 1997.
15. H. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. Technical Report CMU-CS-97-201, CMU, 1997.
16. H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Trans. PAMI*, 20:22–38, 1998.
17. R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
18. H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proc. ICCV*, 2000.
19. P. D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409, 1995.
20. P. Viola and M. Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In *NIPS*, 2001.
21. P. Viola and M. Jones. Robust real-time object detection. In *Proc. ICCV workshop on statistical and computational theories of vision*, 2001.

Recognition of Handwritten Numerals Using a Combined Classifier with Hybrid Features

Kyoung Min Kim[1,4], Joong Jo Park[2], Young Gi Song[3], In Cheol Kim[1], and Ching Y. Suen[1]

[1] Centre for Pattern Recognition and Machine Intelligence (CENPARMI), Concordia University, 1455 de Maisonneuve Blvd. West, Suite GM606, Montreal, Canada H3G 1M8
{kkm,kiminc,suen}@cenparmi.concordia.ca

[2] Department of Control and Instrumentation Engineering, Gyeongsang National University, 900, Gazwa-dong, Chinju, Gyeongnam, 660-701, Korea

[3] Hyundai Information Technology Research Center, 1-8 San, Mabukri, Kusung Myun, Yongin City, Kyonggido 449-716, Korea

[4] Department of Electrical Engineering, Yosu National University, San 96-1, Dunduk-dong, Yeosu, Chonnam, 550-749, Korea

Abstract. Off-line handwritten numeral recognition is a very difficult task. It is hard to achieve high recognition results using a single set of features and a single classifier, since handwritten numerals contain many pattern variations which mostly depend upon individual writing styles. In this paper, we propose a recognition system using hybrid features and a combined classifier. To improve recognition rate, we select mutually beneficial features such as directional features, crossing point features and mesh features, and create three new hybrid feature sets from them. These feature sets hold the local and global characteristics of input numeral images. We also implement a combined classifier from three neural network classifiers to achieve a high recognition rate, using fuzzy integral for multiple network fusion. In order to verify the performance of the proposed recognition system, experiments with the unconstrained handwritten numeral database of Concordia University, Canada were performed, producing a recognition rate of 97.85%.

1 Introduction

As a typical example of complex pattern recognition system, totally unconstrained handwritten numeral recognition is a real challenge. It is difficult to achieve high recognition results using a single feature set and a single classifier, since this system contains many pattern variations which mostly depend upon individual writing styles. Several methods have been proposed and implemented in a number of different ways in the practical applications such as zip code recognition, document analysis and factory automation and accuracy of 95% and above have been reported [1][11]. To improve the recognition rate, current researchers aim at two major tasks, i.e. selection of good feature candidates and realization of universal classifiers. First, the selection of good feature candidates largely depends upon one's experience and their capabilities can only be evaluated by the recognition results. In this paper, we have selected feature extraction methods, which represent pattern variations effectively. Features that we extracted from input numerals include directional features, crossing point features, and zoning features. To improve the recognition rate, we combine mutually beneficial features and create three new feature sets by considering both the local and global characteristics of the input images. Second, the design of classifiers generally falls into three categories depending upon the feature vectors used, i.e., feature vector based classifier, syntactic/structural based classifier, and neural networks classifier [2][3][11]. But it is very difficult to achieve good performances with a single classifier, thus the current trend is to implement pattern multiple classifiers rather than a single classifier to achieve a highly reliable performance[4][11]. In this paper, we have used three neural networks as single classifiers, and used fuzzy integral to combine their outputs to obtain a higher recognition rate.

This paper is organized as follows. In section 2, we present a review of earlier works. Feature extraction methods are introduced in section 3. In section 4, we describe our classification methodology and fuzzy integral for the aggregation of output of the single neural networks. Recognition result of each single classifier and overall recognition results are included in section 5. Finally a brief summary and future study are discussed in the last section.

2 Review of Earlier Works

In this section, we review some representative methods for the recognition of totally unconstrained handwritten numerals. These approaches generally fall into two realms according to the attributes of the feature vectors. The first category includes techniques such as template matching, moments, characteristic loci and mathematical transforms which generally represent global characteristics of the input numerals. In the second category, efforts are aimed at extracting the shape characteristics of the input numerals from their skeletons or contour profiles. Such features include loops, endpoints, junctions, arcs, concavities and convexities that generally represent local characteristics of the input numerals.

Template feature extraction methods have the problem that they are very susceptible to small variations such as translation and rotation, thus the modified version such as affine transform is a good alternative [7]. Structural decomposition methods of feature extraction strive to represent structural properties of the input patterns. Series expansions are methods of representing a signal as a series of coefficients created by projecting the signal onto some basis, but Fourier descriptors can not be applied to fragmented characters, which usually happen in the handwritten numerals because this method extracts only one single closed contour. Furthermore the frequency information of Fourier transform is global, intuitively; a more localized frequency representation should be more effective such as wavelets transform [10]. Moments are invariant to size and rotation, and some moment invariants have invariant characteristics to skewed and mirrored images. Using Zernike moments, we can obtain size and rotation independent features. And KL transform has the best optimal rotation independent features. And KL transform has the best optimal information compactness in terms of mean square error, and is used for object recognition in several application domains, e.g., face recognition and the verification of slab number.

3 Feature Extraction

In the case of handwritten numerals, it seems natural to use some a priori knowledge about the recognition task in order to transform the low level information of the pixel image into a data representation at a higher level. A good feature should represent characteristics of a class ('0'-'9') that helps to distinguish it from other classes, while remaining invariant to pattern diversities within the class. Features also should avoid redundant information since this will lead to a more complex distribution in the feature space and therefore requires a more complex model. Before we extract feature vectors from the input image, we first extract generic data area in the input image by using projections on vertical and horizontal axes. After this segmentation, we undertake preprocessing steps, viz, median filtering for removing small holes inside black pixels which may cause problems especially when extracting crossing point feature and then we perform size normalization to reduce the effect of pattern variations. In this paper, three features are used. The first one consists of directional and global features. Line segments and their directions seem to be an adequate feature. For each location in the image, information about the presence of a line segment of a given direction (4-directions) is stored in a feature map. Among many first-order differential operators, the Kirsch edge detector has been known to detect four directional edges more accurately in comparison to other methods because the Kirsch edge detector considers all eight neighbors. Directional feature maps for horizontal (H), vertical (V), right-diagonal(R), and left-diagonal (L) directions are easily calculated by using Kirsch masks[5]. Some researchers used the four directional feature maps which are normalized to an 8×8 format [5], and others used the four directional feature maps normalized to 4×4 [4][6], and one global feature normalized to 4×4. In this paper, the size of 4×4 is used for directional and global feature maps. From normalized 16×16 input images, four directional images are obtained by Kirsch operation in which 10 is used as a threshold value. After that, these 16×16 directional images and input image are compressed to 4×4 feature maps by accumulating the pixels in each 4×4 subregion. This feature extraction process is depicted in Figure 1.

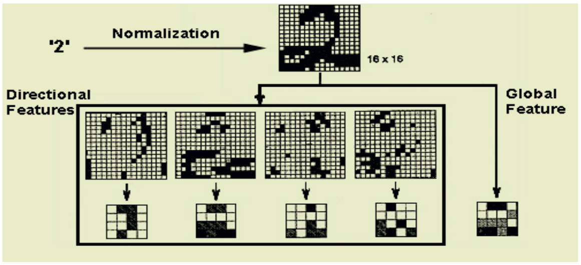


Fig. 1. Overview of feature extraction

The second feature is zoning feature that can be considered as a good candidate for global feature, since zoning feature is obtained by diminishing the image resolution, thus, small variations are easily expressed by the rate of scaling factor although it is susceptible to slant and rotation. In this paper, 10×10 are used as the size of zoning feature. While the first feature considers global and local characteristics of input numerals at the same time, zoning feature represents only global characteristics, which contain more detailed shape information. Finally, the third feature is a crossing point feature. Crossing point is defined as the point at which white pixel changes to black pixel at one scan line. And crossing point feature is obtained by summing the number of crossing points for each vertical and horizontal scan line. In this paper, from a normalized 20×20 input images, we obtain a 20-dimensional crossing point feature vector, 10-dimensions per axis. Each feature element of this feature vector is calculated as follows: one feature element is obtained by considering every two scan lines, i.e., one feature element is calculated by summing the number of crossing points on two adjacent scan lines and dividing it by the maximum number of occurrences in one scan line at each axis. We determine the maximum number of occurrences as 2 for a horizontal scan line and 4 for a vertical scan line. So, in our case, 4 and 8 are used respectively. The purpose of using two scan lines per feature element is to reduce the dimension of the feature vector and avoid confusions caused by shape variation of the numeral. Figure 2 shows this feature extraction process.

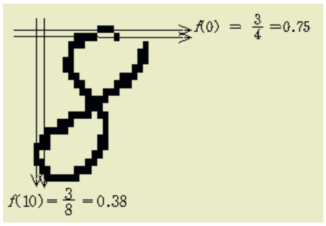


Fig. 2. Extraction of crossing point feature

By using the three features described above, we created three feature sets to improve the recognition rate; feature set 1(FS1) is 4×4 directional and global feature maps which is the first feature itself, feature set 2(FS2) is the combination of zoning feature and crossing point feature and feature set 3(FS3) is the combination of feature set 1 and crossing point feature. Each feature set is used as the input for each single classifier.

4 Multiple Classifications

In this paper, we combine three independent neural networks classifiers to recognize totally unconstrained handwritten numerals. Recently the concept of combining multiple networks has been actively studied for developing highly reliable neural networks system. One of the key issues of this approach is how to combine the results of the various networks to give the best estimate of the optimal result. The basic idea of the multiple network classifiers is to develop *n* independently trained classifiers with particular features, and to classify a given input pattern by obtaining a classification from each replica of the network and then using combination methods [8]. Each classifier uses an independent feature vector as input. The combiner acts as a final decision making processor for selecting the appropriate output class. There are two methods of combining multiple classifiers: loosely coupled and tightly coupled subnetworks. Ryu used tightly coupled subnetworks [2], based on the results of the intermediate layer as the inputs of the final classification stage. Loosely coupled subnetworks use final outputs of each subnetwork and they evaluate the overall classification results using only these values. There are several methods to achieve this goal, such as winner take all, majority voting, Dempster-shafer method and BKS (Behavior knowledge space) method etc. [9][11]. In this paper we use the concept of fuzzy integral to combine multiple classifiers. Fuzzy integral is a nonlinear functional that is defined with respect to a fuzzy measure, especially g_λ-fuzzy measure introduced by Sugeno and provides a useful way for aggregating information. To calculate fuzzy integral, we first select fuzzy measure satisfying boundness, monotonousness and continuity. Fuzzy measure can be determined in several ways. Sugeno introduced the g_λ-fuzzy measure satisfying the following property.

$$\mathrm{g}(\mathrm{A}\cup\mathrm{B})=\mathrm{g}(\mathrm{A})+\mathrm{g}(\mathrm{B})+\lambda\,\mathrm{g}(\mathrm{A})\mathrm{g}(\mathrm{B}) \tag{1}$$

For all $A, B \subset X, A \cap B = \Phi$ and for some $-1 < \lambda < \infty$.
Fuzzy integral is defined as follows; Let Y be a finite set and $h: Y \rightarrow [0,1]$ a fuzzy subset of Y. The fuzzy integral over Y of the function h with respect to a fuzzy measure g is defined by

$$h(x) \circ g(\bullet) = \max_{E\subseteq Y}\left[\min\left(\min_{y\in E} h(y), g(E)\right)\right] = \max_{a\in[0,1]}\left[\min\left(\alpha, g(F_a)\right)\right] \tag{2}$$

where $F_a = \{y \mid h(y) \geq \alpha\}$.

The calculation of the fuzzy integral when Y is a finite set is easily given. Let $Y = \{y_1, y_2, y_3, ..., y_n\}$ be a finite set and let $h: Y \rightarrow [0,1]$ be a function which is the output of each classifier. Suppose $h(y_1) \geq h(y_2) \geq \cdots \geq h(y_n)$, (if not, Y is rearranged so that the relation holds). Then fuzzy integral I, with respect to a fuzzy measure g over Y can be computed by

$$I = \max_{i=1}^{n}\left[\min(\ h(y_i),\ g(A_i))\right] \tag{3}$$

where $A_i = \{y_1, y_2, \cdots, y_n\}$.

Note that when g is a g_λ-fuzzy measure, the values of $g(A_i)$ can be determined recursively as

$$g(A_i) = g^i + g(A_{i-1}) + \lambda g^i g(A_{i-1}), \quad \text{for } 1 < i < n. \tag{4}$$

where λ is given by solving the following equation

$$\lambda + 1 = \prod_{i=1}^{n} (1 + \lambda g^{i}) \tag{5}$$

and $\lambda \in (-1, \infty)$ and $\lambda \neq 0$ and n is the number of subnetworks. This can be easily calculated by solving a $(n-1)$ th degree polynomial and finding the unique root greater than -1. The calculation of the fuzzy integral with respect to a g_{λ}-fuzzy measure would only require the knowledge of density function, where i-th density g^{i} is interpreted as the degree of importance of the source y_i toward the final evaluation. Fuzzy integral shows a good performance when the difference of outputs of subnetworks is minute because the value of g^{i} dramatically changes the performance compared to simple averaging or majority voting method. There is however, a key issue unsolved in the application of fuzzy integral, the determination of density values that determine the fuzzy measure used in the fusion process. In this paper, we determine these fixed densities from the knowledge of recognition rates of three neural networks, which also reflect the importance of each feature set they used.

5 Experimental Results

In this paper, we use three different feature sets to recognize totally unconstrained handwritten numerals. Aforementioned features have certain deficiencies in representing the input numerals, e.g. directional features can be confused when the input numeral is slanted or connected among pixels, and zoning features always lose their local characteristics. To effectively manipulate global and local characteristics of input numerals, we combine two or three mutually complementary features and make three new feature sets. This can make up for the weakness of one feature by the strength of other features. And the network fusion algorithm is used to improve the overall recognition results. The overall recognition system block diagram of our scheme is shown in Fig. 3 where NN1, NN2, and NN3 are feedforward neural network classifiers with one hidden layer, for which backpropagation learning algorithm is used.

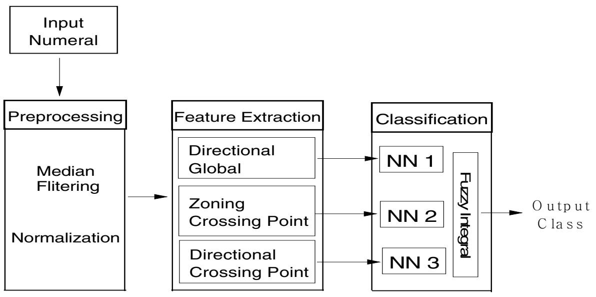


Fig. 3. The overall recognition system

5.1 Results from Single Classifiers

In this section, we use learning rate as 0.9, momentum as 0.7 for each neural network. The rejection criterion used in this paper is

$$RC = \frac{O_1 - O_2}{O_1 + O_2} \tag{6}$$

where O_1 is the highest output node value and O_2 is the second highest of a given input pattern. Rejection takes place if the RC falls below a threshold value that we set as 0.2. And the reliability in the table is computed according to the following equation:

$$Rel. = \frac{Rec.}{Rec. + Sub.} \times 100 \tag{7}$$

where *Rec.* = correct recognition rate, and *Sub.* = substitution error rate.

5.1.1 Feature Set 1

The first feature set (FS1) consists of directional features and global features. FS1 has been used in several studies by itself and in combination with other features. Directional features have robustness with small pattern variations. To extract directional feature, we first normalize the input image to 16×16 using bilinear interpolation. And we obtain the four directional feature maps normalized to a 4×4 and one global feature normalized to 4×4, in which we use the value 10 and 12 respectively for value normalization to the range [0,1]. In the training process, we used several different numbers of hidden layer nodes, and obtained satisfactory error rates (0.8%) when we used the number of hidden layer nodes as about 80% of the dimension of the input feature vector.

From the above recognition result, we see that feature set 1 has poor discriminative capability in number '8'. This is because of slanted and narrow inner spaces of the number '8'. And because we used only normalized 4×4 image as a global feature, this feature is inadequate to represent overall shape variations when distorted. But the overall recognition result is better than other relevant studies.

5.1.2 Feature Set 2

The second feature set (FS2) is composed of zoning features and crossing point features. The training result (0.7% error rate) is a little better than feature set 1. This feature set represents global characteristics very well, but it may lead to confusion among similar numerals. The recognition result of feature set 2 shows a little improvement in comparison with feature set 1. We can observe that this feature set shows lower recognition results for '2' and '3' than other numerals due to the weakness of the zoning feature.

5.1.3 Feature Set 3

To effectively manipulate the minute differences among input numerals, we make feature set 3 (FS3), which is composed of feature set 1, and crossing point feature which was derived in the same manner as used in the feature set 2. The training result (0.01%) is much better than those of previous two feature sets. From Table 1, we see that this feature set shows better recognition results than the previous two recognition results. This is mainly due to the FS3, which represents local and global characteristics of input numerals more effectively than the previous two individual feature sets. This recognition result alone has provided another possibility for practical use. But in numerals '2' and '3', there exist certain confusion in discerning one from the other. We will deal with an aggregation of three neural network outputs by using fuzzy integral to compensate for the weakness of a single classifier's limitations and improve the overall recognition result.

5.2 Results for Using Multiple Classifiers

As introduced in the previous section, the network fusion using fuzzy integral computes an overall recognition result by using fuzzy integral and the results of all the subnetworks. We use $g^1 = 0.31$, $g^2 = 0.32$, and $g^3 = 0.33$ as density values for NN1, NN2 and NN3 according to their recognition rates. The final recognition result using combined multiple classifiers is shown in Table 1. And some examples of misrecognized numerals are shown in Fig. 4.

Comparatively, when we use simple averaging method in which the used weight values are the same as the above density values and majority voting techniques as a network fusion algorithm, we obtained final recognition results of 97.4% and 96.5%, respectively. From these results, we see that network fusion using fuzzy integral outperforms other methods, but we also need to pay more attention to selecting proper feature sets for each subnetwork.

Table 1. Final recognition result using feature sets and fuzzy integral

Methods	Rec.	Sub.	Rej.	Rel.
Feature Set 1	95.15	4.10	0.75	95.88
Feature Set 2	95.65	3.90	0.45	96.09
Feature Set 3	96.95	3.00	0.05	96.97
Multiple Classifier (Fuzzy Integral)	97.85	2.15	0.0	97.85

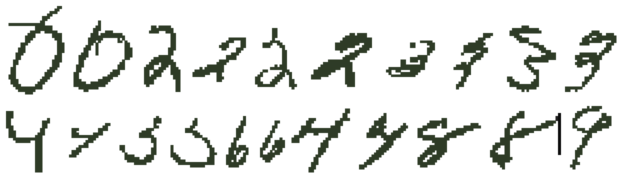


Fig. 4. Examples of misrecognized numerals

6 Conclusion

In this paper, we presented three feature sets, which consisted of several feature vectors to represent input numerals effectively. The recognition result shows that the proposed feature set effectively represents pattern variations such as slant, size and thickness etc. We used multilayer perceptron neural networks (MLP) as each single classifier, and used fuzzy integral to combine the outputs of three single classifiers. Combining of multiple classifiers using fuzzy integral considers the importance of input feature sets and a higher recognition rate was obtained in comparison with other proposed recognition systems. Further studies should be made to design classifiers which have more generalization capabilities and feature extraction methods which are mutually helpful for the recognition of unconstrained handwritten numerals.

References

1. Anil K. Jain and Torfinn Taxt: Feature Extraction Methods for Character Recognition -A Survey. Pattern Recognition 29(4) (1996) 641-662
2. Ryu Gang Soo, Chien Sung Il: Recognition of Handwritten Numerals by Relearning Hidden Layer Outputs of Modular Neural Nertworks. Korea Information Science Society 23(9) (1996) 931-939
3. Cho S. B.: Neural network classifiers for recognizing totally unconstrained handwritten numerals. IEEE Trans. on Neural Networks 8(1) (1997) 43-53
4. Mai T. A. and Suen C. Y.: A Generalized Knowledge Based System for The Recognition of Unconstrained Handwritten Numerals. IEEE Trans. on Systems, Man, and Cybernetics 20(4) (1980) 835-848
5. Knerr S. and Personnaz L. and Dreyfus G.: Handwritten Digit Recognition by Neural Networks with Singlelayer Training. IEEE Trans. Neural Networks 3(6) (1992) 962-968
6. Mohiuddin K. M. and Mao J.: A Comparative Study of Different Classifiers for Handprinted Character Recognition. Pattern Recognition in Practice IV (1994) 437-448
7. Wang J.: Resolving Multifont Character Confusion with Neural Networks. Pattern Recognition. 26(1) (1993) 175-187
8. Huang Y. S. and Suen C. Y.: A Method of Combining Multiple Experts for The Recognition of Unconstrained Handwritten Numerals. IEEE Trans. PAMI 17(1) (1995)
9. Cho S. B. and Kim J. H.: Multiple Network Fusion Using Fuzzy Logic IEEE Trans. Neural Networks 6(2) (1995) 487-501
10. Lee S. W. and Kim Y. J.: Multiresolution Recognition of Handwritten Numerals with Wavelet Transform and Multilayer Cluster Neural Network. 3rd international conference on document analysis and recognition (1995) 1010-1031
11. Suen Ching Y. et al.: Computer Recognition of Unconstrained Handwritten Numerals. Proceeding of the IEEE 80(7) (1992) 1162-1180

Generating Classifier Outputs with Fixed Diversity for Evaluating Voting Methods

Héla Zouari[1], Laurent Heutte[1], Yves Lecourtier[1], and Adel Alimi[2]

[1] Laboratoire Perception, Systèmes, Information (PSI), Université de Rouen
76821 Mont-Saint-Aignan, CEDEX, France
{Hela.Khoufi,Laurent.Heutte,Yves.Lecourtier}@univ-rouen.fr
http://www.univ-rouen.fr/psi
[1] Groupe de Recherche sur les Machines Intelligentes (REGIM), Université de Sfax
Ecole National des ingénieurs, BP W, Sfax, 3038, Tunisie
Adel.Alimi@ieee.org

Abstract. Recently, it has been shown that for majority voting combination methods, (negatively) dependent classifiers may provide better performance compared to that obtained with independent classifiers. The aim of this paper is to analyse the performance of plurality voting according to classifier diversity (agreement). This analysis is conducted in parallel with majority voting in order to show which method is more efficient with dependent classifiers. For this purpose, we develop a new method for the artificial generation of classifier outputs with fixed individual accuracies and pair-wise agreement. A diversity measure is applied for building the classifier teams. The experimental results show that the plurality voting is less sensitive to the correlation between classifiers than majority voting. It is also more efficient in achieving the trade-off between the recognition rate and rejection rate than the majority voting.

1 Introduction

Voting methods, in particular plurality voting and majority voting, are widely used in multiple classifier systems. They operate on the class labels assigned by the classifiers to each pattern. Plurality voting means that the class with the most votes is chosen. In majority voting, the class that receives more than half of votes is chosen. For these voting rules, the use of independent classifiers is considered to be essential to achieve better performance [5, 6, 12, 14]. However, in practice, it is difficult to obtain independent classifiers. Recently, the assumption of independence is questioned "*Is independence good for combination ?*" [7] and some attention has been devoted to the relationship between dependency and accuracy of voting methods especially for the majority voting method. In [8], it is shown that independence is not a universal requirement and negatively dependent classifiers are more beneficial than independent classifiers. In [11], the authors derive the upper and lower limits on the majority vote accuracy with respect to individual classifier accuracy p (the same for all classifiers), the number of classifiers in the pool and the pair wise dependence between the classifiers. They showed that for achieving the highest improvement over p, all pairs of classifiers in the pool should have the same negative dependency. How will the results of the plurality voting be affected when the classifiers are dependent? Is the plurality voting more sensitive to correlation than the majority voting? To seek answers to these questions, a large number of classifier ensembles with desired diversity is needed. However, building such ensembles on real-life data is difficult. In this paper, we are interested in the use of diversity in designing classifier teams in order to examine empirically the relationship between the plurality voting and classifier diversity. The idea behind the proposed approach is to generate classifier outputs with fixed accuracies and fixed agreement. A statistical measure, chosen in advance, is used to determine the pair wise agreement between the classifiers.

The paper is organized as follows. Section 2 presents the techniques suggested to enforce diversity between classifiers. Section 3 discusses the diversity measure used to generate the pair-wise agreement. The process of generating two classifiers according to the predefined agreement measure is presented in section 4. The case of more than two classifiers is addressed in section 5. Experimental results are reported in section 6. Conclusions are drawn in section 7.

2 Enforcing Diversity

Diversity among classifiers, in addition to individual performance of each classifier, is considered as one of the main factors, to explain the behaviour of combination methods. To create a consistent ensemble of classifiers, several implicit and explicit methods have been investigated in the literature. Duin [2] lists the main implicit approaches to build diverse classifiers. The principal one is to use different data representations adapted to the classifiers, for example, by using different feature vectors as inputs of classifiers [6, 13, 16]. The diversity can also be implicitly encouraged either by varying the classifier topology, the learning parameters [5] or by training each classifier on different parts of the data which is done for example by Bagging [1]. On the contrary, the aim of explicit methods is to design a set of classifiers by asserting the diversity measure in the process of building ensembles [4, 8, 13, 14]. The advantage of the incorporation of diversity measure is to control a priori the diversity between the classifier outputs in order to facilitate the analysis of the combination behavior. In [8] for example, the authors propose a comprehensive study of the random generation of binary classifier outputs to evaluate the performance of majority voting. They derive formulas according to how two classifiers can be generated with specified accuracies and dependencies between them using Q statistic as a diversity measure. Based on these formulas, the authors propose an algorithm for generating multiple dependent classifiers. Lecee et al [13] have also studied the influence of correlation among classifiers to evaluate combination methods of class type. For that, teams of classifiers producing binary outputs are created according to fixed recognition rates and predefined similarity measure.

3 Diversity Measure

The problems with diversity started to emerge with the attempts of measuring it [9]. Although used in many recognition problems, there is no analytic proof that a particular measure of diversity is preferable to another. The diversity measures operating on binary classifier outputs have been broadly categorized as pair-wise (measures which are calculated for each pair of classifiers in the team and then averaged) and non pair-wise [9, 15]. Such measures have been studied for artificial datasets [9] or for real-world datasets [15] to analyse the correlation between diversity and majority voting performance. In [9], the authors used ten diversity measures (four averaged pair-wise measures and six non pair-wise measures). Since most of the investigated measures showed very weak correlation with majority voting performance, they recommended the pair-wise Q statistic based on the criteria that it is understandable and relatively simple to implement. To emphasize the relationship between the plurality voting and diversity as well as the difference with the majority voting, we use in this study the kappa measure κ to estimate the level of agreement between the classifier outputs [3]. We slightly favoured this measure for the following reasons: κ depends on the individual accuracies of the classifiers and has a specific value 0 for statically independent classifiers. κ varies between –1 and +1. Values of κ close to +1 indicate that the classifiers tend to recognize the same objects correctly. If the classifiers commit errors for different objects, then κ takes negative values. In this paper, we assess a global agreement on correct and incorrect classification.

More specifically, let A_i and A_j be two classifiers providing S outputs for a N-class classification problem. Each output is made up of input pattern b_s (s=1, …, S) and class labels C_i (i=1 to N). We consider that when A_i and A_j propose the true class label b_s in their outputs, they agree. If they both propose incorrect labels, they also agree. In all the other cases, they disagree. Let p_i be the accuracy or the recognition rate of A_i which stands for the ratio of the number of outputs in which the true class appears on the total number of outputs. We can represent the output of classifier A_i as an S-dimensional binary vector $V_i = [v_{1,i}, \ldots, v_{S,i}]^T$ such that $v_{s,i}$='c' (correct), if the output s contain the true class label b_s, and $v_{s,i}$='w' (wrong) otherwise. Thus S$\times p_i$ elements of V_i have values 'c' and S $\times$ (1- p_i) elements have values 'w'. Thus, the N-class problem is transformed into a 2-class problem depending on the presence or absence of the true class in the classifier outputs.

The pair-wise agreement between A_i and A_j having accuracies p_i and p_j can be obtained by the following function [9]:

$$\kappa_{i,j} = 1 - \frac{p_{cw} + p_{wc}}{2\overline{p}(1-\overline{p})} \tag{1}$$

where $\overline{p}$ is the mean accuracy of the two classifiers, $\overline{p} = \frac{p_i + p_j}{2}$

p_{xy} is the probability that $v_{s,i}$ =x and $v_{s,j}$ =y (see Table 1).

The probabilities estimated in Table 1 are obtained by: $p_{cc} = \frac{n_{cc}}{S}$, $p_{cw} = \frac{n_{cw}}{S}$, $p_{wc} = \frac{n_{wc}}{S}$ and $p_{ww} = \frac{n_{ww}}{S}$ with $n_{cc} + n_{cw} + n_{wc} + n_{ww} = S$

where n_{xy} is the number of outputs in which the classifiers A_i and A_j propose or not the true class b_s .

Table 1. Diversity matrix DM_{ij} representing the percentage of agreement and disagreement between two classifiers A_i and A_j

	A_j correct (c)	A_j wrong (w)
A_i correct (c)	p_{cc}	p_{cw}
A_i wrong (w)	p_{wc}	p_{ww}

4 Output Generation for Two Classifiers

Generating classifiers according to accuracy only is easy. However, when the diversity has to be considered, the generation is not straightforward. Recently Kuncheva and Kountchev propose a method for generating multiple classifier (binary) outputs [8]. They derive formulas according to how two classifiers can be generated with desired accuracies and dependency Q between them. The generation of the classifiers is realized simultaneously. Inspired by this study, we present here another possible solution based on the kappa measure. The idea is to determine a priori the relationship (agreement or disagreement) between the two classifiers through the diversity matrix *DM* according to fixed accuracies and pre-specified agreement level and then to generate, from this matrix, the classifier outputs. The outputs of the first classifier in each team are generated by the classifier simulator developed in our previous work [18]. The generation procedure of these outputs is beyond the scope of this paper and has been discussed earlier in a separate paper [18]. Here we concentrate on the generation of the second classifier based on the outputs of the first classifier and the diversity matrix. Thus, when two classifiers A_1 and A_2 are to be simulated according to a predefined diversity level, the procedure of generation follows 3 steps:

1. Construction of the diversity matrix DM_{12} between A_1 and A_2 according to the desired accuracies p_1 , p_2 and kappa $\kappa_{1,2}$.
2. Generation of the outputs of A_1 according to the fixed accuracy p_1 .
3. Generation of the second classifier from A_1 taking into account the accuracy p_2 and the diversity matrix DM_{12}.

After simple algebraic manipulations using Table 1, we can express the entries p_{cc}, p_{cw}, p_{wc} and p_{ww} using p_1 , p_2 , $\overline{p}$ and $\kappa_{1,2}$ as:

$$\begin{aligned} p_{cc} &= \overline{p}\,((1-\overline{p})(\kappa_{1,2}-1)-1) \\ p_{cw} &= p_1 - p_{cc} \\ p_{wc} &= p_2 - p_{cc} \\ p_{ww} &= 1-2\overline{p} + p_{cc} \end{aligned} \tag{2}$$

After computing the elements of the diversity matrix DM_{12}, we generate the outputs O_1 =[$o_{1,1}$, …, $o_{1,S}$] of the basic classifier A_1 according to the desired accuracy p_1 and we transform it into the binary vector V_1. Based on this vector and DM_{12}, we generate the vector V_2 of the classifier A_2. To do that, we should place the solutions 'c' or 'w' in the outputs of V_2. This generation depends on the number of elements $n_{c.}$ that is the number of outputs for which the first classifier provides 'c' and $n_{w.}$ that is the number of outputs for which the first classifier provides 'w' with $n_{c.} = n_{cc}+ n_{cw}$ and $n_{w.} = n_{wc}+ n_{ww}$. Next, we generate the vector O_2 of the classifier A_2 according to O_1, V_1 and V_2. This procedure is illustrated in Table 2.

Table 2. Generating outputs of the classifier A_2 from the classifier A_1

```
Input: S the number of outputs, $O_1$ the output vector of $A_1$
Outputs: $O_2$ the output vector of $A_2$
Begin
  For each output $s$ =1 to S do
    Draw a number X randomly in the range [1, $n_{c.}+n_{w.}$]
    If X < $n_c$ then $v_{2,s}$='c' Else $v_{2,s}$='w'
    If $v_{1,s}$ = $v_{2,s}$ then $o_{2,s}=o_{1,s}$
    Else
      If $v_{1,s}$='c' then $o_{2,s}$ = $b_s$
      Else
        Choose randomly a class label C_i (i=1, …, N) different from $b_s$
        $o_{2,s}$ = C_i
  End
```

5 The Case of More Than Two Classifiers

The aim of the procedure described in this section is to generate automatically S outputs of L classifiers, for a N-class problem, which can be used for analysing the performance of voting methods. The input is the number of classes N, the number of outputs S, a vector with the desired individual accuracies $\boldsymbol{p}=[p_1,\ldots p_L]^T$, and a Kappa matrix $\kappa=[\kappa_{i,j}]$, where $\kappa_{i,j}$ is the desired agreement between classifiers A_i and A_j (i = 1,…, L-1; j = i+1,…, L). In the first step, the diversity matrices DM_{ij} are determined for each pair of classifiers A_i and A_j using (2). With L classifiers, we obtain L(L-1)/2 possible diversity matrices. The outputs of the first classifier are next generated according to the desired accuracy p_1 . Starting from these outputs, outputs of the other classifiers are generated. For example, take 3 classifiers. For each output s (s=1 to S), we use $o_{1,s}$ to set the output label $o_{2,s}$ of A_2 as presented in Table 2. Next, we use $o_{1,s}$ to generate the output label of A_3 according to the diversity matrices DM_{13} and DM_{23} . Note that there is no reason to use the same order of the classifiers. The selection can be done dynamically by generating for each output *s*, a random permutation of the integers from 2 to L. This is used to pick the order in which the classifiers will be selected. This generation process is repeated S times, so that L-1 output vectors are obtained. (see Table 3).

Table 3. Algorithm for the generation of classifier outputs according to the fixed accuracies and pair-wise agreement κ

```
Inputs: L the number of classifiers, S the number of outputs, N
the number of classes, $\boldsymbol{p}$ = {$p_1$, $p_2$, …, $p_L$) the desired accuracies
and $\kappa=[\kappa_{i,j}]$, where $\kappa_{i,j}$ is the desired agreement between classifiers
$A_i$ and $A_j$ (i=1,…,L-1; j=i+1,…,L)

Outputs: the output vectors $O_h$ (h=1,…, L)

Begin
 For each pair of classifiers $A_i$ and $A_j$ do
   Calculate the matrix $DM_{ij}$ according to $\kappa$ and $\boldsymbol{p}$ using (2)
   Generate the outputs $O_1$ of $A_1$ according to N, S and $p_1$
   For each output $\boldsymbol{s}$ of $A_1$ ($\boldsymbol{s}$ = 1 to S) do
     Choose a random permutation ($k_1$, $k_2$,…, $k_{L-1}$) in {2,…, L}
     Using $A_1$, set the $\boldsymbol{s}^{th}$ output of $A_{k_1}$ according to $DM_{1,k_1}$
     For t =2 to L-1 do
       Using $A_1$, set the $\boldsymbol{s}^{th}$ output of $A_{k_t}$ according to $DM_{1,k_t}$
       Decrement the matrix $DM_{1,k_{t-1}}$
End
```

Note that there are cases where the generation of classifier ensembles with fixed diversity and accuracy are not possible (when the elements of the diversity matrix are negative). This can be explained by the fact that the kappa measure depends on the individual classifier accuracy. For two classifiers having performance equal to 0.5, the diversity can varied between –1 and +1. But, as the accuracy increases, the range of the diversity decreases. For example, the diversity between two classifiers with p=0.9, vary in [-0.1, 1].

6 Experimental Results

In this section, we report two experiments aimed at examining the relationship between the performance of voting methods and diversity in classifier behavior. In these experiments, we simulate ensembles of classifiers with the same pair-wise agreement $\kappa = \{-0.4, -0.2, 0, +0.2, +0.4\}$ for a 10-class problem. For each classifier, S = 10000 outputs are generated. The goal of the first experiment is to study the effect of an additional classifier on the performance of plurality voting method. For this experiment, we simulate two classifiers with the same recognition rate $p_1=p_2$ in {0.5, 0.6, 0.7} to which is added a third one whose recognition rate performance p_3 vary from 0.5 to 0.7 by 0.1 step. The recognition rates of the three individual classifiers and the plurality voting are shown in Figure 1. The results of Figure 1a demonstrate that the addition of the third classifier is interesting for pairs of classifiers having recognition rates higher than 0.5. In this case, as the value of kappa measure increases (i.e. the diversity decreases), the performance of plurality voting decrease. However, the addition of the third classifier does not achieve better performance over the best individual classifier for classifiers having recognition rate equal to 0.5, even in negatively dependent situation. Figure 1b and 1c show that for diverse and independent classifiers, the addition of a third classifier is beneficial for combination whatever its performance. However, for kappa > 0, the combination is not interesting.

The main goal of the second experiment is to determine which voting methods would perform best under the condition that all of the classifiers have the same accuracy p=0.5 (i.e. weak classifiers). Table 4 shows the results of combining five classifiers by plurality and majority voting. These results show that plurality voting outperforms majority voting for different classifier ensembles. The performance of majority voting is quite similar to the individual classifiers. The combination by this rule is not interesting in this case. Consider the combination of three classifiers with $p_1= p_2= p_3$ =0.5 as in the Figure 1a. We can see that the addition of two classifiers in the second experiment improves significantly the recognition rates of the plurality voting. The behaviour of the plurality voting becomes more stable than in the previous experiment. Another advantage of plurality voting over majority voting is its higher rejection efficiency. Plurality voting has a lower rejection rate than the majority voting. This rejection rate increases when the diversity between the classifiers decreases. The experimental results in [14] also support this conclusion when combining independent classifiers.

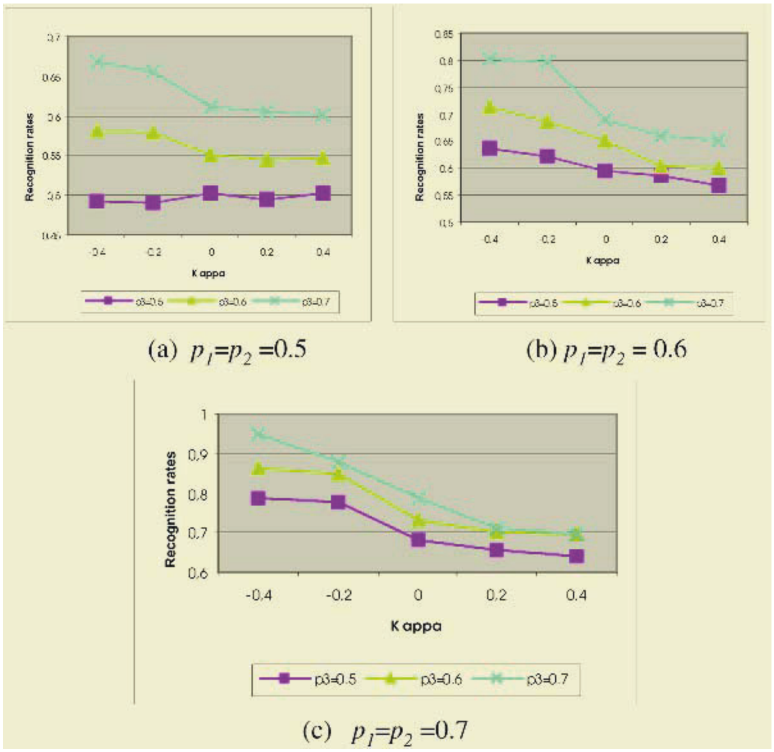


Fig. 1 . Combination results with different performance of classifiers

Table 4. Results of combining five weak classifiers with the two combining rules

	Majority voting		Plurality voting	
Diversity	Recognition rates	Rejection rates	Recognition rates	Rejection rates
-1	0.504 ± 0.001	0.492 ± 0.010	0.812 ± 0.036	0.143 ± 0.045
-0.5	0.501 ± 0.004	0.495 ± 0.002	0.741± 0.009	0.186 ± 0.007
0	0.500 ± 0.030	0.494 ± 0.004	0.737 ± 0.016	0.190 ± 0.018
0.5	0.500 ± 0.008	0.492 ± 0.004	0.682 ± 0.003	0.213 ± 0.005
1	0.500 ± 0.009	0.500 ± 0.011	0.524 ± 0.003	0.286 ± 0.007

7 Conclusion

In this paper, we have proposed a new simulation method for the artificial generation of classifier outputs to examine the relationship between the plurality voting performance and between-classifier diversity. An algorithm for building two classifiers with specified accuracies and a pair-wise agreement κ between them is presented. The algorithm for generating more than 2 classifiers has also been proposed. The experimental results point out the advantages of plurality voting over majority voting when classifiers with different agreement levels are combined. The proposed method will be useful to evaluate any abstract-level combination methods and thus will be helpful to clarify the conditions under which a combination method can be used or is the best for different pattern recognition problems.

References

1. L. Breiman, Bagging predictors, *Machine Learning*, 24:2, (1996), 123-140.
2. R.P. Duin, The combining classifier: to train or not to train?, in: R. Kasturi, D. Laurendeau, C. Suen (eds.), ICPR16, *Proceedings 16th International Conference on Pattern Recognition,* Canada, vol. II, IEEE Computer Society Press, Los Alamitos, (2002), 765-770.
3. J. Fleiss, Statistical methods for rates and proportions, John Wily & sons, (1981).
4. G. Giacinto, F. Roli, Design of effective neural network ensembles for image classification processes, *Image Vision and Computing Journal*, 19:9/10, (2001), 699-707.
5. L. Hansen, P. Salamon, Neural network ensembles, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12, (1990), 993-1001.
6. J. Kittler, M. Hatef, R.P.W. Duin, J. Matas, On combining classifiers, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:3, (1998).
7. L.I. Kuncheva, C.J. Whitaker, C.A. Shipp, R.P.W. Duin, Is independence good for combining classifiers ?, ICPRI 5, *Proceedings 15th International Conference on Pattern Recognition*, 2, (2000), 168-171.
8. L.I. Kuncheva, R.K Kountchev, Generating classifier outputs of fixed accuracy and diversity, *Pattern Recognition Letters*, 23. (2002), 593-600.
9. L.I. Kuncheva, C.J. Whitaker, Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy, *Machine Learning*, 51, (2003), 181-207.
10. L.I. Kuncheva, That elusive diversity in classifier ensembles, Proc IbPRIA 2003, Mallorca, Spain, *Lecture Notes in Computer Science*, *Springer-Verlag*, (2003), 1126-1138.
11. L.I. Kuncheva, C.J. Whitaker, R.P.W. Duin, Limits on the majority vote accuracy in classifier fusion, *Pattern Analysis and Applications*, 6, (2003), 22-31.
12. L. Lam, C.Y. Suen, Application of majority voting to pattern recognition : an analysis of its behavior and performance, *IEEE Transactions on System, Man, and Cybernetics*, 27:5, (1997).
13. V.D. Lecce, G. Dimauro, A. Guerrierro, S. Impedovo, G. Pirlo, A. Salzo, Classifier combination: the role of a-priori knowledge, *In Proceedings of the 7th International Workshop on Frontiers in Handwriting Recognition*, Amsterdam, The Netherlands (2000), 143-152.
14. X. Lin, S. Yacoub, J. Burns, S. Simske, Performance analysis of pattern combination by plurality voting, *Pattern Recognition Letters*, 24, (2003), 1959-1969.
15. D. Ruta, B. Gabrys, New measure of classifier dependency in multiple classifier systems, *In Proceedings of the 3rd International Workshop on Multiple Classifier System (MCS), Lecture Notes Computer Science 2364*, Calgary, Sardinia, Italy, (2002), 127-136.
16. D.M.J. Tax, M.V. Breukelen, R.P.W. Duin, J. Kittler, Combining multiple classifiers by averaging or by multiplying ?, *Pattern Recognition*, 33, (2000), 1475-1485.
17. M. VanErp, An overview and comparison of voting methods for pattern recognition, International Workshop Frontiers Handwritten Recognition, (2002), 195-200.
18. H. Zouari, L. Heutte, Y. Lecourtier, A. Alimi, Simulating classifier outputs for evaluating parallel combination method, *Lecture Notes in Computer Science*, 4th International Workshop, Multiple Classifier Systems (MCS), Guildford, UK, (2003), 296-305.

Feature Selection Using Improved Mutual Information for Text Classification

Jana Novovičová[1,2], Antonín Malík[1,3], and Pavel Pudil[1,2]

[1] Institute of Information Theory and Automation,
Department of Pattern Recognition, Academy of Sciences of the Czech Republic,
Prague, Czech Republic
{novovic,amalik,pudil}@utia.cas.cz
[2] The University of Economics, Faculty of Management,
Prague, Czech Republic
[3] Czech Technical University, Faculty of Electrical Engineering,
Prague, Czech Republic

Abstract. A major characteristic of text document classification problem is extremely high dimensionality of text data. In this paper we present two algorithms for feature (word) selection for the purpose of text classification. We used sequential forward selection methods based on improved mutual information introduced by Battiti [1] and Kwak and Choi [6] for non-textual data. These feature evaluation functions take into consideration how features work together. The performance of these evaluation functions compared to the information gain which evaluate features individually is discussed. We present experimental results using naive Bayes classifier based on multinomial model on the Reuters data set. Finally, we analyze the experimental results from various perspectives, including F_1-measure, precision and recall. Preliminary experimental results indicate the effectiveness of the proposed feature selection algorithms in a text classification problem.

1 Introduction

The goal of text document classification is to assign automatically a new document into one or more predefined classes based on its contents.

An increasing number of statistical classification methods and machine learning algorithms have been explored to build automatically a classifier by learning from previously labelled documents including *naive Bayes*, *k-nearest neighbor*, *support vector machines*, *neural network*, *decision trees*, *logistic regression* (see e.g. [7], [9], [5], [11], [12] and the references therein).

In text classification, usually a document representation using a *bag-of-words* approach is employed (each position in the feature vector representation corresponds to a given word). This representation scheme leads to very high-dimensional feature space. *Feature selection* is a very important step in text classification, because irrelevant and redundant words often degrade the performance of classification algorithms both in speed and classification accuracy.

Methods for feature subset selection for text document classification task use an evaluation function that is applied to a single word. All words are independently evaluated and sorted according to the assigned criterion. Then, a predefined number of the best features is taken to form the best feature subset. Scoring of individual words can be performed using some of the measures, for instance, *document frequency*, *term frequency*, *mutual information*, *information gain*, *odds ratio*, χ^2 *statistic* and *term strength* [10], [8], [3]. Yang and Pedersen [10] and Mladenic [8] give experimental comparison of the above mentioned measures in text classification. The information gain (IG) and a very simple frequency measures were reported to work well on text data. Forman in [3] presents an extensive comparative study of twelve feature selection criteria for the high-dimensional domain of text classification.

In this paper we propose to use sequential forward selection methods based on improved mutual information introduced by Battiti [1] and Kwak and Choi [6], who introduced these criteria for non-textual data. To our knowledge, the improved mutual information has not yet been applied in text classification as a criterion for reducing vocabulary size. We use the simple but effective naive Bayes classifier based on multinomial model.

2 Naive Bayes Classifier

According to the bag-of-words representation, the document d_i can be represented by a feature vector consisting of one feature variable for each word w_t in the given vocabulary $V = \{w_1, \dots, w_n\}$ containing n distinct words. Let $C = \{c_1, \dots, c_{|C|}\}$ be the set of $|C|$ classes. Note, that $|C|$ classes are pre-defined and that document always belongs to at least one class. Given a new document d, the probability that d belongs to class c_j is given by Bayes rule

$$P(c_j|d) = \frac{P(c_j)P(d|c_j)}{P(d)}. \tag{1}$$

If the task is to classify a new document into a single class, simply select the class $c^\star$ with the highest posterior probability.

Assuming a multinomial model [7, 9] and class-conditional independence of words yields the well-known naive Bayes classifier, which computes the most probable class for d as

$$c^\star = \underset{j=1,\dots,C}{\operatorname{argmax}} P(c_j|d) = \underset{j=1,\dots,C}{\operatorname{argmax}} P(c_j) \prod_{t=1}^{n} P(w_t|c_j)^{N(w_t,d)} \tag{2}$$

where $N(w_t, d)$ is the number of occurrences of word w_t in document d. The word probability $P(w_t|c_j)$ are usually estimated using Laplacean prior:

$$P(w_t|c_j) = \frac{1 + \sum_{d_i \in c_j} N(w_t, d_i)}{|V| + \sum_{r=1}^{|V|} \sum_{d_i \in c_r} N(w_t, d_i)} . \tag{3}$$

The class priors $P(c_j)$ are estimated by maximum-likelihood estimates the fraction of documents in each class.

2.1 Classifier Estimation

For evaluating the multi-label classification accuracy we use the standard multi-label measures: precision, recall and F_1 measure. Precision and recall are computed as

$$precision = \frac{\#\ classes\ found\ and\ correct}{\#\ total\ classes\ found}$$

$$recall = \frac{\#\ classes\ found\ and\ correct}{\#\ total\ classes\ correct}$$

where "classes found" means classes c_k with $P(c_k|d) \geq h$. To obtain the single number measure of classification performance we compute the F_1 measure that combine both the precision p and recall r

$$F_1 = \frac{2pr}{p+r}. \tag{4}$$

The closer are the values of precision and recall, the higher is the F_1 measure.

In the case of multi-label classification the document d is classified in the class c_k if the probability $P(c_k|d) \geq h$. The threshold h is estimated to maximize F_1 measure (4) on the training data set. The threshold h_j shifts from 0 to 1 and for each potential value of h_j we make the classification process on the training data set. All training documents d_i are classified according to the equation $P(c_k|d_i) \geq h_j$ and the F_1 measure is computed for each h_j. Then the threshold h with the highest F_1 value is selected.

Given a new document d, the probability $P(c_j|d)$ is computed by applying Bayes rule (1). If the probability $P(c_j|d) > h$, than the document d is assigned to the class c_j. Therefore, the document d can be assigned to one or more classes. If $P(c_j|d) < h$ for each class c_j, the document d is classified in the class with highest probability $P(c_j|d)$.

While the word independence assumption is false in practice with real-world data, there is empirical evidence that the naive Bayes yields surprisingly good classification performance on text data.

3 Feature Selection

Feature subset selection is commonly used when learning on text data, since text documents are characterized by high-dimensionality feature vector.

The focus of this paper is the comparison between *best individual features* and *sequential forward selection* methods. Both methods are based on mutual information $I(C, w_i)$ between classes C and word w_i that is commonly named *information gain* (IG) in text classification

$$I(C, w_i) = \sum_{k=1}^{|C|} P(c_k, w_i) \log \frac{P(c_k, w_i)}{P(c_k)P(w_i)} + \sum_{k=1}^{|C|} P(c_k, \overline{w}_i) \log \frac{P(c_k, \overline{w}_i)}{P(c_k)P(\overline{w}_i)} \qquad (5)$$

where $P(w_i)$ is the probability, that the word w_i occurred, $\overline{w_i}$ means, that the word not occurred, $P(c_j)$ is the probability of the class c_j, $P(c_j, w_i)$ is the joint probability of the class c_j and the occurrence of the word w_i.

3.1 Best Individual Features

Best individual features (BIF) methods [4] evaluate all the n words individually according to a given criterion, sort them and select the best k words.

Since the vocabulary has usually several thousands or tens of thousands of words, the BIF methods are popular in text classification because they are rather fast, efficient and simple. However, they evaluate each word separately and completely ignore the existence of other words and the manner how the words work together. In [2] it has been proven that the best pair of features need not contain the best single features.

Scoring of individual features can be performed using some of the measures, for instance, *document frequency*, *term frequency*, *mutual information*, *information gain*, χ^2 *statistic* or *term strength*. Yang and Pedersen [10] give experimental comparison of the above mentioned measures in text classification. They found information gain and χ^2 statistic most effective in word selection.

In our comparison we include BIF method with information gain criterion (BIF IG) defined in (5).

3.2 Sequential forward Selection

Sequential forward selection (SFS) methods firstly select the best single word evaluated by given criterion. Then, add one word at a time until the number of selected words reaches desired k words. However SFS methods do not result in the optimal words subset but they take note of dependencies between words as opposed to the BIF methods. Therefore SFS often give better results than BIF. The similar strategy is *sequential backward selection* that starts with all n words and successively remove one word at a time.

SFS are not usually used in text classification because of their computation cost due to large vocabulary size. However, in practice we can often both employ calculations from previous steps and make some pre-computations during the initialization. Since feature selection is typically done in an off-line manner, the computational time is not as important as the optimality of words subset or classification accuracy.

We propose two SFS methods based on mutual information (SFS MI) introduced by Battiti [1] and Kwak and Choi [6]. They sufficiently applied these two methods for non-textual data compared with BIF. In contrast to BIF IG, SFS MI uses not only mutual information $I(C, w_i)$ between the set of classes C and a word w_i but also mutual information $I(w_i, w_j)$ between the words w_i and w_j. The SFS MI algorithm is described in the following steps:

1. **Initialization:**
   the set of selected words $S = \emptyset$,
   the set of unselected words $U =$ 'all n words'.

   Pre-computation:
   $I(C, w_i)$ for $i = 1, \ldots, n$,
   I_{ij}, for $i, j = 1, \ldots, n$ and $i \neq j$
   - Battiti: $I_{ij} = I(w_i, w_j)$
   - Kwak-Choi: $I_{ij} = I(w_i, w_j) I(C, w_j)/H(w_j)$
2. **First word selection:**
   Find the word $w^\star$ with maximal $I(C, w_i)$,
   $w^\star = \arg\max_{i=1,\ldots,n} I(C, w_i)$,
   set the sets $S = \{w^\star\}$, $U = U \setminus \{w^\star\}$.
3. **One step:**
   Repeat until the demand k words are selected ($|S| = k$).
   Choose the best word $w^\star$ from the set U.
   $w^\star = \text{argmax}_{i=1,\ldots,|U|} \{I(C, w_i) - \beta \sum_{j=1}^{|S|} I_{ij}\}$
   Set the sets $S = S \cup w^\star$ and $U = U \setminus w^\star$.

$H(w_j)$ is the entropy of the word w_j. The variable $\beta \geq 0$ is typically set to 1. The higher β the stronger impact of the mutual information between words. On the other hand if $\beta = 0$, then the mutual information between words is not considered and the algorithm coincides with the BIF IG selection.

4 Experimental Results

In our experiments we compared the performance of three feature selection methods. The first method is standard BIF algorithm using the IG criterion. Each word is evaluated by IG criterion and then are selected the first best k words. The other two SFS methods Battiti SFS MI [1] and Kwak-Choi SFS MI [6] are based on mutual information criterion between the set of classes C and the word w_i as well as BIF IG method. Moreover SFS MI consider the mutual information between each pair of words and add one word in each step.

All experiments were tested for different number of words on the common used Reuters[1] data set. Since Reuters documents are multi-labelled we employed the *micro-average F_1-measure* and the *precision-recall tradeoff* for evaluating performance.

[1] http://www.daviddlewis.com/resources/testcollections/reuters21578

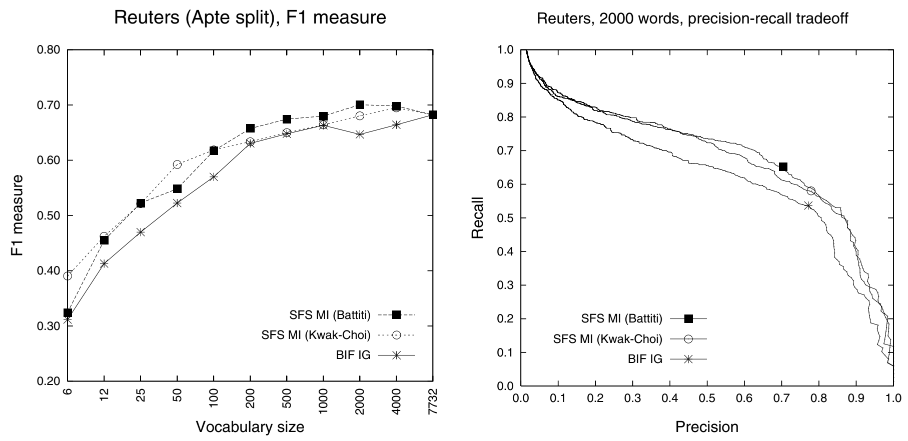


Fig. 1. (a) $F_{\cdot}$ measure of BIF IG, SFS MI (Battiti) and SFS MI (Kwak and Choi) on Reuters data set with Apte split. (b) Identical feature selection methods with (a) but precision-recall tradeoff on 2000 selected words. The highlighted points in (b) show the maximal $F_{\cdot}$-measure.

First, we displaced all unlabelled documents from the data. Second, we removed all uninformative words occurring in stop-list, such as prepositions, conjunctions or articles. Then, Porter stemming algorithm[2] was used. Finally we deleted the words that occurred only once or twice. The data resulted in 7732 words and 11280 documents in 118 classes. We divided this data set in the training and the testing set according to the usually used Apte split.

For classification was used the naive Bayes classifier based on the multinomial model. In addition to training standard parameters, the threshold h for multi-label classification was made to maximize F_1 measure on the training data set.

Figure 1 (a) shows the comparison of all three FS methods on F_1 measure. We can see that both observed SFS MI methods significantly overcome the BIF IG algorithm on the Reuters data. Compared with the BIF IG, the F_1 value of SFS MI algorithms is with some vocabulary sizes even greater than with the full number of 7732 words.

The highest value of F_1 is achieved on 2000 words with the Battiti SFS MI algorithm. The precision-recall tradeoff on 2000 words is depicted on the Figure 1 (b). Figures 2 (a) and (b) presents the similar result like Figure 1 (a) but on the precision and recall measure.

The Kwak-Choi SFS MI has approximately higher value of F1, precision and recall than the Battiti SFS MI on the lower number of words. However, on the greater number of words the Battiti SFS MI overcome it with all three measures.

The time complexity of SFS algorithms is less than $O(kn^2)$ where k is the number of desired words and n is the total number of words. The algorithm adds step by step k words and in each step compute the mutual information between

• http://www.tartarus.org/˜martin/PorterStemmer

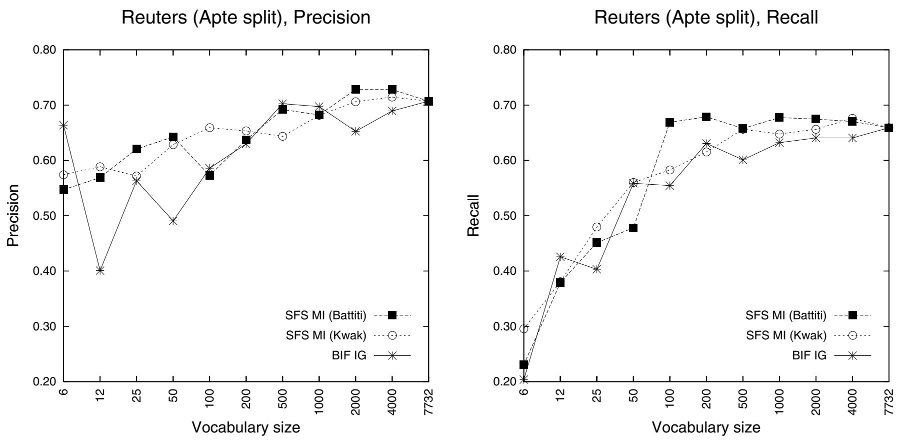


Fig. 2. (a) Precision and (b) recall of BIF IG, SFS MI (Battiti) and SFS MI (Kwak and Choi) on Reuters data with Apte split. The same threshold h was used as in the figure 1.

each word belonging to the set S (selected words) and each word from the set U (unselected words). The required space is $n^2/2$ because we need to store the mutual information between each pair of words. If we compare the BIF and SFS methods, the SFS methods are more time consuming but achieve significantly better results on the testing data.

5 Conclusions and Future Work

In this paper, we have presented sequential forward selection methods based on novel improved mutual information measure. The algorithms are new in the field of text classification and take into consideration how the features work together. These methods significantly overcome standard best individual features method based on information gain on the testing data set.

Many areas of future work remain. Ongoing work includes comparison on the other text classifiers, for example, support vector machines and k-nearest neighbor.

Acknowledgements

The work has been supported by grants of the Grant Agency of the Academy of Sciences of the Czech Republic No A2075302, the Grant Agency of the Czech Republic No 102/03/0049 and the Complex research project of the Academy of Sciences of the Czech Republic K1019101.

References

1. Battiti, R.: Using Mutual Information for Selecting Features in Supervised Neural Net Learning. IEEE Trans. Neural Networks **5** (1994) 537–550
2. Cover, T.M.: The Best Two Independent Measurements are not The Two Best. IEEE Trans. Systems, Man, and Cybernetics **4** (1974) 116–117
3. Forman, G.: An Experimental Study of Feature Selection Metrics for Text Categorization. Journal of Machine Learning Research **3** (2003) 1289–1305
4. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical Pattern Recognition: A Review. IEEE Trans. on Pattern Analysis and Machine Intelligence **22** (2000) 4–37
5. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In: Proceedings of the ECML'98 (1998) 137–142
6. Kwak, N., Choi, C.: Improved Mutual Information Feature Selector for Neural Networks in Supervised Learning. In: Int. Joint Conf. on Neural Networks (IJCNN '99) (1999) 1313–1318
7. McCallum, A., Nigam, K.: A Comparison of Event Models for Naive Bayes Text Classification. In: Proceedings of the AAAI-98 Workshop on Learning for Text Categorization (1998) 41–48
8. Mladenic, D., Grobelnik, M.: Feature Selection for Unbalanced Class Distribution and Naive Bayes. In: Proceedings of the Sixteenth International Conference on Machine Learning (1999) 258–267
9. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text Classification from Labelled and Unlabelled Documents Using EM. Machine Learning **39** (2000) 103–134
10. Yang, Y., Pedersen, J.O.: A Comparative Study on Feature Selection in Text Categorization. In: Proceedings of the 14th ICML97 (1997) 412–420
11. Yang, Y.: An Evaluation of Statistical Approaches to Text Categorization. Journal of Information Retrieval **1** (1999) 67–68
12. Yang, Y., Zhang, J., Kisiel, B.: A Scalability Analysis of Classifier in Text Categorization. In: Proceedings of the 26th ACM SIGIR Conference on Research and Development in Information Retrieval (2003) 96–103

Feature Subset Selection Using an Optimized Hill Climbing Algorithm for Handwritten Character Recognition

Carlos M. Nunes[1], Alceu de S. Britto Jr.[1,2],
Celso A.A. Kaestner[1], and Robert Sabourin[3]

[1] Pontifícia Universidade Católica do Paraná (PUC-PR)
R. Imaculada Conceição, 1155 - Curitiba (PR) 80215-901, Brazil
{carlosmn,alceu,kaestner}@ppgia.pucpr.br
[2] Universidade Estadual de Ponta Grossa (UEPG)
Pç. Santos Andrade S/N, Centro - Ponta Grossa (PR) 84100-000, Brazil
[3] École de Technologie Supérieure (ETS)
1100 Rue Notre Dame Ouest - Montreal (QC) H3C 1K3, Canada
robert.sabourin@etsmtl.ca

Abstract. This paper presents an optimized Hill Climbing algorithm to select a subset of features for handwritten character recognition. The search is conducted taking into account a random mutation strategy and the initial relevance of each feature in the recognition process. The experiments have shown a reduction in the original number of features used in an MLP-based character recognizer from 132 to 77 features (reduction of 42%) without a significant loss in terms of recognition rates, which are 99% for 60,089 samples of digits, and 93% for 11,941 uppercase characters, both handwritten samples from the NIST SD19 database. The proposed method has shown to be an interesting strategy to implement a wrapper approach without the need of complex and expensive hardware architectures.

1 Introduction

Many feature subset selection algorithms [1,2,3] have been developed in the last years, since this procedure can reduce not only the cost of recognition by reducing the number of features that need to be used, but in some cases it can also provide better classification accuracy. Usually, the methods found in the literature [4,5] can be categorized as: Filter or Wrapper-based approach.

In both categories, given a set of candidate features, the objective is to select a subset that performs the best under some classification system. The main difference between them is that in the Filter-based approach the relevance of each feature is defined taking into account statistical information calculated from the training dataset, while in the Wrapper approach [6,7] the classifier is used to evaluate the relevance of each feature during the selection process.

An interesting wrapper-based method was proposed in [8,9] using a genetic algorithm to the subset selection. The authors have achieved a significant feature reduction (from 132 to 100 features), which means about 25% keeping the initial classification rate almost the same (99.16%) for handwritten digits available in the NIST SD19 database. However, their process must to evaluate 128,000 feature subset candidates. For this purpose, the authors use a cluster of 17 personal computers (with 1.1GHz and 512 Mb RAM each).

In order to provide a low-cost solution in terms of architecture needed, this paper presents an optimized Hill Climbing algorithm to select a subset of features for handwritten character recognition. The search is done taking into account a random mutation strategy and a priority associated to each feature by considering its relevance in the recognition accuracy. Different from the method proposed in [8], the proposed method has shown to be an interesting strategy to implement a wrapper approach, which can be executed in more simple hardware architecture. The results presented in this paper are based on 16,000 features subset candidates selected by the proposed algorithm in a single PC (1.3 GHz).

2 Proposed Method

Figure 1 presents an overview of the proposed method, in which is possible to observe 5 stages. The first stage consists of a feature extraction process, which generates 5 files (*train*, *val*$_1$, *val*$_2$, *val*$_3$ and *test*) containing feature vectors extracted from character images.

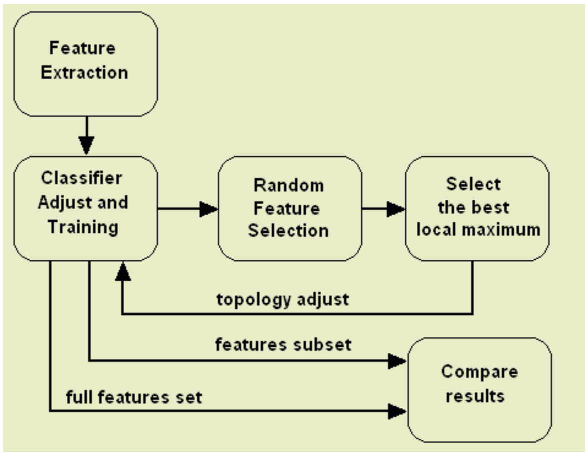


Fig. 1. Overview of the proposed method

In the second stage, a MLP neural network is trained and evaluated considering the original configuration of the feature vector and using *train* and *val*$_1$. In the third stage, an optimized Hill Climbing (OpHC) algorithm uses *val*$_2$ in order to select multiple candidates of feature subsets. A feature randomly selected is removed, or not, taking into account its relevance on the classification accuracy. In addition, a priority is associated to each feature to guide the search.

2.1 Feature Extraction

The same feature extraction method described in [8] is used. It divides a character image into 6 regions and calculates 132 features based on contour, concavity and surface information.

2.2 Classifier

The classifier is an MLP neural network, which was used to allow the comparison with the results reported in [8,9]. The initial topology consists of 132 nodes in the input layer, 100 nodes in the hidden layer, and the output layer contains 10 or 26 nodes for digit or uppercase character recognition, respectively.

2.3 Optimized Hill Climbing Algorithm (OpHC)

This module consists of a modified Hill Climbing algorithm. Figure 2 shows, in italic style, the main differences of the proposed algorithm from the Original Hill Climbing (OrHC).

```
1. Establish priority for each feature;
2. Load neural network previously trained;
3. If (Number of Iteration = MAXITER) then exit;
4. Select a feature, randomly;
5. If (the priority of the selected feature = zero)
      then remove it (e.g. replace it by it's average value) and
           update the current feature set mask;
      else increase the feature priority and go to step 4;
6. Evaluate the classification accuracy with the new feature set configuration;
7. If (current error rate <= previous one)
      then keep the current feature set configuration;
      else backtrack to the previous state;
8. If (number of removed features = TFEAT)
      then save current configuration as a local maximum; go to step 3;
      else go to step 4.
TFEAT – Total of features;          MAXITER – Maximum of iterations.
```

Fig. 2. Optimized Hill Climbing Algorithm (OpHC)

The algorithm starts by defining a priority for each feature (step 1) available on the initial feature vector configuration (132 for this problem). Each feature, or seed, has its priority level calculated as shown in Figure 3. In the second step of the algorithm, the neural network trained using the entire feature set is loaded. In the kernel of the algorithm (step 4), a random process is used to select a feature to be removed. In case the priory related to this feature is zero (step 5), it will be removed, otherwise its priority will be increased and a different feature will be randomly selected. The step 6 provides the current error rate after removing the selected feature. A decision about to keep the current feature set configuration is taken on step 7. For this purpose, the current error rate is compared to the previous one. As we can see, a local maximum is found after evaluating all the features in the current state (see step 8). After that, the OpHC algorithm returns to the initial state instead of backtracking to the previous one. Thus, the complete configuration of the feature set is considered again and a new feature or seed (not processed yet) is selected. The objective is to investigate other areas of the search space. In addition, the priority computed for each feature provides a way of guiding the search taking into account the feature relevance, while the concept of randomized feature removal is maintained.

```
1.  Calculate the error rate when individual features are removed (FERROR),
    see Figure 4;
2.  Select the maximum (MAX) and minimum (MIN) FERROR over all features;
3.  Select a number of levels (NLEVELS), which is the number of priority levels to be considered;
4.  Calculate the range of each level using R = (MAX-MIN) / NLEVELS;
5.  The relevance (priority) of each feature is calculated as:
        P = (- NLEVELS + (MAX–FERROR) / R).
```

Fig. 3. Scheme to calculate the feature priority

As described before, the feature priority takes values between 0 and -NLEVELS. According to Figure 3, the highest error values receive –NLEVEL and the lowest error values receive zero. Each time that a feature is selected to removing, the algorithm evaluates its priority and in case of the priority is 0 (zero) the feature will be removed, otherwise the feature is not removed and the priority is updated (P = P+1). Thus, when the priority reaches the value zero, the corresponding feature is removed. Figure 4 shows the initial individual feature error.

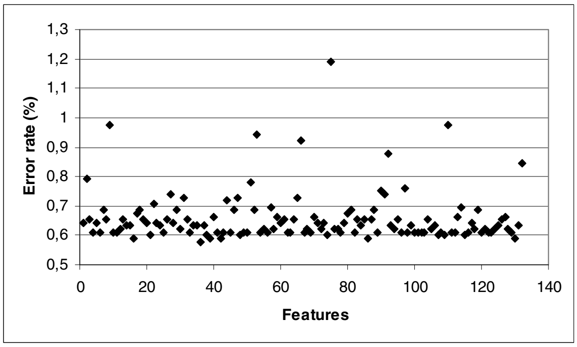


Fig. 4. Error rate after individual removal

The number of levels (NLEVELS) was experimentally defined. After evaluating the values 0, 5, 10, 20 and 50, the best results were obtained by using the value 10. Figure 5 shows that, after 120 iterations the algorithm had already removed 40 features from the original set.

Another important characteristic of the proposed algorithm is the use of sensitivity analysis [8], since to retrain the neural network at each new feature subset is not fea-

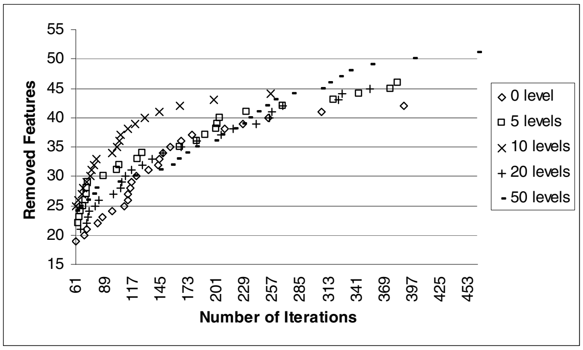


Fig. 5. Selection of the number of priority levels

sible due to the limits imposed by the learning time of the huge training set considered in this work. The sensitivity of the network is used to estimate the relationship between the input features and the network performance. So, in order to evaluate a given feature subset we replace the unselected features by their averages values (step 5 – Figure 3) evaluated on the training database. In this way, we avoid training the neural network and hence turn the wrapper approach feasible for our problem.

Each local maximum found by this module represents a feature subset candidate. The search for feature subsets is done on val_2.

2.4 Selection of the Best Feature Subset Candidate

A different validation dataset (val_3) is used to select one feature subset from those candidates provided by the OpHC algorithm. The selection is done based on the recognition accuracy. Another strategy used to select the best feature set configuration is selecting that with represents the smallest feature subset – also evaluated in our experiments.

2.5 Final Evaluation

In this module, the final feature subset selected in the last stage is used to retrain the neural network, whose topology is adapted to this new configuration of the feature set. A final evaluation is done using the *test* file.

3 Experimental Results and Discussion

The experiments are based on handwritten digits and uppercase characters available in the NIST SD19 database. In the experiment based on digits the following protocol was used: 195,000 samples for training (*train*) and 28008 for validation (9,336 in val_1, 9,336 in val_2 and 9,336 samples in val_3) - all samples from series hsf_0, 1, 2 and 3 series. Other 60,089 samples were used for testing (*test*), which are available on hsf_7 series.

The data sets used in the experiments using uppercase characters are composed of 37,440 samples for training (*train*) and 12,092 for validation (4,031 in val_1, 4,031 in val_2 and 4,030 in val_3) – all samples from hsf_0,1,2 and 3 series. Other 11,941 samples from hsf_4 were used for testing (*test*).

In the experimental protocol, val_1 is used during training of the neural network to avoid an overtraining. Val_2 is used during the search for feature subset candidates performed by the HC algorithm. Finally, val_3 is used to select the subset of features, which provides the best recognition accuracy among the candidates provided by the HC algorithm. The testing set (*test*) is used as a black box just to compare the final recognition results of the classifier after his topology has been adapted to the new configuration of the feature set.

3.1 Experiments on Handwritten Digits

Both algorithms, the original and optimized HC were evaluated. The MAXITER variable was set to 16,000 iterations or solutions. At the end of the iterations the original HC has used only 3 seeds, since it does not returns to the initial configuration of the feature set (initial state) after finding a local maximum or after removing all features in the current state, but it returns to the previous state. From this three seeds or starting points, the original algorithm found 172 local maximum (subset candidates). By contrast, the modified algorithm has investigated with the priority scheme all possible seeds (132) using the same 16,000 iterations and it has generated 47 feature subset candidates. In fact, the proposed algorithm has stopped after all features have been used as seed. This means that all features were removed during the search and a bigger diversity on evaluated solutions was reached.

In Figure 6, each point represents a feature subset candidate of the 172 found by using the Original Hill Climbing (OrHC). As we can observe, all points are very concentrated in a small area of the search space. By contrast, in Figure 7, the 47 feature subset candidates found by using the modified HC are not concentrated as those provided by the original algorithm. The reason is that in the modified algorithm each time a local maximum is found, the initial configuration of the feature set is returned in order to select a new seed. Moreover, the strategy of using the priority scheme has provided to the algorithm a faster convergence to a local maximum.

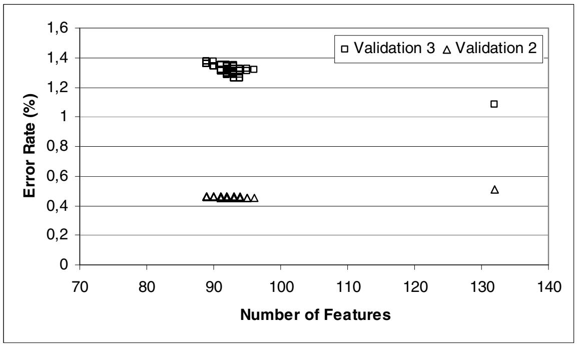


Fig. 6. Feature subset candidates provided by the traditional HC algorithm

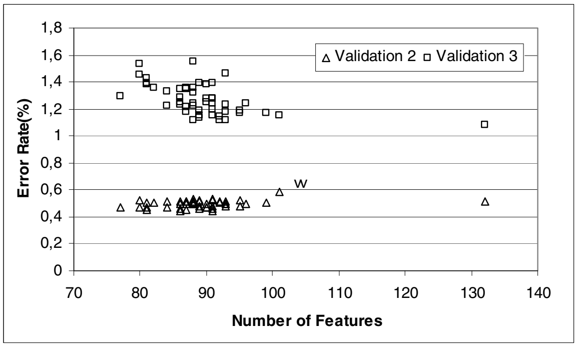


Fig. 7. Feature subset candidates provided by the optimized HC algorithm

In a first set of experiments, the *val*$_3$ dataset is used to evaluate and select the best local maximum (or feature subset) from the 172 provided by the original HC algorithm (OrHC). The same strategy is used to select a local maximum from the 47 provided by the optimized algorithm (OpHC). In addition, we also experiment to consider just the number of features removed, since the classifier has shown a small loss in terms of accuracy. This means that, in this experiment, the best configuration corresponds to the smallest subset of features.

As we can observe in Table 1, there is no significant loss in terms of classification accuracy. However, it is possible to observe a reduction of features when the optimized HC algorithm was used.

Table 1. Experimental results on digits

Experiment	# Features	Training	Testing
Initial configuration of the feature set	132	99.77%	99.10%
OrHC (using *val*$_3$)	92	99.50%	99.04%
OpHC (using *val*$_3$)	87	99.95%	98.95%
OrHC (smallest feature set)	81	99.50%	98.92%
OpHC (smallest feature set)	77	99.86%	98.94%

In order to compare our results with those obtained by Soares [8], we have used the same experimental protocol. The proposed wrapper approach has shown a more significant reduction (42%) than that observed by using GA (25%). Moreover, while the wrapper-based approach proposed by Soares [9] needs to evaluate around 128,000 solutions running in a cluster with 17 machines, the proposed method evaluates just 16,000 solutions using a Pentium III (1.3 GHz).

3.2 Experiments on Handwritten Characters

The method was also applied to handwritten uppercase characters available in the NIST SD19 database. During the experiments, just the optimized HC was used. However, the experiments considers to select the final feature set configuration taking into

Table 2. Experimental results on uppercase characters

Experiment	# Features	Training	Testing
Initial configuration of the feature set	132	97.23%	93.05%
OpHC (validation3)	101	97.57%	92.51%
OpHC (smallest feature set)	79	97.68%	92.50%

account the recognition rate on *val*$_3$ dataset, and also it considers the more significant reduction on the number of features.

The reduction of feature was about 24% in the first experiment and about 40% in the last one.

4 Conclusion

The proposed method has shown to be an interesting strategy to implement a wrapper-based method, which can be executed in low-cost hardware architecture. In the experiments, a reduction in the original number of features 132 to 77 features (around 42%) without a significant loss in terms of digit recognition rate was observed. Similar result was observed for uppercase characters, reduction around 40%.

References

1. Jain, A., Duin, R. P. W., Mao, J.: Statistical Pattern Recognition: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22(1). (2000) 4-37
2. Boz, O.: Feature Subset Selection by Using Sorted Feature Relevance. ICMLA International Conference on Machine Learning and Application, Las Vegas City, (2002)
3. Molina, L. C., Belanche, L., Nebot, À.: Feature Selection Algorithms: Survey Experimental and Evaluation. IEEE International Conference on Data Mining, Vol. 1. Maebashi City (2002) 306-313
4. Blum, A. L., Langlev P.: Selection of Relevant Features and Examples in Machine Learning. special issue Artificial Intelligence, Vol. 97. N.1-2 (1997) 245-271
5. Aha, D. W., Bankert, R. L.: Feature Selection for Case-based Classification of Cloud Types: an empirical comparison. Working Notes of the AAAI, Workshop on Case-Based Reasoning, Vol. 1. (1994) 106-112
6. Raman, B., Ioerger, T. R.: Enhancing Learning Using Feature and Example Selection. Journal of Machine Learning Research, submitted for publication, (2003)
7. Kohavi, Ron, John, George H., Wrappers for Feature Subset Selection. in Artificial Intelligence Journal, Special Issue On Relevance, Vol. 97. N. 1 (1997) 273-324
8. Oliveira, L. S., Sabourin, R., Bortolozzi, F., Suen, C. Y.: Feature Selection Using Multi-objective Genetic Algorithms for Handwritten Digit Recognition. Proc. International Conference on Pattern Recognition, Vol. 1. Quebec City (2002) 568-571
9. Oliveira, L.S., Sabourin, R., Bortolozzi, F., Suen, C.Y.: A Methodology for Feature Selection Using Multi-objective Genetic Algorithms for Handwritten Digit String Recognition. in the International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), Vol. 17. N. 6 (2003) 903-930

Feature Shaving for Spectroscopic Data

Serguei Verzakov, Pavel Paclík, and Robert P.W. Duin

Information and Communication Theory Group
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands
{s.verzakov,p.paclik,r.p.w.duin}@ewi.tudelft.nl

Abstract. High-resolution spectroscopy is a powerful industrial tool. The number of features (wavelengths) in these data sets varies from several hundreds up to a thousand. Relevant feature selection/extraction algorithms are necessary to handle data of such a large dimensionality. One of the possible solutions is the SVM shaving technique. It was developed for applications in microarray data, which also have a huge number of features. The fact that the neighboring features (wavelengths) are highly correlated allows one to propose the SVM band-shaving algorithm, which takes into account the prior knowledge on the wavelengths order. The SVM band-shaving has a lower computational demands than the standard SVM shaving and selects features organized into bands. This is preferable due to possible noise reduction and a more clear physical interpretation.

1 Introduction

In pattern recognition, objects are usually described by a number of features. Not all of them are equally informative for the problem at hand. Therefore, feature selection is an important step in solving a classification problem. It simplifies the classification task offering a faster and cheaper solution and, moreover, it allows to improve the classification performance by avoiding the curse of dimensionality. Feature selection methods can be divided into two groups: a) univariate approaches, where each single feature is tested for its ability to discriminate between classes and b) multivariate approaches, where all features are ranked according to some criterion, which takes all of them into account at once. Univariate approaches are simple to implement, but multivariate approaches give better results as they take into account the feature dependences.

Recently, a number of feature selection methods under the name of 'shaving' have been developed. Among them there are the SVM shaving [1] and the PCA shaving [2, 3]. Shaving approaches are similar but not equivalent to the backward feature elimination technique. In general, shaving algorithms remove a small portion of features at each step based on some criterion calculated on all the features available at that moment. During the backward feature elimination, an importance of each feature is estimated according to a criterion calculated on the feature set present at that moment but this particular feature. The backward feature elimination algorithm should theoretically provide better results than shaving. Yet, the shaving approach is much faster, while it is still able to provide a good solution.

Originally, shaving methods were applied to the microarray data. It has been shown that shaving techniques are useful in finding a small group of features (genes) significant for discrimination in high dimensional microarray data [1–3]. In our research, we are dealing with the problem of spectra classification. Based on the number of channels in the spectrometer, the feature sizes may vary from several hundreds up to a thousand. It is natural to try to apply these methods to the high-resolution spectral data. However, there is an important difference between the spectral data and gene arrays. In spectroscopy, neighboring features (wavelengths) are often highly correlated (more than 90 percent). It makes shaving methods first detect these correlations and only then the actual feature selection starts. In the case of small training set, the estimation of the local correlations may be very imprecise. This leads to a waste of the computational time and it may also result in loss of important features. The outcomes of shaving methods are the sets of original features. However, it is more natural for spectroscopy to select continuous bands of wavelengths and derive (possibly weighted) average representative feature from each band. Such features are easily interpretable from the physical point of view and also more robust to a change of the measurement device.

Another family of feature extraction/selection approaches under the name of GLDB was proposed in [4]. There, the neighboring wavelengths are combined into one feature based on log-odds class posterior probabilities (top-down approach) or on a product of the Fisher criterion and correlation between the features (bottom-up approach). Although these methods take features dependencies into account, all the criteria are applied to the each single region of wavelengths. Thus this family of methods is not fully multivariate.

In this paper, we propose a use of modification of the SVM shaving algorithm, *SVM band-shaving*, which makes use of specific properties of spectral data. Briefly, we combine neighbouring wavelengths into bands at first and then apply the shaving algorithm to them. The paper is organized as follows. In the section 2 we shortly describe the SVM shaving algorithm and our modification of it. Then, in section 3 we present the results of numerical experiments and we summarize with a short conclusion in section 4.

2 Shaving Algorithms

All shaving algorithms rely on a computation of the ranking vector $w \in R^d$, where d is the number of features. The absolute value of the element $w(i)$ estimates the importance of i-th feature e.g. for a discrimination task. After removing the least important feature or some portion of such features, the algorithm recalculates the weight vector for the reduced feature set. Recalculation on the data set with reduced dimensionality is desirable or even necessary, since the algorithm starts from the complete set of features, for which the estimated importance values can be very imprecise due to the curse of dimensionality. The algorithm continues the reduction of feature set size until the specified number of features is reached. It is also possible to estimate the classification error at each step on the current feature set and choose the one which offers a good tradeoff between the small number of features and an acceptable classification error.

It is possible to use as a ranking vector the weight vector w of a linear classifier

$$f(x) = sgn((w, x) + b) \tag{1}$$

In [1], the usage of the SVM classifier was proposed due to its reputation of being robust to the curse of dimensionality and being able to provide better estimations at early steps of shaving.

In [1], the SVM shaving was applied to microarray data. The relations between gene expression levels are either unknown or very complicated. In our case, we have extra prior information about spectra: the order of the features (wavelengths) is meaningful. The spectral values of the neighboring features are typically highly correlated (of course, this is only true for data with a sufficiently high spectral resolution). We use the word *connectivity* to name this property of spectral data sets.

The use of any additional information about a data set is similar to (but more specific than) regularization and in a similar way can help to reach better generalization abilities. In this article, we propose a modification of the SVM shaving technique which takes into account connectivity in the feature set. First we combine features into continuous groups (bands). For this purpose, we use absolute values $|w(i)|$ of the weights $w(i)$ obtained by training linear SVM on all the features. The bands are separated from each other by local minima of $|w(i)|$. To find local minima we estimate the first and the second derivates of $|w(i)|$ using Savitsky-Golay filter [6] with the second order polynomials. By averaging data in each band we create a new feature set to which the standard SVM shaving algorithm will be applied. The small number of features in the new feature set allows us to remove them one by one instead of removing some percentage of them. In all cases we use the ν-SVM algorithm [5], because its parameter ν has a more convenient interpretation (the estimation of the classification error) than the parameter C of the standard C-SVM algorithm.

As input parameters of the algorithm, additionally to the ν parameter of SVM, the minimum number of bands (stopping criterion) and the size of smoothing window for Savitsky-Golay filter should be specified. The selection of the meaningful minimum number of bands is a matter of the experiment and can be only roughly estimated beforehand e.g. as the number of significant principal components. It is worth to notice that classes can overlap substantially for the small numbers of bands which leads to the long execution times of SVM routines. All these problems are also present in the standard SVM shaving. The size of smoothing window should be selected as a largest interval on which $|w(i)|$ (or spectra themselves) can be well fitted by the second order polynomials for any i.

The pseudo-code of the proposed algorithm is presented below:

Input:

```
the training data set $D$,
the complete feature set $F$,
the parameter $\nu$ of the $\nu$-SVM classifier,
minimum number of bands $min\_bn$,
maximum smoothing window size $max\_ws$.
```

Output:

```
the sequence of feature sets $F_1 \supset ... \supset F_n$.
```

Algorithm:

```
1. Calculate the weight vector $w$ for the full feature set $F$
   using $\nu$-SVM algorithm.
2. Calculate the absolute values of the weight vector elements
   $w_a(i) = |w(i)|$.
3. Find the set of bands $B = \{b_1, ..., b_m\}$ which are separated by the
   minima in $w_a(i)$. Use Savitsky-Golay [6] algorithm with the
   second order polynomials and with the smoothing window less or
   equal to $max\_ws$ to estimate the first and the second
   derivatives of $w_a$.
4. Create a new feature set $F_1$ such that each feature $z(i)$ is a
   signed mean value of features $x(j)$ which belong to the band $b_i$.
```

$$z(i) = \frac{1}{|b_i|} \sum_{j \in b_i} sgn(w(j)) * x(j) \tag{2}$$

```
5. Perform the standard SVM shaving (using $\nu$-SVM algorithm) on $F_1$
   removing each time one band producing the sequence of feature
   sets $F_1 \supset F_2...$ until no more than $min\_bn$ bands left.
```

One can use a validation data set to estimate the classification error on the resulting sequence $F_1 \supset ... \supset F_n$ to judge which subset has the smallest number of bands while yielding a suitable performance.

3 Numerical Experiment

For a demonstration of our algorithm we use the data from the CD of [7]. This is a 191-channel airborne multispectral scanner data set which contains a hyperspectral image of Washington DC Mall. The sensor system used in this case measured a response in 0.4 to 2.4 μm region of the visible and infrared spectrum. The task is to discriminate between seven classes of pixels: Roofs, Roads, Paths, Trees, Grass, Water and Shadows. We demonstrate our algorithm on the Roofs/Paths classification. For our calculations we have selected 30 spots of each

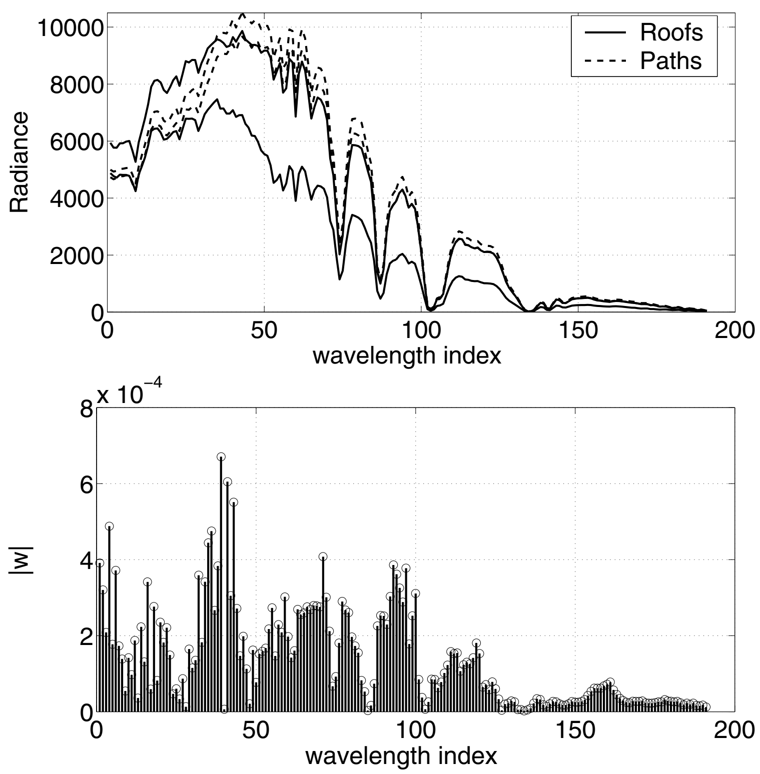


Fig. 1. Upper plot: lower and upper quartiles (25% and 75% levels of a distribution) for each wavelength for both classes. Radiance units are arbitrary. Bottom plot: absolute values of the weights $w(i)$ resulting from training of the SVM classifier on a feature set containing all 191 wavelengths.

class. Each spot consists of 9 pixels. The spots were manually selected to guarantee a representative examples and placed faraway from each other. We used a 5-fold cross-validation to estimate classification errors. The parameter $\nu = 0.05$ of ν-SVM algorithm was selected after a few probe runs and proved to be a good choice. The values of ν greater than $\nu = 0.05$ lead to larger classification errors due to the insufficient penalizing of the classification error. At smaller values, the solutions of the SVM problem start to show early signs of overtraining with the decreasing of the number of features in the shaving procedure. This happens, because in low dimensional data sets classes start to overlap, so unreasonably high penalization of margin errors (ν is much smaller than Bayes error) leads to narrow margin and a bad generalization ability. See [8] for more details.

In Fig. 1, the spectra of both classes are shown, as well as the result of SVM applied to the non-reduced feature set. After a few experiments, we have selected the upper limit for the smoothing window $max\_ws = 11$. The result of the band extraction is shown in Fig. 2.

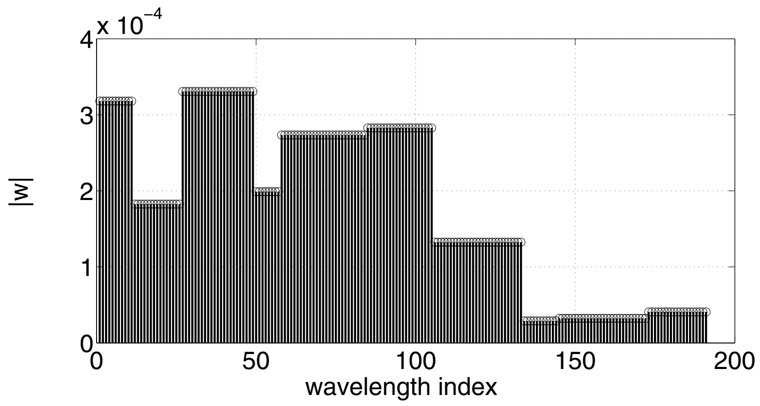


Fig. 2. Absolute values of the weights $w(i)$ after the step 4 of the SVM band-shaving algorithm (band extraction). The cumulative weight of each band is equally distributed among features from this band.

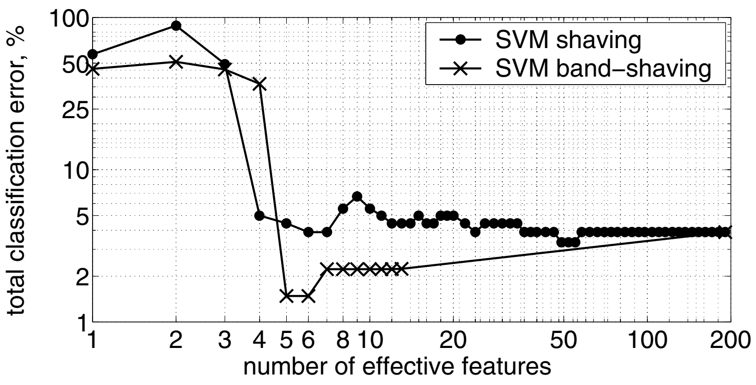


Fig. 3. The classification errors for the feature sets selected by shaving algorithms. The x-axis represents the number of effective features, i.e. the dimensionalities of spaces in which classifiers were trained and tested.

The total classification errors on the feature sets selected by the standard SVM shaving and the SVM band-shaving are shown in Fig 3. Both methods start from the same entire feature set (191 original features). Then, the standard procedure gradually removes the least important features by portions of 5% of the remaining features. The classification error remains almost the same during the shaving. It reaches minimum at 49 features. The number of features equal to 6 seems to be an optimal choice because of a significant dimensionality reduction (from 191 to 6 features) and still a low classification error.

The absolute values of weights $w(i)$ of the SVM trained on the selected 6 features are shown in Fig. 4. The classification performance for the number of features less than 4 is very bad due to overlap.

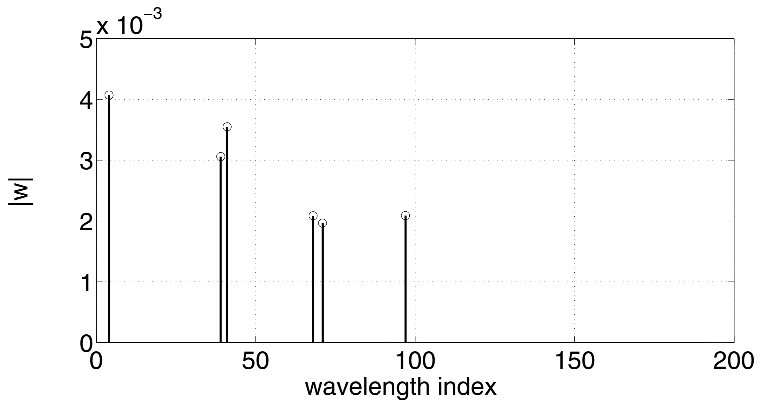


Fig. 4. Absolute values of SVM weights $w(i)$ which are the result of the training SVM classifier on a feature set containing 6 features selected by the SVM shaving procedure.

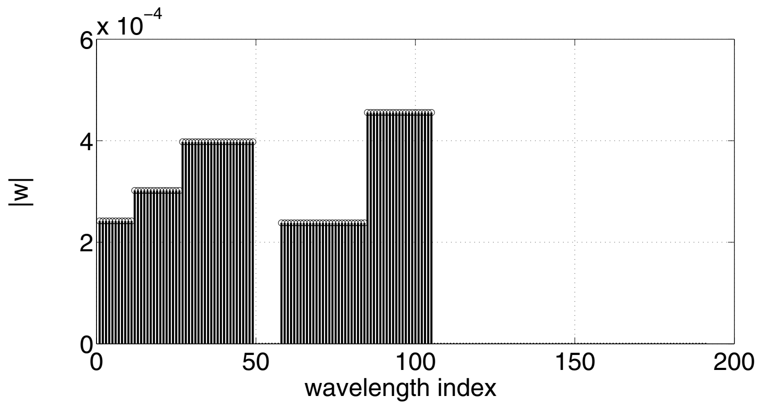


Fig. 5. The absolute values of SVM weights $w(i)$ trained on 5 bands selected by SVM band-shaving. The weight of each band is equally distributed among features from this band.

The SVM band-shaving immediately jumps from the entire feature set to a feature set containing only approximately 10 features. This number varies slightly in each cross-validation run. The results suggest that only by combining the features into the bands, the classification performance can be improved. Moreover, this performance can be better than one of a classifier trained on the same number of features selected by the standard procedure. During the removing the least important bands, the classification error becomes smaller. It reaches the minimum at the number of bands equal to 5 (see Fig. 5). At lower numbers of bands results show the clear signs of a substantial overlap between classes.

4 Conclusion

We proposed a variant of the SVM shaving algorithm, the SVM band-shaving, which takes into account the connectivity property of the spectra. The conducted experiment shows that our algorithm may outperform the standard SVM shaving technique. The SVM band-shaving removes the whole band at once. Thus the number of retrainings of the classifier is smaller than in the standard SVM shaving procedure. On the other hand, our algorithm requires the specification of an additional parameter: the maximum size of smoothing window. A few runs of the whole procedure or an expert knowledge on the nature of data are necessary to select a proper value of this parameter. It is also worth to mention that 5 bands selected by the SVM band-shaving contain in total about 90 original features (wavelengths). So the application of some band shrinking algorithm would be useful. Our results, although preliminary, are very encouraging. We plan to study these techniques further on and apply them to other data sets.

Acknowledgments

For our computations, we used OSU SVM by J. Ma, Y. Zhao, and S. Ahalt [9]. We would like to thank E. Pekalska for the helpful comments on the manuscript. This work was supported by the Dutch Foundation for Applied Sciences (STW).

References

1. I. Guyon, J. Weston, S. Barnhill and V. Vapnik: Gene Selection for Cancer Classification using Support Vector Machines. Machine Learning **46**(13), 389-422, 2002.
2. T. Hastie, R. Tibshirani, et al.: Gene Shaving: a New Class of Clustering Methods for Expression Arrays. Technical report, Department of Statistics, Stanford University, 2000. `http://www-stat.stanford.edu/~hastie/Papers/`.
3. T. Hastie, R. Tibshirani, et al.: "Gene shaving" as a method for identifying distinct sets of genes with similar expression patterns. Genome Biology **1**(2), research0003.1–0003.21, 2000.
4. S. Kumar, J. Ghosh, J. and M. M. Crawford: Best-Bases Feature Extraction Algorithms for Classification of Hyperspectral Data. IEEE Transactions on Geoscience and Remote Sensing **39**(7), 1368-1379, 2001.
5. B. Schölkopf, A. Smola, R.C. Williamson, and P.L. Bartlett: New support vector algorithms. Neural Computation **12**, 1207-1245, 2000.
6. A. Savitzky and M. J. E. Golay: Smoothing and Differentiation of Data by Simplified Least Squares Procedures. Analytical Chemistry **36**(8), 1627-1639, 1964.
7. D. Landgrebe: Signal theory methods in multispectral remote sensing. John Wiley & Sons, 2003.
8. V. Vapnik: The Nature of Statistical Learning Theory. Springer: New York, USA, 2000.
9. J. Ma, Y. Zhao, S. Ahalt: OSU SVM Classifier Matlab Toolbox (version **3.00**), Ohio State University, Columbus, USA, 2002.
`http://www.ece.osu.edu/~maj/osu_svm/`.

Feature Selection by Markov Chain Monte Carlo Sampling – A Bayesian Approach

Michael Egmont-Petersen

Institute of Information and Computing Sciences, Utrecht University,
Padualaan 14, De Uithof, The Netherlands
Michael@cs.uu.nl

Abstract. We redefine the problem of feature selection as one of model selection and propose to use a Markov Chain Monte Carlo method to sample models. The applicability of our method is related to Bayesian network classifiers. Simulation experiments indicate that our novel proposal distribution results in an ignorant proposal prior. Finally, it is shown how the sampling can be controlled by a regularization prior.

1 Introduction

The problem of feature selection has been targeted regularly in publications appearing in the literature on datamining and statistical pattern recognition. Important key references include [1–4]. Whether one wants to learn a statistical classifier from data, a graphical model or perform clustering in high-dimensional space, confining the number of feature variables included in the model by either feature selection or feature transformation, is often necessary. Inclusion of too many feature variables leads to over-fitting. Within the context of feature selection, over-fitting causes the so-called *peaking phenomenon* to occur [5]. Peaking refers to the fact that the performance of a statistical model (e.g., the error rate of a classifier) on an independent test dataset generally peaks when utilizing only a *subset* of the available feature variables. This is counterintuitive, as adding extra non-informative feature variables to a statistical model should not, intuitively, lead to a performance decrease. However, as the model parameters are estimated from a training dataset of a finite size, variance associated with the parameter estimates leads to fitting random variations of the non-informative features and hence to a decrease in performance.

The problem of feature selection is more complex than often stated in the literature, because the use of different feature subsets inevitably imposes different models (e.g., a different topology of a neural network [6]). Hence, feature selection implies model selection. Model selection is a complex problem that, even in the simple case where models are compared with *one* assessment criterion (e.g., the likelihood of the model, its classification error rate or its residual variance), entails a trade-off between bias and variance. On the one hand, allowing the inclusion of a large number of features/parameters, may lead to an accurate model. However, the parameters will because of their large variance result in a model performance that is prone to noise. On the other hand, limiting the number of parameters is more likely to bias model performance, but it results in less variance. In this paper, we suggest to sample different models from a statistical distribution in order to make such trade-offs explicit. So instead of looking for "that one particular model with the best performance", we propose to sample the posterior distribution of models using a Markov Chain Monte Carlo method [7, 8].

2 Background

Feature selection has been approached within statistical pattern recognition at an early stage. In 1971, it was conjectured that the peak in performance (1 – error rate) of statistical classifiers solely occurred when the features were dependent [9]. Later, Trunk [10] managed to prove that even for n_{all} independent normally distributed feature variables, peaking can occur when $n_{all} \to \infty$. It is furthermore clear that increasing the size m of the training dataset solely shifts the position of the peak, allowing the model to utilize an increasing number of feature variables. Also within multivariate statistics, approaches for variable selection for linear regression [11], linear and quadratic discriminant analysis [12] have been developed.

Algorithms for feature selection rely on a *search scheme* and an *assessment criterion* $J_s(X, n)$ for comparing feature subsets. Within the pattern recognition literature, much research focused on search schemes [4, 13] and assessment criteria J_s [6]. Solely exhaustive search is guaranteed to result in an optimal feature subset with the maximal score $J_s(X, n)$ on a test set, when the assumption of monotony of $J_s(X, n)$ with respect to an increase in n does not hold [5].

Conclusively, three interrelated obstacles impede efficient feature selection: the peaking phenomenon, the combinatorial complexity of exhaustive search resulting in $2^{n_{all}} - 1$ nonempty feature subsets and the fact that feature selection entails model selection.

3 A Formalism for Model and Feature Selection

We formalize the joint problem of feature and model selection. In the sequel, individual stochastic variables are denoted with capital letters $A, B, \ldots$, sets of stochastic variables with bold capital letters, $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$. Outcomes of the variables are denoted with small letters, e.g., $P(A = a)$. Correspondingly, $P(\boldsymbol{X} = \boldsymbol{x})$ denotes the probability that the variables $\boldsymbol{X}$ have the outcomes $\boldsymbol{x}$. In general, we use $\boldsymbol{X}$ to denote the set of feature variables and C the outcome variable. The statistical model M consists of a structural part represented by the graph G and a set of parameters $\boldsymbol{\theta}$, $M = (G, \boldsymbol{\theta})$. The model is used to implicitly estimate the class-conditional probability distribution $P(\boldsymbol{X} \mid C, M)$, when the variables $\boldsymbol{X}$ are discrete and the density $p(\boldsymbol{X} \mid C, M)$ when the variables $\boldsymbol{X}$ are continuous. For simplicity, we henceforward solely address the situation where C is *discrete*. With $\boldsymbol{D}$, we denote a training database with m training instances, $\boldsymbol{D} = (\boldsymbol{d}_1, \ldots, \boldsymbol{d}_m)$. Each instance $\boldsymbol{d}$ is represented by the discrete feature vector $\boldsymbol{x}$ and class label c, yielding $\boldsymbol{d} = (\boldsymbol{x}, c)^T$.

The search for the best feature subset can be illustrated with the lattice graph introduced in [14]. Let $G_U = (\boldsymbol{V}, \boldsymbol{E})$ be an *undirected* graph with $\boldsymbol{V}$ indicating the vertices and $\boldsymbol{E}$ the edges connecting pairs of vertices: $E_{ijk} = 1$ if the vertices V_i and V_j are connected (indicating feature subsets with a Hamming distance of 1), with k the model index, whereas $E_{ijk} = 0$ when V_i and V_j are not connected in model k. Figlure 1 shows an example of a nested lattice structure that represents both feature subsets and models. It is clear that model selection can be performed separately, for a given feature subset, but that feature selection necessitates model selection.

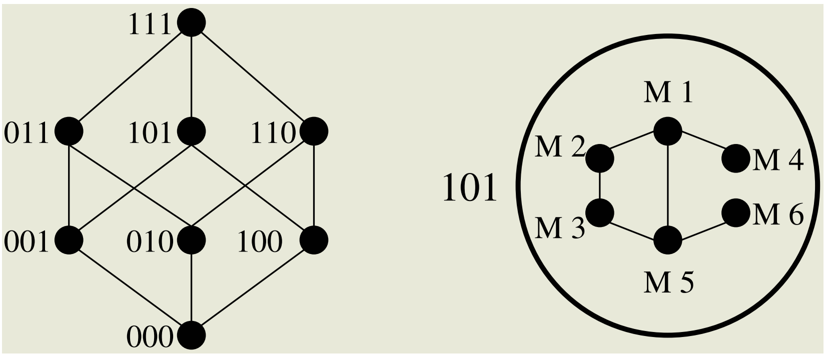


Fig. 1. (a) Lattice structure connecting the feature subsets with a feature set Hamming distance of 1. (b) A close-up of the *model subspace* associated with feature subset '101' reveals that several alternative models ($M1 - M6$) may be applied to the feature subset 101. The edges indicate models with a model Hamming distance of 1. The model subspace associated with the empty set 000 below (not shown) contains solely one model, namely the distribution of the predicted variable C in the training set.

The nested lattice structure depicted in figure 1, with an *upper layer* representing the feature subsets and a *lower layer* representing the possible models that utilize the associated feature subset, is unnecessary complex to work with. Instead, we redefine the feature selection problem as one of model selection. Consequently, we propose considering the score J_s as a random variable and set as goal to sample the joint distribution

$$p(J_s, \boldsymbol{Y}, \hat{M}|\boldsymbol{D}), \;\; \boldsymbol{Y} \in \mathcal{P}(\boldsymbol{X}), \; M \in \mathcal{M} \tag{1}$$

where the score J_s is a continuous stochastic variable. The generic score function J_s may indicate, for example, the likelihood $s = L$, the Bhattacharyya distance $s = \mu$, the error rate $s = \epsilon$, or another measure depending on how the model M should be scored for the particular application at hand. The hat-notation indicates that the parameters $\boldsymbol{\theta}$ of the model M have been estimated from the training set, but they may also be integrated out, see e.g. [15]. $\mathcal{P}(\boldsymbol{X})$ denotes the power set of $\boldsymbol{X}$ and $\mathcal{M}$ the set of valid models that can be learned from it.

4 Markov Chain Monte Carlo Sampling

We will use the Metropolis-Hastings algorithm [16] to perform Markov Chain Monte Carlo sampling from a *target probability function Π*. More specifically, we propose to sample the distribution $P(J, \boldsymbol{Y}, \hat{M}|\boldsymbol{D})$. The Metropolis-Hastings algorithm results in a discrete Markov Chain over a state space, $\mathcal{S}$. The transition probabilities, $P_q(S_k \rightarrow S_l)$, $S_k, S_l \in \mathcal{S}$ specify the probability of making a jump from state S_k (associated with model k) to state S_l.

4.1 Probabilistic Network Classifiers

Markov Chain Monte Carlo techniques can be used to sample the posterior distribution of different types of classifiers. Here, we illustrate MCMC by probabilistic network classifiers [17, 18]. A probabilistic network classifier matches the general description of a statistical model given in Section 3, see Fig. 2.

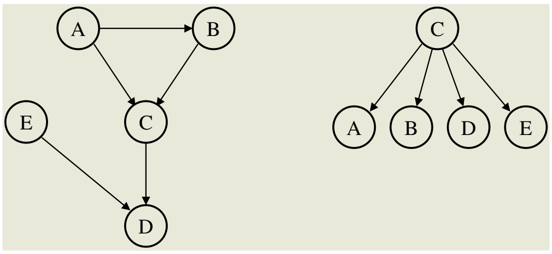


Fig. 2. (a) Directed acyclic graph specifying the direct dependencies in a Bayesian network classifier with 4 feature variables. The chain rule (Eq. (2)) specifies how the joint probability factorizes: $P(A, B, C, D, E) = P(A)P(B|A)P(C|A, B)P(D|C, E)P(E)$. The variables A and B are independent from D and E, given the class label C. (b) Naive Bayesian classifier where the feature variables are independent, given the class label C. The joint probability factorizes into: $P(A, B, C, D, E) = P(A|C)P(B|C)P(C)(D|C)P(E|C)$.

A probabilistic network classifier $M = (G, \boldsymbol{\theta})$ consists of a structural model specification, the directed graph G, and the parameters, $\boldsymbol{\theta}$, with the (un)conditional probability $\theta_{i,j,\pi(i)} = P(D_i = d_j \mid \pi(D_i) = \boldsymbol{d}_{\pi(D_i)})$. The notation $\pi(D_i) = \boldsymbol{d}_{\pi(D_i)}$ indicates the values of the parents of node D_i in the graph G (the parents constitute the nodes with arcs pointing directly to node D_i). Computation of the posterior probability distribution $P(C = c|\boldsymbol{X} = \boldsymbol{x})$ is specified by the directed graph. It follows from the chain rule that the joint probability $P(\boldsymbol{d}) = P(c, \boldsymbol{x})$ is computed from

$$P(\boldsymbol{d}) = \prod_{i=1}^{n+1} P(D_i = d \mid \pi(D_i) = \boldsymbol{d}_{\pi(D_i)}) \tag{2}$$

(see Fig. 2). A little manipulation of Bayes formula yields the posterior probability associated with class label c_j (where we omit denoting the variables)

$$P(c_j|\boldsymbol{x}) = \frac{P(c_j, \boldsymbol{x})}{\sum_k P(c_k, \boldsymbol{x})} \tag{3}$$

4.2 Approach by Madigan & York

Madigan & York have earlier presented an approach for sampling graphical models, based on the Markov Chain Monte Carlo scheme [8]. In their approach, the goal is to sample graphical models $G \in \mathcal{G}$ from the posterior distribution, $P(G \mid \boldsymbol{D})$. The likelihood $L(G \mid \boldsymbol{D}) \propto P(\boldsymbol{D} \mid G)$ with $P(\boldsymbol{D} \mid G)$ the probability that a particular structural model G results in the dataset $\boldsymbol{D}$.

Madigan & York define a homogeneous, stationary and reversible Markov chain. This chain specifies the transition probabilities, $P_q(G \rightarrow G\prime)$, that a jump is made from the model G to the model $G\prime$. The possible jumps from G to $G\prime$ consist of all acyclic graphs that can be constructed by adding one arc to G or by deleting one arc from G. Hence, $G\prime \in NB(G)$, the neighborhood of G defined by

$$NB(G) = \left\{ \begin{array}{ll} \bigcup_{i \,\in\, I(D)}, \bigcup_{j \,\in\, (I(D)\setminus i \,\mid\, (X_i \rightarrow X_j) \,\notin\, G)} & Add_{AC}(G, (X_i \rightarrow X_j)) \;\bigcup \\ \bigcup_{i \,\in\, I(G)}, \bigcup_{j \,\in\, (I(G)\setminus i \,\mid\, (X_i \rightarrow X_j) \,\in\, G)} & Del(G, (X_i \rightarrow X_j)) \end{array} \right\} \tag{4}$$

with $I(D)$ denoting the indices of the nodes representing all $n + 1$ variables, and Add_{AC} a function that adds an arc to G *iff* the resulting graph $G\prime$ is *acyclic*. Together, the requirements of homogeneity and reversibility and the fact that the Markov Chain is stationary, make it feasible to use the transition kernel (proposal distribution) $q(G \rightarrow G\prime)$ in the Metropolis-Hastings algorithm [8]. The *proposal probability* $P_q(G \rightarrow G\prime) = |NB(G)|^{-1}$, whereas the probability of the reverse proposal is $P_q(G\prime \rightarrow G) = |NB(G\prime)|^{-1}$. The *transition probability* $P(G\prime|G)$ is modelled as $P(G\prime|G) = P_q(G \rightarrow G\prime)\alpha(G, G\prime)$, $G \neq G\prime$. The detailed balance, which ensures reversibility, is obtained by using the normalization factor $\alpha(G, G\prime)$

$$\alpha(G, G\prime) = \min\left[1\,,\ \frac{P(\boldsymbol{D}|G\prime)\,P(G\prime)}{P(\boldsymbol{D}|G)\,P(G)}\,\frac{P_q(G\prime \rightarrow G)}{P_q(G \rightarrow G\prime)}\right] \tag{5}$$

Sampling from the proposal distribution $q(G \rightarrow G\prime)$ and normalising by $\alpha(G, G\prime)$ result in a posterior distribution $P(G|\boldsymbol{D})$ where more likely models appear more often than unlikely ones.

4.3 Naive Bayes classifiers

A special type of probabilistic network classifiers are the naive Bayesian classifiers, see Fig. 2 (b). MCMC can be modified to sample naive Bayesian classifiers. Because no model selection takes place in this simple case, each node in the lattice graph (Fig. 1 (a)) contains solely one model. Instead of using the likelihood $P(\boldsymbol{D}|G)$ as assessment criterion, we suggest to use the general criterion J_s, which may be the likelihood, 1–error rate, or another metric that measures discriminative performance. We define the *add-one-delete-one neighborhood* to include only Naive Bayes classifiers

$$NB_I(G) = \left\{ \begin{array}{ll} \bigcup_{i \,\in\, I(X) \,\mid\, (C \rightarrow X_i) \,\notin\, G} & Add(G, (C \rightarrow X_i)) \;\bigcup \\ \bigcup_{i \,\in\, I(G) \,\setminus\, C} & Del(G, (C \rightarrow X_i)) \end{array} \right\} \tag{6}$$

Following the approach by Madigan & York, it would be natural to set the proposal probability to $P_q(G \rightarrow G\prime) = |NB_I(G)|^{-1}$. However, such a choice leads nonuniform proposal distribution $P(N)$ with respect to the size of the feature subsets that are compared. The following theorem formalizes this

Theorem 1 *A Markov Chain Monte Carlo scheme for proposing feature subsets, where each subset in the add-one-delete-one neighborhood NB_I has the same probability of being proposed, $P_q(G \rightarrow G\prime) = |NB_I(G)|^{-1}$, this results in a proposal probability imposing a nonuniform prior $P(N)$ that is maximal for $N = n, n \in \{n_{all}/2 - 1, n_{all}/2, n_{all}/2 + 1\}$, depending on whether n_{all} is even or odd.*

Proof (sketch)
It follows from the Binomial theorem that the number of size $n \in \{1, \ldots, n_{all}\}$ feature subsets of n_{all}, is $\binom{n_{all}}{n}$. As $\binom{n_{all}}{n+1} \geq \binom{n_{all}}{n}$, $n < n_{all}/2$ because $|I(\boldsymbol{X}),\ (C \rightarrow X_i) \notin G| \geq |I(G) \setminus C|$ whereas $\binom{n_{all}}{n+1} \leq \binom{n_{all}}{n}$, $n > n_{all}/2$ because $|I(\boldsymbol{X}),\ (C \rightarrow X_i) \notin G| \leq |I(G) \setminus C|$, it follows that the prior $P(N)$ is maximal for feature subsets with a size $n_{all}/2$.

We will instead use a proposal distribution, that results in each size n having the same (uniform) probability. Establish the partitioning of $NB_I(G)$ into two *disjoint* subsets, $NB_I(G) = \{NB_{IG(+1)}, NB_{IG(-1)}\}$, where $NB_{IG(+1)}(G)$ is the subset of graphical models in $NB_I(G)$ that results from *adding* one arc to G, and $NB_{IG(-1)}(G)$ is the subset of graphical models in $NB_I(G)$ that results from *deleting* one arc from G. We now suggest a two-step proposal distribution q: Draw a uniformly distributed number $u \sim \mathcal{U}(0,1)$. If $u \geq \frac{1}{2}$ then choose a model in $NB_{IG(+1)}$ with the probability $|NB_{IG(+1)}|^{-1}$, otherwise choose a model in $NB_{IG(-1)}$ with the probability $|NB_{IG(-1)}|^{-1}$. The normalization factor that ensures detailed balance, becomes

$$\alpha(G, G\prime) = \min\left[1\ ,\ \frac{P(J|G\prime)\, P(G\prime)}{P(J|G)\, P(G)} \frac{P_q(G\prime \rightarrow G)}{P_q(G \rightarrow G\prime)}\right] \tag{7}$$

with the proposal probability $P_q(G \rightarrow G\prime) = 1$ and

$$P_q(G\prime \rightarrow G) = \begin{cases} \frac{1}{2} : & n(G) = 0 \\ \frac{1}{2} : & n(G) = n_{\max} \\ 1 : & otherwise \end{cases} \tag{8}$$

where $n(G)$ indicates the number of features included in model G. P_q restores detailed balance with respect to the number of features in relation to the two end points, 0 and $n_{\max}$, of the proposal interval. The parameter $n_{\max} \leq n_{all}$ such that the maximal size of a feature subset can be limited. The resulting posterior distribution implies a non-informative (uniform) prior, $P(N)$, on size $N = n$ of any feature subset. We conduct a simulation experiment to illustrate the practical implication of Theorem 1.

4.4 General Bayesian Network Classifiers

We now extend our MCMC approach to sample general Bayesian network classifiers. Hence, the lattice graph (Fig. 1 (a)) represents both different feature subsets and models. Define the *one-step look ahead* neighborhood of the graph G consisting of the directed acyclic graphs resulting in valid probabilistic classifiers that can be constructed by adding one arc to G or deleting one arc from G

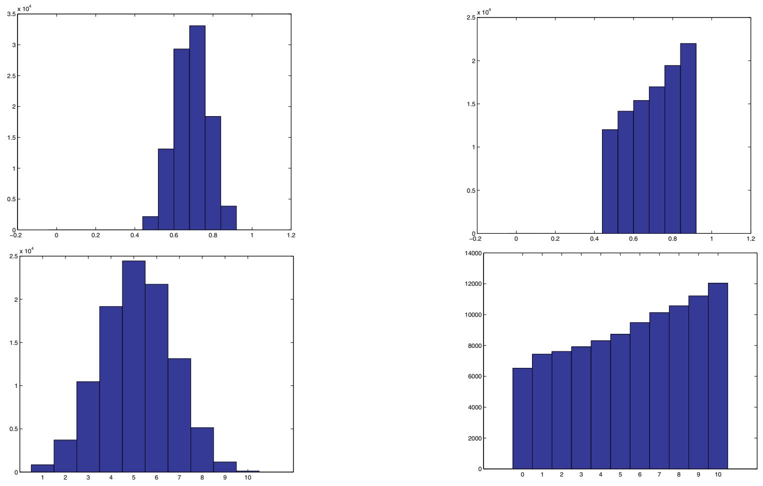


Fig. 3. (Upper left) Shows the posterior distribution of J for MCMC scheme 1. (Lower left) Shows the distribution of feature subsets for MCMC scheme 1. (Upper right) Shows the posterior distribution of J for MCMC scheme 2. (Lower right) Shows the distribution of feature subsets for MCMC scheme 2.

$$NB_C(G) = \left\{ \begin{array}{ll} \bigcup_{i \,\in\, I(\mathbf{X}),\ C \,\notin\, \sigma(X_i),\ C \,\notin\, \pi(X_i)} & Add(G,(X_i \rightarrow C)) \,\bigcup \\ \bigcup_{i \,\in\, I(\mathbf{X}),\ C \,\notin\, \sigma(X_i),\ C \,\notin\, \pi(X_i)} & Add(G,(C \rightarrow X_i)) \,\bigcup \\ \bigcup_{j \,\in\, I(\sigma(C))} \bigcup_{i \,\in\, I(\mathbf{X}) \,\setminus\, \pi(X_j)} & Add(G,(X_i \rightarrow X_j)) \,\bigcup \\ \bigcup_{i \,\in\, I(G)} \bigcup_{j \,\in\, I(G) \,\mid\, (X_i \rightarrow X_j) \,\in\, G} & Del(G,(X_i \rightarrow X_j)) \end{array} \right\} \quad (9)$$

with $\pi(X_i)$ the children of node X_i and $\sigma(X_i)$ the parents of node X_i. The function $I(\sigma(C))$ denotes the indices of the children of the classification node C. The neighborhood $NB_C(G)$ is subdivided into four disjoint subsets

$$NB_C(G) = \{NB_C(G+1_F), NB_C(G-1_F), NB_C(G+1_M), NB_C(G-1_M)\} \quad (10)$$

The subset $NB_C(G+1_F)$ contains the graphical models in $NB_C(G)$ where the addition of an arc implies that G' contains one feature variable more than G. The subset $NB_C(G-1_F)$ contains the models in $NB_C(G)$ where the deletion of an arc implies that G' contains one feature variable less than G. The subset $NB_C(G+1_M)$ contains the models in $NB_C(G)$ where the addition of an arc increases the complexity of G', but where G and G' include the same feature variables. $NB_C(G-1_M)$ contains the models in $NB_C(G)$ where the deletion of an arc decreases the complexity of G', but where G and G' include the same feature variables.

We define the proposal distribution q_C as follows:

$$q_C(G \rightarrow G') = \left\{ \begin{array}{ll} u < \frac{1}{4} & q_1(|NB_C(G+1_F)|^{-1}) \\ \frac{1}{4} \leq u < \frac{1}{2} & q_2(|NB_C(G-1_F)|^{-1}) \\ \frac{1}{2} \leq u < \frac{3}{4} & q_3(|NB_C(G+1_M)|^{-1}) \\ \frac{3}{4} \leq u & q_4(|NB_C(G-1_M)|^{-1}) \end{array} \right. \quad (11)$$

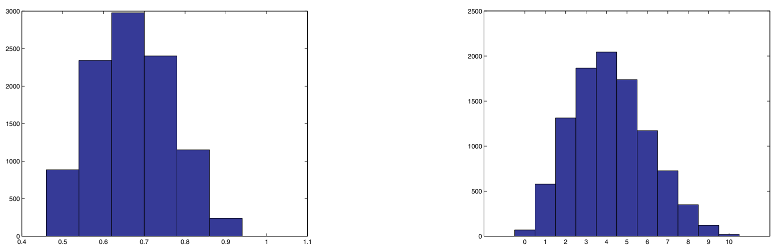


Fig. 4. (a) Shows the posterior distribution of J when using a Poisson prior, $P(G)$. The left tail contains models with the maximal performance 0.9. (b) Shows the distribution of feature subsets when the same prior is used.

with $u \sim \mathcal{U}(0,1)$. So in each proposal, the MCMC-algorithm with the same probability chooses to add a feature, delete a feature, increase the model complexity or simplify the model (the two latter moves keep the same feature subset).

5 Simulation Experiments

We performed a simulation experiment in order to compare the two sampling schemes proposed for the Naive Bayesian classifier in Section 4.3. We set the scoring metric $J(X) = 1 - \epsilon(X)$ and chose to simulate with a feature set of 10 features. Five of the (independent) features could each lead to an increase in $J(X)$ of 0.08, yielding a maximum of 0.9. The performance resulting from the empty feature set is 0.5. We sampled 100.000 feature subsets (naive Bayes classifiers) using the scheme based on the Madigan & York approach (scheme 1), and 100.000 using our novel proposal distribution (scheme 2).

Our second proposal distribution based on the neighborhood NB_I behaves as could be expected and support Theorem 1. The more features a subset contains, the higher the resulting score J will be. To cope with the curse of dimensionality, we experimented with using the discrete Posson distribution as prior, $P(N)$. Setting $\lambda = 4$, we obtained the results as shown in Fig. 4.

In our third experiment, we implemented the proposal distribution for general Bayesian network classifiers, Eq. (11). We sampled 100 training cases from the probability distribution specified by the graph in Fig. 2. MCMC was set to run for 1000 iterations. The simulation resulted in 757 nonempty feature sets. The most frequent nonempty feature set included all 5 features, and was found 166 times. The second most likely feature set consisting only of feature 1, was sampled 63 times. So the correct feature set was also most frequently sampled in the markov chain.

6 Discussion

We have presented a method for sampling statistical models in general, and pattern classifiers in particular, using an ignorant proposal distribution. It was shown how a regularization prior can be used to restrain the maximal dimensionality of the sampled models. Finally, we showed how general Bayesian classifiers can be sampled using the novel proposal distribution.

References

1. Foroutan, I., Sklansky, J.: Feature selection for automatic classification of non-gaussian data. IEEE Transactions on Systems, Man, and Cybernetics **17** (1987) 187–198
2. Kittler, J.: Computational problems of feature selection pertaining to large data sets. In: Proceedings of Pattern Recognition in Practice. (1980) 405–414
3. Jain, A., Zongker, D.: Feature selection: Evaluation, application, and small sample performance. IEEE transactions on pattern analysis and machine intelligence **19** (1997) 153–158
4. Kudo, M., Sklansky, J.: Comparison of algorithms that select features for pattern classifiers. Pattern recognition **33** (2000) 25–41
5. Waller, W.G., Jain, A.K.: On the monotonicity of the performance of a bayesian classifier. IEEE Transactions on Information Theory **24** (1978) 120–126
6. Egmont-Petersen, M., Talmon, J., Hasman, A., Ambergen, A.: Assessing the importance of features for multi-layer perceptrons. Neural networks **11** (1998) 623–635
7. Giudici, P., Castelo, R.: Improving markov chain monte carlo model search for data mining. Machine learning **50** (2003) 127–158
8. Madigan, D., York, J.: Bayesian graphical models for discrete-data. International statistical review **63** (1995) 215–232
9. Chandrasekaran, B.: Independence of measurements and the mean recognition accuracy. IEEE Transactions of Information Theory **17** (2002) 452–456
10. Trunk, G.: A problem of dimensionality: a simple example. IEEE Transactions of Pattern Analysis and Machine Intelligence **1** (1979) 306–307
11. Forsythe, A., Engleman, L., Jennrich, R., May, P.R.A.: A stopping rule for variable selection in multiple regression. Journal of the American Statistical Association **68** (1973) 75–77
12. McLachlan, G.J.: Discriminant analysis and Statistical Pattern Recognition. John Wiley & Sons, New York (1992)
13. Siedlecki, W., Sklansky, J.: On automatic feature selection. Journal of Pattern Recognition and Artificial Intelligence **2** (1988) 197–220
14. Siedlecki, W., Sklansky, J.: A note on genetic algorithms for large-scale feature selection. Pattern Recognition Letters **10** (1989) 335–347
15. Cooper, G.F., Herskovits, E.: A bayesian method for the induction of probabilistic networks from data. Machine learning **9** (1992) 309–347
16. Chib, S., Greenberg, E.: Understanding the metropolis-hastings algorithm. American statistician **49** (1995) 327–335
17. Baesens, B., Egmont-Petersen, M., Castelo, R., Vanthienen, J.: Learning bayesian network classifiers for credit scoring using markov chain monte carlo search. In: Proceedings of the International Conference on Pattern Recognition, Piscataway, IEEE Computer Society (2002) 49–52
18. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine learning **29** (1997) 131–163

Classifier-Independent Visualization of Supervised Data Structures Using a Graph

Hiroshi Tenmoto[1], Yasukuni Mori[2], and Mineichi Kudo[3]

[1] Kushiro National College of Technology,
Kushiro, Hokkaido 084-0916, Japan
tenmo@kushiro-ct.ac.jp
http://www.kushiro-ct.ac.jp/tenmo/
[2] Chiba University, Chiba 263-8522, Japan
yasukuni@faculty.chiba-u.jp
[3] Hokkaido University, Sapporo 060-8628, Japan
mine@main.eng.hokudai.ac.jp

Abstract. Supervised data structures in high dimensional feature spaces are displayed as graphs. The structure is analyzed by normal mixture distributions. The nodes of the graph correspond the mean vectors of the mixture distributions, and the location is carried out by Sammon's nonlinear mapping. The thickness of the edges expresses the separability between the component distributions, which is determined by Kullback-Leibler divergence. From experimental results, it was confirmed that the proposed method can illustrate in which regions and to what extent it is difficult to classify samples correctly. Such visual information can be utilized for the improvement of the feature sets.

1 Introduction

In usual pattern recognition systems, both construction of classifiers and classification of unknown samples are carried out in a high dimensional vector space, called "feature space." The space is spanned by the "features" extracted from raw patterns in the observation space. Therefore, the system's accuracy strongly depends on the feature extraction part. However, the extraction process cannot be generalized, so the system has to be improved heuristically by evaluating the extracted features.

For this purpose, many feature selection method have been proposed. They can find and remove redundant features, while they cannot illuminate what features should be added. Therefore, on the other hand, many trials of visualizations that illustrate the distributions of samples in the feature space have been carried out [1-7]. By using such methods, the following properties can be observed: (1) in what "shape" the samples distribute in the high dimensional space, (2) whether the class regions are separated from the others sufficiently or not, and (3) if it is not, in which regions the classes are not separated well. These matters may help us in improving the system's accuracy, and are also useful for superclass finding problem [8].

2 Visualization Methods

To achieve the purpose mentioned above, many "mapping" methods [1-7] can be applied. These mapping techniques are characterized from the following viewpoints:

1. Using supervise information or not.
2. Displaying the individual points or representative points.
3. Linear mapping or nonlinear mapping.

The relationship among these methods can be illustrated as Fig.1.

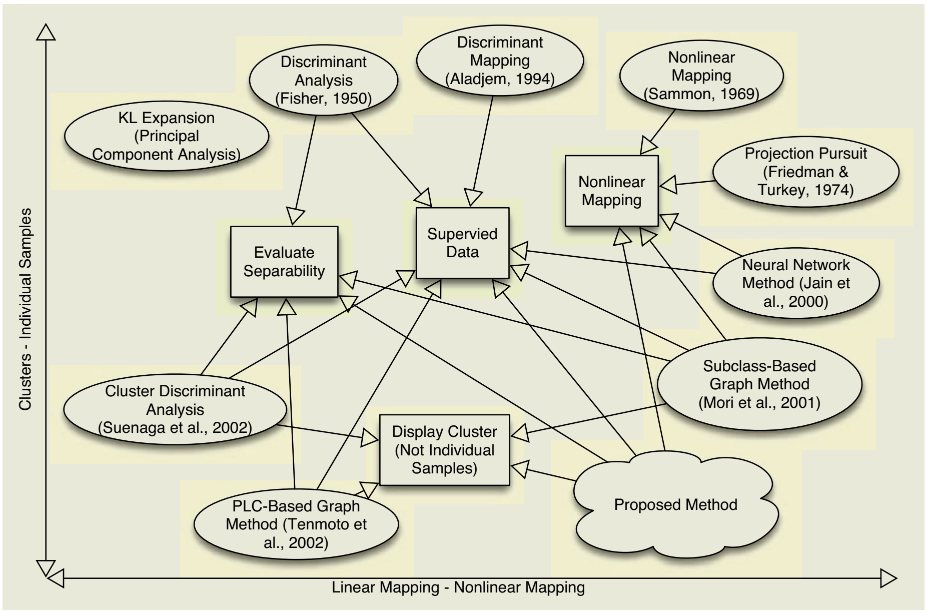

Clusters - Individual Samples
Linear Mapping - Nonlinear Mapping
KL Expansion (Principal Component Analysis)
Discriminant Analysis (Fisher, 1950)
Discriminant Mapping (Aladjem, 1994)
Nonlinear Mapping (Sammon, 1969)
Nonlinear Mapping
Projection Pursuit (Friedman & Turkey, 1974)
Evaluate Separability
Supervied Data
Neural Network Method (Jain et al., 2000)
Subclass-Based Graph Method (Mori et al., 2001)
Cluster Discriminant Analysis (Suenaga et al., 2002)
Display Cluster (Not Individual Samples)
PLC-Based Graph Method (Tenmoto et al., 2002)
Proposed Method

Fig. 1. Visualization methods of high-dimensional data.

When the supervise information, i.e., the class labels, are used, the separation status between the class regions can be observed. Therefore, almost latest methods are included in that group. In addition, the latest studies display representative points instead of the individual points. This is effective especially in the case of numerous samples. Once the points in a high dimensional space were mapped onto two dimensional space, the visual can trick us that the near points on the resultant map are also near in the feature space. This phenomenon can be happened both in linear mapping and nonlinear mapping methods.

To avoid these misunderstandable situations, Mori *et al.* proposed a novel visualization method [6]. In that technique, the individual samples in the feature space are organized to special type of overlapping clusters by a nonparametric classifier called subclass method [9], and such clusters are displayed as the nodes of a graph. The clusters called "subclasses" are formed so as to include the samples belonging to a certain class (positive samples) as many as possible and exclude the samples belonging to the other classes (negative samples) completely. In addition, the method connects the nodes according to the overlapping hypervolumes between the subclasses in order to display the separability.

However, the method tends to produce much subclasses in order to exclude the negative samples, then the resultant graph becomes complex. Although such a parameter that is used to eliminate too small subclasses in the graph is prepared, it is difficult to judge whether such nodes can be removed or important for showing the separability of the class regions.

From these points of view, in this study, normal mixture distributions are employed for the analysis of the structures in order to absorb the effects of the noise or outlier samples. Hence, Kullback-Leibler divergence is used to illustrate the separability between the component distributions.

3 Proposed Method

3.1 Overview

The supervised data structure is displayed as a "graph" by the following procedure:

1. Estimate normal mixture distributions on the samples in the feature space.
2. Project the mean vectors of the component distributions onto two dimensional space by Sammon's nonlinear mapping [1], and let them the nodes of a graph.
3. Calculate Kullback-Leibler divergences between the component distributions, and connect the nodes according to the values of the divergences.

The details of the steps are described in the following sections.

3.2 Normal Mixture Distributions

In this study, normal mixture distributions on the samples are estimated by EM algorithm [10] that maximizes the following likelihood:

$$L = \sum_{i=1}^{N} \log p(\boldsymbol{x}_i) = \sum_{i=1}^{N} \log \left\{ \sum_{k=1}^{K} c_k \mathrm{N}(\boldsymbol{m}_k, \Sigma_k)(\boldsymbol{x}_i) \right\},$$

where $\boldsymbol{x}_i$ is the ith feature vector (sample), N is the number of the samples, K is the number of the components, $\mathrm{N}(\boldsymbol{m}_k, \Sigma_k)(\cdot)$ is a normal distribution with a mean vector $\boldsymbol{m}_k$ and a covariance matrix Σ_k, and c_k is the weight of the kth component $\left(\sum_{k=1}^{K} c_k = 1\right)$. Such mixture models are estimated for the individual classes.

Here, the appropriate value of K is determined by MDL (minimum description length) criterion [11]. Each hypothesis is evaluated by the following formula:

$$\mathrm{MDL}(K) = -L + \frac{1}{2}\left[(K-1) + K\left\{D + \frac{D(D+1)}{2}\right\}\right]\log N,$$

where D is the number of the features. The appropriate number of components $\hat{K}$ is determined by varying K from 1 through a fixed number K_{max}, e.g., 10, as:

$$\hat{K} = \arg\min_{K} \mathrm{MDL}(K).$$

3.3 Nonlinear Mapping

In this study, the nodes of a graph are located by Sammon's nonlinear mapping [1]. This method projects the points in a high dimensional space so as to maintain the distances of every pair as far as possible, that is achieved by minimizing the following evaluation formula:

$$E = \frac{1}{\gamma}\sum_{i<j}^{N} \frac{\{\delta(\boldsymbol{x}_i, \boldsymbol{x}_j) - \delta(\boldsymbol{y}_i, \boldsymbol{y}_j)\}^2}{\delta(\boldsymbol{x}_i, \boldsymbol{x}_j)},$$

where $\boldsymbol{x}_i$ is the original vector and $\boldsymbol{y}_i$ is the corresponding projected vector,

$$\gamma = \sum_{i<j} \delta(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

and $\delta(\cdot,\cdot)$ is the Euclidean distance.

Unfortunately, the original Sammon's method cannot project additional points after the calculation of the mapping. Therefore, in this study, the mean vectors of the component distributions are tentatively added to the samples before performing the projection, and are picked up from the result.

3.4 Kullback-Leibler Divergence

In order to take into account the scattering information of the samples, the separability between the classes are evaluated by Kullback-Leibler divergence.

Because, in this study, normal mixture distributions are used for the analysis of the structure, the divergence between two components p, q is calculated directly from the distribution parameters as follows:

$$D(p||q) = \frac{1}{2}\left\{\log\det(\Sigma_q \Sigma_p^{-1}) - D + \mathrm{tr}(\Sigma_q^{-1}\Sigma_p) + (\boldsymbol{m}_p - \boldsymbol{m}_q)^t \Sigma_q^{-1}(\boldsymbol{m}_p - \boldsymbol{m}_q)\right\}.$$

The thickness of the edge between two nodes is determined proportional to $1/D(p||q)$. As a result, well-separated nodes have no or thin edges, while nonseparated nodes have thick edges.

Here, a threshold parameter θ for $D(p||q)$ is introduced to eliminate the edges with too large value of $D(p||q)$, because the resultant too thin edges are redundant and obstacle for the observation.

4 Experiments

The proposed method was tested on SHIP dataset [12]. This dataset aims to distinguish eight types of navy ships with eleven features, that were extracted from the ships' silhouette images. The total number of samples is 2545.

The result of Sammon's nonlinear mapping is shown in Fig.2, and the graphs obtained by the proposed method are shown in Fig.3 (a) and (b). In addition, the graphs obtained by Mori *et al.*'s method are also shown in Fig.4(a) and (b), corresponding to two different types of location methods. The parameters in their method were adjusted according to their guideline.

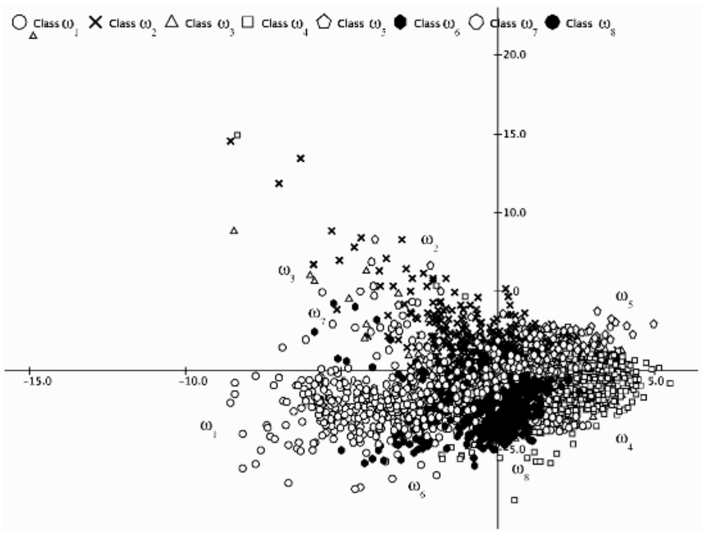
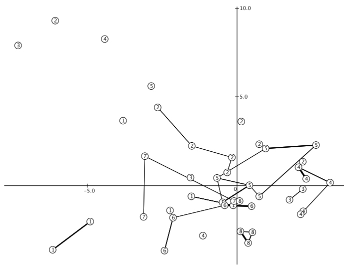


Fig. 2. Result for SHIP data by Sammon's nonlinear mapping.

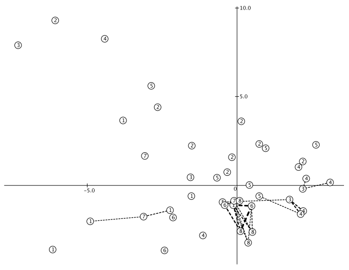
The result of Sammon's nonlinear mapping is very condensed, therefore it is difficult to find where the class $\omega_6, \omega_7, \omega_8$ samples distribute.

On the other hand, from the result of the proposed method (a), the essential distribution structure of each class is easily observed. For example, the class ω_5 forms "ring" structure, the class ω_8 forms unimodal cluster, and the other classes forms nonlinear belt structure and some isolated clusters.

In addition, from the result of the proposed method (b), it can be observed that in which regions correct classification is difficult. Here, two distant nodes in the graph does not necessarily mean that the separation of the corresponding samples is easy, and *vice versa*. For example, the distance between the class ω_1–ω_7 nodes at the left-bottom region in Fig.3(b) is far, but they should be nonseparable because there is the edge between the nodes. On the other hand, the class ω_2–ω_5 nodes at the right-upper region are close to each other, so it

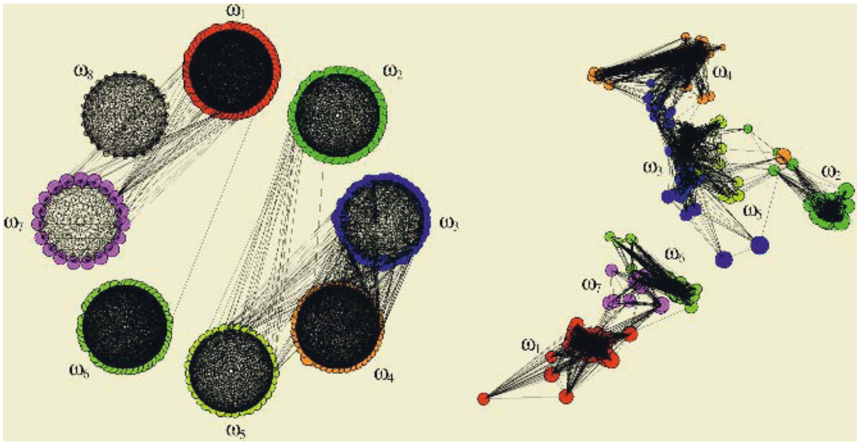


(a) Within-class edges with the threshold $\theta = 70$.



(b) Between-class edges with the threshold $\theta = 35$.

Fig. 3. Result for SHIP data by the proposed method. The numbers in the circles correspond the class labels.



(a) Polygon display type. (b) Principal component display type.

Fig. 4. Result for SHIP data by Mori *et al.*'s method.

seems very difficult to separate them. However, there is no edge between the nodes. This means the separation may be easy in the feature space.

In order to confirm that the nonseparable regions illustrated by the proposed method is right, the resultant graphs were compared to the confusion matrices obtained by nearest neighbor method and SVM with soft-margin. The hold-out method was used for the calculation. The results are shown in Table 1 (a) and (b).

Table 1. Confusion matrices for SHIP data. Too much erroneous results are underlined.

(a) by nearest neighbor method.

I\O	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	Error
$\omega\cdot$	143	1	0	0	1	$\underline{12}$	$\underline{11}$	2	0.159
$\omega\cdot$	0	214	5	6	1	0	1	0	0.057
$\omega\cdot$	0	7	72	$\underline{9}$	5	0	0	0	0.226
$\omega\cdot$	0	5	$\underline{10}$	225	5	0	0	0	0.082
$\omega\cdot$	2	6	0	5	154	5	1	1	0.115
$\omega\cdot$	$\underline{8}$	1	0	0	3	114	2	$\underline{12}$	0.186
$\omega\cdot$	$\underline{9}$	5	0	0	1	0	102	2	0.143
$\omega\cdot$	1	0	0	0	1	4	0	98	0.058

(b) by SVM with soft-margin.

I\O	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	$\omega\cdot$	Error
$\omega\cdot$	140	3	0	0	0	$\underline{19}$	6	2	0.176
$\omega\cdot$	0	222	1	3	1	0	0	0	0.022
$\omega\cdot$	1	6	68	$\underline{9}$	$\underline{9}$	0	0	0	0.269
$\omega\cdot$	0	4	0	235	6	0	0	0	0.041
$\omega\cdot$	0	3	1	5	161	3	0	1	0.075
$\omega\cdot$	5	2	0	0	2	120	0	$\underline{11}$	0.143
$\omega\cdot$	$\underline{17}$	5	0	0	2	1	92	2	0.227
$\omega\cdot$	0	0	0	0	1	5	1	97	0.067

By comparing Fig.3 to the tables, it can be confirmed that the understanding of the resultant graph is almost consistent with the results of those classifiers. For example, in the confusion matrices, the most erroneous relationships between classes are ω_1–ω_6, ω_3–ω_4, ω_6–ω_8 and ω_1–ω_7. Such classes are also connected strongly in the result of the proposed method. This means that such class regions are very closed and/or the local variance of each class is large, then the local distributions are overlapped. Therefore, new features should be added in order to improve the system's accuracy at these classes. On the other hand, the other classes, for example, ω_1–ω_3 has no error in the tables, and there is also no edge between the classes in the graph. This means the current feature set is sufficient for such classes, and the features may be reduced by feature selection methods.

While, the resultant graphs obtained by Mori *et al.*'s method do not necessarily match the results of the proposed method and the two classifiers. For example, Fig.4 insists that there are much overlaps between the class ω_3–ω_4, ω_1–ω_7 and ω_2–ω_5, but no or not so much overlaps between the class ω_1–ω_6 and ω_6–ω_8. Two options can be considered for the reason: (1) the subclass classifier is so strong that succeeded to separate these class regions completely, (2) Important but small nodes were removed by the threshold parameter. As a result, the corresponding edges did not appear in the graph.

5 Conclusion

A new graph type visualization method for supervised data structures was proposed. The method uses normal mixture distributions for the analysis of the distribution structures, and uses Kullback-Leibler divergence for the evaluation of the separability between the class regions.

From the experimental results, it was confirmed that the proposed method can display the essential distribution structure by within-class edges, and also can show by between-class edges in which regions and to what extent the class separation is not achieved sufficiently by the current feature set.

Compared to Mori *et al.*'s method, the proposed method shows the within-class structure simply and the between-class relationships appropriately. This property comes from the normal mixture distributions and MDL criterion in the complexity selection.

However, the proposed method also has a drawback in nature. The estimation of normal mixture models is difficult when the number of samples is relatively smaller than the number of features. Therefore, the estimation should be carried out with special care as far as possible. For example, it may be effective to substitute the basic EM algorithm to SMEM (Split and Merge EM) algorithm [13]. In addition, MDL framework for the selection of the appropriate number of component distributions does not work well if the number of samples is insufficient. Also at this point, good substitutions should be researched for the visualization purpose.

Acknowledgements

The authors are grateful to Professor Dr. Sklansky of University of California Irvine for providing the SHIP dataset.

This work was partly supported by the Ministry of Public Management, Home Affairs, Posts and Telecommunications of Japan under grant Strategic Information and Communications R&D Promotion Programme (SCOPE-S).

References

1. Sammon, J. W.: A Nonlinear Mapping for Data Structure Analysis. IEEE Transactions on Computers **18** (1969) 401–409
2. Friedman, J. H., Tukey, J. W.: A Projection Pursuit Algorithm for Exploratory Data Analysis. IEEE Transactions on Computers **23** (1974) 1974 881–889
3. Aladjem, M.: Multiclass Discriminant Mappings. Signal Processing **35** (1994) 1–18
4. Mao, J., Jain, A. K.: Artificial Neural Networks for Feature Extraction and Multivariate Data Projection. IEEE Transactions on Neural Networks **6** (1995) 296–317
5. Jain, A. K., Duin, P. W., Mao, J.: Statistical Pattern Recognition: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 4–37
6. Mori, Y., Kudo, M., Toyama, J., Shimbo, M: Comparison of Low-Dimensional Mapping Techniques Based on Discriminary Information. Proceedings of the Second International ICSC Symposium on Advances in Intelligent Data Analysis (2001) CD-ROM Paper #1724-166
7. Tenmoto, H., Mori, Y., Kudo, M.: Visualization of Class Structures using Piecewise Linear Classifiers. Proceedings of Logic Applied to Technology (LAPTEC) (2002) 104–111.
8. Taylor, P. C., Hand, D. J.: Finding 'Superclassifications' with an Acceptable Misclassification Rate. Journal of Applied Statistics **26** 579–590
9. Kudo, M., Torii, Y., Mori, Y., Shimbo, M.: Approximation of Class Regions by Quasi Convex Hulls. Pattern Recognition Letters **19** (1998) 777–786

10. Dempster, A. P., Laird, N. M., Rubin, D. B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society, Series B **39** (1977) 1–38
11. Rissanen, J.: A Universal Prior for Integers and Estimation by Minimum Description Length. Annals of Statistics **11** (1983) 416–431
12. Park, Y., Sklansky, J.: Automated Design of Multiple-Class Piecewise Linear Classifiers. Journal of Classification **6** (1989) 195–222
13. Ueda, N.: EM Algorithm with Split and Merge Operations for Mixture Models (Invited). Transactions of IEICE, **E83-D** (2000) 2047–2785.

Texel-Based Texture Synthesis with Bunch Sampling

Dongxiao Zhou and Georgy Gimel'farb

CITR, Tamaki Campus, Department of Computer Science,
The University of Auckland,
Auckland, New Zealand

Abstract. Texture is frequently considered as a repetitive spatial arrangement of texels, the primitive elements of texture. We define the texel as a bunch of image signals that has a particular geometric structure (shape and size). This provides for fast synthesis of a spatially homogeneous texture by bunch sampling. First, the structure of the texels and a placement grid to spatially arrange them are estimated from a training image using a generic Gibbs random field model of the texture. Then at the synthesis stage, the structure serves as a sampling mask to capture the texels from the training image. Random positions for replicating texels to form a synthetic large-size texture are selected according to the placement grid.

1 Introduction

Most of the known texture synthesis methods use Markov random field (MRF) models to describe spatially homogeneous textures with translation invariant signal statistics. With respect to sampling techniques, there are two main groups of such methods, namely, model-based probabilistic synthesis and non-parametric sampling. Model-based methods [1, 3, 4, 7, 17] construct an explicit MRF model of a given training image. The model is specified by a joint Gibbs probability distribution (GPD) of image signals. New textures are generated by iterative Markov Chain Monte Carlo (MCMC) probabilistic sampling of the GPD based pixel-wise stochastic relaxation e.g., the Gibbs or Metropolis sampler. These methods are successful in modelling a broad range of textures, but are inefficient in realistic texture synthesis because the stochastic relaxation is too computationally complex.

Non-parametric sampling [5, 6, 11] achieves much faster texture synthesis by avoiding slow pixel-wise relaxation. The MRF model of texture is assumed only implicitly, and the training texture itself is used as a sampling source. To preserve local features of the training texture, the sampling in [6], for instance, chooses sampling candidates by minimising the distance (e.g., in the $L2$ metric) between the pixel neighbourhoods in the training image and the like neighbourhood in the synthetic texture. As each sampling candidate is copied to the synthetic texture, the pixel neighbourhood structure and thus the texture feature is expected to be inherited by the synthetic texture. However, without an explicit texture model,

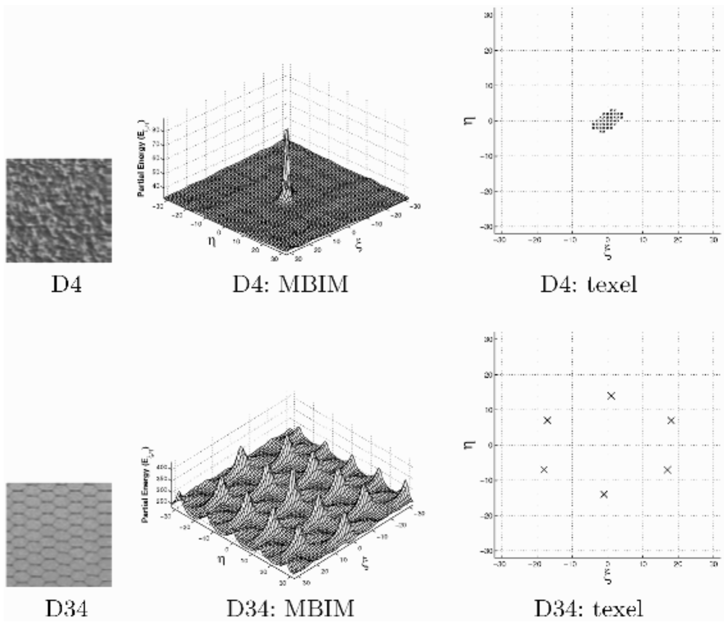


Fig. 1. Brodatz's training textures D4 and D34 (128×128), the corresponding MBIMs and the structure of texels. The MBIMs for the 65×65 search window are scaled up for better visual perception.

the non-parametric sampling lacks knowledge of internal structure of signal dependencies varying from texture to texture. This hinders adaptive selection of the shape and size of the pixel neighbourhood for a particular texture. Also, the distance-based pixel-wise copying may accumulate local errors. More stable block- or patch-based copying in [5, 11] results in false borders between adjacent permuted blocks (patches) and in verbatim replicas of each training singularity. Most of these problems are overcome by "smart copying" [12] that cuts from the training image arbitrary shaped patches forming a "borderless" synthetic texture rather than attempts to post-fix the false borders.

Our paper describes an alternative method, called bunch sampling in [9, 16], for realistic texture synthesis. It combines the strengths of both the model-based probabilistic synthesis and the non-parametric sampling to provide fast synthesis of large-size textures. Compared to [9, 16], this paper introduces more elaborated techniques for estimating the geometric structure of texels and their placement grid for each particular texture. The same bunch sampling is used here also to rectify grayscale and colour textures with at least weak translational periodicity. The single texel for a rectified prototype of the periodic texture is estimated using marginal posterior probabilities of the corresponding signals in all the superposed training bunches.

2 Bunch Sampling

Structural approach to texture analysis [10] considers a homogeneous texture as the result of regular and repetitive spatial arrangement of its primitive elements or texels. Bunch sampling synthesises textures in line with this idea, that is, a new texture is generated by permuting and replicating the texels. The replication is guided by a placement grid specifying the periodic spatial arrangement of texels in the texture. Experiments in [16] have shown that the bunch sampling forms realistic large-scale synthetic textures for two separate classes of images, namely, periodic or almost periodic mosaics and aperiodic stochastic textures. The classes have different notions of the texels and placement grids. Textures of the first class contain spatially uniform repetitive subimages (graphical patterns) that may be considered as noisy or transformed replicas of a single texel prototype or a few such prototypes. The grid puts strict limits on possible relative positions of the neighbouring texels. Textures of the second class has spatially uniform marginal distributions of signal co-occurrences in the bunches. The texel-based description is still possible, but now it involves a very large set of different texels, and there are no too strict limitations on their relative placement except for the image continuity (that is, the absence of "false borders").

Loosely speaking, in both cases the texel is a bunch of image signals having a geometric structure with particular shape, orientation, and size. Due to assumed translational invariance, all the texels share the same geometric structure but may differ by particular signal combinations. Spatial repetitiveness results in non-uniform frequencies of signal co-occurrences. Thus the bunch structure can be approximately related to most characteristic pairwise gray level cooccurrences in a generic Gibbs random field (GGRF) model of translation invariant textures [7,8]. Once that the structure is found for a given training image, the placement grid is specified accordingly in terms of relative spatial positions of the texels.

The GGRF texture model allows to rank families of translation invariant pixel pairs according to their partial interaction energies in a particular texture. Let $\mathbf{R}$ denote an arithmetic lattice supporting digital images. Let $\mathbf{C}_{\xi,\eta} = \{(x,y),(x+\xi,y+\eta) : (x,y) \in \mathbf{R}, (x+\xi,y+\eta) \in \mathbf{R}\}$ be a family of pixel pairs having the same relative spatial displacement. Then the first approximation of the partial energy for these pixel pairs in the training image $\mathbf{g}$ is proportional to the variance of the normalised grey level co-occurrence histogram (GLCH) collected over the family $\mathbf{C}_{\xi,\eta}$ [7,8]:

$$E_{\xi,\eta}(\mathbf{g}) \propto (\mathbf{F}_{\xi,\eta}(\mathbf{g}) - \mathbf{F}_{\mathbf{IRF}}) \bullet \mathbf{F}_{\xi,\eta}(\mathbf{g}) \tag{1}$$

where $\mathbf{F}_{\xi,\eta}(\mathbf{g})$ and $\mathbf{F}_{\mathbf{IRF}}$ are the normalised GLCHs for the image and the independent random field, respectively, and $\bullet$ denotes the dot product.

The partial energy of Eq. (1) specifies the contribution of the family $\mathbf{C}_{\xi,\eta}$ to the overall interaction energy, that is, to the probability of the training image. The higher the partial energy, the more probable the image and the more characteristic the family itself for this texture. The top-rank partial energies specify the most characteristic structure of pairwise interactions [8].

The partial energies of all the families in a large search set of displacements, $\mathbf{W} = \{(\xi, \eta) : |\xi| \leq \Delta, |\eta| \leq \Delta\}$ of the size $(2\Delta + 1) \times (2\Delta + 1)$, specify a symmetric model based interaction map (MBIM) [8]. Points (ξ, η) and $(-\xi, -\eta)$ in the MBIM correspond to the family $\mathbf{C}_{\xi,\eta}$, and their scalar values represent the relative partial energy $E_{\xi,\eta}(\mathbf{g})$ in Eq. (1). Figure 1 demonstrates the MBIMs with the search window $\mathbf{W}$ of the size 65×65 for the Brodatz's stochastic texture D4 and periodic mosaic D34.

2.1 Geometric Structure of Texels

As shown in Fig. 1, the top-rank partial energies form specific clusters in the MBIMs. These most energetic clique families determine both statistical features of the texture and the geometric structure of the texels. Stochastic textures (e.g. D4) with dominant close-range pixel interactions have only one central cluster. The like central cluster for periodic textures such as D34 relates mainly to uniform background. The periodic structure of these textures is reflected by the peripheral clusters. Thus, the MBIM allows bunch sampling to distinguish between the aperiodic and periodic textures and estimate the geometric structure of texels using different strategies. For a stochastic texture, the central cluster in the MBIM specifies the shape of the texel (usually it is a connected region). For a periodic mosaic, the peak points of the peripheral clusters being the nearest neighbours to the MBIM's centre form the geometric structure of the texels. These texels are of the relatively small size, e.g., only six pixels for the texture D34. As shown later, even these simple structures are adequate for realistic texture synthesis.

The following algorithm is used to analyse MBIMs.

1. Find the threshold that separates the most energetic families of pixel pairs from others, using the algorithm described in [14]. The energy histogram for the MBIM is usually unimodal, with only one main peak at the lower end representing the majority of families with the low partial energy.
2. Segment the MBIM using the obtained threshold and identify the energy clusters using a general connected component labeling technique.
3. Find the peak of every cluster (it can be approximated by the gravity centre of the cluster if the partial energies are considered as the weights).

2.2 Placement Grid for Texels

Assuming that non-overlapping bunches are conditionally independent, bunch sampling tessellates the textured image with a guiding grid derived according to the estimated geometric structure of the texel. Each cell of the grid is a compact parallelogram enclosing the texel's structure. The parallelogram is specified with four parameters, $(\theta_x, \theta_y, m, n)$ where the angles θ_x and θ_y give the guiding orientation of the cell sides with respect to the image coordinate axes, and the side sizes m and n are the maximum spans of the texel along the guiding directions(Fig. 3).The enclosing parallelogram is computed here using the Voss–Suesse's algorithm in [15].

The guiding grid reflects the underlying structure that forms the repetitive pattern of a homogeneous texture. It defines a reference coordinate system so that the spatial dependencies between the texels are specified in terms of the relative positional shift of each texel with respect to the closest grid cell. As shown in Fig. 4, the relative shift, (δ_x, δ_y), for the texel centred on point $p = \{x, y\}$ in the image is as follows:

$$\begin{aligned} \delta_x &= (y \cdot \sin\theta_y + x \cdot \cos\theta_x) \ mod\, m \\ \delta_y &= (y \cdot \cos\theta_y - x \cdot \sin\theta_x) \ mod\, n \end{aligned} \tag{2}$$

2.3 Texture Synthesis with Bunch Sampling

Given a training texture $\mathbf{g}^0$, texture synthesis begins with a blank goal image, $\mathbf{g}^1$, that can be of arbitrary size. For each pixel, p, in the goal image, bunch sampling computes its relative shift (δ_x, δ_y) with respect to the placement grid by Eq. (2), and then searches the training image for a proper sampling position p_s with the same relative shift. Usually there are multiple candidate sampling positions satisfying the condition, so one of them is chosen randomly. With p_s as the centre and the geometric texel structure as the sampling mask, a bunch of image signals is picked up from the training image, and then it is copied into the goal image $\mathbf{g}^1$ with the centre on p. During the synthesis, signal collisions happen when a new bunch has to be placed into an area that has been occupied by a previously placed bunch. A simple heuristic rule that preserves the already placed signals to resolve the collisions [9] results in the most of cases in realistic borderless textures.

The pixels in the goal image can be generated in an arbitrary order, and the texture synthesis completes after all the pixels are covered. The synthesis has linear computational complexity with respect to the size of the synthetic image $|\mathbf{g}^1|$. It depends linearly on the size of the bunch, too.

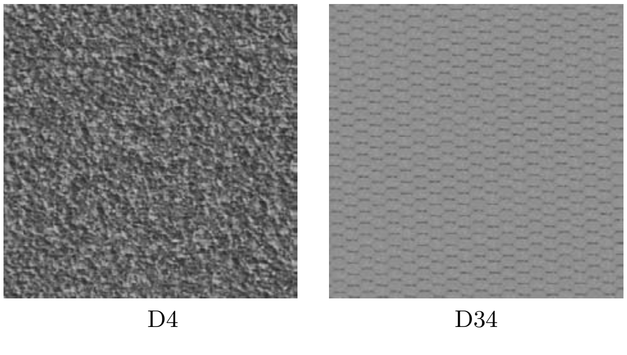


Fig. 2. Synthetic textures D4 and D34 (360×360).

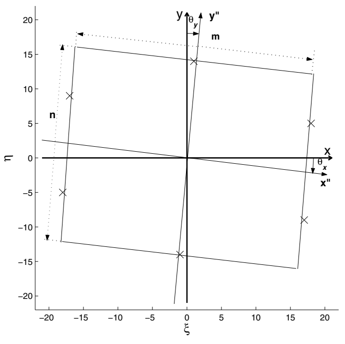


Fig. 3. Parameters of the placement grid for the texture D34.

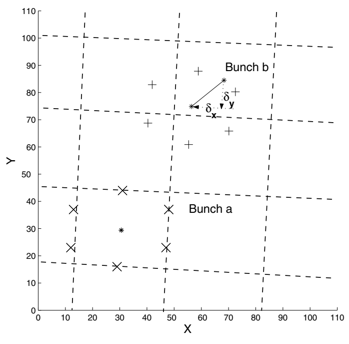


Fig. 4. Tessellation of D34 and relative positions of the bunches: $(0, 0)$ for Bunch a and (δ_x, δ_y) for Bunch b.

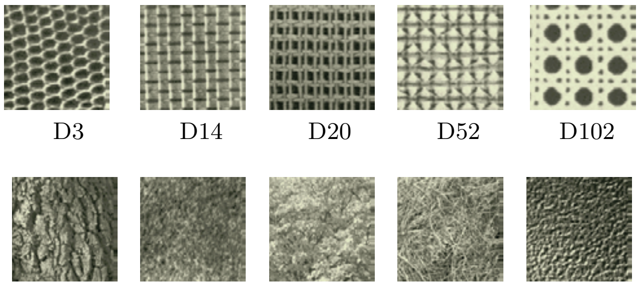


Fig. 5. Training Brodatz's and MIT VisTex textures 128 × 128.

3 Results and Discussion

Bunch sampling has been tested on a variety of textures, mainly from the digitised Brodatz album [2] and the MIT VisTex texture database [13]. Figure 2 shows the synthetic textures D4 and D34 generated with the texels in Fig 1. Figure 6 shows synthetic textures for the training images in Fig. 5. In these cases, the synthetic and training textures have good visual similarity.

But deviations from periodicity may hinder the synthesis. Figure 7,a shows the visually unsatisfactory synthetic Brodatz's texture D3. Because the bunch sampling assumes the fixed geometric structure of each texel, it fails under local deformations of such a weakly homogeneous texture. As a result, random distortions appear in the synthetic texture due to mismatches between the estimated fixed structure and its local deformation at the sampling position.

However, in this case, the bunch sampling can at least rectify the distorted training image and produce its idealised periodic prototype. The approximate Bayesian estimate of a single texel based on the maximum marginal posterior (MMP) probabilities of signals in each pixel is easily obtained by superposing all signals with the same relative shift with respect to the placement grid. Figure 7,b shows the resulting synthetic prototype. In spite of less noise and better visual appearance, it is far from a realistic D3-like texture. Such a "homogenised" prototype allows to recover local geometric deformations of the training image, and their spatial model can be later used to convert the prototype into a more realistic texture.

The GGRF model is usually restricted to only the second order GLCHs as its sufficient statistics. Moreover, to restrict the computational complexity we deal in practice with only 16 grey levels. Nevertheless, the bunch sampling still

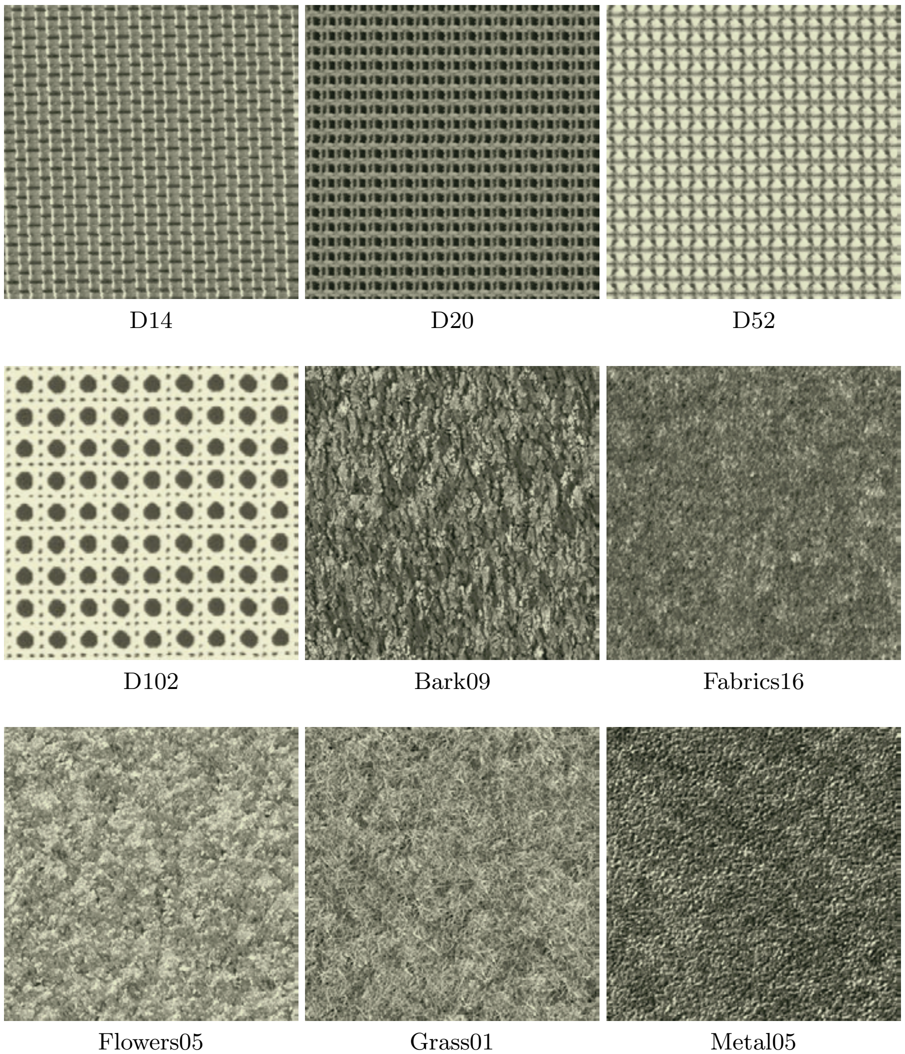


Fig. 6. Examples of synthetic textures (360×360) generated with bunch sampling.

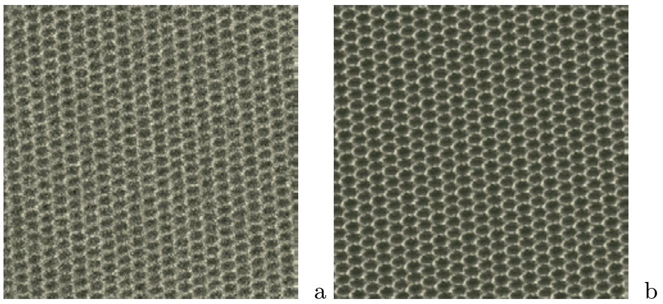


Fig. 7. Synthetic textures D3 (360×360) generated under different selection of texels.

handles colour textures fairly well due to the separate analysis and synthesis stages. At the analysis stage, a colour texture is reduced to the greyscale one for extracting the texel structure and placement grid. During the synthesis, the signals sampled from the original colour images are used to generate new textures. Figure 8 shows examples of the training and rectified synthetic colour textures.

4 Conclusions

Bunch sampling uses a "placement rule + texels" scheme to model homogeneous textures. The size, shape, and placement rule for texels are derived from image statistics of the sample texture using a GGRF model. During the synthesis, texels are sampled form the training texture and are directly moved to the synthetic one guided by the placement rule. In such a way, bunch sampling provides a fast model-based texture synthesis. However, bunch sampling is less efficient in handling inhomogeneous deformation in texture because the assumption of the uniform geometric structure for texels is too rigid for weakly homogeneous and inhomogeneous textures. Our future work is to relax the geometric structure of texels with some geometric adaption so that we can model the local deformation in those textures more effectively.

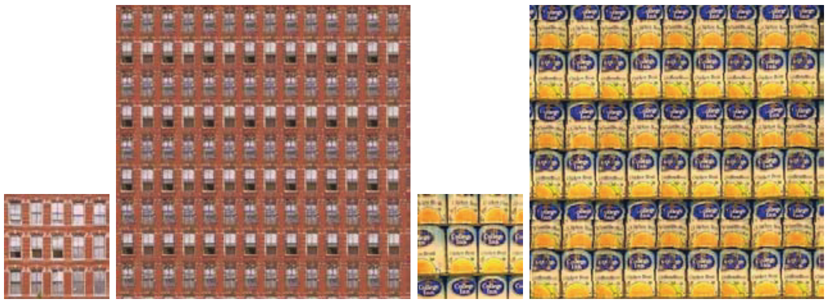


Fig. 8. Synthesis of colour textures, "windows" and "cans".

Acknowledgements

This work was partially supported by the RSNZ Marsden Fund grant UOA122.

References

1. J. E. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, Vol. B48, pp. 259–302, 1986.
2. P. Brodatz. *Textures: A Photographic Album for Artists and Designers.* Dover, New York, 1966.
3. R. Chellappa. Two dimensional discrete Gaussian Markov random field models for image processing. In *Progress in Pattern Recognition*, Vol. 2, pp. 79–112. Elsevier Science Publishers B. V., 1985.
4. G. R. Cross and A. K. Jain. Markov random field texture models. *IEEE Trans. Pattern Anal. Machine Intell.*, Vol.5, pp. 25–39, 1983.
5. A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH 2001, Computer Graphics Proceedings*, pp. 341–346. ACM Press / ACM SIGGRAPH, 2001.
6. A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proc. Int. Conf. on Computer Vision (ICCV'99), Kerkira, Corfu, Greece, 20–25 September 1999*, Vol. 2, pp.1033–1038, 1999.
7. G. L. Gimel'farb. Texture modeling with multiple pairwise pixel interactions. *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 18, pp. 1110–1114, 1996.
8. G. L. Gimel'farb. *Image Textures and Gibbs Random Fields.* Dordrecht, Kluwer Academic Publishers, 1999.
9. G. L. Gimel'farb and D. Zhou. Fast synthesis of large-size textures using bunch sampling. In *Proc. Image and Vision Computing New Zealand (IVCNZ 2002), Auckland, New Zealand, 26–28 November 2002*, pp. 215–220, 2002.
10. R. Haralick. Statistical and structural approaches to texture. *Proc. IEEE*, Vol. 67, no.5, pp. 786–804, 1979.
11. L. Liang, C. Liu, and H. Y. Shum. *Real-time texture synthesis by patch-based sampling.* Technical Report MSR-TR-2001-40, Micorsoft Research, 2001.
12. A. Neubeck, A. Zalesny, and L. van Gool. Cut-primed smart copying. In Proc. 3rd Int. Workshop "Texture 2003", Nice, France, October 17, 2003. Heriot-Watt Univ., pp. 119–124, 2003.
13. R. Picard, C. Graszyk, S. Mann, *et al. VisTex Database.* MIT Media Lab, Cambridge, Massachusettes, 1995.
14. P.L. Rosin. Unimodal thresholding. *Pattern Recognition*, vol. 34, no. 11, pp. 2083-2096, 2001.
15. K. Voss and H. Suesse. Invariant fitting of planar objects with primitives. *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 19, pp. 80–84, 1997.
16. D. Zhou and G. L. Gimel'farb. Bunch sampling for fast texutre synthesis.. In *Proc. Int. Conf. Computer Analysis of Images and Patterns, Groningen, The Netherlands, 25–27 August 2003.* (*Lecture Notes in Computer Science 2756*), Springer-Verlag, Berlin, pp. 124–131, 2003.
17. S. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (FRAME): To a unified theory for texture modeling. *International Journal of Computer Vision*, Vol. 27, no. 2, pp. 107–126, 1998.

Towards 3-Dimensional Pattern Recognition

Ken-ichi Maeda, Osamu Yamaguchi, and Kazuhiro Fukui

Corporate Research & Development Center, TOSHIBA Corporation
1, Komukai-Toshiba-cho, Saiwai-ku, Kawasaki, 212-8582, Japan
{ken.maeda,osamu1.yamaguchi,kazuhiro.fukui}@toshiba.co.jp

Abstract. 3-dimensional pattern recognition requires the definition of a similarity measure between 3-dimensional patterns. We discuss how to match 3-dimensional patterns, which are represented by a set of images taken from multiple directions and approximately represented by subspaces. The proposed method is to calculate the canonical angles, in particular the third smallest angle between two subspaces. We demonstrate the viability of the proposed method by performing a pilot study of face recognition.

1 Introduction

We aim to extend pattern recognition coverage from 2-dimensional (2-D) applications to 3-dimensional (3-D). One expected application of such 3-D pattern recognition is differentiation between an object and its photograph as well as identification of the object.

In general, for pattern recognition, we define a similarity measure between two patterns and use this as the criterion for performing recognition. Thus, for our purpose, we need a definition of similarity between two 3-D patterns.

A set of images taken from multiple directions are used to describe a 3-D object[1] and the set is approximately represented by a subspace. A subspace representation of 3-D objects was introduced, for example, in the *parametric eigenspace representation* [1], although the subspace representation of a pattern set had been known as the *subspace method* [2]. In most cases, the *principal component analysis* (PCA) is used for making the subspace as an approximation of the distribution of the patterns in the set. However, the pattern matching method in [1] is limited to measure the nearest distance between an input represented by a vector and a reference that is the nearest vector in a class.

The largest problem with such a framework is that a single photograph which happens to be identical to an image that is captured from a particular direction matches exactly even though the actual objects are different. More concretely, it is impossible to differentiate between an object and its photograph using conventional methods.

[1] 3-D modeling provides a representation of 3-D objects, but it takes a long time to construct such models for real objects. It is therefore currently impractical to acquire real-time results with this approach.

We present a 3-D pattern matching method to solve this problem. The proposed method is an extension of a 2-D pattern matching method, the *mutual subspace method* (MSM) [3]. The extension is to calculate angles between two subspaces, in particular the third smallest one rather than the smallest one that is used in the MSM. We also show the results of a pilot study where faces and their photographs are dealt with.

2 Approach to the Task

2.1 Use of the Subspace Method Framework

It is desirable that the 3-D pattern matching method is an extension of a 2-D method. Among existing 2-D methods, we chose the subspace method framework [2] because most of its calculations are linear operations that are executable at high speed, in particular with the SIMD instructions of today's CPUs. The subspace method framework has been widely applied in various pattern recognition tasks, such as character, speech, and face recognition. However, if we try to apply the subspace method directly to the 3-D problem, a 3-D model is required: which is expensive.

Among the subspace methods, the MSM has the most interesting properties. It was initially developed in order to provide greater tolerance for hand-printing deformation of Kanji in early 80's, and outperformed the classical subspace method in a Kanji recognition experiment [3]. It was later applied to face recognition, which is originally a form of 3-D recognition, and achieved satisfactory accuracy [4].

2.2 Review of the Mutual Subspace Method

The MSM[2] was the first method to utilize the angle between two subspaces for defining the similarity between an input and a reference[3], though they are originally two *sets* of vectors. Given subspaces, U and V, the angle between these is defined as the minimum angle between vectors $\boldsymbol{u}$ and $\boldsymbol{v}$, where $\boldsymbol{u} \in U$ and $\boldsymbol{v} \in V$, according to [7]. Let θ be the angle. Then θ is calculated as

$$\cos^2\theta \doteq \sup_{\substack{\boldsymbol{u} \in U, \boldsymbol{v} \in V \\ \|\boldsymbol{u}\| \neq 0, \|\boldsymbol{v}\| \neq 0}} \frac{(\boldsymbol{u}, \boldsymbol{v})^2}{\|\boldsymbol{u}\|^2 \|\boldsymbol{v}\|^2}. \tag{1}$$

This is also used as the definition of the similarity of the subspaces, i.e. the sets of the vectors, in the MSM. In order to calculate the similarity, we apply the following theorem.

• Since the first MSM paper [3] is only in Japanese, we summarize the major topics in this subsection.

• The concept of using the angle between subspaces was also discussed in [5] and [6], but the aim was other than defining similarity between an input and a reference.

Theorem 1:
Let U and V be two subspaces and P and Q be orthogonal projection operators onto U and V, respectively. Then the angle between U and V is calculated as the maximum eigenvalue of PQP or QPQ [3] [8] [9]. Let μ and ν be the maximum eigenvalues of PQP and QPQ, i.e.

$$PQP\boldsymbol{x} = \mu\boldsymbol{x}, \tag{2}$$

and

$$QPQ\boldsymbol{x} = \nu\boldsymbol{x}, \tag{3}$$

respectively. Then

$$\begin{aligned} \cos^2\theta &= \|QP\|^2, & (4)\\ &= \|PQ\|^2, & (5)\\ &= \mu, & (6)\\ &= \nu, & (7) \end{aligned}$$

where the norm of an operator A is defined as

$$\|A\| \doteq \sup_{\|\boldsymbol{z}\|\neq 0} \frac{\|A\boldsymbol{z}\|}{\|\boldsymbol{z}\|}. \tag{8}$$

According to this theorem, μ or ν itself can be used as the similarity in the sense that it is the angle between the two subspaces. However, as the eigenvalue calculation of PQP or QPQ is costly, the matrices basically being large, we actually calculate the eigenvalue of a smaller matrix X whose eigenvalues are identical.

Let $X = (x_{ij})$ be

$$x_{ij} = \sum_{m=1}^{M} (\boldsymbol{\psi}_i, \boldsymbol{\phi}_m)(\boldsymbol{\phi}_m, \boldsymbol{\psi}_j), \tag{9}$$

or

$$x_{ij} = \sum_{n=1}^{N} (\boldsymbol{\phi}_i, \boldsymbol{\psi}_n)(\boldsymbol{\psi}_n, \boldsymbol{\phi}_j), \tag{10}$$

where $\{\boldsymbol{\phi}_m\}_{m=1}^{M}$ and $\{\boldsymbol{\psi}_n\}_{n=1}^{N}$ are the bases of U and V, respectively. Then the eigenvalues of X are equal to those of PQP and QPQ [3].

In the sense that an input is represented by a subspace, the MSM is an extension of the classical subspace method where the input is represented by a vector: which is also a basis of a 1-D subspace.

2.3 Use of Other Canonical Angles

Canonical Angles in Previous Methods. The *canonical angles* are defined as the angles between two subspaces which are measured from the directions that are orthogonal to one another [9]. The angle calculated in the MSM is the smallest one, calculated as the largest eigenvalue. It is sufficient for application to 2-D pattern matching, but there exists a serious problem for 3-D pattern matching, as described earlier.

The problem can be modeled as follows; if we imagine a case where two subspaces share a smaller subspace, e.g. two 2-D subspaces sharing a line, as shown in Fig. 1, the smallest angle is zero. In such a case, the two patterns will be evaluated as identical in the MSM framework.

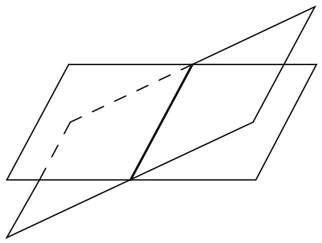


Fig. 1. Two 2-D subspaces in 3-D space.

As an alternative definition of distance between two subspace, Oja and Parkkinen proposed to use the smallest eigenvalue, $\mu_{\min}$, i.e. the largest canonical angle [5]. This method does not have the above problem, because it uses another angle of the shared line in Fig. 1.

Let P and Q be the projections to the subspaces U and V, respectively. Then the equation in [5] is

$$PQP\boldsymbol{x} = \mu_{\min} P\boldsymbol{x}, \tag{11}$$

where

$$\|P\boldsymbol{x}\| = 1. \tag{12}$$

As the eigenvectors of PQP are on the subspace U, (11) is equivalent to (2). But for the condition of (12), the smallest eigenvalue would be zero in most cases because it corresponds to a direction that is out of the subspaces, so it would not represent the canonical angle.

Canonical Angles for 3-D Recognition. We have found a possibility of further extension of the subspace method framework, using other angles[4], referring to the two previous methods. The methods adopted the opposite definitions of subspace distance; the smallest angle in the MSM and the largest angle in [5]. However we can have more canonical angles by solving the eigenvalue problem, (2) or

$$X\boldsymbol{y} = \mu\boldsymbol{y}. \tag{13}$$

Now the question is determining which one is suitable for 3-D recognition.

• A similar approach was presented in [10], but the meanings of the other angles and similarity were not clearly discussed.

In order to simplify the issue of 3-D recognition, we assume the following conditions, so that a linear approximation can be applied:

– The moving object is captured using a fixed camera.
– Motion primarily consists of small rotations.
– An appropriate normalization is performed.

These conditions ensure that variations in the recognition patterns are small.

When variation, e.g. lateral rotation, in the patterns is small, the distribution of the patterns caused by the variation can be assumed to be on a 2-D subspace, reflecting the dimension of the pattern itself and that of the variation. A different variation, e.g. vertical rotation[5], makes another 2-D subspace that shares a dimension of the pattern itself with the previous one.

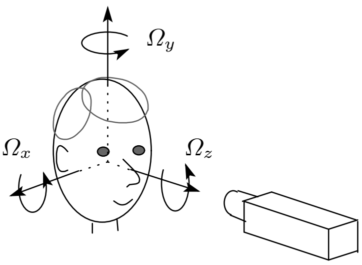


Fig. 2. Rotation of a 3-D object – a face –.

We should consider both lateral and vertical rotations, so that the rotations of a different direction or multiple directions can be described as a combination of these two rotations. Therefore a 3-D subspace is a good approximation of distributions due to rotation[6].

Given two 3-D objects that are identical, the 3-D subspaces for them should also be identical, or more precisely, the 3-D intersecting parts of two higher-dimensional subspaces should be identical. This means that the three smallest canonical angles should be zero. Since some actual objects like human faces have other variations, the canonical angles may have small values. However the three angles are still good measures of the similarity between the objects.

Referring to the definition of distance used in [5], we propose to use just the third smallest angle or the third largest eigenvalue among the three. The fourth largest eigenvalue should be nearly zero because we assume that the distribution is approximated by a 3-D subspace. The second largest eigenvalue can be 1 or near 1 if the photograph has a cylinder shape whose side view is the same as that of the object. This makes the third largest more appropriate.

• There is yet another rotation for 3-D objects; a rotation around the axis of the lens. However, considering the task of differentiating an object from its photograph, this the rotation is of no use, we take only Ω_x and Ω_y into account, and not Ω_z (see Fig. 2).

• Since the actual distribution lies in a higher-dimensional subspace due to many variations such as facial expressions, the input and reference subspaces should be represented in higher dimensions, e.g. in five or seven dimensions, shown empirically in [11].

The proposed method corresponds to calculating the largest canonical angle between two 3-D subspaces that are the closest part of the two original subspaces. Assume that $\{\mu_k\}_{k=1}^{K}$ are the eigenvalues of matrix X in (9) or (10) which are sorted in descending order. Then $\mu_3 (= \cos^2 \theta_3)$ is the answer. Since the calculation of the eigenvalues is in descending order and we need only the third largest one, the condition of (12) is no longer required.

3 A Pilot Study on Differentiating an Object from Its Photograph

We performed a pilot study on differentiating an object from its photograph in face recognition, which we consider to be a typical application in this paper (see Fig. 3). As described earlier, it is impossible to make this differentiation if we use a single image that is taken head-on.

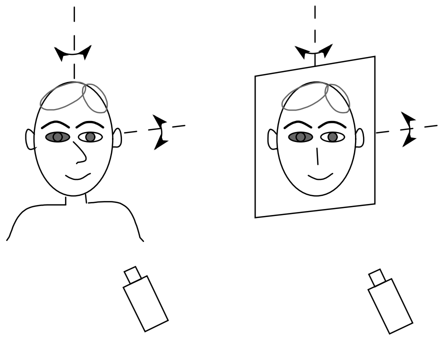


Fig. 3. Differentiation between an object – a face – and its photograph.

If we use the images of a 3-D object taken from multiple directions, some images may exhibit occlusions, or the lighting and shading conditions may be different among the images. Differently, all the images of a 2-D object are affine transformed ones of any single image. Even with the difference between these two situations, the largest eigenvalues of the two cases in the MSM framework are identical.

We experimented with using μ_3 as well as μ_1. Due to the normalization process [4], the variation in the normalized 3-D patterns is small as shown in Fig. 4. The experimental set therefore fulfills the conditions assumed in the last section.

We registered a set of face images (subject P0), and performed recognition for 11 people's faces including the registered person's and their photographs. Figure 5 shows an example set of face images, and Figure 6 shows an example set of photograph images of the same subject. Table 1 shows the experimental results for face inputs, and Table 2 shows the experimental results for photograph inputs.

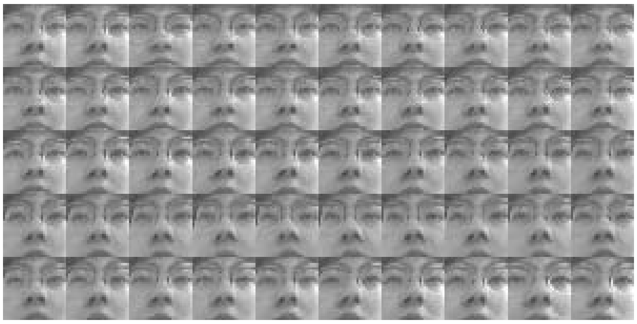


Fig. 4. Normalized recognition area.

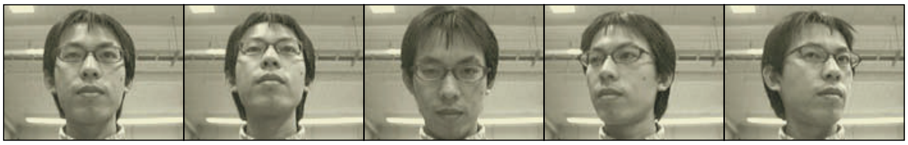


Fig. 5. Example set of face images.

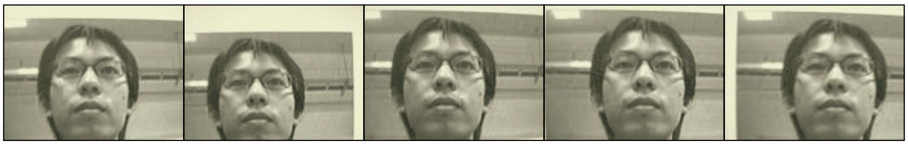


Fig. 6. Example set of photograph images.

Table 1. Recognition results for face inputs.

Person	$\mu.$	$\mu.$
P0	0.989	0.937
P1	0.702	0.256
P2	0.707	0.520
P3	0.786	0.488
P4	0.701	0.457
P5	0.643	0.459
P6	0.730	0.227
P7	0.554	0.334
P8	0.750	0.557
P9	0.716	0.545
P10	0.772	0.435

Table 2. Recognition results for photograph inputs.

Person	$\mu.$	$\mu.$
P0	0.977	0.204
P1	0.591	0.165
P2	0.619	0.237
P3	0.741	0.123
P4	0.665	0.075
P5	0.626	0.124
P6	0.612	0.055
P7	0.678	0.238
P8	0.732	0.246
P9	0.600	0.154
P10	0.648	0.075

The experimental results show that μ_1 is not suitable for differentiating between the face and the photograph since the values for P0's face and photograph are 0.989 and 0.977, respectively. It is difficult to determine a threshold of rejection between such similar values.

Conversely, μ_3 makes a good criterion because the values for the face and the photograph are 0.937 and 0.204, respectively. As well as μ_1, μ_3 is also effective for rejecting other subjects (P1, ..., P10).

4 Conclusion

We have shown that the MSM framework can be extended to 3-D object recognition by using the third largest eigenvalue or the third smallest canonical angle. The viability of the proposed method has been demonstrated by a face recognition pilot study. Quantitative evaluation of this method with a large data set will be considered in future work.

Acknowledgments

We thank Professor (Emeritus) T. Iijima, Professor H. Ogawa, and Professor M. Sato at Tokyo Institute of Technology, and Professor E. Oja at Helsinki University of Technology for fruitful discussions on subspace distance. We thank Professor R. Cipolla at Cambridge University, Professor A. Maki at Kyoto University, and Dr. P. Wyatt at Toshiba Corporation for their help in writing this paper.

References

1. Murase, H. and Nayar, S. K.: Illumination planning for object recognition in structured environments. *Proc. of CVPR* (1994) 31–38
2. Oja, E.: *Subspace Method of Pattern Recognition*, Research Studies Press (1983)
3. Maeda, K. and Watanabe, S.: A Pattern Matching Method with Local Structure *Trans. IECE* **J68-D 3** (1984) 345–352 (in Japanese)
4. Yamaguchi, O., Fukui, K., and Maeda, K.: Face Recognition Using Temporal Image Sequence. *Proc. of FG98* (1998) 318–323
5. Oja, E. and Parkkinen, J.: On Subspace Clustering. *Proc. of 7th ICPR* (1984) 692–695
6. Riittinen, H.: Short-cut Algorithms for the Learning Subspace Method. *Proc. of ICASSP 84* (1984) 17.2.1–17.2.4
7. Dixmier, M.: Etude sur les Varietes et les Operaterns de Julia, avec Quelques Applications. *Bull. Soc. Math. France* **77** (1949) 11–101
8. Björck, Å. and Golub, G. H.: Numerical Methods for Computing Angles between Linear Subspaces. *Mathematics of Computation* **27** (1975) 579–594
9. Chatelin, F.: *Valeurs Propres de Matrices*, Masson (1988)
10. Fukui, K. and Yamaguchi, O.: Face Recognition Using Multi-viewpoint Patterns for Robot Vision. *Proc. of ISRR03* (2003)
11. Epstein, R. et al: 5 ± 2 Eigenimages Suffice: An Empirical Investigation of Low-Dimensional Lighting Models. *Proc. of IEEE Workshop on Physics-based Modeling in CV* (1995) 108-116

A General Strategy for Hidden Markov Chain Parameterisation in Composite Feature-Spaces

David Windridge, Richard Bowden, and Josef Kittler

Centre for Vision, Speech and Signal Processing
Dept. of Electronic & Electrical Engineering, University of Surrey,
Guildford, GU2 5XH Surrey, UK
Tel.: +44 1483 876043
d.windridge@eim.surrey.ac.uk

Abstract. A general technique for the construction of hidden Markov models (HMMs) from multiple-variable time-series observations in noisy experimental environments is set out. The proposed methodology provides an ICA-based feature-selection technique for determining the number, and the transition sequence, of underlying hidden states, along with the statistics of the observed-state emission characteristics. In retaining correlation information between features, the method is potentially far more general than Gaussian mixture model HMM parameterisation methods such as Baum-Welch re-estimation, to which we demonstrate our method reduces when an arbitrary separation of features, or an experimentally-limited feature-space is imposed.

1 Introduction

1.1 Objective

We shall set out a generic methodology for the automated generation of maximally generalising hidden Markov models given an initial time-sequence (or possibly Markovian) model of observed feature-space transitions within a noisy experimental environment. That is, for any experimental situation where multiple measurements are associated with individual temporally-ordered observation states, and where the observed state-transition mechanisms are governed to any extent by stochastic processes detrimental to classification, we propose to derive a method for automated hidden Markov model (HMM) generation that will provide maximal generalisation under the stated experimental assumptions.

In order to qualify as more general than existing HMM parameterisation techniques, the elaborated methodology might be expected to incorporate existing methods as a limiting subset: when measurement modalities are artificially constrained, we will therefore, in section 2.21, demonstrate an equivalence between the results of our technique and that of the Gaussian mixture-model estimation methods (such as Baum-Welch re-estimation [1]).

Other issues arise in the application of this method, particularly the possibility of sub- and super-sampling of hidden states, with their attendant consequences for classification performance; these, however, fall beyond the scope of the current paper, and will be dealt with in a future publication. Also reserved for future publication is experimental evidence for the utility of this method, demonstrating an approximately 50 percent performance improvement over 'naive' Markovian modelling of observed transitions in typical experimental spaces: indeed, the method may be considered as a technique for transforming naturally between Markovian and hidden-Markovian models of a particular feature-space.

1.2 Overview of Technique

The mechanics of the proposed methodology will centre on determining both the number of, and the transition sequence of, a set of putative hidden states underlying experimental scenarios characterised by a certain common type of noise model, in addition to determining the emission characteristics of the corresponding observation states. More specifically, the technique for generating the HMM structure derives from the many-to-one mapping that spontaneously arises between the space of observed feature-space transitions and the corresponding 'hidden' state-transitions occurring in a subspace defined by the underlying 'operational' manifold specific to the nature of the experimental system under consideration. Most importantly, this mapping will be of an essentially stochastic nature, as required by the relation of hidden and observed states in hidden Markov modelling [cf eg. 2]. It is this stochasticity, in particular, that accounts for the failure of purely Markovian descriptions of observed feature-space transitions to generalise within such experimental environments, requiring the utilisation of hidden Markov techniques for optimal classification.

Since there will, in general, be several distinct noise sources represented within an observed feature-space, characterised by their collective stochastic independence, the technique adopted to bring about the distinction of the signal and noise components consists in an initial independent component analysis (ICA) [enacted via coordinate transformation], followed by a secondary feature-selection (FS) procedure, this process being sufficient to completely eliminate all of the independent noise features. It is, in particular, only the prior ICA-transformation stage that permits feature-section to remove the independent noise sources. Without this step, since the original features contain both information and noise signals, feature-selection would, in removing both simultaneously, invariably act to degrade overall classification performance.

Once this ICA+FS procedure has taken place, there then exists an implicit many-to-one mapping between the observed feature-space and the optimally-compactly described signal-space arising from the procedure. However, since the observed state-space into which the signal subspace is mapped is defined by the noise features removed by feature-selection, these mappings are dictated purely by chance, and not by the *specific* configuration of the noise vectors in the training set. Thus it becomes possible to ignore the particular transition structure of the mapped states and consider rather a stochastic model of the behaviour of observed states with respect to the 'hidden' states in the ICA+FS space. We have thus all the ingredients required, given an observed set of feature-space transition sequences, to arrive at a maximally generalising hidden Markov model in which a naturally occurring distinction between hidden states and observed states is modelled by some emission probability. Moreover, once the ICA+FS procedure has taken place, it becomes possible to eliminate it entirely from the classifying process, working only with the HMM model so derived. Setting out precisely how this process takes place will be the concern of the next section of this paper.

2 Details of Methodology

2.1 ICA Description

The first requirement of our technique is the ICA transformation of the observed pattern space: we shall give a very brief overview of this process as follows (with more detailed and general treatments being available in, for instance, [3] and [4]): Let $\boldsymbol{t}$ be a vector describing the state of the f_n individual feature channels over the complete temporal range, t. We wish to describe this vector in terms of the fully independent feature-set vector, $\boldsymbol{t'}$, of presently unknown dimensionality. The two vector quantities are related by a mixing matrix, M, via $\boldsymbol{t'} = M\boldsymbol{t}$. The determination of the matrix M is hence the objective of our calculation, the first stage of which is the decorrelating (or 'whitening', 'sphering') of the original input space. That is, we shall require a linear transformation $\boldsymbol{t_w} = W\boldsymbol{t'}$ such that the expectation $E(\boldsymbol{t_w}\boldsymbol{t_w}^T)$ is equal to I (I being the identity matrix). A simple solution to this constraint exists via the expansion:

$$I = E(\boldsymbol{t_w}\boldsymbol{t_w}^T) = E(W\boldsymbol{t'}[W\boldsymbol{t'}]^T) \tag{1}$$

$$= E(W\boldsymbol{t'}\boldsymbol{t'}^T W^T) = E(W[\boldsymbol{t'}\boldsymbol{t'}^T]W^T) \tag{2}$$

Setting $S = E(\boldsymbol{t'}\boldsymbol{t'}^T)$, we observe that $W = S^{\frac{1}{2}}$ fulfils the terms of the constraint, the last term in the equation becoming $S^{\frac{1}{2}}SS^{\frac{1}{2}}(= I)$. Having found a suitable W, it only remains to perform a rotation of the whitened pattern-space axes such that the non-Gaussianity of the probability distribution of the individual variates of the transformed space is maximised (linear mixtures of variates being invariably more Gaussian than their components via the central limit theorem). This is usually achieved via an appropriate statistical, information theoretic or morphological measure of non-Gaussianity, and any one of a number of algorithms for finding global maxima/minima.

2.2 Experimental Noise in the Context of Independent Component Analysis

A necessary preliminary for the detailing of our technique is to consider the manner in which experimental noise relates to the above processes. Perhaps the paradigmatic illustration of the application of ICA techniques in this context is the 'blind' discrimination of spatially-distinct sound sources by multiple randomly-placed microphones. Within the feature-space (technically a Hilbert space) defined by the simultaneous composition of the microphone outputs, each of the various signal sources has associated with it a characteristic vector, the orientation of which is determined by their placement relative to the various microphones. This is not an exact replication of the physical geometry of the situation, since each sound source's amplitude is attenuated by the inverse-square of the distance between itself and the microphone in question; however, it is a linear transformation thereof. Critically, when more than one source is considered, the individual fixed-orientation vectors are combined via linear addition of the components (it is this quality, along with the presence of an inner product term [to be introduced later], that defines the feature-space as a Hilbert space). That is:

$$\boldsymbol{S} = S^1\hat{S}_1 + S^2\hat{S}_2 + \ldots + S^s\hat{S}_s \tag{3}$$

denotes the total source feature vector, with the $\hat{S}_i$ unit vectors denoting individual signal source 'orientations'. The coefficients S^i vary over the range $[0 : S^i_{\max}]$, defined by the maximum signal output and source/microphone geometry.

Since the s sound sources are completely independent, the *noiseless* signal vector consequently describes a uniform Euclidean subspace $\mathcal{S}$ within the original feature-space, defined by tensor multiplication of the various source vectors:

$$\mathcal{S} = S^1_{\max}\hat{S}_1 \otimes S^2_{\max}\hat{S}_2 \otimes \ldots \otimes S^s_{\max}\hat{S}_s \tag{4}$$

(written henceforth as: $\mathcal{S} = \bigotimes_{i=1}^{s} S_i\hat{S}_i$). In our example, the origin of this subspace embedding is straightforwardly inferred from the geometry already inherent in the nature of the experimental situation, with the signal/microphone positioning characterising the unit vectors composing the feature subspace. This behaviour is very much more general, however, and precisely describes *any* situation in which features are composed of differing linear signal combinations. Crucially, it also approximates the very much wider class of situations in which the feature subspace *topology* is identical to that of a projective hyperplane.

We wish now to establish exactly how stochastic noise modifies this signal geometry. This can be encompassed within the above regime by considering white-noise emitting speakers placed at random locations within the experimental geometry. Each of the noise sources has again associated with it a characteristic orientation vector determined by its relative geometry, written $N^i\hat{N}_i$, where $\hat{N}_i$ is the noise source unit vector and N^i a uniform random variate distributed over the range $[0 : N^i_{\max}]$. These orientation vectors combine via tensor multiplication in the same manner as the signal vectors, increasing the dimensionality of the signal manifold $\mathcal{S}^{\mathcal{N}}$ by the number of independent noise sources, n:

$$\mathcal{S}^{\mathcal{N}} = \left(\bigotimes_{i=1}^{s} S_i\hat{S}_i\right) \otimes \left(\bigotimes_{i=1}^{n} N_i\hat{N}_i\right) \tag{5}$$

The actual, measured feature vector $\boldsymbol{V}$ is thus composed of the vector sum; $\boldsymbol{S} + \boldsymbol{N}$, the latter term denoting the pure noise vector.

Applying ICA-transformation to the *distribution* over $\mathcal{S}^{\mathcal{N}}$ performs an optimal coordinatisation of the manifold, in the sense of generating a hyper-space over which state vectors are uniformly distributed, and entirely eliminating the vector magnitude coefficients:

$$\mathcal{S}^{\mathcal{N}'} = \left(\bigotimes_{i=1}^{s} \hat{S}_i{}'\right) \otimes \left(\bigotimes_{i=1}^{n} \hat{N}_i{}'\right) \tag{6}$$

Here the $\hat{S}_i{}', \hat{N}_i{}'$ are *orthogonal* unit vectors, which is to say they obey the constraints $\hat{S}_i.\hat{S}_j = 0$, $\hat{S}_i.\hat{N}_j = 0$, $\hat{N}_i.\hat{N}_j = 0 \; \forall i \neq j$: dashed terms will in general indicate ICA-transformed quantities throughout the paper.

Hence we see that signal and noise sources, being independent at both the collective and individual levels, are orthogonalised with respect to each other via ICA-transformation. Under these circumstances it is possible to *selectively* eliminate individual signal and noise features from the space. In particular, it becomes possible to perform feature-selection on the transformed space via a classification-performance based criterion such as forward searching [5]. Since the noise dimensions can only degrade classification performance, we would naturally expect the terminating output of the feature selection procedure to be the subspace $\mathcal{S}' = \hat{S}_1{}' \otimes \hat{S}_2{}' \otimes \ldots \otimes \hat{S}_s{}'$, representing perfect noise elimination. This immediately implies a projective mapping of observational states:

$$\bigotimes_{i=1}^{s} \hat{N}_i{}' \hat{N}_1{}' \to \mathbf{1} \tag{7}$$

In fact there is a very much more inclusive projection than this (in the sense of a mapping of states beyond those present in the operational manifold) implied by the feature-selection from f_n-dimensions to s-dimensions, given by the orthogonal complement space mapping:

$$\left(\bigotimes_{i=1}^{s} S^i_{\max} \hat{S}_i\right)^{\perp} \to \bigotimes_{i=1}^{s} S^i_{\max} \hat{S}_i \tag{8}$$

This does not represent a difference in state mappings from equation 7 for the training data, but does so for test-data with noise characteristics differing from the training set (caused, perhaps, by insufficiently large training sets). From the point of view of the construction of hidden Markov models, however, we shall consider only the identified noise features in defining the emission characteristics of the hidden states in order to give precise statistical descriptions.

It is thus our central contention that the many-to-one mapping of the observed states to signal states represented by equation 7, without which observational state transitions would be partially stochastic, represents (in reverse) the emission spectrum for the 'hidden' state transitions that take place within the embedded signal space, $S^1_{\max}\hat{S}_1 \otimes S^2_{\max}\hat{S}_2 \otimes \ldots \otimes S^s_{\max}\hat{S}_s$. That is, critically, the various transitions that take place between the noise vector component of the total observed feature vector $\boldsymbol{S} + \boldsymbol{N}$, ie $\boldsymbol{N}^1 \to \boldsymbol{N}^2 \to \ldots$ for times $T = 1, 2, 3$, are of a purely random nature, and may be modelled by an emission probability density over states determined by the signal vector $\boldsymbol{S}$. In essence, by removing ICA noise *components* in the feature-selection, we have thus replaced the $\overline{\text{particular}}$ mapping of observed-to-signal-subspace states present in the training-set by a more general mapping that can only be interpreted statistically, which is to say via a *hidden* Markov process of random emission of observation states.

In more conventional Markovian terminology [cf eg 2] a Markov chain of observed state transition coefficients, $a_{(\boldsymbol{S}+\boldsymbol{N})^t(\boldsymbol{S}+\boldsymbol{N})^{t+1}}$, is here transformed to an HMM with *hidden* state transition coefficients;

$$a'_{(\boldsymbol{S})^t(\boldsymbol{S})^{t+1}} = \int_{\forall \boldsymbol{N}_1} \int_{\forall \boldsymbol{N}_2} a_{(\boldsymbol{S}^t+\boldsymbol{N}_1)(\boldsymbol{S}^{t+1}+\boldsymbol{N}_2)} d\boldsymbol{N}_1\boldsymbol{N}_2 \tag{9}$$

and state emission probabilities:

$$\left\{ \begin{array}{c} b_{\boldsymbol{S}^t}(\boldsymbol{V}) = 1/|\bigotimes_{i=1}^n N^i_{\max}\hat{N}_i| \\ \text{if } \boldsymbol{V} - \boldsymbol{S}^t \in \{\bigotimes_{i=1}^n N^i_{\max}\hat{N}_i\};\ N^i \in \Re \end{array} \right.$$

$$\left\{ \begin{array}{c} b_{\boldsymbol{S}^t}(\boldsymbol{V}) = 0 \\ \text{if } \boldsymbol{V} - \boldsymbol{S}^t \notin \{\bigotimes_{i=1}^n N^i_{\max}\hat{N}_i\};\ N^i \in \Re \end{array} \right.$$

We will demonstrate in the next section that when the observed features are considered as separate data streams (which is to say, when the vector decomposition $\boldsymbol{V} \rightarrow \{\boldsymbol{V_1}, \boldsymbol{V_2}, \boldsymbol{V_3} \ldots\}$ takes place), the above emission spectrum defaults exactly to the Gaussian emission spectra typically assumed by conventional hidden Markov model parametrisation techniques, and hence that the above method serves as a Gaussian mixture model parameteriser under these circumstances.

We reiterate that while the above analysis deals with an idealised noise model, to the extent that the underlying conditions of additive linear noise hold, substantial benefit may still be gained by application of the technique in the fully general case. Indeed, empirical testing on data for which no *a priori* guarantee of the validity of the model assumptions can be made (to be published at a later date), nonetheless yields very substantial performance improvement on application of the ICA+FS technique.

2.3 Relation to Gaussian Mixture-Model Parameterisation Methods

It is frequently found in the experimental domain that the total feature-space is naturally divided into separate subspaces (for instance, division of the physical configuration-space into position and velocity vectors). We will aim to demonstrate in this subsection that, under these circumstances, our methodology approximates that of the Gaussian mixture-model parameterisation methods (such as Baum-Welch re-estimation). In carrying out this exercise, it should be emphasised that the generality of our method means that optimal results are always to be obtained by generating a composite feature-space (ie, one in which the contributing subspaces are appended) prior to application of the ICA+FS method:

however it is reassuring to note that imposing this limitation reproduces familiar, empirically-validated methodologies.

We shall initially consider only a single dimensional marginal projection for simplicity of demonstration, later providing an argument for its generalisation to higher dimensionality feature space projections (such as those required to specify independent position and velocity vectors). A more general indication of the tendency of oblique projections of uniform multivariate distributions to approximate Gaussian distributions is given explicit proof in [6].

Firstly, let the chosen marginal feature, D_c, be characterised by its unit vector, $\hat{D}_c$, in the composite observed space. Measurements of observed states (defined by the projection $\hat{D}_c.\boldsymbol{V}$) are thus characterised, for a given signal vector $\boldsymbol{S}_g$, by the sum of the individual random variates $\hat{D}_c.N^i\hat{N}_i$ and the signal vector marginal projection, $\hat{D}_c.\boldsymbol{S}_g$. That is to say:

$$\hat{D}_c.\boldsymbol{V} = \sum_{i=1}^{i=n} \hat{D}_c.N^i\hat{N}_i + \hat{D}_c.\boldsymbol{S}_g$$

These random noise variates are uniformly distributed over the range $[\hat{D}_c.\boldsymbol{S}_g : \hat{D}_c.\boldsymbol{S}_g + \hat{D}_c.\hat{N}_iN^i]$, and hence have a mean $\hat{D}_c.\boldsymbol{S}_g + \hat{D}_c.\hat{N}_iN^i_{\text{max}}/2$ and variance $(\hat{D}_c.\hat{N}_iN^i_{\text{max}})^2/12$. Now, according to the central limit theorem, the composite variate:

$$X = \frac{\sum_{i=1}^n x_i - \sum_{i=1}^n \mu_i}{\sum_{i=1}^n \sigma_i^2} \tag{10}$$

with *individual* means, μ_i, and variances, σ_i , obeys a Gaussian distribution with mean 0 and variance 1 in the limit $n > \infty$, on the proviso that a number of constraints on the individual variate probability density functions are observed, all of which are fulfilled by the uniform distribution.

Comparing this to the variate sum $\hat{D}_c.\boldsymbol{V}$ above, it is clear that we only require a coordinate transform of:

$$\hat{D}_c.\boldsymbol{V} \rightarrow \frac{(\hat{D}_c.\boldsymbol{V} - \sum_{i=1}^n \hat{D}_c.\hat{N}_iN^i_{\text{max}}/2 - \hat{D}_c.\boldsymbol{S}_g)}{\sum_{i=1}^n (\hat{D}_c.\hat{N}_iN^i_{\text{max}})^2/12}$$

to bring about an equivalence. Hence, our marginal variate $\hat{D}_c.\boldsymbol{V}$ becomes asymptotically normally-distributed with mean $\hat{D}_c.\boldsymbol{S}_g + \sum_{i=1}^n \hat{D}_c.\hat{N}_iN^i_{\text{max}}/2$ and variance $\sum_{i=1}^n (\hat{D}_c.\hat{N}_iN^i_{\text{max}})^2/12$, obeying the overall formula:

$$P(\hat{D}_c.\boldsymbol{V}|\boldsymbol{S}_g) = \frac{e^{\left[\frac{(\hat{D}_c.\boldsymbol{V} - (\hat{D}_c.\boldsymbol{S}_g + \sum_{i=1}^n \hat{D}_c.\hat{N}_iN^i_{\text{max}}/2))^2}{2\left(\sum_{i=1}^n (\hat{D}_c.\hat{N}_iN^i_{\text{max}})^2/12\right)}\right]}}{\left[2\sum_{i=1}^n (\hat{D}_c.\hat{N}_iN^i_{\text{max}})^2/12\pi\right]^{\frac{1}{2}}} \tag{11}$$

Thus our mapping of the marginal observational states onto the 'hidden' states of the ICA+FS transformed *total* observational configuration space implies the existence of a hidden Markovian model of Gaussian observed state emission characteristics (of the kind familiar from, in particular, automated speech recognition). It should be emphasised that this form is in no way imposed *a priori*; the relationship between, and distribution of, hidden and observed states is determined purely by the output of the ICA+FS procedure.

The f_n-dimensional generalisation of the above argument for single-dimensional marginal projections is complicated by the issue of covariance. However, it is possible to see that there must always exist a coordinate transformation of the f_n-dimensional space such that each of the noise source vectors $N^i\hat{N}_i$ supplies an equivalent aggregate contribution to the signal vector: that is, a transformation such that unit vectors in the transformed space, $\hat{u}_i$, satisfy the constraint: $< \hat{u}_i.\hat{N}_j >=$ constant $\forall i,j : 1 \leq i \leq f_n,\ 1 \leq j \leq n$. Full details of the derivation of this transformation, T, are set out in [7]: critically, the coordinate transformation is both linear and invertible.

That the signal-vector distribution in the transformed space is now composed of normalised uniform distributions again implies, via the central limit theorem, that they collectively approximate symmetric Gaussian distributions of variance: $1/(\sqrt{2^{f_n-1}}12)$ for each marginal distribution of the transformed space. Since these marginal distributions are independent of each other, we can take their tensor product in order to recover the distribution over the full transformed space. Consequently, since the tensor product of Gaussian distributions is itself Gaussian, we find that the overall distribution is a covariance-less $(F-n)$-dimensional Gaussian distribution over the vectorial range of the transformed space. The bijective linear transformation back into the original space, T^{-1}, maintains this Gaussian form, but generates a unique covariance matrix Σ defined by the various shear and rotation deformations implicit in T^{-1} [1].

It is hence this potential to constrain unique Σ and $\boldsymbol{\mu}$ values that constitutes the effective Gaussian parameterisation of the ICA+FS method in restricted spaces, and brings it in to an operational equivalence with the Gaussian mixture-model estimators such as Baum-Welch re-estimation, which typically [cf eg. 2] assume models of the form:

$$b_j(\boldsymbol{o}_t) = \prod_{w=1}^{W} \left[\sum_{m=1}^{M_w} c_{jwm} \mathcal{N}(\boldsymbol{o}_{wt}; \boldsymbol{\mu}_{jwm}, \Sigma_{jwm}) \right] \tag{12}$$

with;

$$\mathcal{N}(\boldsymbol{o}; \boldsymbol{\mu}, \Sigma) = \frac{1}{((2\pi)^n |\Sigma|)^{\frac{1}{2}}} e^{\frac{1}{2}(\boldsymbol{o}-\boldsymbol{\mu})' \Sigma^{-1} (\boldsymbol{o}-\boldsymbol{\mu})} \tag{13}$$

(M is the number of Gaussians in the mixture, W the number of distinct observational feature-spaces, and $\boldsymbol{o}$ the within-stream observational vector).

[1] It is additionally the case that multiple Gaussians may by formed by this type of marginal projection where there exist distinct signal regions within the observational space, as might be obtained by regional application of the ICA+FS procedure. The key requirement for Gaussianisation of the emission states in either scenario is that there exists a *projective* mapping from a higher to a lower dimensional space.

3 Conclusions

We have set out a technique for automatically obtaining parameterised hidden Markov models from observed feature transition sequences through a process of feature-selection in an ICA-transformed space, on the assumption of the presence of independent noise sources within the data. This involves the mapping of observed states onto a space of hidden states induced by the orthogonalisation and removal of the noise sources within the observed features. The stochasticity of this mapping accounts for the generated entity being a *hidden*, rather than a simply Markovian model, and thus allows a much greater generalising capacity than would a simple Markovian description of the observed transitions.

When an artificial distinction between data streams (such as position and velocity) is assumed, or when the observed feature-space represents an incomplete description of the free experimental parameters, the observed state distribution in the HMM models generated by the ICA+FS technique was shown to default to a Gaussian emission model. The proposed methodology under these circumstances thus reproduces the familiar Gaussian mixture model parameterises, without, however, having to assuming this form on an *a priori* basis. The technique is thus very much more general than existing HMM parameterisation techniques, having fewer underlying assumptions, and can be treated as first recourse when the nature of the emission state/hidden state transition relationships are unknown. Empirical studies suggest a consistent 50% performance improvement over equavalent 'naive' Markovian methods.

Acknowledgements

This paper gratefully acknowledges the financial support of INRIA within the context of the Cognitive Vision Systems framework collaboration.

References

1. L. Baum and J. Eagon, American Mathematical Society Bulletin, 73:360-363, 1967.
2. Steve Young, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland. The HTK Book for HTK Version 2.1. Cambridge University, 1997.
3. A. Hyvärinen, E. Oja, "Independent Component Analysis: Algorithms and Applic ations", Neural Networks, 13(4-5):411-430, 2000
4. P. Comon, "Independent component analysis, A new concept?," IEEE Signal Processing Mag., vol. 36, no. 3, 1994.
5. P.A. Devijver and J. Kittler, "Pattern Recognition: A Statistical Approach", Prentice-Hall, 1982
6. Asymptotics of graphical projection pursuit, P. Diaconis & D. Freedman, Ann. Stat., vol. 12, 793-815
7. D. Windridge, R. Bowden, J. Kittler, (Univ. of Surrey Technical Report: VSSP-TR-3/2004), UNIS, UK

Tracking the Evolution of a Tennis Match Using Hidden Markov Models

Ilias Kolonias, William Christmas, and Josef Kittler

Center for Vision, Speech and Signal Processing,
University of Surrey,
Guildford GU2 7XH, UK
{i.kolonias,w.christmas,j.kittler}@surrey.ac.uk
http://www.ee.surrey.ac.uk/CVSSP/

Abstract. The creation of a cognitive perception systems capable of inferring higher-level semantic information from low-level feature and event information for a given type of multimedia content is a problem that has attracted many researchers' attention in recent years. In this work, we address the problem of automatic interpretation and evolution tracking of a tennis match using standard broadcast video sequences as input data. The use of a hierarchical structure consisting of Hidden Markov Models is proposed. This will take low-level events as its input, and will produce an output where the final state will indicate if the point is to be awarded to one player or another. Using hand-annotated data as input for the classifier described, we have witnessed 100% of the points correctly awarded to the players.

1 Introduction

The area of automatic data parsing from multimedia sequences has been one of great research interest for many years now. Such applications can be very useful for a wide range of purposes – most of them concerning automatic annotation and indexing of these sequences according to the criteria one might wish to specify. Especially for today's broadcasters, where the enormous amount of audiovisual information coming in to them can *only* mean more time spent on indexing what they store in their archives, such tools would greatly improve their ability to manage those archives in an efficient manner. Another important reason for this is the fact that automatic annotation of multimedia content will help us preserve it and re-use it through time.

More specifically though, sports events are very common in broadcast television, and also usually have a quite well-defined structure. Sports will either run against the clock and have some distinctive events as reference points (like football or basketball have scoring) or will be solely based on specific events which will define their evolution through time (like tennis or volleyball have scoring). Therefore, the underlying structure of each sport, which is largely dictated by its rules, is strong enough to allow for such approaches to be designed and more efficiently implemented.

In this work, we focus on trying to extract high-level information from processing low-level input data for a specific kind of content – video sequences of tennis matches. The choice of a clearly event-driven sport will make things easier in this analysis by not explicitly involving time. Besides, time-constrained sports can be considered as event-driven ones if we consider a set of *'time'* events being triggered at regular intervals throughout the sequence. The layout of this paper is as follows: in the following section, a short summary of some relevant work on scene evolution tracking in video sequences and event recognition in sport videos is presented; Section 3 is a presentation of the proposed system; the results from the application of the proposed system to a hand-annotated data set are presented and discussed in Section 4; and finally, conclusions and ideas for future work are given in Section 5.

2 Related Work

As we can see in literature, HMMs and other forms of Dynamic Bayesian Networks (DBNs) in general are very powerful tools in the area of machine learning. Hence, a great number of cognitive visions applications have deployed such tools in order to perform information extraction from video sequences. Some of those attempts would include those of [1] for automatic annotation of Formula 1 race programs, [2] for recognising various types of strokes from tennis players during a tennis match, [5] for the recognition of general hand gestures, [3] for the recognition of American Sign Language and [4] for general human pose estimation. Apparently, such pieces of work may prove to be extremely useful in our case as well, and they can certainly serve as a guideline of what is feasible in cognitive vision, and what issues in this area are still open.

In the work by Ivanov and Bobick [5], we can see that the authors introduced a framework by which they analyse each complex event into its constituent elementary actions; in one of the examples the authors have used, a gesture is broken down into simple hand trajectories, which can be tracked more successfully via HMMs. Then, they apply *Stochastic Context-Free Grammars* to infer the full gesture. The results reported show that the proposed system performed quite well on real-world data. Such a paradigm can be considered as quite similar to that of a tennis match; if we consider all elementary events leading up to the award of a point in a tennis match to be the equivalent of the elementary gestures in this work, and the tennis rules related to score keeping as an equivalent of the grammar-based tracking of the full gesture the authors have implemented, we can easily see the underlying similarities between the authors' work and reasoning on tennis video sequences.

Another piece of work that deals with sets of body movements that are by definition constrained is that reported by Starner *et al.* in [3]. In this case, the objective was to develop a system that will be capable of recognising a gesture language that is the American Sign Language (ASL). The main idea behind this is the use of Hidden Markov Models to 'learn' a small lexicon of words in ASL, as they are expressed through gestures. The low-level visual information (in this case, the hands that has been segmented out of the input sequence) are separately analysed in terms of their shape, and this information is fed into the HMMs. The output is the word recognised by the system, and it can be seen that (at least for a small vocabulary of gesture-words) the system has performed very well. This piece of work, along with that discussed in the previous paragraph, show us that HMMs can be quite successfully applied in recognising shapes and tracking object trajectories – which are very important parts of analysing tennis video sequences at a lower level.

Moreover, in the work of Rosales and Sclaroff [4], a more general problem concerning human body pose estimation has been addressed. Through the use of an *artificial neural network* (ANN), the authors have initially attempted to classify the various possible configurations of the human body by initially capturing 3-D body positions, which they the project into 2-D (namely, a set of image planes). Given these 2-D projections, they have formulated exclusive subsets via unsupervised clustering, using the *Expectation-Maximisation* (EM) algorithm. The results of this approach have proven to be quite promising as well, showing us that we *can* use reasoning tools to distinguish between different body poses.

In [1], we can see how an overall cognitive vision system for sport videos has been developed and deployed. In this case, the authors have attempted to isolate semantic information from *both* the audio and the visual content of the sequence, and tried to annotate the video sequences processed by detecting events perceived as highly important; for example, and bearing in mind what events can occur in Formula 1 racing, they attempted to cover visual events such as overtaking, cars running out of the road etc. In addition, as the sequences used came from live broadcasts, they also included textual information about the race, like drivers' classification and times; that information was also extracted and used. The audio part of the sequences, since they came from normal broadcasts, was dominated by the race commentary; out of that, features like voice intensity and pause rates were also used. Having performed all these operations, the authors attempted to infer events of semantic importance through the use of Dynamic Bayesian Networks, attempting to infer content semantics by using audio and video information separately or combining this information in the temporal domain; both approaches yielded promising results when tested on simple queries (like finding shots in the Formula 1 race where a car runs out of the race track)

In another piece of work by Petkovic *et al.* [2], we can see a piece of work that is somewhat more constrained that that described in the previous paragraph, but which is still quite useful in terms of understanding the scene described, and which is obviously much more relevant to the present work. In this case, the authors have chosen to use HMMs in order to determine how the player hits the tennis ball; hit types would include forehands, backhands, serves, etc. To do this, the authors initially segmented the players out of the background (that is, the tennis court); then, they used Fourier Descriptors in order to describe the players' body positioning; finally, they trained a set of Hidden Markov Models to recognise each type of hit. The results of this work show that this method can be quite successful in performing the recognition task it was designed for.

Finally, another relevant piece of work is that of Kijak *et al.* [6], where the authors have attempted to analyse the structure of a tennis video through the use of Hidden Markov Models, as well as fuse audio and visual cue data in order to perform reasoning. Their objective was to separate a tennis video sequence into a set of scenes, each of which had to be classified under one of the following categories:

– First missed serve
– Rally
– Replay
– Break – that is, a *commercial* break

The visual cues include a vector of dominant colours and their respective spatial coherencies, and a measure of camera motion, whereas the audio cues form a binary feature vector where speech, applause, ball hits, noise and music are shown to be detected (or not). The results of the semantic segmentation system the authors proposed in this paper also seem to be quite promising.

Nonetheless, this is just a small subset of the applications these inference tools have been tested upon, and we can observe that all of these systems tend to perform the recognition and/or reasoning operations they were designed to do quite well. Therefore, it can safely be assumed that similar methods are capable of producing satisfactory results in other similar problems, like the one we are asked to address in this paper.

3 Proposed Scheme

In our context (the analysis of tennis video sequences), the rules of the game of tennis provide us with a very good guideline as to what events we will have to be capable of tracking efficiently, so as to follow the evolution of a tennis match properly. The full graphical model for the evolution and award of a point in a tennis match is given in the graph of Figure 1.

As we can readily see from this diagram, it is a graphical model where a number of loops exist; the state transitions drawn with bolder lines indicate where these loops close. Moreover, as it has already been mentioned, this graphical model *only* tackles the problem of awarding a single point in the match; there is some more detail to be added to it if we wish to include the awarding of games, sets or the full match. How these stages are going to be implemented will be discussed in more detail later in this section. Finally, this figure also shows us that, in order to address the problem of 'understanding' the game of tennis more effectively and robustly, we will have to convert this complex evolution graph into a set of simpler structures.

In the game of tennis, we have a case of a hierarchical evolution model. That is, we first need to identify and examine elementary events within the tennis sequence. Such events would include, among others:

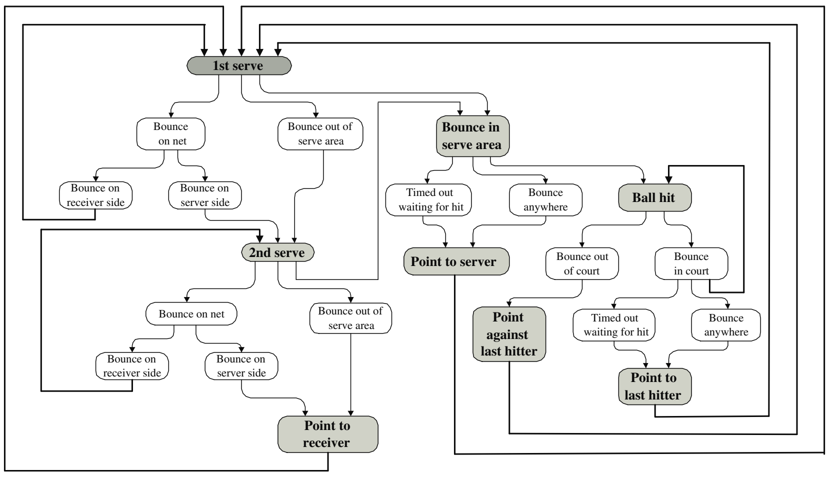


Fig. 1. Graphical model for awarding a point in a tennis match.

- The tennis ball being hit by the players
- The ball bouncing on the court
- The players' positions and shapes (that is, body poses)
- Sounds related to the tennis match (not implemented yet – however, preliminary experiments show that it *does* contain important semantic information as well)

These events can then be used as a basis on which to perform reasoning for events of higher importance, like awarding the current point from the events witnessed during play. Having successfully performed this step, we can then move on to the award of games, sets and the match to the players involved. Obviously, since the detection of such elementary events will be done using machine vision algorithms and techniques; however, we are bound to encounter event detection errors from this process – a fact which leads us to the conclusion that we will need some kind of correction at a higher level of reasoning to address such errors. Since probabilistic reasoning tools (such as HMMs) can address this problem effectively, they are a natural choice to perform reasoning processes where the input data comes *directly* from the low-level event detectors – which is the case in awarding points.

Therefore, the reasoning process described here is best represented by a hierarchical model. Using such a model will allow us to properly decompose the evolution of the tennis match into smaller events, which can be more concisely defined and tracked with greater accuracy. For example, it would be best if we modelled events of lower conceived importance through an HMM, which would then trigger another HMM to infer on more important events within the game; that would also help us prevent spurious data from low-level feature extraction modules from propagating to higher levels of the inference engine.

Moreover, we can also implement parts of this graph by using a *'switch'* variable to select which, among a number of acyclic sub-graphs, will be appropriately modelling the scene at that moment. Those sub-graphs will contain subsets of the initial, full-scale graph we have just seen. In more detail, we can consider the start of a tennis point (that is, the serve) as the starting state of a *set* of Hidden Markov Models, each of which will follow a particular scenario within the game; for example, the serve could either be successful (so we then 'switch' to the model that deals with a rally of balls) or unsuccessful (where we switch to the model dealing with second serves and double faults). Another useful application of a 'switching' model in this context is the fact that the initial state switches from one player serving to the other (or one player hitting the ball, then the other); therefore, a 'switch' variable between two models that are identical in structure but exactly opposite as to which player they address could help simplify the design and training of all models quite considerably.

Thus, we propose to replace the original scene evolution model with a set of smaller models, each one trying to properly illustrate a certain scenario of the match evolution. The most important thing we need to ensure during this procedure is that, when we combine all of the models in this set, we *must* have a model equivalent to the original one (so that it reflects the rules of tennis). The set of sub-graphs proposed to replace the original one is illustrated in Figure 2.

As we can see from the set of models above, we have opted for a more 'perceptual' way of selecting the set of event chains that will formulate the new

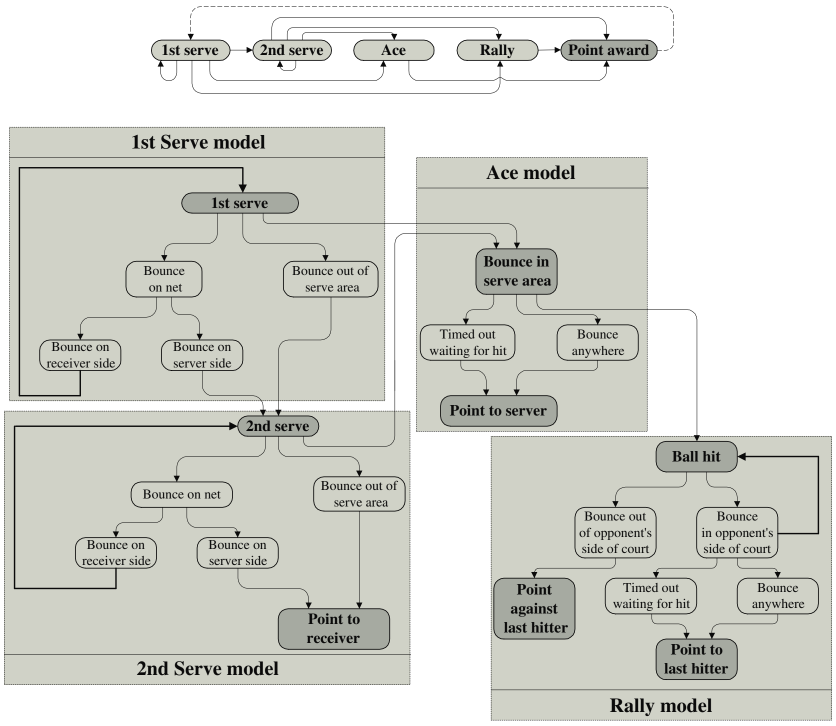


Fig. 2. Switching model and its respective set of sub-models for awarding a point in a tennis match, as separated out from the original graphical model.

model set for our purposes. This design strategy is going to be particularly helpful in the (quite frequent, as it is expected) case where the system receives a query to extract sequences which contain some specific event; since the scene description will consist of a series of events occurring in the match, such queries can be dealt with relatively easily. Moreover, choosing to break the initial graph down to a number of sub-graphs and train each one of them separately will be beneficial in many more different ways. Some of the resulting benefits are:

– Using probabilistic reasoning tools (like HMMs) will allow for correction of wrongly detected elementary events within the game – so we can have a point correctly awarded even if we haven't tracked *all* events accurately from the start.
– Since the models are simpler, we will *not* have to acquire a huge training data set; a relatively small amount of data per model will suffice for our analysis.
– Training will *certainly not* be as time-consuming if we break down the initial graph as it would be if we did not do so. This is because, apart from the training needing more input training data if it were to be trained as a whole, it would also need to calculate more event transition probabilities compared to what would be required if we broke the initial graph to smaller models.
– In some cases, we can *considerably speed up* the training process due to prior knowledge for this type of content – as the amount of statistics available for some events in a tennis match helps us get a very reasonable initial estimate without the need to go through a training procedure – we only need to pick up some existing measurements for this purpose.
– It will be *far easier* for us to determine which specific areas of reasoning need to be improved to boost the overall performance of the system, and which low-level feature extraction modules are more suspect to produce misleading results – so that we can either improve them or altogether discard them.

As soon as we have determined which way the points are going to be awarded, we can move on to the award of games and sets in the match. This can either be done through the use of probabilistic reasoning tools (such as HMMs), or with simpler, rule-based tools – such as grammars. The latter is possible due to the fact that at this level of abstraction in modelling tennis game sequences, it is the rules of the game of tennis that stipulate the evolution of the scene rather than low-level events in it. Anyway, the uncertainty stemming from the successful (or unsuccessful) detection of the elementary events mentioned above is considered to have been effectively addressed in the lower-level stages of the reasoning process – up to the level of point awarding. Therefore, and since it will be an easier and more natural approach to record the rules of tennis via a rule-based tool, we could just as easily opt for using grammars to perform higher levels of reasoning for these video sequences as we could use probabilistic reasoning approaches. An example of the point illustrated in this paragraph could be the way games are awarded in a tennis match out of points won by both sides (assuming that we record the score correctly at all times); if a player has scored *4 or more* points in the current game *and* his/her opponent has *at least 2 points less*, then this player has *won the game* – otherwise the game goes on.

4 Results and Discussion

The architecture described above has been tested on one hour's play from the men's Final of the 2003 Australian Tennis Open. The sequence contained a total of 100 points played – which was the equivalent of approximately one and a half sets of the match. Out of these 100 exchanges, a total of 36 were played on a second serve. The data that was used as input in this experiment were *ground-truth, hand-annotated event chains* from the broadcast match video. Therefore, we have *not* examined the robustness of the proposed method that in this work; we were more interested in its ability to actually model the evolution of the match accurately. To do that, we had to introduce four sets of models – one for every combination of which player serves and which side of the court he/she serves from (left or right).

In those 100 exchanges, we have intentionally left in a few *unfinished* points, so as to examine whether the selection of the hidden states for these models can lead to an accurate representation of the scene at *any* given time – *not only* at the end of the scene. They were 4 in total – 2 leading to a second serve and 2 were cut short while still on play. An overall view of the results is given in the table below.

Table 1. Total results.

	Ground Truth	Correctly Awarded	Wrongly Awarded	Not awarded (still on play)
Near Player Points	59	59	0	0
Far Player Points	37	37	0	0
Unfinished Points	4	4	0	n/a
TOTAL	100	100	0	0

As we can see in Table 1, *all* of the points were successfully tracked by the proposed system. Therefore, it can be easily seen that the performance of this method allows us to use any kind of decision-making scheme (either rule-based or probabilistic) to make decisions about events of higher semantic importance – since that is what is finally intended.

5 Conclusions

As we can readily see from the results shown above, the proposed system has tackled the problem of tracking the evolution of a tennis match very effectively. However, there are still some issues to be addressed in this area. First of all, is is obvious that the excellent performance of the proposed method can partly be attributed to the fact that the input events were came from manual annotation of a tennis video sequence. Thus, it has been verified that they were correct chains of events, either leading to points or not. However, the aim of developing such an automatic evolution tracking system is mainly to be used in conjunction with a set of fully automatic low-level feature extraction tools that will be able to detect the basic events required for input to the proposed system, so that the combined system can be effectively used as an automatic video annotation system. Obviously, as in any fully automatic computer vision system, the low-level feature extraction tools are bound to produce some recognition errors, which will propagate to the proposed decision-making scheme. Therefore, it is essential that the proposed system is tested to effectively address such situations and provide accurate information about the evolution of the game to higher levels of the inference engine.

Moreover, the system can only be considered as the first step in a hierarchical model that will fully describe the evolution of a tennis match – it will only cover the award of a single point in the match. The creation of a full system will include a system to award games, sets and finally the match to the players – which will all rely on the efficiency of this method – on top of it. Therefore, a full system will also have to include a system similar to the one proposed here for point award, which will address that problem effectively as well.

Acknowledgements

This work has been supported by the EU IST-2001-34401 Project VAMPIRE.

References

1. Petkovic, M., Mihajlovic, V., Jonker, W., Djordjevic-Kajan, S.: Multi-modal extraction of highlights from TV Formula 1 programs. In: Proceedings of the IEEE International Conference on Multimedia and Expo. Volume 1. (2002) 817–820
2. Petkovic, M., Jonker, W., Zivkovic, Z.: Recognizing strokes in tennis videos using Hidden Markov Models. In: Proceedings of Intl. Conf. on Visualization, Imaging and Image Processing, Marbella, Spain. (2001)
3. Starner, T., Weaver, J., Pentland, A.: Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 1371–1375
4. Rosales, R., Sclaroff, S.: Inferring Body Pose without Tracking Body Parts. In: Proceedings of the IEEE International Conference Computer Vision and Pattern Recognition (CVPR). Volume 2. (2000) 714–720
5. Ivanov, Y., Bobick, A.: Recognition of Visual Activities and Interactions by Stochastic Parsing. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 852–872
6. Kijak, E., Gravier, G., Gros, P., Oisel, L., Bimbot, F.: HMM based structuring of tennis videos using visual and audio cues. In: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME'03). (2003)

Local Features for Speaker Recognition[⋆,⋆⋆]

Roberto Paredes, Enrique Vidal, and Francisco Casacuberta

Instituto Tecnológico de Informática
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera, s/n, 46022 Valencia, Spain

Abstract. An approach combining a *simple local representation method* with a k-nearest neighbors-based *direct voting scheme* is proposed for speaker recognition. This approach rises computational problems that we effectively solved through an approximate fast k-nearest neighbors search technique. Experimental results with the EuTrans and SIVAspeech databases are reported showing the effectiveness of the proposed approach.

Keywords: Speaker Recognition, Local Features, Nearest Neighbor

1 Introduction

Biometric identification (BI) is presently one of the most active areas in pattern recognition. The local features (LF) approach have been used before to solve problems related to the BI field satisfactorily [1, 2].

Local features are explicitly cited mainly in the image database retrieval literature [5–7], where invariances to scale, rotation and illumination changes are needed. A good BI system has to show a high degree of robustness with respect to the *variability* that the biometrics signals use to reflect. Usually, the biometric signal considered to identify a person can be very different of those used to build the identification system. On the other hand, these signals are often composed by several small objects that bear discriminative information by themselves, almost independently of the other parts. In this context the use of the LF approach is clearly recommended.

Among BI systems *speaker recognition* is one of the most unobtrusive methods, well tolerated by users, and with a wide range of applications. In this paper we propose a combination of a simple local representation along a specially devised feature extraction technique and a direct decision scheme based on k-nearest neighbors. This approach has shown to be most effective if the set of local-feature vectors from the training data is large. This entails a high computational cost, which is avoided by using an approximate fast k-nearest neighbors search technique.

⋆ Work partially supported by the Spanish "Ministerio de Ciencia y Tecnología" under grants TIC2003-08496-C04-02 and DPI2001-0880-CO2-02.

⋆⋆ The authors would like to thank the FUB - Fondazione Ugo Bordoni, for providing the SIVA corpus.

2 Proposed Approach

2.1 Preprocessing and Feature Extraction

The audio speech signal is sampled and represented into vector sequences of Mel-cepstral coefficients. To this end, short-term spectral analysis is performed on short overlapping signal segments (frames). The resulting power spectrum is warped according to the Mel-scale and 11 cepstral-coefficients are derived from each log power spectrum [3].

Figure 1 shows a speech sentence. In the top of the figure, the speech signal of an utterance is represented in the temporal domain. In the middle, the corresponding spectrogram is shown, with frequency presented in the vertical axis, and time in the horizontal axis. Finally, the cepstral coefficients appear in the bottom of the figure. The coefficient number is presented in the vertical axis, and time in the horizontal axis. These cepstral-coefficient vectors are normalized between 0 and 255 in all the experiments carried out in this work.

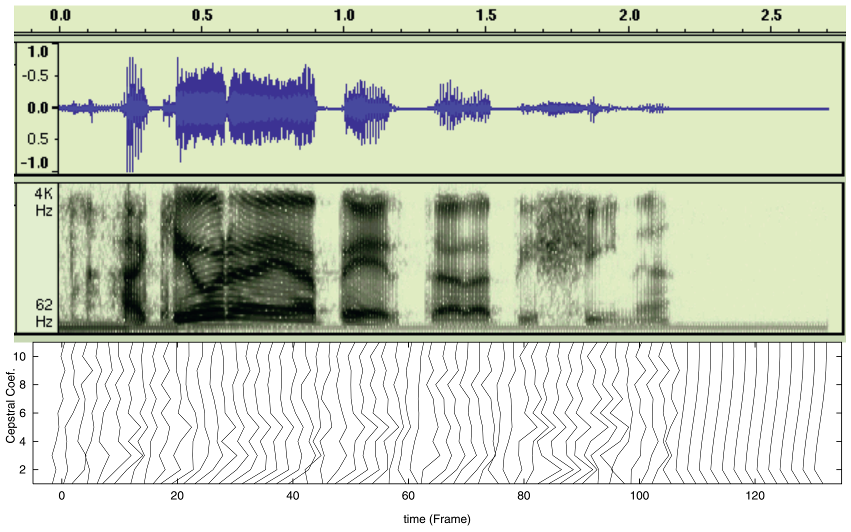


Fig. 1. Top, the original speech signal of a utterance. Middle, the corresponding spectrogram. Bottom, the temporal evolution of the Mel-cepstral coefficients.

Preprocessing consists of two steps. The first step aims at *selecting* those parts of the speech signal with high information content. We have chosen a simple and fast method: the local variance in a small window of the speech signal is measured. Those parts of the signal having local variance above a certain global threshold are selected. The proposed approach based on local variance is applied to these cepstral-coefficient vectors. Around each selected vectors a small window of size w is applied, obtaining a local representation vector of $11 \times w$ cepstral-coefficient components.

The second step aims at reducing the resulting dimensionality using *principal component analysis* (PCA). In this way, a compact local representation of a region of the object is obtained. Finally, we label each vector with an identifier of the class, i.e. the person who uttered the speech sentence considered.

In a classical classifier, each object is represented by a single feature vector, and a discrimination rule is applied to classify a test object that is also represented by a single feature vector. Local features approach, however, implies that each test object is scanned to compute many feature vectors. Each of these vectors can be classified into a different class, and therefore a consensus scheme is required to finally decide a single class for the whole test object.

2.2 Classification through a *k*-NN Based Voting Scheme

The classification procedure used in this work is closely related to a family of techniques often referred to as *"direct voting schemes"* [6]. It is in fact based on applying the well known k-nearest neighbor rule to the set of vectors representing a test utterance, using the local-feature vectors obtained from the training utterances as reference or training set. More formally, we can present the proposed classification technique under the statistical framework of "*classifier combination*" [9].

Let Y be a test speech signal. Following the conventional probabilistic framework, Y can be optimally classified in a class $\hat{w}$ having the maximum posterior probability among all the C classes:

$$\hat{w} = \arg\max_{1 \le j \le C} P(\omega_j | Y) \tag{1}$$

By applying the feature extraction process described in the previous section to Y, a set of m_Y feature vectors, $\{\mathbf{y}_1, \ldots, \mathbf{y}_{m_Y}\}$ is obtained. Thus, we can see the classifier (1) as a *combination of m_Y classifiers*, each for every feature vector of Y. Assuming independence between each $\mathbf{y}_i$, $P(\omega_j|Y)$ could be written as the product of the posterior probabilities associated to every feature vector and (1) becomes:

$$\hat{w} = \arg\max_{1 \le j \le d} \prod_{i=1}^{m_Y} P(\omega_j | \mathbf{y}_i)$$

This is commonly called the *"product rule"* for classifier combination [9].

In order to *smooth* the (poorly estimated) small probabilities the so called "*sum rule*" can alternatively be used for classifier combination [2]:

$$\hat{w} = \arg\max_{1 \le j \le d} \sum_{i=1}^{m_Y} P(\omega_j | \mathbf{y}_i) \tag{2}$$

In our case, posterior probabilities are directly estimated by k-nearest neighbors. Let k_{ij} be the number of neighbors of $\mathbf{y}_i$ belonging to class ω_j. Assuming the average number of reference vectors representing all the training local features of each class is more or less constant, a well known estimate of $P(\omega_j|\mathbf{y}_i)$ is:

$$\hat{P}(\omega_j|\mathbf{y}_i) = \frac{k_{ij}}{k}$$

and, using this estimate in (2), our classification rule becomes:

$$\hat{w} = \arg\max_{1 \le j \le d} \sum_{i=1}^{m_Y} k_{ij} \tag{3}$$

That is, a class $\hat{\omega}$ is selected with the largest number of *"votes"* accumulated over all the vectors belonging to the test biometric signal. This justifies why techniques of this type are often referred to as *"voting schemes"*.

2.3 Efficient Approximate Search for Matching Feature Vectors

The nearest neighbor search is performed by a fast approximate nearest neighbor search algorithm [4]. This algorithm uses a *kd*-tree structure to store the set of local features from the training objects. In a *kd*-tree, the search of the nearest neighbor of a test point is performed starting from the root, which represents the whole space, and choosing at each node the sub-tree that represents the region of the space containing the test point. When a leaf is reached, an exhaustive search of the b prototypes contained in the associated region is performed. Since the closest point may also be a member of some other region, the algorithm needs to backtrack until all possible regions are checked.

If a guaranteed exact solution is not needed, as can be assumed in our case, the backtracking process can be aborted as soon as a certain criterion is met by the current best solution. In [4], the concept of $(1 + \epsilon)$-approximate nearest neighbor query is introduced. A point p is a $(1 + \epsilon)$-approximate nearest neighbor of q if the distance from p to q is less than $1 + \epsilon$ times the distance from p to its nearest neighbor. This concept is used here to obtain an efficient approximate search that can easily cope with very large sets of reference vectors at significantly lower runtime.

3 Experiments

The speaker recognition experiments were carried out with two different corpus, the EuTrans and SIVA. Both experiments were carried out varying the variance threshold t used to decide if a cepstral-coefficient vector is selected as a center of a local feature, the window w of cepstral-coefficient vectors considered, and the PCA dimensionality reduction applied. A lower value of t means a large number of local features obtained from the speech signal. Figure 2 shows the relation between the total number of training local features and the parameter t for EuTrans and SIVA.

The parameter w is related with the dimensionality of the original representation space while the PCA parameter is the final dimensionality of the feature vectors.

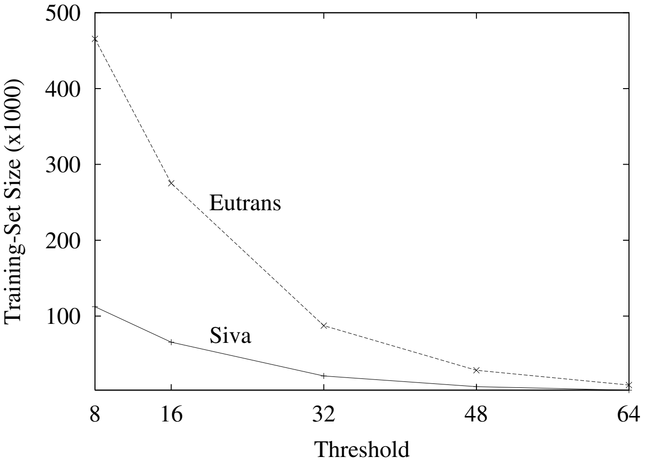


Fig. 2. Training set size with respect to the variance threshold.

3.1 EUTRANS

Experiments were carried out with Italian speech sentences acquired in the EuTrans project [17]. This database has a total of 2,757 sentences and 213 speakers (approximately 7.9h of speech). The database was splited into 2,161 sentences for training and 596 sentences for test. The speech corpus consisted of acquisitions of real phone calls to the front desk of a hotel, simulated using *Wizard of Oz* techniques [16]. This corpus is highly spontaneous and contains many non-speech artifacts. The speech signal was sampled at 8 kHz.

Figure 3 shows the results obtained for $pca = 40$. Similar results were obtained for $pca = 20$ and $pca = 30$. In the experiments the threshold value, t, was varied from 8 to 64. We obtained 465,527 local vectors from the training speech sentences by setting the variance threshold $t = 8$. The window size w, was varied from 5 to 11.

For a window size of $w = 9$ samples and a variance threshold of $t = 8$ the best result is obtained leading to an 85.9% of accuracy. This can be considered a good result taking into account the large number of speakers and the high degree of spontaneity of the corpus. This results show again that this approach is more effective if the set of local-feature vectors obtained from the training data is large.

3.2 SIVA

The Italian speech database SIVA (Speaker Identification and Verification Archives) [18], contains the recordings of more than 2,000 speakers. This corpus consists of four speaker categories: male users, female users, male impostors and female impostors.

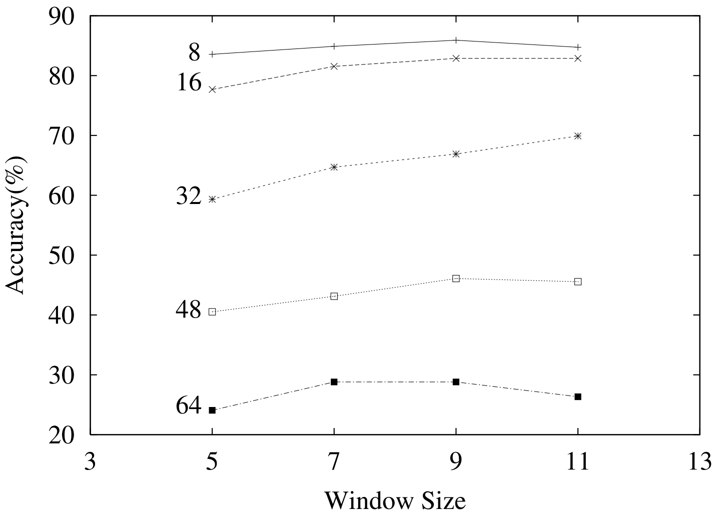


Fig. 3. Accuracy on the EuTrans data base for different window sizes and variance thresholds.

The speakers access the recording system calling a 'toll free' number. An automatic answering system guides them along the three sessions that complete the recording. This corpus has a controlled utterance scenario involving non spontaneous sentences.

For speaker recognition experiments a reduced part of this corpus was selected. We used only utterances from the female and male impostors sets and only of those speakers having more than one utterances. Under these restrictions our final set has 102 different speakers and 612 utterances. We used half utterances for training and the other half error estimation.

In the experiments the threshold value, t, was varied from 8 to 64 and the window size, w, from 5 to 11.

Figure 4 shows the results obtained. As in the previous experiment, it is important to remark that the most important parameter of this approach is the size of the local feature set obtained from the training data, controlled by the variance threshold, t. The best accuracy, 99.7%, is obtained for the largest data set extracted with the variance threshold $t = 8$ and with a window size $w = 7$, using $pca = 40$. Again, similar results were obtained for different pca values tested. This result is significantly better than the results obtained for the EuTrans corpus, mainly it is due to the lower degree of spontaneity of the utterances involved.

Table 1 shows comparative results for the EuTrans and SIVA corpora. The accuracy is reported for different values of the variance threshold, with window sizes $w = 7$ and $w = 9$ for SIVA and EuTrans respectively.

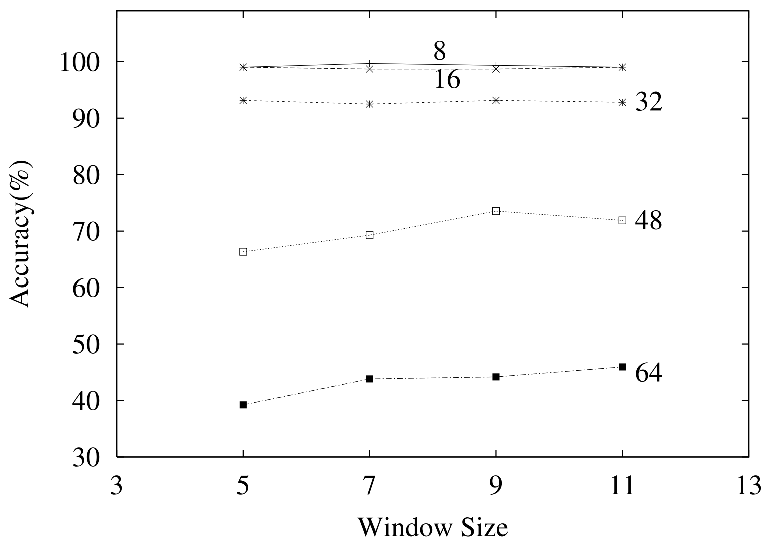


Fig. 4. Accuracy on the SIVA data base for different window sizes and variance thresholds.

Table 1. Comparative results for EuTrans and SIVA corpora. Results in boldface are the best results obtained for each corpus.

Corpus/Threshold	8	16	32	48	64
SIVA $w = 7$	**99.7%**	98.7%	92.5%	69.3%	43.8%
EuTrans $w = 9$	**85.9%**	82.9%	66.9%	46.1%	28.8%

Both experiments show that the most important parameter is the number of local features used to represent the speech sentences. This behavior, under the proposed probability estimation scheme, reflects the importance of using a fast k-nearest neighbors search technique. In our experiments the system takes less than one second to recognize a speaker in a conventional PC computer.

4 Conclusions

A local feature approach is proposed for speaker recognition which combines a simple local representation method with a direct voting scheme based on k-nearest neighbors. The results confirm that the most important parameter is the number of local features extracted from the speech signals. This number of local features is controlled by a variance-threshold parameter in the proposed feature selection approach. Large number of local features implies slow k-nearest neighbors searches, this problem is effectively solved through an approximate fast k-nearest neighbors search technique.

Current work is under way to test the proposed approach on other public-domain databases. We are also interested in studying other voting schemes and local feature extraction methods.

References

1. K. Messer, J. Kittler, M. Sadeghi, S. Marcel, C. Marcel, S. Bengio, F. Cardinaux, C. Sanderson, J. Czyz, L. Vandendorpe, S. Srisuk, M. Petrou, W. Kurutach, A. Kadyrov, R. Paredes, B. Kepenekci, F. B. Tek, G. B. Akar, F. Deravi, N. Mavity. Face Verification Competition on the XM2VTS Database. *Proc. 4th International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA)*, Guildford, 2003.
2. R. Paredes, J. C. Perez-Cortes, A. Juan, and E. Vidal. Local Representations and a Direct Voting Scheme for Face Recognition. In *Workshop on Pattern Recognition in Information Systems*, Setúbal, Portugal, July 2001.
3. L.R. Rabiner, and R.W. Shafer. Digital processing of speech signals. *Ed. Prentice Hall.* 1978
4. S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *JACM*, 45:891–923, 1998.
5. C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. on PAMI*, 19(5):530–535, 1997.
6. R. Mohr, S. Picard, and C. Schmid. Bayesian decision versus voting for image retrieval. In *Proc. of the CAIP-97*, 1997.
7. C. Shyu et al. Local versus Global Features for Content-Based Image Retrieval. In *Proc. of the IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 30–34, 1998.
8. R. Deriche and G. Giraudon. A Computational Approach to Corner and Vertex Detection. *Int. Journal of Computer Vision*, 10:101–124, 1993.
9. R.P Duin J. Kittler, M. Hatef and J. Matas. On combinig classifiers. *IEEE Trasn. on PAMI*, 1998.
10. R. Liao and S. Z. Li. Face Recognition Based on Multiple Facial Features. In *Proc. of the 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition*, 2000.
11. Z. Zhang, R. Deriche, O. Faugeras, and Q. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78:87–119, 1995.
12. M. Lhuillier and L. Quan. Robust Dense Matching Using Local and Global Geometric Constraints. In *Proc. of ICPR-2000*, volume 1, pages 968–972, 2000.
13. K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. XM2VTSDB: The Extended M2VTS Database. In *Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 964–966, March 1999.
14. F. Samaria and A. C. Harter. Parameterisation of a Stochastic Model for Human Face Identification. In *Proc. of the 2nd IEEE Workshop on Applications of Computer Vision*, pages 138–142, 1994.
15. J. Ben-Arie and D. Nandy. A volumetric/iconic frequency domain representation for objects with application for pose invariant face recognition. *IEEE Trans. on PAMI*, 20:449–457, 1998.
16. D. Aiello, L. Cerrato, C. Delogu, and A. Di Carlo. The acquisition of a speech corpus for limited domain translation. In *Proceedings of the European Conference on Speech Communication and Technology*, Budapest, 1999.

17. EuTrans. Example-based language translation systems. Final report. Technical report, Instituto Tecnológico de Informática, Fondazione Ugo Bordoni, Rheinisch Westfälische Technische Hochschule Aachen Lehrstuhl für Informatik VI, Zeres GmbH Bochum: Long Term Research Domain, Project Number 30268, 2000.
18. Falcone M., Gallo The SIVA speech database for speaker verification: description and evaluation *ICSLP'96*, Philadelphia, USA, October, pp. 1902–1905.

Selection/Extraction of Spectral Regions for Autofluorescence Spectra Measured in the Oral Cavity

Marina Skurichina[1], Pavel Paclík[1], Robert P.W. Duin[1], Diana de Veld[2], Henricus J.C.M. Sterenborg[2], Max J.H. Witjes[3], and Jan L.N. Roodenburg[3]

[1] Information and Communication Theory Group, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, P.O. Box 5031, 2600GA Delft, The Netherlands
m.skurichina@ewi.tudelft.nl
[2] Photodynamic Therapy and Optical Spectroscopy Programme, Department of Radiation Oncology, Erasmus MC, Rotterdam, The Netherlands
[3] Department of Oral and Maxillofacial Surgery, University Hospital Groningen, The Netherlands

Abstract. Recently a number of successful algorithms to select/extract discriminative spectral regions was introduced. These methods may be more beneficial than the standard feature selection/extraction methods for spectral classification. In this paper, on the example of autofluorescence spectra measured in the oral cavity, we intend to get deeper understanding what might be the best way to select informative spectral regions and what factors may influence the success of this approach.

1 Introduction

In medical applications, one often faces the small sample size problem: the number of measurements is smaller than or comparable with the data dimensionality. In such conditions, it is difficult (or even impossible) to construct a good classification rule [1]. One has to reduce the data dimensionality. When having spectral data, using standard feature selection/extraction methods may be inconvenient. The standard approaches assume the independency of data features while in spectra the features (neighbouring wavelengths/pixels/bins) are correlated. Therefore, some useful information may be lost if the connectivity of spectral neighbouring pixels is not taken into account when extracting/selecting features informative for discrimination between data classes. During the last few years, a number of novel methods for selection/ extraction informative spectral regions/bands has been developed. One example of such feature extraction algorithms is an Optimal Region Selector (ORS) [2] guided by a genetic algorithm. Another example is a top-down multiresolution feature extraction algorithm proposed by Kumar, Ghosh and Crawford [7].

In this paper we pretend neither to introduce fundamentally new algorithms for selection/extraction of informative spectral regions, nor to perform an extensive comparison of already existing algorithms with a standard feature selection/extraction methods. Our goal is to understand what happens exactly when extracting informative spectral regions by different methods, what underlines the success of this process and what factors influence the benefit of this approach.

In order to perform our study we have selected a real data set, which represents autofluorescence spectra measured in the oral cavity. This data set introduces a 2-class problem: lesions against healthy tissues. It is described in section 2. Different feature selection/extraction techniques used for finding informative spectral regions/bands are introduced in section 3. The results of our simulation study are presented in section 4. Conclusions can be found in section 5.

2 Data

We perform our study on the example of autofluorescence spectra measured in the oral cavity. The data consist of the autofluorescence spectra acquired from healthy and diseased mucosa in the oral cavity. The measurements were performed at the Department of Oral and Maxillofacial Surgery of the University Hospital of Groningen [3]. Autofluorescence spectra were collected from 97 volunteers with no clinically observable lesions of the oral mucosa and 137 patients having lesions in the oral cavity. The measurements were taken at 11 different anatomical locations with excitation wavelength equal to 365 nm. The previously performed study [3] has shown that spectra measured at different anatomical locations are similar, and the location of a probe affects only the intensity of the spectra but not the shape. By this, it was possible to use a larger data set for our study. In total, 856 spectra representing healthy tissue and 132 spectra representing diseased tissue were obtained. After preprocessing [3], each spectrum consists of 199 bins (pixels/wavelengths).

In order to get rid of a large deviation in a spectral intensity within each data class, we normalized spectra by the Unit Area (UA)

$$a_i^{UA} = \frac{a_i}{U}, \quad U = \sum_{j=1}^{199} a_j, \quad i = 1, \ldots, 199, \tag{1}$$

where a_i is an intensity of a spectrum $\boldsymbol{A}=\{a_1, \ldots, a_{199}\}$ at bin i, i=1,...,199. Normalized autofluorescence spectra representing healthy and diseased tissues and their median spectra are illustrated in Fig. 1.

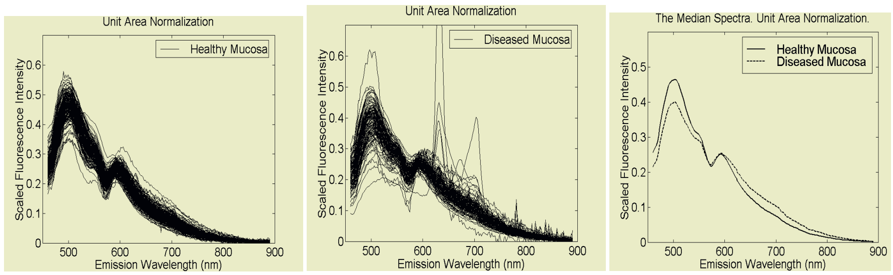


Fig. 1. Normalized autofluorescence spectra for healthy and diseased mucosa in oral cavity.

For our simulation study, training data sets with 2/3 of available samples per class are chosen randomly from the total set. The remaining data are used for testing. The prior class probabilities are set to be equal as the data are very unbalanced and the real prior class probabilities are unknown. To evaluate the performance of lesion diagnostics when different feature selection/extraction methods are used, we have chosen the Linear Discriminant Analysis (LDA) [4] which was the best performing classifier for this application. In particular, we apply the regularized linear classifier [5] which constructs a linear discriminant function assuming normal class distributions and using a joint class covariance matrix for both data classes. The value of the regularization parameter used is equal to 10^{-10}. All experiments are repeated 20 times on independent training sample sets. In all figures the averaged results over 20 trials are presented and we do not mention that anymore. The standard deviations of the reported mean generalization errors (the mean per two data classes) is approximately 0.01 for each considered case.

3 Selection/Extraction of Spectral Regions

Inspired by the success of approaches suggested by Nikulin [2] and Kumar [7], we became interested in what actually happens when selecting/extracting spectral regions, why and when it is beneficial and not.

The first approach, Optimal Region Selector by Nikulin [2], is based on a genetic algorithm. First, one randomly generates few sets of non-overlapping spectral regions of arbitrary size. For each region, a new feature (for instance, the mean of the spectral intensities in the region) is derived. Then the goodness of each set of new features is evaluated by some criterion (Nikulin has used the mean square error between the true labels and the posterior class probabilities calculated on the training dataset by the linear classifier having the averaged coefficients over all linear classifiers constructed on leave-one-out cross-validation). According to this criterion, the best subsets of spectral regions are selected. Further, one again increases the number of these subsets by random mutations and crossovers of the region definitions, and the procedure repeats. At the end, a suboptimal solution (due to a randomness of the whole procedure) for the best set of spectral regions is found.

The second approach, a top-down multiresolution feature extraction algorithm proposed by Kumar et al. [7], partitions the original p-dimensional spectra into smaller subspaces by using a top-down recursive algorithm. First, the best place to split spectra into two parts is found by computing a discriminant measure between data classes (for instance, Bhattacharya distance, Kullback-Leibler divergence or log-odds of class posterior probabilities used by Kumar can be applied). The discriminant measure obtained on the parent space is compared with the discriminant measures calculated on the children subspaces. If the child subspace has a higher discrimination than the parent space, then it is partitioned further. If the child subspace does not show any improvement in its discrimination capacity compared to the parent space, then this child subspace is not partitioned any further. Finally, one finds a set of spectral regions/ bands with high discrimination. However, the optimization is performed only in a one-dimensional way: a discrimination capacity is evaluated for each spectral region separately but not for a total set of selected spectral regions.

We consider here the Top-Down variant of Generalized Local Discriminant Bases algorithm (GLDB-TD), which is conceptually close to the algorithms, described in this paper. The GLDB-TD algorithm represents each group of wavelengths by a mean of corresponding intensities.

In our study on the usefulness of spectral regions extraction/selection approach and on the benefits of different ways to extract/select discriminative spectral regions, we do not use exactly the two algorithms discussed above. First, we would like to simplify the procedures for selection/extraction of discriminative spectral regions in order to get more insight what does actually happen. Second, it is convenient to use the same discriminant measure between data classes when comparing different ways of selecting/extracting the best spectral bands. And finally, we prefer to use multi-dimensional optimization when looking for the informative spectral regions.

In our study we consider few approaches to extract/select the informative/discriminative spectral regions. In all of them, first we perform the dimensionality reduction for each considered spectral band. Namely, from each spectral region considered we derive one new feature by taking the average of intensities in this region. This new feature (the averaged intensity of the spectral band) is used further to introduce the spectral region.

In order to evaluate a discriminative capacity of extracted spectral regions, we use the Mahalanobis Distance (MD) between data classes:

$$MD = (\mu_A - \mu_B)'(p\Sigma_A + (1-p)\Sigma_B)^{-1}(\mu_A - \mu_B), \qquad (2)$$

where μ_A, μ_B and Σ_A, Σ_B are the means and the covariance matrices of data classes A and B, respectively; p is the prior probability of the data class A. The larger Mahalanobis distance, the larger discriminative capacity between data classes. In order to perform the multi-dimensional optimization of S spectral regions, we calculate the Mahalanobis distance on the whole set of S features (the averaged intensities of spectral bands), each representing one of S spectral regions. By this, we find the optimal set of spectral regions providing the best discrimination (according to MD) between data classes.

In our study we consider the following ways to extract/select the informative spectral regions.

Approach 1.

A. *Sequential Partition of Spectra into Non-overlapping Bands Using All Spectral Pixels.*

First, we split spectra into two spectral regions by finding the best split which gives the largest MD (over all possible partitions) in the space of two features extracted from the two spectral bands. Then, the first found split is fixed and we look for the next optimal split in such a way that the MD in a three-dimensional space (on three features extracted from the three spectral bands) is the largest over all possible locations for the second split (when the location of the first split is anchored). Again, we fix the location for the first two found splits and repeat the procedure while the desired number of spectral regions S is found (see top plots in Fig. 2). In this approach, all spectral emission wavelengths are used in the partitioning of spectra. However, some spectral bins can be uninformative - introducing only noise. Hence, they may deteriorate the classification when they are included in the extracted spectral regions. Therefore, it is good to remove them from the spectral bands. One way to do this is described below.

B. *Sequential Partition of Spectra into Non-overlapping Bands Excluding Uninformative Spectral Pixels.*

After a desired number of spectral regions S is found by Approach 1A, we can shrink the spectral bands removing uninformative emission wavelengths. We reduce the number of bins in each spectral region in a sequential way moving from the most left spectral region to the most right one. For shrinking the spectral band, we consider all possible subregions of the reduced size in this band and find the one with the largest MD in S-dimensional space (one feature, the averaged intensity, calculated from a shrunk subregion of the spectral band under consideration and the rest S-1 features extracted from the other S-1 spectral bands which definitions are fixed for a moment). After the optimal shrinking for the first spectral band is found, we anchor its new definition and move to the next spectral band in order to exclude uninformative pixels (see middle plots in Fig. 2). We should mention that the proposed method is highly dependent on the spectral regions proceeding order and therefore it does not guaranty the optimal shrinking for all regions in general.

Approach 2.

A. *Sequential Selection/Extraction of Discriminative Spectral Regions.*

In order to find a set of the most discriminative spectral bands, at each step s, s = 1, 2, … S, we consider all possible definitions of spectral regions (of arbitrary size) in spectra. For each of them we calculate the MD criterion in s dimensions: one

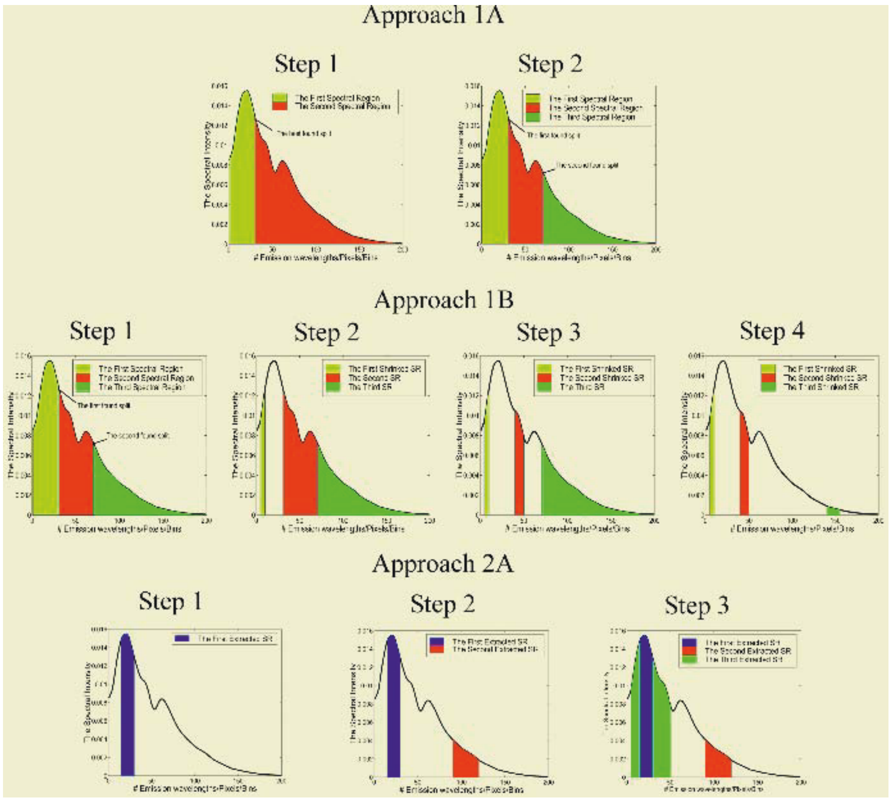


Fig. 2. Approach 1A (top plots), Approach 1B (middle plots) and Approach 2A (bottom plots) for selection/extraction of informative spectral regions (SR) to discriminate between healthy and diseased tissues.

feature is the averaged intensity of a current potential pretender for the most informative band and other *s*-1 features are extracted from the previously found optimal spectral regions. The spectral band (a potential pretender) with the largest MD is picked as the most discriminative spectral band (in combination with the *s*-1 previously found optimal regions). Let us mention that in this approach overlapping as well as non-overlapping spectral bands are possible (see bottom plots in Fig. 2).

B. *Sequential Selection/Extraction of Non-overlapping Discriminative Spectral Regions.*

This approach is identical to Approach 2A with the exception that overlapping spectral bands are not allowed: when looking for an additional discriminative spectral region, the regions overlapping with the previously selected optimal spectral bands are excluded from consideration.

4 Simulation Study

Let us now study the benefits of extracting/selecting the discriminative spectral regions. In order to judge the success of this approach, we compare different ways to extract/select spectral regions with one of the most successful standard feature extraction methods - the Principal Component Analysis (PCA) [4]. In Fig. 3, we present the mean generalization errors of the LDA (top plots) (over 20 independent trials) and the mean Mahalanobis distances (bottom plots) obtained on the autofluorescence spectral

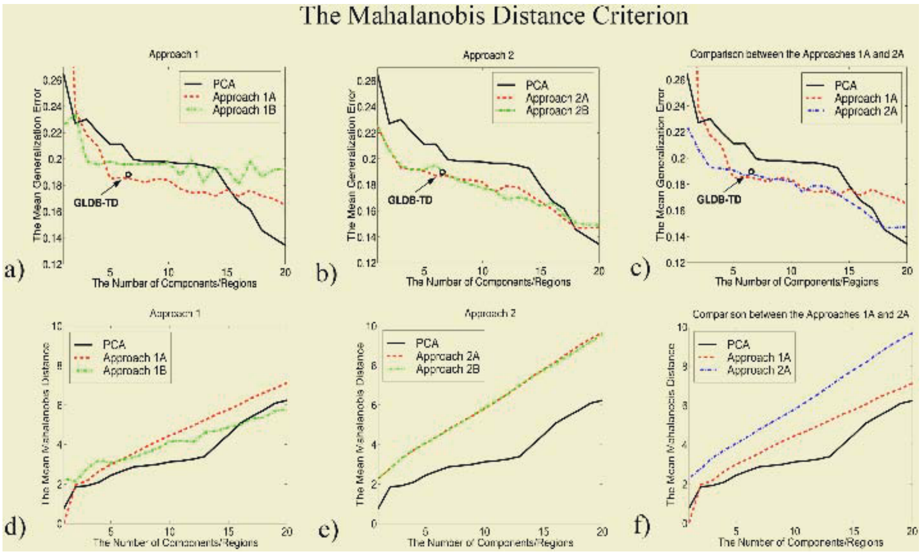


Fig. 3. The mean generalization error (GE) of LDA (a-c) and the mean Mahalanobis distance (MD) (d-f) when the MD criterion is used in Approaches 1 (a,d) and 2 (b,e) to select/extract discriminative spectral bands for autofluorescence spectral data. In plots (c,f) the comparison is made between Approaches 1A and 2A. The standard deviation of the mean GE is around 0.01. GLDB-TD denotes the performance of a Top-Down variant of the Generalized Local Discriminant Bases using a Mahalanobis distance criterion.

data when feature extraction is performed by the PCA and Approaches 1A, 1B, 2A and 2B for discriminative spectral bands extraction. We see that almost all introduced approaches for selection/extraction of informative spectral regions outperform the PCA. The exceptions are the cases when the whole spectrum is taken as one spectral band (the averaged intensity over the whole spectrum is not very informative) or too many spectral regions (some of them contain only noise) are picked to discriminate between data classes.

In order to evaluate our approach we compare different feature extraction techniques, proposed by us, to two existing methods. The first is a Principal Component Analysis (PCA). The second is a Top-Down Generalized Local Discriminant Bases algorithm (GLDB-TD) of Kumar et al. [7].

Similarly to the proposed feature extraction techniques, the GLDB-TD algorithm uses the Mahalanobis distance as a criterion.

Because the GLDB-TD algorithm terminates automatically using a data-driven criterion, only a single point is given in each plot. The point represents the mean error of 20-fold cross-validation. Because each fold results, in general, in a different number of wavelength groups, we plot a median over these 20 results.

Comparing the performance of the linear classifier for Approaches 1A and 1B, we notice that shrinking the spectral regions in the optimal partition of spectra is not useful (see Fig. 3a). One reason underlies in the proposed method 1B that is not optimal in general. Another reason is that the MD criterion is not equivalent to the LDA: the covariance matrices of data classes are assumed to be different in the MD criterion while they are considered to be the same for both data classes in the LDA. Therefore, optimizing the MD in selection/extraction of the most discriminative spectral regions does not guaranty the optimal performance for the linear classifier. For instance, for 20 spectral features extracted, Approach 1A provides the largest MD (see Fig.3d) but worse performance of the LDA than for the PCA (see Fig. 3a).

For our autofluorescence spectral data, we do not observe any difference between Approaches 2A and 2B: it is not important whether the extracted spectral regions do overlap or do not (see Fig. 3b). However, we see that Approach 2A is more beneficial than Approach 1A: using less spectral wavelengths in selected spectral bands is better than when all spectral wavelengths are used in the partitioning of spectra (see Fig. 3c). In addition, we should mention that often both Approaches 1B and 2B tend to select/ extract very narrow spectral bands consisting only of 1-3 pixels/wavelengths which may introduce more noise fluctuations than a real difference between data classes. This may happen due to the overtraining that occurs when the same training set is used to construct the LDA and to optimize the MD criterion in the spectral regions extraction.

In order to avoid the shortages of the MD criterion mentioned above and the overtraining when extracting discriminative spectral regions, other criterion should be used. The alternative may be the apparent error (AE) (the classification error on the training set) of the linear classifier. This criterion is more close to the LDA than the MD criterion what concerns assumptions on the data classes distributions. The elusion of overtraining can be done by bootstrapping [6] the training set when calculating the apparent error. Namely, we bootstrap the training set B times constructing a linear classifier on each bootstrap replicate of the training set and calculate the average classification error of these B bootstrapped classifiers on the original training set.

Let us now consider the differences in the performance of the LDA caused by applying the AE criterion instead of the MD to extract informative spectral bands. In our

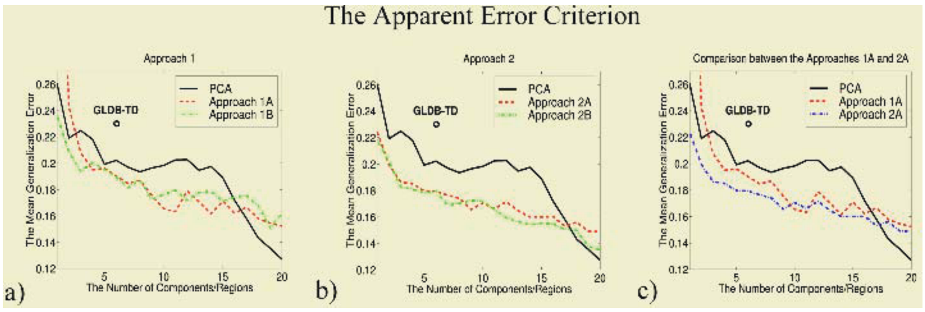


Fig. 4. The mean generalization error (GE) of LDA when the discriminative spectral regions for autofluorescence spectra are extracted/selected by using the apparent error (AE) criterion in Approach 1 (plot a) and Approach 2 (plot b). In plot (c) the comparison is made between Approaches 1A and 2A. The standard deviation of the mean GE is around 0.01. GLDB-TD denotes the performance of a Top-Down variant of the Generalized Local Discriminant Bases using an apparent error criterion.

simulations we use *B*=5 bootstrap replicates of the training set to calculate the apparent error. Figures 4 and 5 illustrate that using the AE criterion is indeed more beneficial than using the MD criterion: the performance of all four Approaches (1A, 1B, 2A and 2B) is improved. In figure 4 also the AE result for the GLDB is shown. In Approaches 1B and 2B, wider spectral bands on average are found to be optimal when the AE criterion is applied instead of the MD criterion for extracting discriminative spectral regions. Also, the previously made observations hold on: using a subset of spectral wavelengths in extracted spectral bands is more preferable than using them all in the optimal partition of spectra; it seems to be unnecessary to extract uncorrelated spectral bands in order to achieve the best classification results.

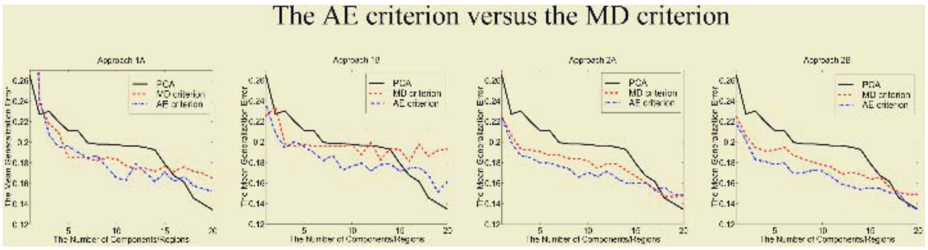


Fig. 5. The performance of LDA calculated on principal components (PCA) and on spectral regions selected/extracted by using the Mahalanobis Distance (MD) and the Apparent Error (AE) criteria for autofluorescence spectra.

5 Conclusions

Summarizing simulation results presented in the previous section, we can conclude the following. The selection/extraction of discriminative spectral bands may be more beneficial than the standard unsupervised feature selection/extraction methods (e.g., PCA) that do not assume the connectivity of the neighbouring wavelengths in spectral data.

In order to gain more advantage from the selection/extraction of informative spectral regions, the optimization criterion to select/extract discriminative spectral bands should be adjusted to the classification rule used to solve the problem. When selecting informative spectral regions, overtraining may be avoided if the evaluation criterion measures a performance on a different than the training dataset. For this purpose, the bootstrapped training set or an additional validation dataset can be used.

It follows from our experiments that it is advantageous to perform a multi-variate selection of wavelength groups, compared to a uni-variate approach such as Generalized Local Discriminant Bases (GLDB) utilizing the same criterion. It is also useful to exclude uninformative wavelengths from the spectral regions.

Interestingly, we have found out that extracted informative spectral bands do not need to be uncorrelated. Classifiers such as a linear discriminant assuming normal densities will find a good separation boundary even for correlated features. Feature extraction techniques like GLDB identify groups of non-overlapping wavelengths. Our finding suggests that overlapping groups of wavelengths may provide discriminative representation as well.

Acknowledgments

This work was supported by the Dutch Technology Foundation (STW), grant RRN 5316, and the Dutch Cancer Society (Nederlandse Kanker Bestrijding), grant 99-1869.

References

1. Jain, A.K., Chandrasekaran, B.: Dimensionality and Sample Size Considerations in Pattern Recognition Practice. In: Krishnaiah, P.R., Kanal, L.N. (eds.): Handbook of Statistics, Vol. 2. North-Holland, Amsterdam (1987) 835-855.
2. Nikulin, A., Dolenko, B., Bezabeh, T., Somorjai, R.: Near Optimal Region Selection for Feature Space Reduction: Novel Preprocessing Methods for Classifying MR Spectra, *NMR in Biomedicine* **11** (1998) 209-216.
3. De Veld, D.C.G., Skurichina, M., Witjes, M.J.H., et.al. Autofluorescence Characteristics of Healthy Oral Mucosa at Different Anatomical Sites, *Lasers in Surgery and Medicine*, **32** (2003) 367-376.
4. Fukunaga, K.: Introduction to Statistical Pattern Recognition. Academic Press (1990) 400-407.
5. Friedman, J.H.: Regularized Discriminant Analysis. Journal of the American Statistical Association (JASA) **84** (1989) 165-175.
6. Efron, B., Tibshirani, R.: An Introduction to the Bootstrap. Chapman and Hall, New York (1993).
7. Kumar, S., Ghosh, J. and Crawford, M.M., Best-Bases Feature Extraction Algorithms for Classification of Hyperspectral Data, IEEE Transactions on Geoscience and Remote Sensing, **39** (2001) 1368 - 1379.
8. P. Paclik, S. Verzakov, and R. P. W. Duin. Hypertools: the toolbox for spectral image analysis. Technical report, Pattern Recognition Group, TU Delft, The Netherlands, Dec. 2003.

Kernel Relative Principal Component Analysis for Pattern Recognition

Yoshikazu Washizawa[1], Kenji Hikida[2], Toshihisa Tanaka[3,4], and Yuhikiko Yamashita[5]

[1] Toshiba Solutions Corporation, Tokyo, 183-8511, Japan
washizawa.yoshikazu@toshiba-sol.co.jp
[2] Wireless Terminals, Texas Instruments Japan Limited, Tokyo 160-8366, Japan
hikida@ti.com
[3] Department of Electrical and Electronic Engineering
Tokyo University of Agriculture and Technology, Tokyo 184-8588, Japan
tanakat@cc.tuat.ac.jp
[4] Laboratory for Advanced Brain Signal Processing, RIKEN Brain Science Institute
Saitama 351-0198, Japan
[5] Graduate School of Science and Engineering, Tokyo Institute of Technology
Tokyo 152-8552, Japan
yamasita@ide.titech.ac.jp

Abstract. Principal component analysis (PCA) is widely used in signal processing, pattern recognition, etc. PCA was extended to the relative PCA (RPCA). RPCA provides principal components of a signal while suppressing effects of other signals. PCA was also extended to the kernel PCA (KPCA). By using a mapping from the original space to a higher dimensional space and its kernel, we can perform PCA in the higher dimensional space. In this paper, we propose the kernel RPCA (KRPCA) and give its solution. Similarly to KPCA, the order of matrices that we should calculate for the solution is the number of samples, that is 'kernel trick'. We provide experimental results of an application to pattern recognition in order to show the advantages of KRPCA over KPCA.

1 Introduction

Principal component analysis (PCA) or Karhunen-Loève transform is widely used in signal processing, pattern recognition, etc [1], [2], [3]. We can extract important components of a signal that minimize the mean square error between the extracted and the original signals. Consider that input vectors are in a D-dimensional real vector space R^D with the inner product $\langle \boldsymbol{f}, \boldsymbol{g} \rangle$ and the norm $\|\boldsymbol{f}\| = \sqrt{\langle \boldsymbol{f}, \boldsymbol{f} \rangle}$ for vectors $\boldsymbol{f}$ and $\boldsymbol{g}$. Let P be a matrix. PCA is defined by minimizing the following criterion

$$E_{\boldsymbol{f}} \|P\boldsymbol{f} - \boldsymbol{f}\|^2 \tag{1}$$

under the condition that $\mathrm{rank}(P) \leq d$ for any $d \leq D$, where $E_{\boldsymbol{f}}$ is the ensemble average with respect to a signal $\boldsymbol{f}$. Let $R_{\boldsymbol{f}} = E_{\boldsymbol{f}} \boldsymbol{f}\boldsymbol{f}^T$ be the correlation matrix of $\boldsymbol{f}$, where $\boldsymbol{f}^T$ is the transpose of $\boldsymbol{f}$. The vector $\boldsymbol{\phi}_n$ is given as the eigen vector corresponding to the n-th largest eigenvalue of $R_{\boldsymbol{f}}$. The solution of the PCA P is given as

$$P = \sum_{n=1}^{d} \boldsymbol{\phi}_n \boldsymbol{\phi}_n^T. \tag{2}$$

We consider the case that there exist two signals $\boldsymbol{f}$ and $\boldsymbol{g}$ and the principal component of $\boldsymbol{f}$ is obtained while the effect of $\boldsymbol{g}$ is suppressed. When $\boldsymbol{g}$ is noise, we can extract the principal components of a signal $\boldsymbol{f}$ in the presence of noise. When $\boldsymbol{g}$ is a pattern in other categories, we can extract the features of $\boldsymbol{f}$ that are not closed to the features of $\boldsymbol{g}$.

By extending eq.(1) the relative PCA (RPCA) or the relative Karhunen-Loève transform was proposed [4], [5]. Consider that a matrix X minimizes the following criterion:

$$E_{\boldsymbol{f}} \|X\boldsymbol{f} - \boldsymbol{f}\|^2 + \alpha E_{\boldsymbol{g}} \|X\boldsymbol{g}\|^2 \tag{3}$$

under the condition that $\mathrm{rank}(X) \leq d$ for any d. The parameter $\alpha(> 0)$ controls the weight for suppressing the effect of $\boldsymbol{g}$. With a large α, the effect of $\boldsymbol{g}$ is suppressed well. With a small α, the approximation error between $X\boldsymbol{f}$ and $\boldsymbol{f}$ is decreased.

The solution of RPCA is given as the form:

$$X = \sum_{n=1}^{d} \boldsymbol{\phi}_n (\boldsymbol{\varphi}_n)^T. \tag{4}$$

Here, we describe its solution when $R_{\boldsymbol{f}} + \alpha R_{\boldsymbol{g}}$ is not singular. Let μ_n ($\mu_1 \geq \mu_2 \geq \cdots \geq \mu_D$) be eigenvalues and let $\boldsymbol{\psi}_i$ be corresponding eigen vectors of $R_{\boldsymbol{f}}(R_{\boldsymbol{f}} + \alpha R_{\boldsymbol{g}})^{-1} R_{\boldsymbol{f}}$. Then, the solution is given as $\boldsymbol{\phi}_n = \boldsymbol{\psi}_n$ and $\boldsymbol{\varphi}_n = (R_{\boldsymbol{f}} + \alpha R_{\boldsymbol{g}})^{-1} R_{\boldsymbol{f}} \boldsymbol{\psi}_n$. From eq.(4) the n-th relative principal component of $\boldsymbol{f}$ is given as $\langle \boldsymbol{f}, \boldsymbol{\varphi}_n \rangle \boldsymbol{\phi}_n$. When $d = D$, X is reduced to Wiener filter, which provides the best approximation of a signal in sense of mean square error. A similar criterion with eq.(3) was provided for the rank reduced Wiener filter. However, the reason why they restrict the rank is not for obtaining principal components but for robustness [6].

The advantages of RPCA over PCA were shown by experiments of data compression in the presence of noise [4] and handwritten character recognition [5].

As for PCA with two kinds of signals, Fisher discriminant [7] and Oriented PCA (OPCA) [3] were proposed. OPCA is defined by vectors $\boldsymbol{\phi}_n$ that minimize

$$\frac{E_f \langle f, \boldsymbol{\phi}_n \rangle^2}{E_g \langle g, \boldsymbol{\phi}_n \rangle^2} \tag{5}$$

under the condition that $\langle \boldsymbol{\phi}_m, R_f \boldsymbol{\phi}_n \rangle = 0$ and $\langle \boldsymbol{\phi}_m, R_g \boldsymbol{\phi}_n \rangle = 0$ for $m \neq n$. The theoretical advantage of RPCA over them is in that the criterion of RPCA compares the approximation error between principal components and the original signal directly. Then, it provides the principal components of which accuracy is guaranteed.

PCA was extended to another direction based on a kernel. The kernel PCA (KPCA) is defined as follows [8], [9], [10], [11], [12], [13]. Let Φ be a mapping from a input vector space R^D to a real Hilbert space $\mathcal{H}$. The inner product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is also defined for vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathcal{H}$. Furthermore, we assume $\langle \Phi(\boldsymbol{f}), \Phi(\boldsymbol{g}) \rangle = k(\boldsymbol{f}, \boldsymbol{g})$, where $k(\boldsymbol{f}, \boldsymbol{g})$ is a kernel function. Let $\{\boldsymbol{f}_l\}_{l=1}^{L}$ be a set of samples of a signal. When $\mathcal{H}$ is a vector space, a sample correlation matrix $S_{\boldsymbol{f}}$ in $\mathcal{H}$ is given as

$$S_{\boldsymbol{f}} = \frac{1}{L}\sum_{l=1}^{L} \Phi(\boldsymbol{f}_l)\Phi(\boldsymbol{f}_l)^T. \tag{6}$$

Let V^n be the eigen vector corresponding to the n-th largest eigenvalue of $S_{\boldsymbol{f}}$. For an input vector $\boldsymbol{f}$, the magnitude of the n-th principal component is given as $\langle V^n, \Phi(\boldsymbol{f})\rangle$. However, for obtaining the magnitude we don't need to calculate the eigenvalue problem in $\mathcal{H}$, which is a very high or infinite dimensional space. The advantage of the theory of KPCA is that we can reduce the dimension for the calculation to the number of samples L. Fisher discriminant was also extended to the kernel Fisher discriminant [14], [15], [16].

Since a signal contains noise in almost all cases or suppression of the effect of other categories is useful in many cases, the kernelization of RPCA is very important as well as the kernelization of PCA. In this paper, we propose the kernel relative PCA (KRPCA) which is a kernel based extension of RPCA. We provide the definition and its solution in this paper. The dimension of a space for the calculation is the number of samples. Different from other many kernelized problems such as support vector machine (SVM), the solution that minimizes the criterion of KRPCA is given by a closed form, that is, it can be provided by eigen vectors and inversion of matrices similarly to KPCA. Furthermore, since it provides the relative principal components, it can be applied not only to discrimination but also to many problems such as feature extraction and dimensional reduction. In order to show the advantage of KRPCA, we show experimental results of an application to pattern recognition.

2 Kernel Relative PCA

In this section, we provide the definition and a solution of KRPCA.

Schatten product $\boldsymbol{x} \otimes \overline{\boldsymbol{y}}$ for vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathcal{H}$ is defined as a linear operator from $\mathcal{H}$ to $\mathcal{H}$ such that $(\boldsymbol{x} \otimes \overline{\boldsymbol{y}})\boldsymbol{z} = \langle \boldsymbol{z}, \boldsymbol{y}\rangle \boldsymbol{x}$ for any $\boldsymbol{z} \in \mathcal{H}$ [17]. It is an abstract notation of $\boldsymbol{x}\boldsymbol{y}^T$ for vectors in a Hilbert space.

Let $\{\boldsymbol{f}_l\}_{l=1}^{L}$ be a set of samples, of which principal components are extracted. Let $\{\boldsymbol{g}_m\}_{m=1}^{M}$ be a set of samples of which effect is suppressed. We define the criterion of KRPCA X as minimizing

$$J = \frac{1}{L}\sum_{l=1}^{L} \|X\Phi(\boldsymbol{f}_l) - \Phi(\boldsymbol{f}_l)\|^2 + \frac{\alpha}{M}\sum_{m=1}^{M} \|X\Phi(\boldsymbol{g}_m)\|^2 \tag{7}$$

under the conditions that $\mathrm{rank}(X) \leq d$ and its null space includes the orthogonal subspace of the subspace spanned by $\{\boldsymbol{f}_l\}_{l=1}^{L}$ and $\{\boldsymbol{g}_m\}_{m=1}^{M}$.

For brief, let $\boldsymbol{f}_{i+L} = \boldsymbol{g}_i$ $(i = 1, 2, \cdots, M)$ and $N = L + M$. Since all vectors in the criterion(7) are in a subspace spanned by $\Phi(\boldsymbol{f}_i)$ $(i = 1, 2, \cdots, L, L+1, \cdots, N)$, we can assume that X is expressed with an (N, N)-matrix $A = (a_{ij})$ as

$$X = \sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}\Phi(\boldsymbol{f}_i) \otimes \overline{\Phi(\boldsymbol{f}_j)} \tag{8}$$

Let $\mathbf{0}_{mn}$ be (m, n)-matrix of which all elements are zero. We define (L, L)-matrix K_0, (N, L)-matrix K_1, (N, M)-matrix K_2, (N, N)-matrix $\tilde{K}_1$, (N, N)-matrix $\tilde{K}_2$, (N, N)-matrix K, and (N, N)-matrix $\tilde{K}$ as

$$K_0 = \begin{pmatrix} k(\boldsymbol{f}_1, \boldsymbol{f}_1) & \cdots & k(\boldsymbol{f}_1, \boldsymbol{f}_L) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{f}_L, \boldsymbol{f}_1) & \cdots & k(\boldsymbol{f}_L, \boldsymbol{f}_L) \end{pmatrix},$$

$$K_1 = \begin{pmatrix} k(\boldsymbol{f}_1, \boldsymbol{f}_1) & \cdots & k(\boldsymbol{f}_1, \boldsymbol{f}_L) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{f}_N, \boldsymbol{f}_1) & \cdots & k(\boldsymbol{f}_N, \boldsymbol{f}_L) \end{pmatrix}, \quad K_2 = \begin{pmatrix} k(\boldsymbol{f}_1, \boldsymbol{f}_{L+1}) & \cdots & k(\boldsymbol{f}_1, \boldsymbol{f}_N) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{f}_N, \boldsymbol{f}_{L+1}) & \cdots & k(\boldsymbol{f}_N, \boldsymbol{f}_N) \end{pmatrix},$$

$$\tilde{K}_1 = \begin{bmatrix} \frac{1}{\sqrt{L}} K_1 & \mathbf{0}_{NM} \end{bmatrix}, \quad \tilde{K}_2 = \begin{bmatrix} \mathbf{0}_{NL} & \sqrt{\frac{\alpha}{M}} K_2 \end{bmatrix}, \quad K = [K_1 \quad K_2], \quad \tilde{K} = \tilde{K}_1 + \tilde{K}_2.$$

It is clear that $K_1 K_1^T = L\tilde{K}_1\tilde{K}_1^T$, $\alpha K_2 K_2^T = M\tilde{K}_2\tilde{K}_2^T$, and $\tilde{K}_1\tilde{K}_1^T + \tilde{K}_2\tilde{K}_2^T = \tilde{K}\tilde{K}^T$. For a matrix B there exists an unique matrix C such that $BCB = B$, $CBC = C$, $(BC)^T = BC$, and $(CB)^T = CB$. The matrix C is called the Moore-Penrose generalized inverse matrix and denoted by $B^\dagger$ [18]. A symmetric matrix B is called the *positive semi-definite* if and only if $\langle B\boldsymbol{x}, \boldsymbol{x}\rangle \geq 0$ for any $\boldsymbol{x}$. For a positive semi-definite matrix B, there exists a positive semi-definite matrix C such that $CC = B$. We denote the matrix C by $B^{1/2}$. We define a matrix A_0 as

$$A_0 = K^\dagger \tilde{K}_1 \tilde{K}_1^T (\tilde{K}\tilde{K}^T)^\dagger. \tag{9}$$

Let λ_i ($\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$) be eigenvalues of $K^{1/2}A_0\tilde{K}(K^{1/2}A_0\tilde{K})^T$, which is a symmetric matrix, and let $\boldsymbol{u}_i$ be the corresponding eigen vectors such that $\{\boldsymbol{u}_n\}_{n=1}^N$ is an orthonormal basis. When $\lambda_n \neq 0$, let $\boldsymbol{v}_n = ((K^{1/2}A_0\tilde{K})^T\boldsymbol{u}_n)/\sqrt{\lambda_n}$. When $\lambda_n = 0$, we can chose any $\boldsymbol{v}_n$ such that $\{\boldsymbol{v}_n\}_{n=1}^N$ is an orthonormal basis. The i-th element of an N-dimensional vector $\boldsymbol{w}$ is denoted by $(\boldsymbol{w})_i$.

Theorem 1. *A KRPCA X is given as*

$$X = \sum_{i=1}^N \sum_{j=1}^N \left(\sum_{n=1}^d \left((K^{1/2})^\dagger \boldsymbol{u}_n\right) \tilde{\boldsymbol{v}}_n^T \right)_{ij} \Phi(\boldsymbol{f}_i) \otimes \overline{\Phi(\boldsymbol{f}_j)}, \tag{10}$$

where

$$\tilde{\boldsymbol{v}}_n = \sqrt{\lambda_n}(\tilde{K}^\dagger)^T \boldsymbol{v}_n. \tag{11}$$

For an input vector $\boldsymbol{f}$*, let*

$$\boldsymbol{h} = (k(\boldsymbol{f}, \boldsymbol{f}_1), k(\boldsymbol{f}, \boldsymbol{f}_2), \cdots, k(\boldsymbol{f}, \boldsymbol{f}_N))^T. \tag{12}$$

Then, $X\Phi(\boldsymbol{f})$ *is given as*

$$X\Phi(\boldsymbol{f}) = \sum_{i=1}^N \sum_{n=1}^d \langle \boldsymbol{h}, \tilde{\boldsymbol{v}}_n \rangle \left((K^{1/2})^\dagger \boldsymbol{u}_n\right)_i \Phi(\boldsymbol{f}_i). \tag{13}$$

The n-th kernel relative principal component of $\boldsymbol{f}$ *with respect to* $\boldsymbol{g}$ *is given as*

$$\sum_{i=1}^{N}\langle \boldsymbol{h}, \tilde{\boldsymbol{v}}_n\rangle \left((K^{1/2})^{\dagger}\boldsymbol{u}_n\right)_i \Phi(\boldsymbol{f}_i). \tag{14}$$

Furthermore, we have

$$\|X\Phi(\boldsymbol{f})\|^2 = \sum_{n=1}^{d} |\langle \tilde{\boldsymbol{v}}_n, \boldsymbol{h}\rangle|^2 \tag{15}$$

and

$$\|X\Phi(\boldsymbol{f}) - \Phi(\boldsymbol{f})\|^2 = k(\boldsymbol{f},\boldsymbol{f}) + \sum_{n=1}^{d}\{|\langle \tilde{\boldsymbol{v}}_n, \boldsymbol{h}\rangle|^2 - \langle \tilde{\boldsymbol{v}}_n, \boldsymbol{h}\rangle\langle (K^{1/2})^{\dagger}\boldsymbol{u}_n, \boldsymbol{h}\rangle\}. \tag{16}$$

Outline of the Proof
We can expand the sum in eq.(7) and simplify to

$$J = \|K^{1/2}A\tilde{K} - K^{1/2}A_0\tilde{K}\|_2^2 - \mathrm{tr}[\tilde{K}^T A_0^T K A_0 \tilde{K}] + \frac{1}{L}\mathrm{tr}[K_0]. \tag{17}$$

where $\|\cdot\|_2$ is the Frobenius norm of a matrix. From eq.(17), J is minimum subject to rank$(A) \leq d$ if and only if $\|K^{1/2}A\tilde{K} - K^{1/2}A_0\tilde{K}\|_2^2$ is minimum with the condition. From the definitions of $\boldsymbol{u}_n$ and $\boldsymbol{v}_n$, by considering SVD of $K^{1/2}A_0\tilde{K}$, J is minimum subject to rank$(A) \leq d$ if and only if

$$K^{1/2}A\tilde{K} = \sum_{n=1}^{d} \sqrt{\lambda_n}\boldsymbol{u}_n\boldsymbol{v}_n^T. \tag{18}$$

Note that the sum of (18) is truncated by d. Then, we have

$$A = (K^{1/2})^{\dagger}\sum_{n=1}^{d}\boldsymbol{u}_n\tilde{\boldsymbol{v}}_n^T. \tag{19}$$

Then, eq.(10) is proved. Furthermore, $X\Phi(\boldsymbol{f})$ is given as

$$X\Phi(\boldsymbol{f}) = \sum_{i=0}^{N}(A\boldsymbol{h})_i\Phi(\boldsymbol{f}_i) = \sum_{i=1}^{N}\sum_{n=1}^{d}\langle \boldsymbol{h}, \tilde{\boldsymbol{v}}_n\rangle\left((K^{1/2})^{\dagger}\boldsymbol{u}_n\right)_i \Phi(\boldsymbol{f}_i). \tag{20}$$

The rest of the proof is clear. □

3 Application to Pattern Recognition

In order to show the advantage of KRPCA, we provide an experimental result of hand-written character recognition.

Let Ω_c be the learning sample set for a category c ($c = 1, 2, \cdots, N_C$). The matrix P_c and the operator P'_c of PCA and KPCA for the category Ω_c are decided as minimizing

$$\frac{1}{|\Omega_c|}\sum_{\boldsymbol{f}\in\Omega_c}\|P_c\boldsymbol{f} - \boldsymbol{f}\|^2, \qquad \frac{1}{|\Omega_c|}\sum_{\boldsymbol{f}\in\Omega_c}\|P'_c\Phi(\boldsymbol{f}) - \Phi(\boldsymbol{f})\|^2$$

subject to $\mathrm{rank}(P_c)$ and $\mathrm{rank}(P'_c)$ are fixed, respectively, where $|\Omega_c|$ is the number of samples in Ω_c.

The matrix X_c and the operator X'_c of the RPCA and the KRPCA for a category c is decided as minimizing with a parameter α

$$\frac{1}{|\Omega_c|}\sum_{\boldsymbol{f}\in\Omega_c}\|X_c\boldsymbol{f}-\boldsymbol{f}\|^2+\alpha\frac{1}{|\Omega_x|}\sum_{\boldsymbol{g}\in\Omega_x}\|X_c\boldsymbol{g}\|^2, \tag{21}$$

$$\frac{1}{|\Omega_c|}\sum_{\boldsymbol{f}\in\Omega_c}\|X'_c\Phi(\boldsymbol{f})-\Phi(\boldsymbol{f})\|^2+\alpha\frac{1}{|\Omega_x|}\sum_{\boldsymbol{g}\in\Omega_x}\|X'_c\Phi(\boldsymbol{g})\|^2 \tag{22}$$

subject to $\mathrm{rank}(X_c)$ and $\mathrm{rank}(X'_c)$ are fixed, respectively, where Ω_x is the set of samples which are suppressed. We call Ω_x the suppression set. In the above criterion, the notations $\boldsymbol{f}$ and $\boldsymbol{g}$ express patterns in the own and the others categories, respectively.

An unknown pattern $\boldsymbol{h}$ is discriminated as the category c for each method, when for all $b \neq c$ we have

$$\begin{aligned}
&\|P_c\boldsymbol{h}-\boldsymbol{h}\|^2<\|P_b\boldsymbol{h}-\boldsymbol{h}\|^2, &&\|P'_c\Phi(\boldsymbol{h})-\Phi(\boldsymbol{h})\|^2<\|P'_b\Phi(\boldsymbol{h})-\Phi(\boldsymbol{h})\|^2,\\
&\|X_c\boldsymbol{h}-\boldsymbol{h}\|^2<\|X_b\boldsymbol{h}-\boldsymbol{h}\|^2, &&\|X'_c\Phi(\boldsymbol{h})-\Phi(\boldsymbol{h})\|^2<\|X'_b\Phi(\boldsymbol{h})-\Phi(\boldsymbol{h})\|^2.
\end{aligned}$$

In cases of PCA and KPCA since P_c and P'_c are orthogonal projection matrix and operator, $\|P_c\boldsymbol{h}-\boldsymbol{h}\|^2$ and $\|P'_c\Phi(\boldsymbol{h})-\Phi(\boldsymbol{h})\|^2$ are minimum if and only if $\|P_c\boldsymbol{h}\|^2$ and $\|P'_c\Phi(\boldsymbol{h})\|^2$ are maximum, respectively. Usually the latter rules are used as the discriminant laws.

In point of view of the learning set, KPCA and KRPCA use the same learning set. In point of view of calculation complexity, the dimension of the space where we have to calculate is the number of learning samples used for evaluations. For KPCA and KRPCA the numbers are given as $|\Omega_c|$ and $|\Omega_c| + |\Omega_x|$, respectively. Therefore, it is difficult to use all samples, which do not belong to Ω_c, for the suppression set Ω_x. Then, we fix $|\Omega_x|$ as N_x. Consider a value $t(\boldsymbol{g}) = \|P'_c\Phi(\boldsymbol{g})\|/\|P'_b\Phi(\boldsymbol{g})\|$ for a pattern $\boldsymbol{g}$ in Ω_b $(b \neq c)$. Let t_{N_x} be the N_x-th largest value among $t(\boldsymbol{h})$ for all patterns $\boldsymbol{h}$ in Ω_b $(b \neq c)$. We add the pattern $\boldsymbol{g}$ to the suppression set Ω_x with respect to Ω_c when $t(\boldsymbol{g})$ is not less than t_{N_x}.

3.1 Data

We use US Postal Service database (USPS) which contains 7291 training patterns and 2007 test patterns collected from real-life zip codes. It has ten categories from '0' to '9' ($N_c = 10$). For a preprocessing we use the weighted direction index histogram method and the variable transformation [19].

3.2 Result

The following kernel is used in this experiment for KPCA and KRPCA.

$$k(\boldsymbol{f},\boldsymbol{g}) = (\langle\boldsymbol{f},\boldsymbol{g}\rangle+1)^{20}. \tag{23}$$

Table 1. Result of handwritten character recognition.

METHOD	ERROR RATE (%)	RANK	α	$\lvert\Omega_x\rvert$
PCA	5.38	9	–	
RPCA	4.91	12	$10^{-2.5}$	25
KPCA	4.43	150	–	
KRPCA	3.49	20	$10^{-4.0}$	150

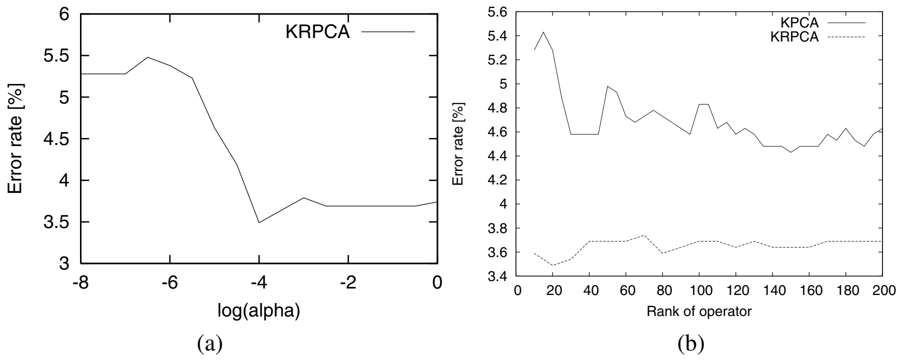


Fig. 1. (a) Parameter α and error rate by KRPCA(rank=20) ($|\Omega_x| = 150$). (b) Rank and error rate in KRPCA($\alpha = 10^{-4.0}$, $|\Omega_x| = 150$) and KPCA.

We show the best error rate of the test set for each method among various ranks and values of the parameter α, $|\Omega_x|$ in Table 1. We show the relation between the parameter α and the error rate of KRPCA in Figure 1 (a). We also show the relation between the ranks and the error rates of KPCA and KRPCA in Figure 1 (b).

We can see from Table 1 that KRPCA performs the best recognition rate. We can also see from Figures 1 (a) and (b) that KRPCA outperforms KPCA for any $\alpha > 10^{-4}$ and for any rank of operators, respectively.

4 Discussion

4.1 Computational Complexity

In the construction stage, the computational complexity to calculate a matrix of KRPCA is several times higher than that of KPCA. The dominant parts of computations are as follows.

– KPCA

the kernel function:	L^2 times
the eigen value problem of an (L, L)-matrix:	1 times

– KRPCA

the kernel function:	N^2 times
the inversion of an (N, N)-matrix:	2 times
the SVD of an (N, N)-matrix:	1 times
the square root of a symmetric (N, N)-matrix:	1 times

In the recognition stage, since many terms in eq.(16) can be calculated in advance, the dominant parts of computations are as follows.

- KPCA
  the kernel function: L times
  multiplication of elements: DL times
- KRPCA
  the kernel function: N times
  multiplication of elements: $2DN$ times

where D is the rank of each operator. In case of the experiment in Section 3, $L = 729$ and $N = 879$ in average and $D = 150$ for KPCA and $D = 20$ for KRPCA to achieve minimum error rates. In this case, the computational complexity of KRPCA is less than that of the KPCA in the recognition stage.

4.2 Comparison with Other Kernel Machines

SVM has the problem that training needs enormous computational cost because the optimization problem becomes very large. It depends on the complexity of the problem and the number of samples belonging to both its own and all rival classes. If the total number of samples is very large, kernel fisher discriminant (KFD) also has the same problem. On the other hand, KPCA and KRPCA are trained by the samples of its own class only and by those and some samples of rival classes, respectively, and can be solved only by matrix computations. Therefore, even if the number of classes is very large, KPCA and KRPCA can be obtained easily compared to SVM and KFD. Since SVD has a convex criterion and SVM does not have, sample selection for SVM also needs much computational cost. In practice, the KRPCA with a few thousands of samples for a class can be obtained by a present personal computer. Furthermore, since KRPCA extracts features of which mean square error is minimized, it can be used not only for discrimination but also for analysis.

5 Conclusion

We proposed the theory of the kernel relative principal component analysis (KRPCA). KRPCA can extract the principal components of a signal while suppressing the effects of other signals. We provided its definition and a solution. In order to show the advantage of KRPCA, we provided an experimental example of handwritten character recognition.

Acknowledgments

The authors wish to thank members in USPS for providing the date of the handwritten numeral characters. This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (C), 1550101, 2003.

References

1. Watanabe, S., Pakvasa, N.: Subspace method in pattern recognition. Proc. 1st Int. J. Conf on Pattern Recognition, Washington DC (1973) 25–32
2. Oja, E.: Subspace Methods of Pattern Recognition. Research Studies Press, Hertfordshire (1983)
3. Diamantaras, K.I., Kung, S.Y.: Principal Component Neural Networks. John Wiley & Sons, Inc., Yew York (1996)
4. Yamashita, Y., Ogawa, H.: Relative Karhunen-Loève transform. IEEE Trans. on Signal Processing **44** (1996) 371–378
5. Ikeno, Y., Yamashita, Y., Ogawa, H.: Relative Karhunen-Loève transform method for pattern recognition. Proc. of the 14th International Conference on Pattern Recognition, Brisben, Austraria **2** (1998) 1031–1033
6. Scharf, L.: The SVD and reduced rank signal processing. Signal Processing **25** (1991) 113–133
7. Fisher, R.A.: The use of multiple measurements in taxonomic problems. Annal Eugenics **7** (1936) 179–188
8. Schölkopf, B., Smola, A., Müller, K.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation **10** (1998) 1299–1319
9. Mika, S., Schölkopf, B., Smola, A.J., Müller, K.R., Scholz, M., Rätsch, G.: Kernel PCA and de-noising in feature spaces. In Kearns, M.S., Solla, S.A., Cohn, D.A., eds.: Advances in Neural Information Processing Systems 11, MIT Press (1999) 536–542
10. Schölkopf, B., Mika, S., Burges, C., Knirsch, P., Müller, K.R., Rätsch, G., Smola, A.: Input space vs. feature space in kernel-based methods. IEEE Transactions on Neural Networks **10** (1999) 1000–1017
11. Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Smola, A.J., Müller, K.R.: Invariant feature extraction and classification in kernel spaces. In Kearns, M.S., Solla, S.A., Cohn, D.A., eds.: Advances in Neural Information Processing Systems 12, MIT Press (2000) 526–532
12. Maeda, E., Murase, H.: Kernel based nonlinear subspace method for multi-category classification. Tech. Rep., Information Science Laboratory **ISRL-98-1** (1998)
13. Tsuda, K.: Subspace classifier in the Hilbert space. Pattern Recognition Letters **20** (1999) 513–519
14. Mika, S., Rätsch, G., Weston, J., Schölkopf, B., Müller, K.R.: Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E.W., Douglas, S., eds.: Neural Networks for Signal Processing IX, IEEE (1999) 41–48
15. Mika, S., Rätsch, G., Müller, K.R.: A mathematical programming approach to the kernel Fisher algorithm. In T.K. Leen, T.D., Tresp, V., eds.: Advances in Neural Information Processing Systems 13, MIT Press (2001) 591–597
16. Mika, S., Smola, A.J., Schölkopf, B.: An improved training algorithm for kernel Fisher discriminants. In Jaakkola, T., Richardson, T., eds.: Proceedings of Eighth International Workshop on Artificial Intelligence and Statistics, Morgan Kaufmann (2001) 98–104
17. Schatten, R.: Norm Ideals of Completely Continuous Operators. Springer-Verlag, Berlin (1970)
18. Ben-Israel, A., Greville, T.N.E.: Generalized Inverses: Theory and Applications. John Wiley & Sons, New York (1974)
19. T. Wakabayashi, S. Tsuruoka, F.K., Miyake, Y.: Increasing the feature size in handwritten numeral recognition to improve accuracy. Systems and Computers in Japan (Scripta Technica) **26** (1995) 35–44

Dynamic Character Model Generation for Document Keyword Spotting

Beom-Joon Cho[1] and Bong-Kee Sin[2]

[1] Department of Computer Engineering, Chosun University,
Dong-ku, Gwangju 501-759 Korea
bjcho@chosun.ac.kr

[2] Department of Computer Multimedia, Pukyong National University,
Dayon-dong 599-1, Nam-ku, Busan 608-737 Korea
bkshin@pknu.ac.kr

Abstract. This paper proposes a novel method of generating statistical Korean Hangul character models in real time. From a set of grapheme average images we compose any character images, and then convert them to P2DHMMs. The nonlinear, 2D composition of letter models in Hangul is not straightforward and has not been tried for machine-print character recognition. It is obvious that the proposed method of character modeling is more advantageous than whole character or word HMMs in regard to the memory requirement as well as the training difficulty. In the proposed method individual character models are synthesized in real-time using the trained grapheme image templates. The proposed method has been applied to key character/word spotting in document images. In a series of preliminary experiments, we observed the performance of 86% and 84% in single and multiple word spotting respectively without language models. This performance, we believe, is adequate and the proposed method is effective for the real time keyword spotting applications

1 Introduction

In the field of OCR the neural network is a highly successful model for recognizing machine-printed characters. However, one problem with the neural network is that the sequential nature of texts running left to right is not well captured without sophisticated network architectures like that of TDNN [1]. As a result, most of the neural network systems with ordinary architectures assume external segmentation of character blocks prior to recognition. In this case the overall system performance is usually limited by the performance of the segmenter and the quality of the resulting segments. Another problem with the neural network model is that it is a purely wholistic model that cannot be decomposed, analyzed nor synthesized; therefore training thousands of character models is extremely difficult, if not impossible.

Since the early nineties one model has come into the arena of document analysis; it is the hidden Markov model or HMM. Stimulated by the success in speech recognition, the modeling capability of the variability and sequential flow of a pattern, HMM has been used in diverse areas successfully [2]. The application of HMMs benefits from the wide range of experience accumulated in speech recognition and many other fields.

Since document texts run sequentially and, mostly, left to right, it is natural that the idea of using HMM occurs to researchers. To date, the HMM application to English has been reported in several places in the literature. But, although texts run linearly, individual character patterns are not linear but two-dimensional. This fact has not been a barrier to the modeling of Latin alphabet-based texts that run strictly left-to-right down at the letter level. In fact the idea is simply straightforward. But in the case of Korean Hangul characters the problem is not so simple. At the character level or above, texts run linearly. One problem is that there are thousands of characters used in Hangul texts, and we may need the corresponding number of models. An observation below the character level is that a Korean Hangul character is composed of either two or three graphemes arranged two-dimensionally in a way to fit into a rectangle. The two-dimensional composition of grapheme models in Hangul is not straightforward and thus the HMM has not been tried for machine-print character recognition. Without doubt, however, the composition method of character modeling is more advantageous than that of designing thousands of whole character models in regard to the memory requirement as well as the training difficulty.

This research is focused on the application of the HMM method to the analysis of document text images. The basic idea lies in the real time generation of Korean Hangul character models for spotting key characters in the content analysis of optical documents. In the proposed method individual character models are synthesized in real-time using the trained grapheme image templates.

Since characters are two-dimensional, it is natural to believe that a 2D HMM, an extension to the standard HMM, will be helpful and offer a great potential for analyzing and recognizing character patterns. But a fully connected 2D HMM leads to an algorithm of exponential complexity [3]. To avoid the problem, the connectivity of the network has been reduced in several ways, two among which are Markov random field and its variants [4] and pseudo 2D HMM [5]. The latter model, called P2DHMM, is a very simple and efficient 2D model that retains all of the useful HMM features. The basic idea of this paper is about the real time construction of the Hangul character pseudo 2D HMM using trained grapheme image templates. We believe the proposed method is feasible and particularly appropriate thanks to the absence of natural italic fonts corresponding to the English italics, a rationale for using P2DHMM.

In the proposed method, we prepared a set of grapheme patterns for each grapheme and obtained their average, the grapheme template. By superposing appropriate grapheme templates, we can compose a character image template. This character template is converted to a P2DHMM in a systematic way. In this method the new idea of location-preserving 2D superposition is very simple but highly elegant and efficient for real-time processing. The idea of character composition is not new, but the application to strictly 2D model design is. It is especially true in 2D HMM framework. Another feature of the proposed method is the conversion of the gray-scale template into P2DHMM, which is theoretically correct in the sense of maximum likelihood estimation. Additional noteworthy feature is model reduction by noting the information redundancy in the templates; successive HMM states are merged based on the similarity between their output PDs. The resulting models are often much smaller than the original and thus speed up the spotting task, and sometimes, improves the performance.

The rest of the paper consists as follows. In Section 2 we will briefly review the HMM and P2DHMM. In Section 3 the pseudo 2D HMM and its algorithm are described; and then a procedure for developing character models is discussed in detail. Section 4 describes auxiliary models needed for the proposed method of key character spotting. Section 5 presents results from preliminary experiments. Section 6 concludes the paper.

2 HMM Theory

This section reviews briefly the theories of HMM and pseudo 2D HMM.

2.1 HMM

The hidden Markov model is a doubly stochastic process that can be described by three sets of probabilistic parameters as $\lambda = (A, B, \pi)$. Given a set of N states and a set V of observable symbols, the parameters are formally defined by [2]:

- Transition probability: $A = \{ a_{ij} = p(q_t = j \mid q_{t-1} = i), 1 \le i, j \le N \}$, $\sum_j a_{ij} = 1$.
- Output probability: $B = \{ b_i(v) = p(x_t = v \mid q_t = i), 1 \le i \le N, v \in V \}$, $\sum_v b_i(v) = 1$.
- Initial transition probability: $\pi = \{ \pi_i = p(q_1 = i), 1 \le i \le N \}$, $\sum_i \pi_i = 1$.

The most frequent task with an HMM is the evaluation of the model matching score for an input sequence $X = x_1 x_2 \dots x_T$. It is given by the likelihood function of the sequence generated from the model

$$P(X \mid \lambda) = \sum_Q \pi_{q_1} b_{q_1}(x_q) \prod_{t=2}^{T} a_{q_{t-1}q_t} b_{q_t}(x_t) \qquad (1)$$

Although simple in form, the time requirement is exponential. Thanks to the use of the DP technique, this can be computed in linear time in T. However when it comes to 2D HMM formulation, even the DP technique alone is not enough. One research direction is the structural simplification of the model, and the pseudo 2D HMM is one solution.

2.2 P2DHMM

Pseudo 2D HMM in this paper is realized as a horizontal connection of vertical sub-HMMs (λ_k). But it is not the only one. The alternative realization is the vertical connection of horizontal sub-HMMs as in the work of Xu and Nagy [6]. In order to implement a continuous forward search method and sequential composition of word models, the former type has been used in this research.

Let us consider a *t*-th vertical frame $X_t = x_{1t}\, x_{2t} \dots x_{St}$, $1 \le t \le T$, in a text line image. This is a one-dimension sequence like that of X in Equation (1). This is modeled by a sub-HMM λ_k with the likelihood $P(X_t|\lambda_k)$. You may regard each sub-HMM λ_k as a super-state whose observation is a vertical frame of pixels.

$$P_{r_t}(X_t \mid \lambda_{r_t}) = \sum_{Q} \pi_{q_1} b_{q_1}(x_{1t}) \prod_{s=2}^{S} a_{q_{t-1}q_t} b_{q_t}(x_{st}) \tag{2}$$

Now let us consider a bitmap image which we define as a sequence of such vertical frames as $X = X_1\, X_2 \dots X_T$. Each frame will be modeled by a super-state or a sub-HMM. Let Λ be a sequential concatenation of sub-HMMs. Then the evaluation of Λ given the sample image X is

$$P(X \mid \Lambda) = \sum_{R} P_1(X_1) \prod_{t=2}^{T} \vec{a}_{r_{t-1}r_t} P_{r_t}(X_t) \tag{3}$$

where it is assumed that super-state process starts only from the first state. The P_{r_t} function is the super-state likelihood. Note that both of the Equations (2) and (3) can be effectively approximated by the Viterbi score.

One immediate goal of the Viterbi search is the calculation of the matching likelihood score between X and an HMM. The objective function for an HMM λ_k is defined by the maximum likelihood as

$$\Delta(X_t, \lambda_k) = \max_{Q} \prod_{s=1}^{S} a_{q_{s-1}q_s} b_{q_s}(x_{st}) \tag{4}$$

where $Q = q_1 q_2 \cdots q_S$ is a sequence of states of λ_k, and $a_{q_0 q_1} = \pi_{q_1}$. $\Delta(X_t, \lambda_k)$ is the similarity score between two sequences of different length. The basic idea behind the efficiency of DP computation lies in formulating the expression into a recursive form

$$\delta_s^k(j) = \max_{i} \delta_{s-1}^k(i) a_{ij}^k b_j^k(x_{st}) \quad j = 1, \dots, M_k,\ s = 1, \dots, S,\ k = 1, \dots K \tag{5}$$

where $\delta_s^k(j)$ denotes the probability of observing the partial sequence $x_{1t} \dots x_{st}$ in model k along the best state sequence reaching the state j at time/step s. Note that

$$\Delta(X_t, \lambda_k) = \delta_S^k(N_k) \tag{6}$$

where N_k is the final state of the state sequence. The above recursion constitutes the DP in the lower level structure of the P2DHMM. The remaining DP in the upper level of the network is similarly defined by

$$D(X, \Lambda) = \max_{k} \prod_{t=1}^{T} \vec{a}_{r_{t-1}r_t} \Delta(X_t, \lambda_{r_t}) \tag{7}$$

that can similarly be- reformulated into a recursive form. Here $\vec{a}_{r_1 r_2}$ denotes the probability of transition from super-states r_1 to r_2. According to the formulation described thus far, a P2DHMM adds only one parameter set, the super-state transitions, to the conventional HMM parameter sets. Therefore it is a simple extension to conventional HMM.

3 Character Modeling

One of the most important tasks in hidden Markov modeling is estimating the probabilistic parameters. For this task we assume a set of typical samples of character images X = { $X^{(1)}$, …, $X^{(D)}$ } of an equal dimension. Different size raises no problem if we scale the images bilinearly. Moreover, the scale difference in test images is naturally resolved with HMM method.

The focus of the section lies in the construction of the P2DHMM for a Korean Hangul character. A Hangul character consists of either two or three graphemes of phonetic consonant and vowel letters. The composition follows a general rule to fit the graphemes into a rectangle. There are six types of combination (Fig. 1) according to the shape of the vowel (horizontal, vertical, or both) and the presence of consonant suffix. The proposed method of model creation is based on the given set of bitmap images. The overall procedure is shown in Fig. 2, and explained as follows:

(1) Grapheme segmentation. This step involves extracting the individual graphemes from character samples while retaining the location inside the box enclosing the character. As illustrated in Fig. 3, the graphemes are separated while retaining the position with the box. In its simplest form this step is the most costly in the proposed method. But the problem can be avoided by using a bootstrapping strategy or a little more sophisticated prototyping idea [6].

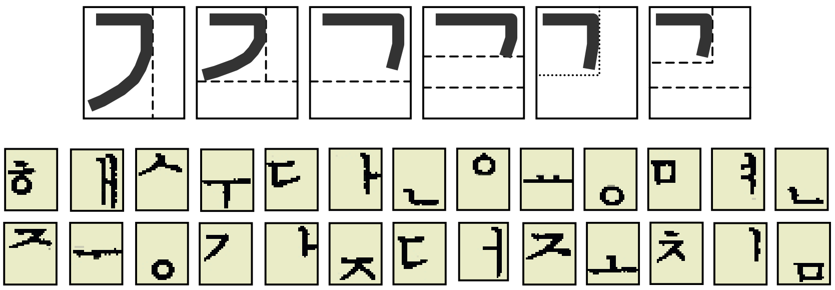


Fig. 1. (Top) Six types of grapheme arrangement inside a Korean syllable character box. A grapheme changes its shape according to the type. (Bottom) Grapheme segment samples.

(2) Average the extracted samples. Now there is a set of grapheme samples. First we classify the samples according to the type of the grapheme arrangement pattern of the original character. For the initial consonant grapheme there are six types (see Fig. 1), and two for each vowel grapheme. Then, take the sample average of the set of categorized images pixel by pixel so that a smooth grayscale-like image is obtained (see Fig. 3). Assuming binary samples, the average intensity of the pixel at (i, j) is

$$\bar{x}_{ij} = \frac{N_{ij}}{N}$$

where N_{ij} is the number of samples whose (i, j) pixel is black (or white) and N is the total number of samples. Essentially the training phase is finished at this stage.

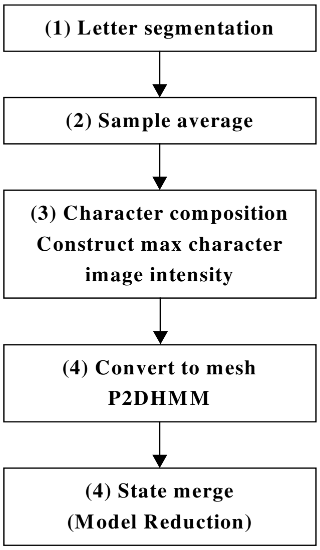


Fig. 2. Model design procedure.

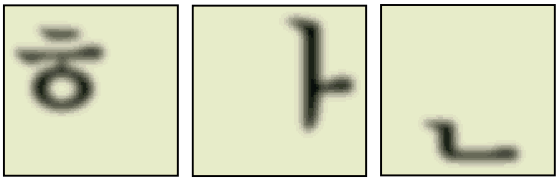


Fig. 3. Korean graphemes separated out from a syllabic character for */han/*. From left to right: the initial consonant, the vowel, and the suffix consonant. Note that the grapheme position in the original character block is retained.

(3) Character image construction. From this step on the process belongs to the decoding or recognition phase, and is performed in real time. Here the given task is to spot or recognize a character. The image template of the character is synthesized in the image domain from the component grapheme images generated in the previous step(Fig. 4). The value of the (i, j)-th pixel of the character template takes the maximum of the two or three pixels (i, j) from each grapheme plane.

(4) Conversion into P2DHMM. Given a character image, it is straightforward to construct a P2DHMM. First assign a state to every pixel with the output probability being the intensity value. Then the states are linked according to the topological constraint of P2DHMM: vertical sub-state transition, and then horizontal transition between super-states. Note that all the transition probabilities are one without self-transitions. There is no space-warping in the current model.

(5) State merge. When two or more successive states are similar in the output probability (gray scale), they are replaced with a new node with a modified output probability

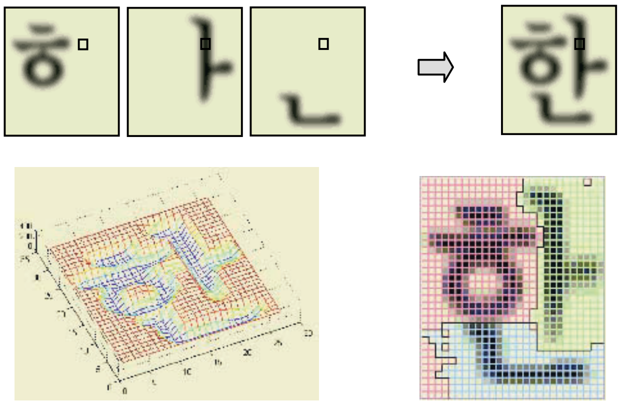


Fig. 4. Character image construction and the result.

$$\bar{x}_{ij} = w_i x_{ij} + w_{i+1} n_{i+1,j}$$

where w_i is the weight as a function of the duration in the state *i*. The state similarity is measured by the output probability of the states. In the case of super-states, the distance measure is

$$d(x, n_k) = |x - n_k|^{\alpha}$$

where $\alpha \in R$. If $\alpha = 2$, then this measures the dissimilarity in the least square sense. Then we estimate the transition probability similarly, or the whole transition parameter set may be replaced by state duration probabilities.

The proposed procedure of creating statistical model is theoretically correct in the sense of maximum likelihood sense. One problem with the method is found in the final stage of merging states. But it is justified because, although the method of state merging itself is coarse yet, the idea of merging is correct information-theoretically.

4 Keyword Spotting

For keyword spotting task, we developed two more classes of P2DHMMs in addition to key character models, and then combined then into a network model for continuous decoding of input streams.

4.1 Filler Model

In keyword spotting task, a filler corresponds to something between interesting things or keywords. It is also called a non-key character. Then a filler model is defined as the model for all non-key characters. For convenience sake, however, it does not discriminate keys and non-keys but models all kinds of character patterns statistically. The desired characteristic of the filler model λ_F is

$$p(x^K \mid \lambda_F) < p(x^K \mid \lambda_K)$$
$$p(x^F \mid \lambda_F) > p(x^F \mid \lambda_K)$$

where λ_K is a key model for the key pattern x^K, and x^F is a non-key pattern. In general, however, the character patterns are not completely random and there is a certain degree of similarity between some characters. In addition it is not easy to design a single good model for numerous patterns of all characters. According to the work of Lee and Kim [7], the filler model can behave as a threshold. For better thresholding capability in Korean Hangul characters, we defined six fillers, one for each of the six character composition types as of Fig. 1. Fig. 5 shows the filler images before conversion to P2DHMMs. They are simple arithmetic averages over a large set of character samples. Unlike the key character models, the filler models are not synthesized in real-time. Rather they are prepared once and for all from the image templates. Compared to the key model construction, filler model creation is very simple.

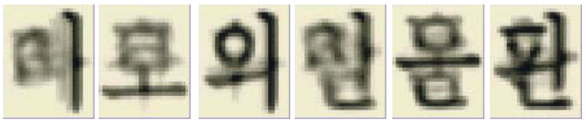


Fig. 5. Bitmap images for filler models.

4.2 White Space Model

The region excluding the text is white space. The white space will be limited to the white frames between characters. It is modeled with a small number of nodes. Actually the state merge step reduced the nodes to one or two most of the time in practice.

4.3 Spotting Network

For character spotting task we have designed a network-based transcription model(Fig. 6). It is a circular digraph with a backward link via the space model so that it can model arbitrary long sequence of non-key as well as key patterns. Given such a network, an input sequence of will be aligned to every possible path circulating the network. One circulation is called a level. An *l* level path hypothesis represents a string of *l* characters [8]. The result is retrieved from the best hypothesis.

4.4 Search Method

The spotter network models a small set of key patterns and used to locate them while ignoring the rest of the words of no interest. One efficient search is the one stage DP. For the continuous spotting with forward scanning, we applied a modified form of two-level one-stage DP; this performs a single forward pass consisting of alternating partial forward search and output. The time requirement for the DP is $O(N^2ST)$ where N is the total number of states or nodes, and S and T are the frame length and the number of vertical frames in a line [9] respectively. In the proposed method, this is reduced to $O(NST)$.

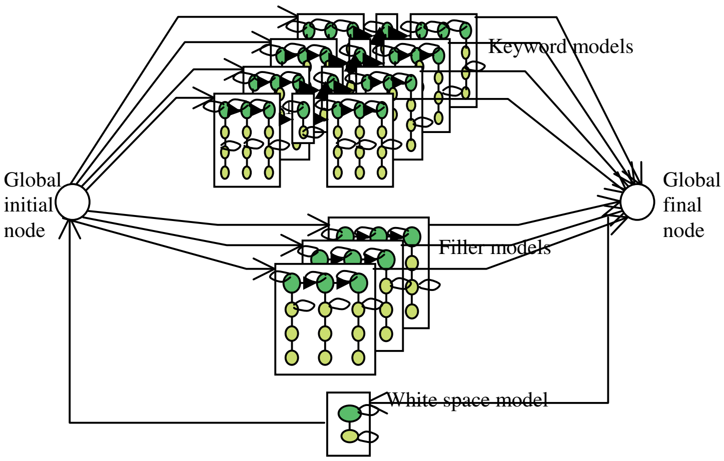


Fig. 6. The circular network of P2DHMMs for spotting keywords.

5 Experiments

5.1 Hangul Character Spotting

One significant characteristic of Korean text is that there are no natural italic fonts. This observation justifies the use of simple image-based modeling scheme. In the initial experiment a limited test has been performed using 10 point (*Myongjo* font) character images scanned in 200dpi resolution. The letter models were created from the hand-segmented letter images. In this test we prepared only a single image for each letter and blurred it by a Gaussian filter to the effect of averaging multiple images. The most frequently used 97 character classes were used in character (not word) spotting task. The character set constitutes approximately the half of the test text.

The test result has been analyzed in terms of correct spottings(H), false positives(P) and false negatives(N). The overall spotting performance was 79.7 percent as shown in Table 1. In order to better understand the performance and weakness, we detailed the result into character type hits and failures in the same table. The character Types I to VI correspond to the six different arrangements (see Fig. 1) of Hangul vowels and consonants. Here the type hit means that the type of the character is correct regardless of the correctness of the character label. According to the table, the hit ratios of Type III and V are relatively low, for which false acceptance and rejection are high. We noted this fact to refine the models and tune merge parameters for the next set of experiments.

Table 1. Korean Hangul character spotting result. (h = the number of hits, p = the number of false positives, and n = the number of false negatives; H = h/(h+p+n), P = p/(h+p+n), and N = n/(h+p+n)) All figures are in %.

	H	*FA*	*FR*	#Classes
Character	79.7%	10.9%	9.4%	107
Type average	87.0%	5.6%	7.4%	6
Type I	90.9%	0.0%	9.1%	(20)
Type II	91.7%	8.3%	0.0%	(22)
Type III	81.3%	12.5%	6.3%	(17)
Type IV	88.9%	11.1%	0.0%	(18)
Type V	80.0%	0.0%	20.0%	(19)
Type VI	87.5%	0.0%	12.5%	(11)

5.2 Word Spotting

A word is a linear left-to-right concatenation of characters in Hangul system. For word spotting task we tested a mixture of fourteen keyword models on a set of one hundred journal papers' abstract images. In this test we fixed the filler models optimized previously since they need not be created dynamically at run time.

Fig. 7 shows the performance change by varying the state merging thresholds. In the graph the highest hit(H) reaches 66.7% at the threshold of 0.03. In this case the recall is very high but the precision is sacrificed a lot; the best precision score is obtained at threshold 0.01.

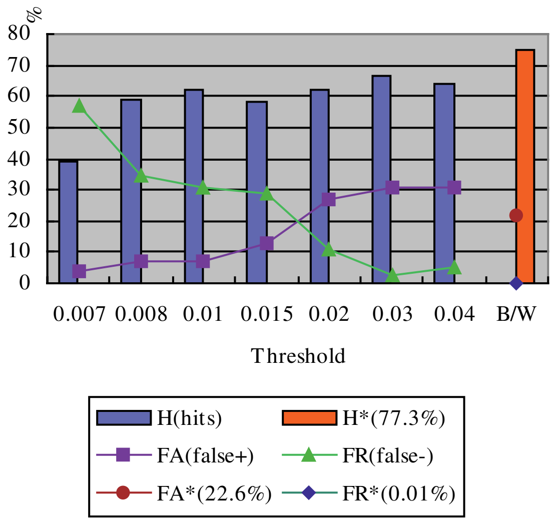


Fig. 7. Word spotting result.

Let us compare the performance figures of the proposed method with those of spotting with Baum-Welch-trained P2DHMMs which are assumedly optimal. The latter models record 77.3%(H* for the hit rate), 22.6%(P* for the false positives) and 0.1%(N* for the false negatives), as are separately marked at threshold 1.0. The Baum-Welch modeling method for the P2DHMM, although superior, cannot be used for large vocabulary keyword spotting tasks that require training tens of thousand P2DHMMs and preparing a huge number of character samples. This implies that the proposed method of dynamic synthesis of key character P2DHMMs has a definite advantage over the traditional Baum-Welch modeling. Furthermore, if a higher precision is desired, we can pass the spotted word images to a high-performance recognizer for a more accurate spotting. This method will be far faster than the full recognition of the whole documents.

Fig. 8. A sample result, part of screen shot, showing correct spotting and filler type classification.

Fig. 8 gives a sample result, a part of screen shot, showing correct spotting and filler type labeling. Note the small gaps between fillers. They denote white spaces between characters.

5.3 Keyword Set Spotting

In the final experiment we compared the hit ratio by varying the number of keywords sought at a time. Table 2 summarizes the result. When the number of keywords $N = 1$, the hit ratio reached peak, above from character spotting performance. When N increases, the confusion among words also increases thus degrading the performance gradually. The last column corresponds to the highest hit ratio in the preceding experiment. But when N is moderately large, the word spotting task is more successful than the individual character spotting task where we used about four character models at a time, about two Korean Hangul syllable characters in a word.

Table 2. Character and word spotting performance with increasing number of keyword models spotted at a time.

	Character spotting ($N = 2$)	word spotting (N = # keywords)		
		$N = 1$	$N = 2.5$	$N = 14$
Hit ratio(H)	75.0%	86.3%	83.8%	66.7%

6 Conclusion

Using a set of letter image templates, we proposed a very effective method for real time synthesis of key word P2DHMMs. The method is based on the principle of composing Hangul syllable characters. The composition itself is very efficient and its conversion to a P2DHMM is highly intuitive considering that we are dealing with machine printed character images. With experimental results form the application to key word spotting tasks, we consider that the method is highly feasible and meets our ultimate demand for the application to content-based document image indexing and retrieval.

References

1. Lang, K., Waibel, A., Hinton, G.: A time delay neural network architecture for isolated word recognition, Neural Networks, 3(1990), 23–44
2. Rabiner, L. R.: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE, *77*(1989), 257–286
3. Levin, E., Pieraccini, R.: Dynamic planar warping for optical character recognition, Proc. ICASSP, 3(1992), San Fransisco, CA, 149–152
4. Chellapa, R., Chatterjee, S.: Classification of textures using Gaussian Markov random fields, IEEE Trans. ASSP*,* 33(1985), 959–963
5. Agazzi, O. E., Kuo, S.: Hidden Markov model based optical character recognition in the presence of deterministic transformations, Pattern Recognition, 26(1993), 1813–1826
6. Xu, Y., Nagy, G.: Prototype extraction and adaptive OCR, IEEE Trans. PAMI, 21(1999), 1280–1296
7. Lee, H.-K., Kim, J.H.: An HMM-based threshold model approach for gesture recognition, IEEE Trans. PAMI, 21(1999), 961–973
8. Meyers, C. S., Rabiner, L. R.: A level building dynamic time warping algorithm for connected word recognition, IEEE Trans. ASSP, 29(1981), 284–297
9. Sakoe, H.: Two-level DP-matching - a dynamic programming-based pattern matching algorithm for connected word recognition, IEEE Trans. on ASSP, 27(1979), 588–595

Texture Classification by Combining Wavelet and Contourlet Features

Shutao Li[1,2] and John Shawe-Taylor[2]

[1] College of Electrical and Information Engineering, Hunan University,
Changsha, 410082, P.R. China
shutao_li@yahoo.com.cn

[2] ISIS Research Group, School of Electronics and Computer Science,
University of Southampton, SO17 1BJ, UK
jst@ecs.soton.ac.uk

Abstract. In the recent decades, many features used to represent a texture were proposed. However, these features are always used exclusively. In this paper, a novel approach is presented that combines two types of features extracted by discrete wavelet transform and contourlet transform. Support vector machines (SVMs), which have demonstrated excellent performance in a variety of pattern recognition problems, are used as classifiers. The algorithm is tested on four different datasets, selected from Brodatz and VisTex database. The experimental results show that the combined features result in better classification rates than using only one type of those.

1 Introduction

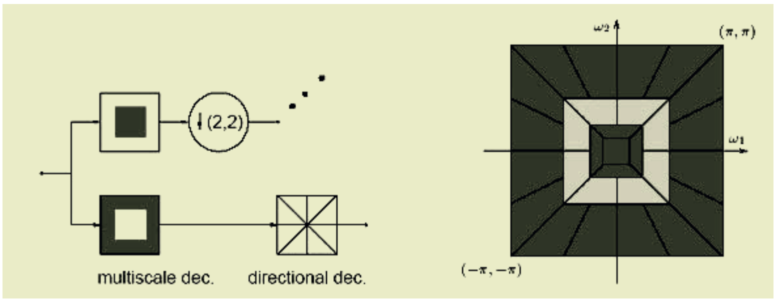
Texture analysis is important in many applications, such as object recognition, image retrieval, remote sensing, and biomedical image analysis, scene interpretation and segmentation.

The method for feature extraction from texture is critical to the success of the texture classification. Many methods have been proposed to extract texture features, such as the co-occurrence matrices [1], the Markov random fields [2], fractals [3], and the Gabor filters [4], wavelet transforms [5,6] and quadrature mirror filters [7]. Recently, Randen and Husøy did an extensive review and comparative study for texture classification on most major filtering-based approaches [8]. They concluded that various filtering approaches yield different results for different images.

However, Most of these previous studies have used on the features individually. In this paper, a novel approach is presented that combines two types of features extracted by discrete wavelet transform and contourlet transform. Compared to wavelet transform, which can only offer limited directional information in representing image edges, the contourlet proposed by Do and Vetterli can capture the intrinsic geometrical structure in images [9]. Unlike other transforms, such as curvelets, that were initially developed in the continuous-domain and then discretized for sampled data, the contourlet starts with a discrete-domain transform. The discrete contourlet transform has a fast-iterated filter bank algorithm that requires an order N operation for N-pixel images.

The classifier is also clearly critical in texture classification. The support vector machine [10,11], which has found important applications in image classifications, is used as classifiers for texture classification.

This paper is structured as follows. Section 2 briefly introduces the contourlet. In section 3 the feature extraction methods using wavelet and contourlet are described. Section 4 describes the datasets, experiments and their results. Conclusions can be found in section 5.



(a) (b)

Fig. 1. (a) Block diagram of the contourlet filter bank. First, a standard multiscale decomposition into octave bands is computed, where the lowpass subband is subsampled and iterated, while a directional filter bank is applied the to the bandpass subband. (b) Resulting frequency division, where the number of directions is increased with frequency

2 Contourlet

Do and Vetterli developed the contourlet representation based on an efficient two-dimensional nonseparable filter banks that can deal effectively with images having smooth contours [9]. The block structure for the contourlet filter bank is shown in Figure 1 together with an example of its frequency partition. In contourlet transformation, the Laplacian pyramid is first used to capture the point discontinuities, then followed by a directional filter bank to link point discontinuities into linear structures. Contourlets possess not only the main features of wavelets, namely multiresolution and time-frequency localization, but they also show a high degree of directionality and anisotropy. Precisely, contourlet transform involves basis functions that are oriented at any power of two's number of directions with flexible aspect ratios. With such richness in the choice of bases, contourlets can represent any one-dimensional smooth edges with close to optimal efficiency.

3 Feature Extraction and Combination

The original texture sample is first decomposed using wavelet and contourlet. For each filtered subband image (except the one from the approximation subband) from wavelet, its mean and standard deviation are used to identify the textures based on the common belief. Denote the M×N image obtained in subband i by I_i, its mean and standard deviation are shown as follows.

Mean:

$$MEAN = \frac{1}{M \times N} \sum_{x=1}^{M} \sum_{y=1}^{N} |I_i(x, y)| \tag{1}$$

Standard deviation:

$$STD = \sqrt{\frac{1}{M \times N} \sum_{x=1}^{M} \sum_{y=1}^{N} (I_i(x, y) - MEAN)^2} \tag{2}$$

By decomposing the image to d levels using wavelet, we thus obtain a feature vector of length 6×d.

To contourlet, only the means of absolute decomposition values in subbands are used as feature vector. Figure 2 shows the contourlet representation of one texture image with [2 3] decomposition levels, from which we can extract 13 features.

Commonly, the exploitation of different information sources for the same recognition task often leads to different errors in the recognition results, which are very often complementary. This means that an appropriate exploitation of these sources can effectively reduce the error rate. Thus, we have chosen the strategy of using the combination of features from wavelet and contourlet for texture classification. Here the Combination of features from wavelet and contourlet are achieved by simply concatenating them.

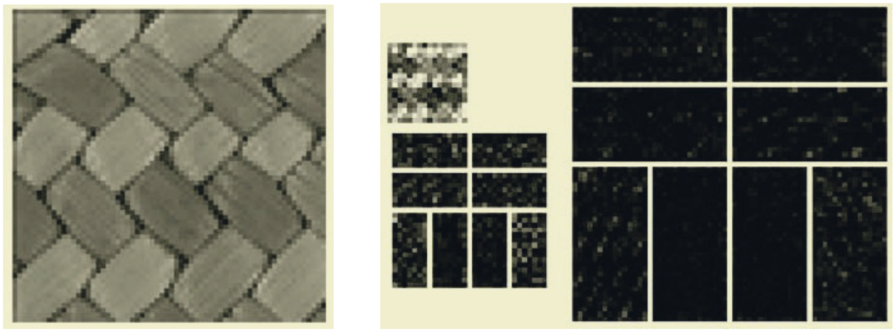

(a) (b)

Fig. 2. (a) One texture region; (b) Its contourlet transform

4 Experimental Results

4.1 Data

In this experiment, we have applied our method to four datasets from two commonly used texture sources: the Brodatz album and the MIT Vision Texture database.

The dataset 1 has 40 textures with size of 256×256, ten of which are shown in Fig.3. This dataset is challenging because there are significant variations within some textures and some of them are very similar to each other [12]. The dataset 2 and 3, shown in Fig. 4 (a) and (b), both have 10 textures are challenging too, which with size of 128×128, used in [8] and [12]. For these two groups, due to the inhomogeneity and large variations, texture types in local windows are perceptually close. The dataset 4 has 28 textures with size of 256×256 used in [5]. All textures are gray-scale images when presented to the methods. The dynamic ranges are represented by eight bits per sample.

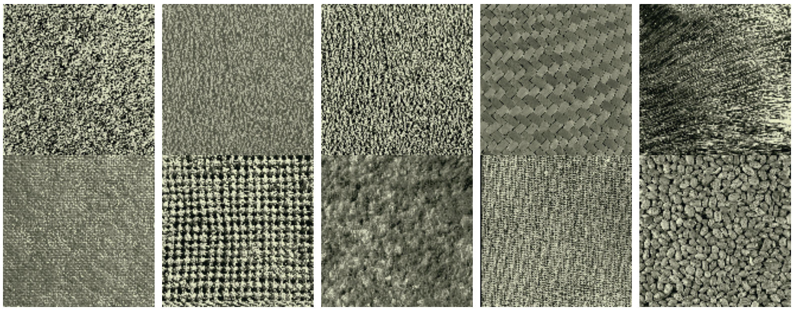
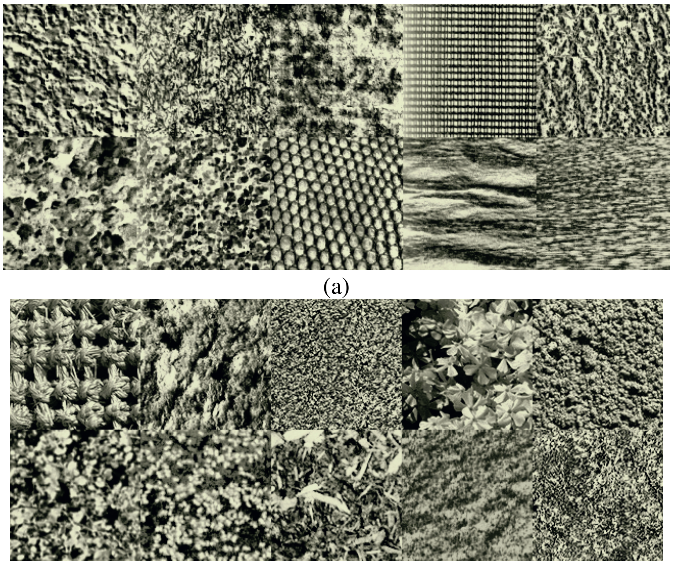


Fig. 3. Ten of the dataset 1 with 40 textures. The input image size is 256×256. These images are available at http://www-dbv.cs.uni-bonn.de/image/texture.tar.gz



(a)

(b)

Fig. 4. (a) Dataset 2 with 10 textures; (b) Dataset 3 with 10 textures. The image size is 128×128. These images are available at http://www.ux.his.no/~tranden/

In this experiment, we use a complete separation between the training and test sets and repeat the experiment 100 times and compute the average performance.

The classification gain defined by [12] is adopted as a measure to test the classification performance,

$$G = \frac{1 - C_{err}}{\frac{1}{M}} = M(1 - C_{err}) \tag{3}$$

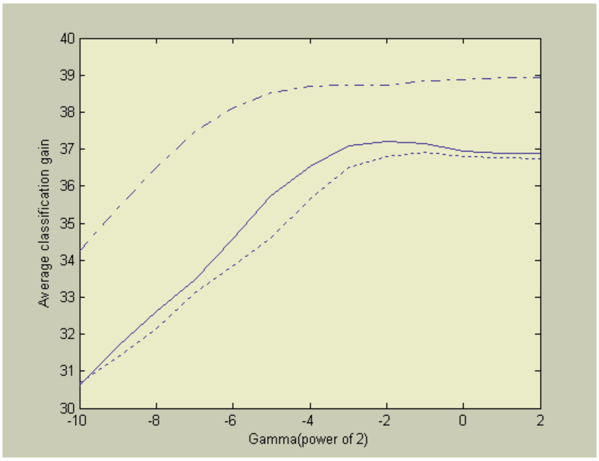
where C_{err} is the classification error rate, M is the total number of classes in the database. G measures the effectiveness of classifiers more objectively than C_{err} because is C_{err} closely related to M.

4.2 Selection of SVM Parameters

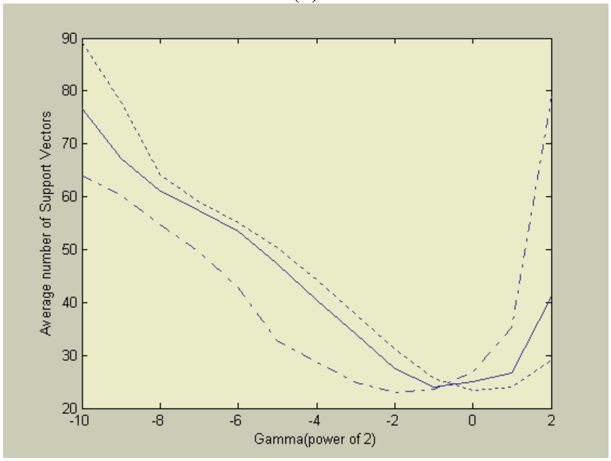
SVMs were originally designed for binary classification. Dealing with multi-class problem, SVMs can use one-against-the rest, one-against-one, error-correcting output coding and other methods. In this paper, we adopt the one-against-the rest. For training a class-C_i, the one-against-the rest method labels the data for the class-C_i as +1 and the data for other classes as –1. On classifying a new sample, the classifier with the largest output will be selected as the winner, and this new sample is assigned to the winner's corresponding texture class.

In the experiment, the Gaussian kernel will be used in the SVM, because preliminary results suggest that the Gaussian kernel outperforms the polynomial kernel. To select a suitable gamma value for Gaussian kernel function, we use the dataset 1, where size is 256×256, is subdivided into 64 nonoverlapping samples of size 32×32, resulting in a total of 2560 samples. For each sample, 3 level wavelet using “coif4” filters and [2 3] level contourlet with “pkva” filters are separately applied, so 18 and 13 features are obtained, respectively, resulting in the combined feature vector with length of 31. In the total samples, 50% are used for training and the rest of 50% are used for testing. The figures of average classification gain and the average number of support vectors versus gamma value are shown in Fig.5(a) and Fig.5(b), respectively, where C value is set to 1000.

It can be found that the classification gain on the combined feature is much better than those on wavelet feature and contourlet feature individually from Fig.5(a). The classification gains using individual wavelet features and contourlet features reach the maximum when gamma equals to 2^{-3} and 2^{-2}, respectively. And the classification gain using the combined features is increasing when the gamma value increases. But when gamma be larger than 2^{-5}, the increasing trend is slow. As shown in Fig.5(b), to the classifiers using wavelet feature and contourlet feature individually, the average number of support vectors is the fewest when the gamma value is about 0.5 and 1, respectively. The classifiers with combined features have fewest average number of support vectors when gamma equals to 0.25. After gamma be larger than 2, the average number of support vectors are increasing dramatically. So during the following experiments, considering the relationship between classification gain and computation complexity, the gamma is set to 1 and C value is 1000.



(a)



(b)

Fig. 5. (a) Average classification gain versus gamma (b) Average number of SVs versus gamma. Solid line-wavelet based; Dotted line-contourlet based; Dash-dot line-combined features based

4.3 Experimental Results and Comparisons

Table 1 shows the classification gain on dataset 1 using window size of 32×32. The experimental setup is same to the previous section. Different proportions of samples for training are applied to investigate the generalization capability of the proposed method. From the table, we can see that the classification result by using the combined features does not change too much for the training samples ranging from 12.5% to 75% of the total samples. The best result in [12] on that dataset is given in Table 1 for comparison. From the result we can see that, for the case 50% for training and 50% for testing, the classification gains of individual wavelet and contourlet features are slightly worse than that of the spectral histogram-based method, but the classifiers using combination of these two features can obtain much better results.

Table 1. Average classification gains of the proposed method on dataset 1 with window size of 32×32

Ratio	Wavelet	Contourlet	Combined	Method in [12] (best)
12.5%	33.95	33.62	37.27	N/A
25%	35.82	35.54	38.27	N/A
50%	**37.00**	**36.81**	**38.87**	**>=37**
75%	37.41	36.42	39.13	N/A

In order to analyze the effect of the window size, samples of size 64×64 pixels are also be used for experiment. So the original image is subdivided into 16 nonoverlapping windows, resulting in a total of 640 samples. The classification results are shown in Table 2. It can be seen the result is also very good.

Table 2. Average classification gains of the proposed method on dataset 1 with window size of 64×64

Ratio	Wavelet	Contourlet	Combined
12.5%	33.13	34.89	36.74
25%	35.85	37.09	38.30
50%	38.18	38.62	39.44
75%	38.77	38.94	39.57

The experimental results on the dataset 2 and 3 are given in Table 3 and Table 4, respectively. The original image is with size of 128×128, which is divided into 16 32×32 regions. The best results of the methods in [8] and [12] are also given in the tables. As we can see, in the case of 50% samples for training and 50% for testing, the proposed method can get significantly improvement to those methods.

Table 3. Average classification gains of the proposed method on dataset 2

Ratio	Wavelet	Contourlet	Combined	Method in [8] (best)	Method in [12] (best)
12.5%	8.04	7.31	8.51	N/A	N/A
25%	8.53	8.23	9.14	N/A	N/A
50%	**9.01**	**8.76**	**9.52**	**6.77**	**8.31**
75%	9.28	9.09	9.72	N/A	N/A

To further illustrate our method, we have done a comparison with a method proposed by [5], which on the dataset 4. The original image, where size is 256×256, is subdivided into 64 nonoverlapping samples of size 32×32, resulting in a total of

1792 samples. Eighty percent of samples are used for learning and the rest are used for testing the classifier. The classification performance is evaluated using five different randomly selected learning and testing sets. The results shown in Table 5, from which we can see that the proposed method can get better results than the method in [5].

Table 4. Average classification gains of the proposed method on dataset 3

Ratio	Wavelet	Contourlet	Combined	Method in [8] (best)	Method in [12] (best)
12.5%	7.19	6.62	7.37	N/A	N/A
25%	7.88	7.08	7.90	N/A	N/A
50%	**8.33**	**7.66**	**8.58**	**7.22**	**7.91**
75%	8.78	7.87	8.92	N/A	N/A

Table 5. Average classification gains of the proposed method on dataset 4

Wavelet	Contourlet	Combined	Method in [5] (best)
27.42	**27.50**	**27.88**	**27.36**

5 Conclusions

In this paper, we have proposed a method for texture classification by using combined features from wavelet and contourlet. SVMs with Gaussian kernel are used as classifiers. Experiments are performed on four different texture datasets selected from Brodatz and VisTex database. Experimental results demonstrated the combination of the two feature sets always outperformed each method individually. Comparative results to other methods show that the proposed method can get better results than other methods.

References

1. Gibson, J.J.: The Perception of The Visual World. Riverside Press, Cambridge (1950)
2. Cesmeli, E. and Wang, D. L.: Texture Segmentation using Gaussian-Markov Random Fields and Neural Oscillator Networks. IEEE Trans. Neural Networks 12 (2001) 394–404
3. Keller, J.M., Chen, S., and Crownover, R.M.: Texture Description and Segmentation through Fractal Geometry. Computer Vision and Graphical Image Processing 45 (1989) 150–166
4. Jain, A.K. and Farrokhnia, F.: Unsupervised Texture Segmentation using Gabor Filters. Pattern Recognition 24 (1991) 1167–1186
5. Kim, N.-D. and Udpa, S.: Texture Classification using Rotated Wavelet Filters. IEEE Trans System, Man, and Cybernetics-PART A: Systems and Humans 30 (2000) 847–852

6. Arivazhagan, S. and Ganesan, L.: Texture Classification using Wavelet Transform. Pattern Recognition Letters 24 (2003) 1513–1521
7. Randen, T. and Husøy, J.H.: Multichannel Filtering for Image Texture Segmentation. Optical Engineering 33 (1994) 2617–2625
8. Randen, T. and Husøy, J.H.: Filtering for Texture Classification: A Comparative Study. IEEE Trans Pattern Analysis and Machine Intelligence 21 (1999) 291–310
9. Do, M. N. and Vetterli, M.: Contourlets: A Directional Multiresolution Image Representation. Proc. of IEEE International Conference on Image Processing, Rochester, September 2002.
10. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
11. Cristianini, N. and Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press, Cambridge (2000)
12. Liu, X.W., and Wang, D.L.: Texture Classification using Spectral Histograms. IEEE Trans Image Processing 12 (2003) 661–670

Distance Measures between Attributed Graphs and Second-Order Random Graphs

Francesc Serratosa[1] and Alberto Sanfeliu[2]

[1] Universitat Rovira I Virgili, Dept. d'Enginyeria Informàtica i Matemàtiques, Spain
Francesc.Serratosa@etse.urv.es
http://www.etse.urv.es/~fserrato
[2] Universitat Politècnica de Catalunya, Institut de Robòtica i Informàtica Industrial, Spain
sanfeliu@iri.upc.es

Abstract. The aim of this article is to purpose a distance measure between Attributed Graphs (AGs) and Second-Order Random Graphs (SORGs) for recognition and classification proposes. The basic feature of SORGs is that they include both marginal probability functions and joint probability functions of graph elements (vertices or arcs). This allows a more precise description of both the structural and semantic information contents in a set (or cluster) of AGs and, consequently, an expected improvement in graph matching and object recognition. The distance measure is derived from the probability of instantiating a SORG into an AG.
SORGs are shown to improve the performance of other random graph models such as FORGs and FDGs and also the direct AG-to-AG matching in two experimental recognition tasks.

1 Introduction

Some attempts have been made to try to reduce the computational time of matching the unknown input patterns to the whole set of models from the database. Assuming that the AGs that represent a cluster or class are not completely dissimilar in the database, only one structural model is defined from the AGs that represent the cluster, and thus, only one comparison is needed for each cluster.

One of the most common methodologies are based on keeping the probabilistic information in the structure that represent the cluster of AGs. These models, which are usually called *Random Graphs* (RGs), are described in the most general case through a joint probability space of random variables ranging over graph vertices and arcs. They are the union of the AGs in the cluster, according to some synthesis process, together with its associated probability distribution. In this manner, a structural pattern can be explicitly represented in the form of an AG and an ensemble of such representations can be considered as a set of outcomes of the RG. The most important probabilistic methods are First-Order Random Graphs (FORGs) [4], the Sengupta method [3], Function-Described Graphs (FDGs) [1,6,7] and Second-Order Random Graphs (SORGs), which can be seen as a generalisation of both of them [5].

In the following section, we introduce the formal definitions used throughout the paper. In section 3, we recall the general formulation for estimating the joint probability of the random elements in a RG synthesised from a set of AGs. In section 4, we present the new distance measure between AGs and SORGs derived from the joint probabilities of the random elements. Finally, we present a comparative study between SORGs and FORGs, FDGs and direct AG-to-AG matching. In the last section, we provide some discussion about our distance measure.

2 Formal Definitions of Random-Graph Representation

Definition 1: Attributed Graph (AG). Let Δ_v and Δ_e denote the domains of possible values for attributed vertices and arcs, respectively. These domains are assumed to include a special value Φ that represents a *null value* of a vertex or arc. An AG G over (Δ_v,Δ_e) is defined to be a four-tuple $G=(\Sigma_v,\Sigma_e,\gamma_v,\gamma_e)$, where $\Sigma_v=\{v_k \mid k=1,...,n\}$ is a set of vertices (or nodes), $\Sigma_e=\{e_{ij} \mid i,j\in\{1,...,n\}, i\neq j\}$ is a set of arcs (or edges), and the mappings $\gamma_v:\Sigma_v\rightarrow\Delta_v$ and $\gamma_e:\Sigma_e\rightarrow\Delta_e$ assign attribute values to vertices and arcs, respectively.

Definition 2: Random Graph (RG). Let Ω_v and Ω_e be two sets of random variables with values in Δ_v (random vertices) and in Δ_e (random arcs), respectively. A *random-graph structure R* over (Δ_v,Δ_e) is defined to be a tuple $(\Sigma_v,\Sigma_e,\gamma_v,\gamma_e,P)$, where $\Sigma_v=\{\omega_k \mid k=1,...,n\}$ is a set of vertices, $\Sigma_e=\{\varepsilon_{ij} \mid i,j\in\{1,...,n\}, i\neq j\}$ is a set of arcs, the mapping $\gamma_v:\Sigma_v\rightarrow\Omega_v$ associates each vertex $\omega_k\in\Sigma_v$ with a random variable $\alpha_k=\gamma_v(\omega_k)$ with values in Δ_v, and $\gamma_e:\Sigma_e\rightarrow\Omega_e$ associates each arc $\varepsilon_{ij}\in\Sigma_e$ with a random variable $\beta_k=\gamma_e(\varepsilon_{ij})$ with values in Δ_e. And, finally, P is a joint probability distribution $P(\alpha_1,\ldots,\alpha_n,\beta_1,\ldots,\beta_m)$ of all the random vertices $\{\alpha_i \mid \alpha_i=\gamma_\omega(\omega_i), 1\leq i\leq n\}$ and random arcs $\{\beta_j \mid \beta_j=\gamma_\varepsilon(\varepsilon_{kl}), 1\leq j\leq m\}$.

Definition 3: Probability of a RG instantiation. Given an AG G and a RG R, the joint probability of random vertices and arcs is defined over an instantiation that produces G, and such instantiation is associated with a structural isomorphism $\mu:G'\rightarrow R$, where G' is the extension of G to the order of R. G' represents the same object than G but some vertices or arcs have been added with the null value Φ to be μ bijective. Let G be oriented with respect to R by the structurally coherent isomorphism μ; for each vertex ω_i in R, let $\mathbf{a}_i=\gamma_v(\mu^{-1}(\omega_i))$ be the corresponding attribute value in G', and similarly, for each arc ε_{kl} in R (associated with random variable β_j) let $\mathbf{b}_j=\gamma_e(\mu^{-1}(\varepsilon_{kl}))$ be the corresponding attribute value in G'. Then the *probability of G according to* (or given by) *the orientation* μ, denoted by $P_R(G|\mu)$, is defined as

$$P_R(G|\mu)=\Pr\left(\wedge_{i=1}^{n}(\alpha_i=\mathbf{a}_i)\wedge\wedge_{j=1}^{m}(\beta_j=\mathbf{b}_j)\right)=p(\mathbf{a}_1,\ldots,\mathbf{a}_n,\mathbf{b}_1,\ldots,\mathbf{b}_m) \qquad (1)$$

We define $\mathbf{d}_i$ to represent a vertex or arc attribute value ($\mathbf{a}_i$ or $\mathbf{b}_i$). Thus, if s is the number of vertices and arcs, $s=m+n$, eq. (1) can be rewritten as,

$$P_R(G|\mu) = p(\mathbf{d}_1,\ldots,\mathbf{d}_s) \tag{2}$$

3 Second-Order Random-Graph Representation

If we want to represent the cluster of AGs by a RG, it is impractical to consider the high order probability distribution defined in the RGs $P(\alpha_1,\ldots,\alpha_n,\beta_1,\ldots,\beta_m)$ (definition 2), where all components and their relations in the structural patterns are taken jointly due to time and space costs. For this reason, some other more practical approaches have been presented that propose different approximations [3,4,5,7]. All of them take into account in some manner the incidence relations between attributed vertices and arcs, i.e. assume some sort of dependence of an arc on its connecting vertices. Also, a common ordering (or labelling) scheme is needed that relates vertices and arcs of all the involved AGs, which is obtained through an optimal graph mapping process called synthesis of the random graph representation. We showed in [5] that all the approximations in the literature of the joint probability of an instantiation of the random elements in a RG (eq. 1) can be described in a general form as follows:

$$P_R(G|\mu)=p(\mathbf{a}_1,,,\mathbf{a}_n,\mathbf{b}_1,,,\mathbf{b}_m)=\prod_{i=1}^{n}p_i(\mathbf{a}_i)\prod_{i=1}^{m}p_i(\mathbf{b}_i)\prod_{i=1}^{n-1}\prod_{j=i+1}^{n}r_{ij}(\mathbf{a}_i,\mathbf{a}_j)\prod_{i=1}^{n}\prod_{j=1}^{m}r_{ij}(\mathbf{a}_i,\mathbf{b}_j)\prod_{i=1}^{m-1}\prod_{j=i+1}^{m}r_{ij}(\mathbf{b}_i,\mathbf{b}_j) \tag{3}$$

where p_i are the marginal probabilities of the s random elements γ_i, (vertices or arcs) and r_{ij} are the Peleg compatibility coefficients [2] that take into account both the marginal and 2nd-order joint probabilities of random vertices and arcs.

According to eq. (2), we can generalise the joint probability as,

$$P_R(G|\mu) = p(\mathbf{d}_1,,,\mathbf{d}_s) = \prod_{i=1}^{s} p_i(\mathbf{d}_i)\prod_{i=1}^{s}\prod_{j=i+1}^{s} r_{ij}(\mathbf{d}_i,\mathbf{d}_j) \tag{4}$$

and define the Peleg coefficient,

$$r_{ij}(\mathbf{d}_i,\mathbf{d}_j) = \frac{p_{ij}(\mathbf{d}_i,\mathbf{d}_j)}{p_i(\mathbf{d}_i)p_j(\mathbf{d}_j)} \tag{5}$$

The Peleg coefficient, with a non-negative range, is related to the “degree” of dependence between two random variables. If they are independent, the joint probability, p_{ij}, is defined as the product of the marginal ones, thus, $r_{ij} = 1$ (or a value close to 1 if the probability functions are estimated). If one of the marginal probabilities is null, the joint probability is also null. In this case, the indecisiveness *0/0* is solved as 1, since this do not affect the global joint probability, which is null.

4 Distance Measure between AGs and SORGs

The distance measure presented in this section provides a quantitative value of the match between an AG *G* (data graph) and a SORG *S* (model graph) similar to the one presented in [1]. It is related to the probability of G according to the labelling function $\mu : G \to S$, denoted $P(G|\mu)$ in eq. (4). We may attempt to minimise a *global cost* measure *C* of the morphism μ in the set *H* of allowable configurations, by taking the cost as a monotonic decreasing function of the conditional probability of the data graph given the labelling function, $C = f(P(G|\mu))$. For instance, $C = -\ln(P(G|\mu))$ would be a possible choice. Thus, considering eq. (4),

$$C(G|\mu) = -\ln\left(\prod_{i=1}^{s} p_i(\mathbf{d}_i)\prod_{i=1}^{s-1}\prod_{j=i+1}^{s} r_{ij}(\mathbf{d}_i,\mathbf{d}_j)\right) = -\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) - \sum_{i=1}^{s-1}\sum_{j=i+1}^{s}\ln(r_{ij}(\mathbf{d}_i,\mathbf{d}_j)) \quad (6)$$

and using the definition of the Peleg coefficient (eq. 5) we obtain the following equation in the case that $p_i(\mathbf{d}_i) > 0$ and $p_j(\mathbf{d}_j) > 0$

$$C(G|\mu) = -\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) - \sum_{i=1}^{s-1}\sum_{j=i+1}^{s}\left[\ln(p_{ij}(\mathbf{d}_i,\mathbf{d}_j)) - \ln(p_i(\mathbf{d}_i)) - \ln(p_j(\mathbf{d}_j))\right] \quad (7)$$

Rearranging the second term of the expression we arrive at the equation

$$C(G|\mu) = -\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) + (s-1)\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) - \sum_{i=1}^{s-1}\sum_{j=i+1}^{s}\left[\ln(p_{ij}(\mathbf{d}_i,\mathbf{d}_j))\right] \quad (8)$$

And we obtain the final expression in which the cost of the labelling monotonically depends on the probabilities provided that $\forall i : p_i(\mathbf{d}_i) > 0$.

$$C(G|\mu) = (s-2)\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) - \sum_{i=1}^{s-1}\sum_{j=i+1}^{s}\left[\ln(p_{ij}(\mathbf{d}_i,\mathbf{d}_j))\right] \quad (9)$$

In the case that there is only one random element any joint probability is not defined and *s*=1. Then, the global cost of the matching is $C(G|\mu) = -\ln(p_i(\mathbf{d}_i))$. Moreover, in the case that there is a couple of random elements (*s*=2), the cost depends only on the joint probability, $C(G|\mu) = -\ln(p_{12}(\mathbf{d}_1,\mathbf{d}_2))$, although it has to be considered that $p_1(\mathbf{d}_1) > 0$ and $p_2(\mathbf{d}_2) > 0$. And finally, in the case that there are 3 random elements (*s*=3), the cost is defined as, $C(G|\mu) = \ln(p_1(\mathbf{d}_1)) + \ln(p_2(\mathbf{d}_2)) + \ln(p_3(\mathbf{d}_3)) - \ln(p_{12}(\mathbf{d}_1,\mathbf{d}_2)) - \ln(p_{13}(\mathbf{d}_1,\mathbf{d}_3)) - \ln(p_{23}(\mathbf{d}_2,\mathbf{d}_3))$. In the last case (and all the cases that *s*≥3, the marginal probabilities make the global cost decrease and the joint probabilities make the cost increase.

4.1 Assuming Independence between Random Elements

In the case that the random elements are independent, the joint probability is defined as the product of the marginal ones, $p_{ij}(\mathbf{d}_i,\mathbf{d}_j) = p_i(\mathbf{d}_i)p_j(\mathbf{d}_j)$. Thus, eq. (9) is rewritten as,

$$\begin{aligned} C(G|\mu) &= (s-2)\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) - \sum_{i=1}^{s-1}\sum_{j=i+1}^{s}\left[\ln(p_i(\mathbf{d}_i)p_j(\mathbf{d}_j))\right] = \\ &= (s-2)\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) - \sum_{i=1}^{s-1}\sum_{j=i+1}^{s}\left[\ln(p_i(\mathbf{d}_i)) + \ln(p_j(\mathbf{d}_j))\right] = \\ &= (s-2)\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) - (s-1)\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) \end{aligned} \quad (10)$$

Thus, the final equation is,

$$C(G|\mu) = -\sum_{i=1}^{s}\ln(p_i(\mathbf{d}_i)) \quad (11)$$

Note that this expression could be obtained by considering eq. (4) and (6) with $r_{ij} = 1$, that is, assuming independence between the random elements.

4.2 Approximating the Distance Using Bounded Individual Costs

Using the above distance, only that one graph element had a probability of zero, the global joint probability would be zero and *C* would be infinite. Since this may happen due to the noisy presence of an unexpected element or the absence of a model's element, only that one graph element were not properly mapped, the involved graphs would be wrongly considered to be completely different. We must therefore admit the possibility of both extraneous and missing elements in the data graphs, since the data extracted from the information sources (e.g. images) will usually be noisy, incomplete or uncertain. As a consequence, the matches for which $P(G|\mu) = 0$ should not be discarded since they could be the result of a noisy feature extraction and graph formation. In addition, a model (SORG) should match to a certain degree not only the objects (AGs) in its learning set but also the ones that are "near".

Hence, it is more appropriate for practical purposes to decompose the global cost *C* into the sum of some *bounded* individual costs, one for each of the graph element matches (first-order costs on the marginal probabilities) and one for each relation between a pair of element matches (second-order costs on the joint probabilities)

$$C(G|\mu) = -(s-2)\sum_{i=1}^{s} C_i^1(\mathbf{d}_i) + \sum_{i=1}^{s-1}\sum_{j=i+1}^{s} C_{i,j}^2(\mathbf{d}_i, \mathbf{d}_j) \quad (12)$$

where first-order costs are given by

$$C_i^1(\mathbf{d}_i) = Cost(p_i(\mathbf{d}_i)) \quad (13)$$

and second-order costs are given by

$$C_{i,j}^2(\mathbf{d}_i, \mathbf{d}_j) = Cost(p_{i,j}(\mathbf{d}_i, \mathbf{d}_j)) \quad (14)$$

and the function *Cost(Pr)* yields a bounded normalised cost value between 0 and 1 depending on the negative logarithm of a given probability *Pr* and parameterised by a positive constant $K_{pr} \in [0,1]$, which is a threshold on low probabilities that is introduced to avoid the case ln(0), which would give negative infinity. This is,

$$Cost(\mathrm{Pr}) = \begin{cases} \dfrac{-\ln(\mathrm{Pr})}{-\ln(K_{\mathrm{Pr}})} & \text{if } \mathrm{Pr} \geq K_{\mathrm{Pr}} \\ 1 & \text{otherwise} \end{cases} \tag{15}$$

Once a cost measure C is defined, a distance measure between an AG and a SORG and the optimal labelling μ^* are defined respectively as

$$d = \min_{\mu \in H} \{C(G|\mu)\} \quad \text{and} \quad \mu^* = \arg\min_{\mu \in H} \{C(G|\mu)\} \tag{16}$$

The algorithm we use to calculate d and μ^* is a classical recursive tree search procedure, where the search space is reduced by a *branch and bound* technique (not described here due to lack of space).

5 Results

We carried out two different types of experiments to assess the usefulness of our new representation and to compare it with some other representations presented in the literature. In the first experiments, the AGs were synthetically generated varying some parameters such as the number of vertices or the distance between the AGs in their clusters. In the second experiments, we used a real application in which AGs represent coloured 3D objects. They were extracted and recognised from some 2D images. The first experiments are useful to study our representation from the theoretical point of view and the second ones are useful to apply our methods on noisy, real and complex images.

We present the experiments in the following three sections. In each experiment, we compare SORGs with three other methods: FDGs, FORGs and AG-to-AG matching. First, we show some information of the AGs and the structures obtained in the synthesis process and then we show the run time and ratio of correctness of the classification processes for each method. SORGs, FDGs and FORGs were synthesised using the *dynamic clustering* in which the models are incrementally updated from a sequence of AGs that represent the same cluster or 3D-object [6] (We used the order of presentation of AGs that obtained the best results). In the SORG method, AGs were classified using the distance measure described in this paper. In the FDG method, the AGs were classified applying the *distance measure between AGs and FDGs relaxing second-order constraints* (moderate costs on the antagonisms, existences and occurrences), without the *efficient module*, presented in [7]. FORGs were compared using the methods presented in [4]. Finally, in the direct AG-to-AG matching method, we used the *edit-operations distance* between AGs presented in [8]. The algorithms presented here were implemented in visual C++ and run on a Pentium IV (1.6Ghz).

5.1 Experiments with Randomly Generated AGs

The AGs used in this section were generated by the random graph generator process shown in figure 1 (this graph generator was also used and explained in depth in [6]).

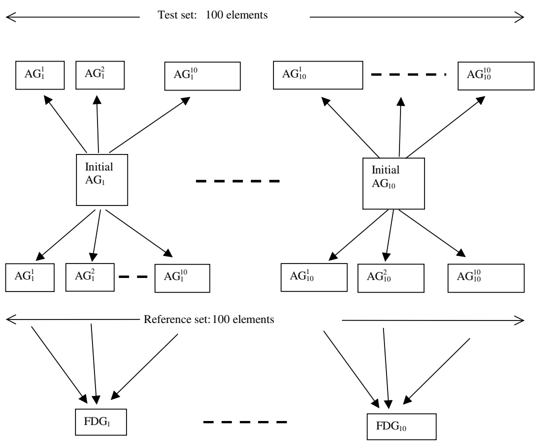


Fig. 1. Random generation of reference and test sets and FDG synthesis.

We first generated 10 initial AGs randomly, one for each model, that had 15 vertices and 5 arcs per vertex. From these AGs, the reference and test sets were derived in the following way. For each initial AG, a reference and a test set of 10 AGs was built by randomly deleting 3 vertices and replacing the attribute of the other vertices by adding gaussian noise with variance *V* to the attribute values. Then, from each set of 10 reference AGs, an FDG was synthesised.

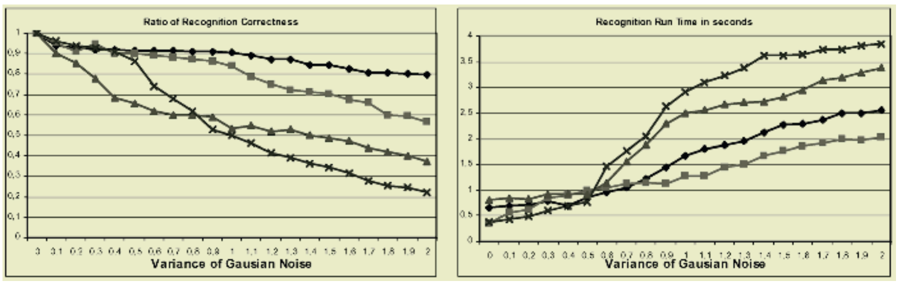


Fig. 2. (a) Ratio of recognition correctness (b) run time spent in the classification. SORG: ◆; FDG: ■; FORG: ▲; AG-AG: ✕

Figure 2 shows in (a) the ratio of recognition correctness and in (b) the time in seconds spent to compute an AG classification in average applying 4 different classification methods: SORGs, FDGs, FORGs and direct AG-AG matching. We have seen that the second-order knowledge kept in the SORGs is higher than in the FDGs and than in the FORGs. We see, through the results, that this knowledge is useful to represent the cluster of AGs and so to increase the recognition ratio. The direct AG-AG matching methods have similar results than SORGs and FDGS only when there is few noise in the test set. When the variance of the noise increases, the AGs in the tests set are very different from the AGs in the reference sets and then the ratio of classification decreases. While considering the run time, we see that the higher differences appear when the variance of the noise is large. FDGs is the fastest method since the antagonisms are useful to prune the search tree (see 10] for more details). Nevertheless, the Peleg coefficients computed in the distance between AGs and SORGs are also useful to prune the search tree. For this reason, SORGs obtain better results than FORGs. Finally, the direct AG to AG matching is the slowest method when the variance is bigger than 0.6. This is due to the fact that the AGs in the test set are very different to those in the reference set and so the *branch and bound* algorithm can scarcely prune the search tree.

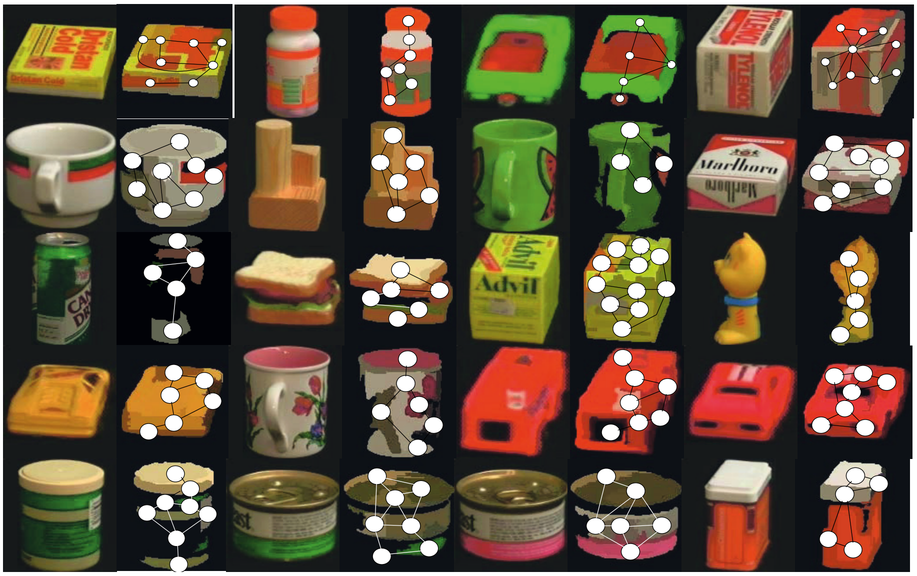


Fig. 3. The 20 selected objects at angle 100 and the segmented images with the AGs.

5.2 Application of Graph Structures to 3D Object Recognition

Finally, we present a real application to recognise coloured objects using 2D images. Images were extracted from the database COIL-100 from Columbia University (www.cs.columbia.edu/CAVE/research/ softlib/coil-100.html). It is composed by 100 isolated objects and for each object there are 72 views (one view each 5 degrees). AGs are obtained by the segmentation process presented in [9]. AG nodes represent regions and their attribute value is their average hue and arcs represent adjacent regions and their attribute value is the distance between average hues. Figure 3 shows the 20 objects at angle 100 and their segmented images with the AGs. These AGs have from 6 to 18 vertices and the average number is 10. The test set was composed by 36 views per object (taken at the angles 0, 10, 20 and so on), whereas the reference set was composed by the 36 remaining views (taken at the angles 5, 15, 25 and so on). We made 6 different experiments in which the number of clusters that represents each 3D-object varied. If the 3D-object was represented by only one cluster, the 36 AGs from the reference set that represent the 3D-object were used to synthesise the SORGs, FORGs or FDGs. If it was represented by 2 clusters, the 18 first and consecutive AGs from the reference set were used to synthesise one of the SORGs, FORGs or FDGs and the other 18 AGs were used to synthesise the other ones. A similar method was used for the other experiments with 3, 4, 6 and 9 clusters per 3D-object.

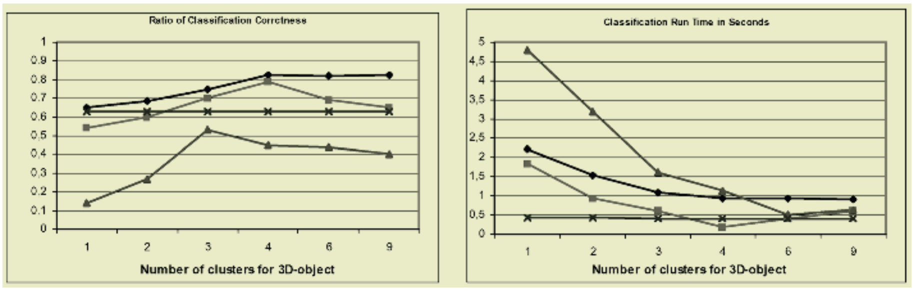


Fig. 4. (a) Ratio of recognition correctness (b) run time spent in the classification. SORG: ◆; FDG: ■; FORG: ▲; AG-AG: ✕

Figure 4.a shows the ratio of correctness of the four classifiers varying the number of clusters per each object. When objects are represented by only 1 or 2 clusters, there are too much spurious regions (produced in the segmentation process) to keep the structural and semantic knowledge of the object. For this reason, different regions or faces (or vertices in the AGs) of different views (that is, AGs) are considered to be the same face (or vertex in the AGs). The best result appears when each object is represented by 3 or 4 clusters, that is, each cluster represents 90 degrees of the 3D-object. When objects are represented by 9 clusters, each cluster represents 40 degree views of the 3D-object and 4 AGs per cluster, there is poor probabilistic knowledge and therefore there is a lack of discrimination between objects.

Figure 4.b shows the average run time spent to compute the classification. When the number of clusters per object decreases, the number of total comparisons also decreases but the time spent to compute the distance increases since the structures that represent the clusters (SORGs, FORGs or FDGs) are bigger.

6 Conclusions and Future Work

SORGs are a general formulation of an approximation of the joint probability of random elements in a RG, that describes a set of AGs, based on 2nd order joint probabilities and marginal ones. FORG and FDG approaches are two specific cases of SORGs. A new distance measure between AGs and SORGs has been presented. It is related to the probability of the AG according to the function that matches the graph elements. We have commented the features of the new distance measure and we have applied to two pattern recognition applications. We show that in both cases the use of the 2nd order probabilities is useful to increase the recognition ratio and decrease the run time.

References

1. R. Alquézar, F. Serratosa, A. Sanfeliu, “Distance between Attributed Graphs and Function-Described Graphs relaxing 2nd order restrictions”. *Proc. SSPR’2000 and SPR’2000,* Barcelona, Spain, Springer LNCS-1876, pp. 277-286, 2000.
2. S. Peleg and A. Rosenfeld, “Determining compatibility coefficients for curve enhancement relaxation processes”, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 8, pp. 548-555, 1978.
3. K. Sengupta and K. Boyer, “Organizing large structural model bases”, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 321-332, 1995.
4. A.K.C. Wong and M. You, “Entropy and distance of random graphs with application to structural pattern recognition”, *IEEE Trans. on PAMI.*, vol. 7, pp. 599-609, 1985.
5. F. Serratosa, R. Alquézar y A. Sanfeliu, “Estimating the Joint Probability Distribution of Random Vertices and Arcs by means of Second-order Random Graphs”, *Proc. Syntactic and Structural Pattern Recognition, SSPR’2002, LNCS 2396*, Windsor, Canada, pp: 252-262, 2002.
6. F. Serratosa, R. Alquézar y A. Sanfeliu, “Synthesis of function-described graphs and clustering of attributed graphs”, *International Journal of Pattern Recognition and Artificial Intelligence*,Vol. 16, No 6, pp.621-655, 2002.
7. F. Serratosa, R. Alquézar y A. Sanfeliu, “Function-described for modeling objects represented by attributed graphs”, *Pattern Recognition*, 36 (3), pp. 781-798, 2003.
8. A. Sanfeliu and K. Fu, “A distance measure between attributed relational graphs for pattern recognition”, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 13, pp. 353-362, 1983.
9. P.F. Felzenswalb and D.P. Huttenlocher, “Image Segmentation Using Local Variation”, *Proc. of the IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition,* pp. 98-104, 1998

On Not Making Dissimilarities Euclidean

Elżbieta Pękalska[1], Robert P.W. Duin[1], Simon Günter[2], and Horst Bunke[2]

[1] ICT Group, Faculty of Electrical Engineering,
Mathematics and Computer Sciences,
Delft University of Technology, The Netherlands
{e.pekalska,r.p.w.duin}@ewi.tudelft.nl

[2] Department of Computer Science, University of Bern, Switzerland
{gunter,bunke}@iam.unibe.ch

Abstract. Non-metric dissimilarity measures may arise in practice e.g. when objects represented by sensory measurements or by structural descriptions are compared. It is an open issue whether such non-metric measures should be corrected in some way to be metric or even Euclidean. The reason for such corrections is the fact that pairwise metric distances are interpreted in metric spaces, while Euclidean distances can be embedded into Euclidean spaces. Hence, traditional learning methods can be used.

The k-nearest neighbor rule is usually applied to dissimilarities. In our earlier study [12, 13], we proposed some alternative approaches to general dissimilarity representations (DRs). They rely either on an embedding to a pseudo-Euclidean space and building classifiers there or on constructing classifiers on the representation directly. In this paper, we investigate ways of correcting DRs to make them more Euclidean (metric) either by adding a proper constant or by some concave transformations. Classification experiments conducted on five dissimilarity data sets indicate that non-metric dissimilarity measures can be more beneficial than their corrected Euclidean or metric counterparts. The discriminating power of the measure itself is more important than its Euclidean (or metric) properties.

1 Introduction

For learning purposes, objects can be described by dissimilarities to some chosen examples. Such representations can be derived from raw (sensor) measurements, e.g. images or spectra [10, 7], feature-based representations, e.g. for objects represented by mixed variables, or they can result from structural descriptions, e.g. when objects are defined by strings or trees [2].

Assume a collection of objects, a representation set $R := \{p_1, p_2, \ldots, p_r\}$ and a dissimilarity measure d, capturing the notion of closeness between two objects. d is required to be nonnegative and to obey the reflexivity condition, $d(x,x) = 0$, yet, it might be non-metric. A dissimilarity representation (DR) of an object x is defined as a vector of dissimilarities between x and the objects of R, i.e. $D(x,R) = [d(x,p_1), d(x,p_2), \ldots, d(x,p_r)]$. Hence, for a set of objects from T, it extends to a dissimilarity matrix $D(T, R)$. The set R ($R \subseteq T$ or $R \cap T = \emptyset$), consisting of representative objects for the domain, should be relatively small.

A direct approach to dissimilarities leads to the k-nearest neighbor (k-NN) method. This rule is applied here to $D(T_t, R)$, so test objects of T_t become members of the class the most frequently occurring among the k nearest neighbors from R. The k-NN rule can learn complex boundaries and generalize well for large representation sets, yet, at high computational costs. In practice, it might also be difficult to get a sufficiently large R to reach a satisfactory accuracy. Moreover, the performance of the k-NN rule may be affected by presence of noisy examples.

Alternative approaches to DRs can be more computationally advantageous than the k-NN method, especially for a small R. The *embedding* approach builds an embedded pseudo-Euclidean configuration such that the dissimilarities are preserved. In the *dissimilarity space* approach, $D(x, R)$ is considered as a data-depending mapping to the so-called dissimilarity space, where each dimension corresponds to a dissimilarity to a particular object from R [12]. Various classifiers can be constructed in both embedded and dissimilarity spaces [11–13].

The k-NN method is often applied to metric distances, where based on metric properties also fast approximating NN rules can be constructed; see e.g. [9]. Our approaches to DRs can handle quite arbitrary measures. Still, an open question refers to possible benefits of correcting a measure to make it metric or even Euclidean [4, 14]. Metric or Euclidean distances can be interpreted in appropriate spaces, which posses many algebraical properties and where an arsenal of discrimination functions exists. Here, we investigate some ways of making a dissimilarity measure 'more' Euclidean (or 'more' metric) and the influence of such corrections on the performance of some classifiers. We will show that the corrected measures do not necessarily guarantee better performances.

2 Interpretations of the Dissimilarity Data

Embedding. Given any symmetric $D(R, R)$, a configuration X can be found such that the distances between the vectors of X reflect the original ones. In general, a Euclidean space is not 'large enough' for such a distance-preserving mapping, but a pseudo-Euclidean space is [5]. It is a $(p+q)$-dimensional nondegenerate indefinite inner product space $\mathcal{E} := \mathcal{R}^{(p,q)}$ such that the inner product $\langle\cdot,\cdot\rangle_{\mathcal{E}}$ is positive definite (pd) on $\mathcal{R}^p$ and negative definite on $\mathcal{R}^q$. Therefore, $\langle \boldsymbol{x}, \boldsymbol{y}\rangle_{\mathcal{E}} = \sum_{i=1}^{q} x_i y_i - \sum_{i=p+1}^{p+q} x_i y_i = \boldsymbol{x}^T \mathcal{J}_{pq} \boldsymbol{y}$,, where $\mathcal{J}_{pq} = \mathrm{diag}\,(I_{p\times p}; -I_{q\times q})$ and I is the identity matrix. Consequently, $d^2_{\mathcal{E}}(\boldsymbol{x}, \boldsymbol{y}) = \langle \boldsymbol{x}-\boldsymbol{y}, \boldsymbol{x}-\boldsymbol{y}\rangle_{\mathcal{E}} = d^2_{\mathcal{R}^p}(\boldsymbol{x}, \boldsymbol{y}) - d^2_{\mathcal{R}^q}(\boldsymbol{x}, \boldsymbol{y})$. Since $\mathcal{E}$ is a linear space, many inner product based properties can be appropriately extended from the Euclidean case. Yet, the interpretations are different [5, 11].

The inner product (Gram) matrix S of the underlying configuration X can be expressed by using the square dissimilarities $D^{*2} = (d^2_{ij})$ as $S = -\frac{1}{2} J D^{*2} J$, where $J = I - \frac{1}{r}\mathbf{1}\mathbf{1}^T$ [5, 13, 11]. So, X is determined by the eigendecomposion of $S = Q\Lambda Q^T = Q|\Lambda|^{1/2}\mathrm{diag}(\mathcal{J}_{p'q'}; 0)\,|\Lambda|^{1/2}Q^T$, where $|\Lambda|$ is a diagonal matrix of first decreasing p' positive eigenvalues, then decreasing magnitudes of q' negative eigenvalues, followed by zeros. Q is a matrix of the corresponding eigenvectors. X is then uncorrelated [5, 13] and represented in $\mathcal{R}^k$, $k = p' + q'$, as $X = Q_k |\Lambda_k|^{1/2}$. Since only some eigenvalues are large (in magnitude), the remaining ones, if close to zero, can be disregarded as non-informative. By their removal, the data are not only de-noised, but the curse of dimensionality is also avoided. So, the reduced representation $X_{red} = Q_m\, |\Lambda_m|^{1/2}$, $m = p + q < k$, is determined by the largest p positive and the smallest q negative eigenvalues. New objects $D(T_t, R)$ are orthogonally projected onto $\mathcal{R}^m$; see [5, 13, 11] for details.

Inner product based classifiers can appropriately be redefined in a pseudo-Euclidean space. A linear classifier $f(\boldsymbol{x}) = \boldsymbol{v}^T \mathcal{J}_{pq} \boldsymbol{x} + v_0$ is e.g. constructed by addressing it as $f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + v_0$, where $\boldsymbol{w} = \mathcal{J}_{pq} \boldsymbol{v}$; see also [5, 13, 11].

Dissimilarity Spaces. In a dissimilarity space, each dimension corresponds to a dissimilarity $D(\cdot, p_i)$. The property that dissimilarities should be small for similar objects (belonging to the same class) and large for distinct objects, gives a possibility for a discrimination. Thereby, $D(\cdot, p_i)$ can be interpreted as an attribute. This reasoning justifies the usage of traditional classifiers, e.g. linear ones, built in dissimilarity spaces. They can outperform the k-NN rule since they become more global in their decisions by making use of a larger training set T, while maintaining a small R. By using weighted combinations of dissimilarities, such classifiers suppress the influence of noisy examples [12, 13].

3 Going More Euclidean or More Metric

The Gram matrix $S = -\frac{1}{2} J D^{*2} J$ is pd iff D is Euclidean [12, 11, 5]. If S has negative eigenvalues, then D is non-Euclidean and a Euclidean configuration X preserving the distances perfectly cannot be constructed. However, D can be corrected to be Euclidean, which makes the corresponding S pd. Some possible approaches to address this issue are [4, 14, 11]:

- *Clipping* - only p positive eigenvalues are considered yielding a p-dimensional configuration $X = Q_p\, \Lambda_p^{1/2}$. Now, after neglecting the negative contributions, the resulting Euclidean representation overestimates the actual dissimilarities.
- *Adding* 2τ - there exists a positive $\tau \geq -\lambda_{\min}$, where $\lambda_{\min}$ is the smallest (negative) eigenvalue of S, such that $D_{corr} = [D^{*2} + 2\,\tau\,(\mathbf{1}\mathbf{1}^T - I)]^{*1/2}$ is Euclidean [6, 13, 11]. This means that the corresponding S_{corr} is pd. In practice, the eigenvectors of S and S_{corr} are identical, but the value τ is added to the eigenvalues, giving rise to the new diagonal eigenvalue matrix $\Lambda_{cor} := \Lambda_k + \tau\, I$. The distortion is significant if τ is large. If reduced representations of a fixed dimensionality are considered, different eigenvectors will be selected (based on significant eigenvalues) for the original and corrected dissimilarities.
- *Adding* κ - there exists a positive $\kappa \geq \lambda_{\max}$, where $\lambda_{\max}$ is the largest eigenvalue of $\begin{bmatrix} O_{n\times n} & 2\,S\,(D^{*2}) \\ -I_{n\times n} & -4\,S\,(D) \end{bmatrix}$, $S(A) := -\frac{1}{2} JAJ$, such that $D_{corr} = D + \kappa\,(\mathbf{1}\mathbf{1}^T - I)$ is Euclidean [6, 13, 11]. The corresponding Gram matrix S_{corr} yields eigenvectors which are different than these of S.

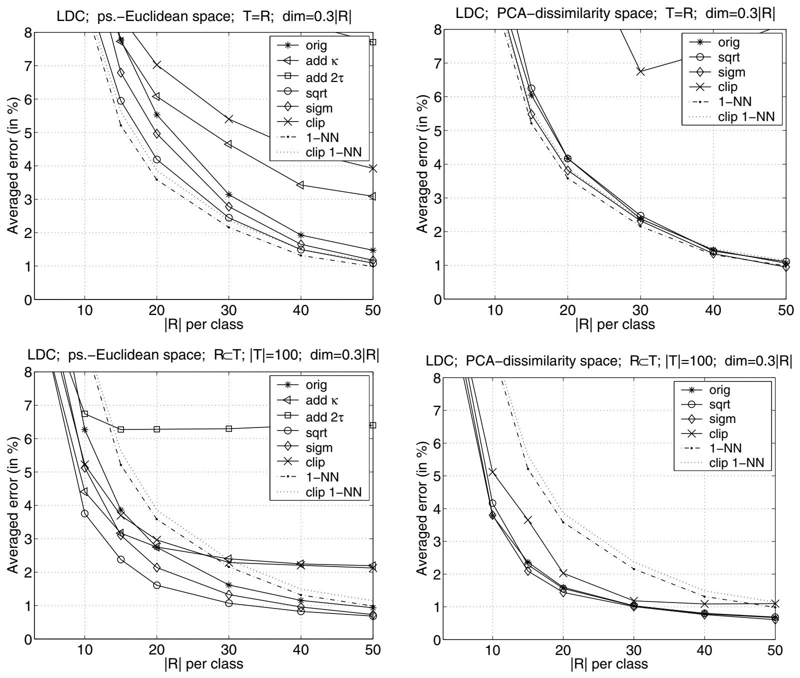


Fig. 1. The performance of the LDC for the Pen-angle dissimilarity data.

– *Power/Sigmoid* - there exists a parameter p such that $D_p = (g(d_{ij};p))$ is Euclidean for a concave function g such as $g(x)=x^p$ with $p<1$ or a sigmoid $g(x)=2/(1+e^{-x/s})-1$ [4, 11]. In practice, p is determined by trial and error.

These approaches transform D such that a Euclidean configuration X can be found. It is, however, still possible that the corrections applied are less than required for Euclideaness. In such cases, the measure is simply made 'more' Euclidean (hence, also 'more' metric), since the influence of negative eigenvalues will become smaller after applying the above transformations.

4 Experiments

Five dissimilarity data sets are used in our study. The first two refer to DRs built on the contours of pen-based handwritten digits [1]. All digits are represented by strings of vectors between the contour points for which an edit distance with a fixed insertion and deletion costs and with some substitution cost is computed. The substitution costs such as an angle and a Euclidean distance between vectors lead to two different DRs, denoted as Pen-dist and Pen-angle, respectively; see also [2]. Here, only a part of the data of 3488 examples, is considered. The values are also scaled by some constant to bound the dissimilarities. The digits are unevenly represented; the class cardinalities vary between 334 and 363.

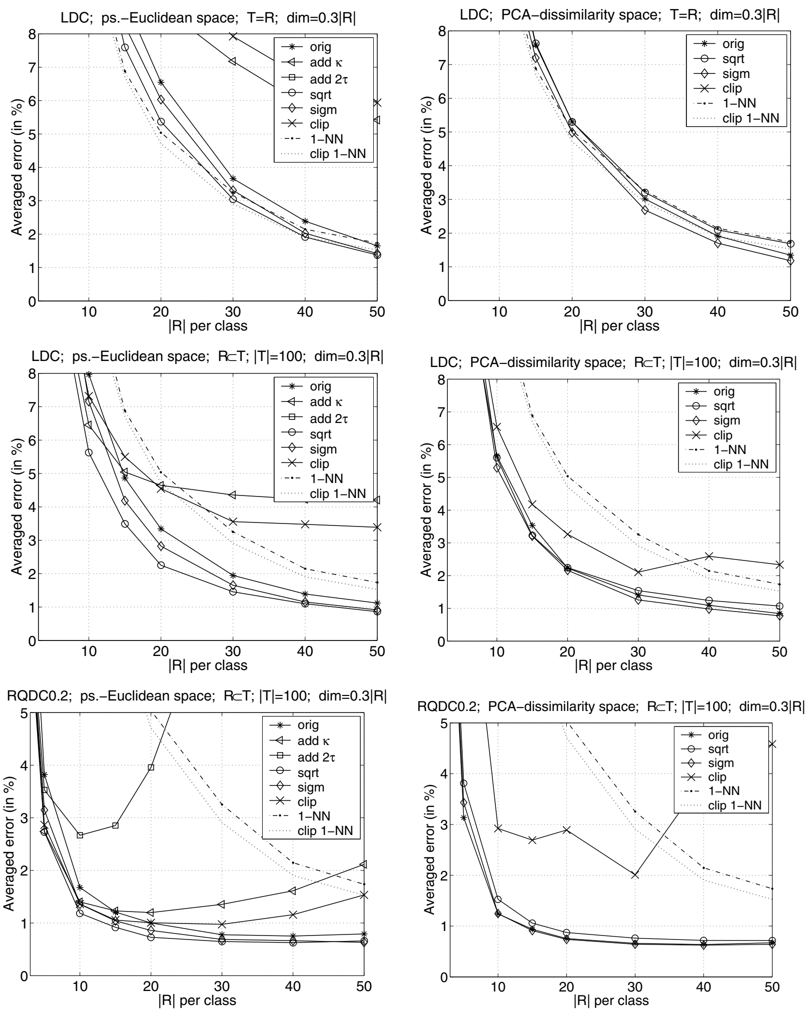


Fig. 2. The performances of the LDC and RQDC for the Pen-dist dissimilarity data.

Another dissimilarity data set, consisting of 2000 examples evenly distributed in ten classes, represents the NIST digits [15]. Here, the asymmetric similarity measure, based on deformable template matching, as defined in [8], is used. Let $S = (s_{ij})$ denote the similarities. The symmetric dissimilarities $D = (d_{ij})$ are derived as $d_{ij} = (s_{ii} + s_{jj} - s_{ij} - s_{ji})^{1/2}$ for $i \neq j$ and $d_{ii} = 0$.

The last two DRs are derived for randomly generated polygons. They consist of convex quadrilaterals and general heptagons. The polygons are first scaled and then the Hausdorff and modified Hausdorff distances [10] between their corners are computed. The two classes are equally represented by 2000 objects.

If a dissimilarity d is Euclidean, then for a symmetric $D = (d_{ij})$, all eigenvalues λ_i of the corresponding Gram matrix S are non-negative. Hence, the magnitudes

Table 1. Non-Euclidean and non-metric aspects of some DRs. The ranges of r_{mm}, r_{neg} and c indicate the smallest and largest values found for $D(R,R)$, where $|R|$ varies between $30-500$ or $10-200$ for the digit and polygon data, respectively. As a reference, the last two columns show the average and maximum dissimilarity for the complete data.

DR	r_{mm} (in %)	r_{neg} (in %)	c	avr. dissim.	max dissim.
Pen-angle	$[10.6, 12.2]$	$[\ 9.4, 24.1]$	$[0.0, 0.3]$	7.1	20.0
Pen-dist	$[13.8, 14.3]$	$[14.2, 27.8]$	$[0.3, 1.0]$	4.0	12.5
NIST-matching	$[27.5, 35.5]$	$[10.6, 35.5]$	$[0.1, 0.5]$	0.6	1.0
Polygon-hausd	$[13.0, 25.5]$	$[\ 5.4, 31.6]$	0	1.2	3.1
Polygon-mhausd	$[\ 5.0, 13.0]$	$[\ 1.8, 24.6]$	$[0.0, 0.1]$	0.7	1.6

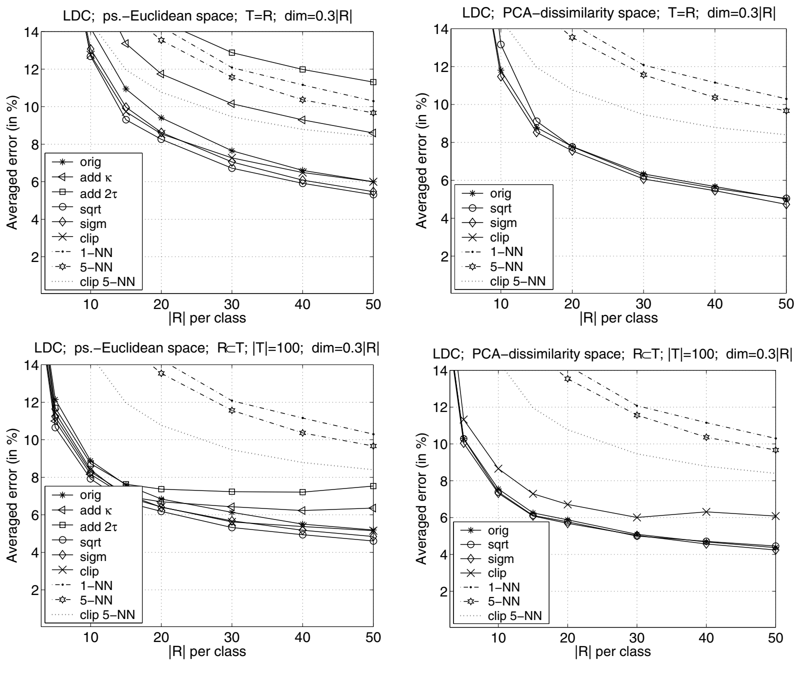


Fig. 3. The LDC performance for the NIST-matching dissimilarity data.

of negative eigenvalues manifest the deviation from Euclideaness. An indication of such a deviation is given by $r_{mm} := |\lambda_{min}|/\lambda_{max}$, i.e. the ratio of the smallest negative eigenvalue to the largest positive one. The overall contribution of negative eigenvalues can be estimated by $r_{neg} := \sum_{\lambda_i<0} |\lambda_i| / \sum_{j=1}^{r} |\lambda_j|$. Any symmetric D can also be made metric by adding a suitable value c to all off-diagonal elements of D. Such a constant can be found as $c = \max_{p,q,t} |d_{pq} + d_{pt} - d_{qt}|$. A smaller value making D metric was determined by us in a binary search. Table 1 provides suitable information on the Euclidean and metric aspects of the measures considered:

– Pen-angle is moderately non-Euclidean and nearly metric.
– Pen-dist is both moderately non-Euclidean and non-metric.
– NIST-matching is highly non-Euclidean and highly non-metric.
– Polygon-hausd is highly non-Euclidean, yet metric.
– Polygon-mhausd is moderately non-Euclidean and slightly non-metric.

The experiments are repeated 50 times for representations sets of various sizes and the results are averaged. $|R|$ varies from 3 to 50 examples per class (ten classes) for the digit DRs and from 5 to 100 examples per class (two classes) for the polygon DRs. For each $|R|$, two cases for the training set T are considered: $T = R$ or T consists of 100/200 objects per class for the digit/polygon DRs, respectively. In the latter case, the ratio of $|T|/|R|$ becomes smaller with a growing $|R|$. The test sets consist of 2488/1000/3600 examples for the pen-digit/NIST/polygon data, correspondingly. For each DR, the k-NN rule is considered, as well as the linear discriminant built in both embedded and dissimilarity spaces. The embedding is derived from $D(R, R)$, but additional objects $T\backslash R$, if available, are projected there and used for constructing classifiers. To denoise the data and avoid the curse of dimensionality, the dimensionality of the embedded space was fixed to $0.3|R|$, so the dimensions corresponding to small eigenvalues (in magnitude) are neglected. Also the principal component analysis was applied in the dissimilarity space $D(\cdot, R)$ to reduce the dimensionality to $0.3|R|$. In both cases, although the dimensionalities are reduced, the spaces are still defined by all the objects of R.

Adding a constant to the dissimilarities or applying a concave transformation preserves their order, hence it does not influence the k-NN rule. However, by clipping (neglecting all negative eigenvalues in the embedding), the re-computed Euclidean distances differ non-monotonically from the original ones, hence the k-NN rule behaves differently. Also both embedded and dissimilarity spaces change, so a linear classifier will change as well. (Adding a constant is not worth doing in dissimilarity spaces, since a constant shift is applied to all D_{ij}, but the self-dissimilarity $D_{ii} = 0$. This is expected to worsen a classifier performance). In our experiments, we study the influence of such corrections on the given measures for various R. For this purpose, proper κ and τ guaranteeing Euclideaness are chosen. Two concave transformations are considered: the square root (which makes the measures close to Euclidean, yet still not Euclidean) and the sigmoid with the slope $s := avr(D(R, R))$. Such measures are non-Euclidean, but less than the original ones as judged by magnitudes of negative eigenvalues in the embeddings.

The results of our experiments compare the averaged performance of the linear discriminant (LDC) and 1-NN rule and the best k-NN rule. They are presented in Fig. 1-5. The standard deviations (for all the data) reach on average 0.3% and maximally $0.8 - 1.4\%$ for very small R. Due to lack of space, the performance of the RQDC02 (regularized quadratic classifier with a relative regularization of 0.2) is shown in Fig. 2 for the Pen-dist data only to indicate that such a classifier can reach even better accuracy than the LDC. The notation in figures refers to:

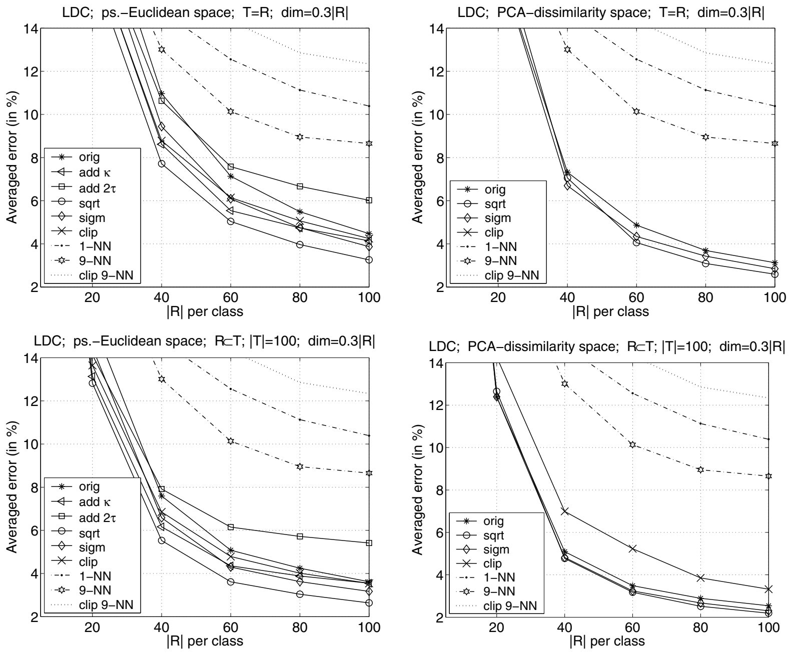


Fig. 4. The LDC performance for the Polygon-hausd dissimilarity data.

– *orig* - original dissimilarities; no transformation applied.
– *add* $\kappa/2\tau$ - a constant added to the dissimilarities; makes $D(R,R)$ Euclidean.
– *sqrt/sigm* - a square root/sigmoid transformation of the dissimilarities; makes $D(R,R)$ 'more' Euclidean.
– *clip* - only positive eigenvalues are used; a new Euclidean D_{eu} is derived from D.

The following general conclusions can be made by analyzing our results:

1. The correction by adding 2τ yields worse results than by adding κ (the former results are missing on some plots since they are out of the given scales).
2. The LDC and the RQDC in (corrected or not) dissimilarity spaces perform similarly or better than in pseudo-Euclidean spaces (compare right vs. left columns in all the figures).
3. For larger T and smaller R, the LDC/RQDC in both embedded and dissimilarity spaces (original or transformed by a square root or a sigmoid function) significantly outperform the k-NN and clip k-NN rules (bottom rows in all the figures). For $T = R$, this phenomenon is much less pronounced; the k-NN might even become somewhat better as observed for the Pen-angle data, Fig. 1.
4. Concave transformations of dissimilarities have a minor effect on the LDC/RQDC constructed in dissimilarity spaces. On the contrary, 'clipping' can deteriorate their performance.

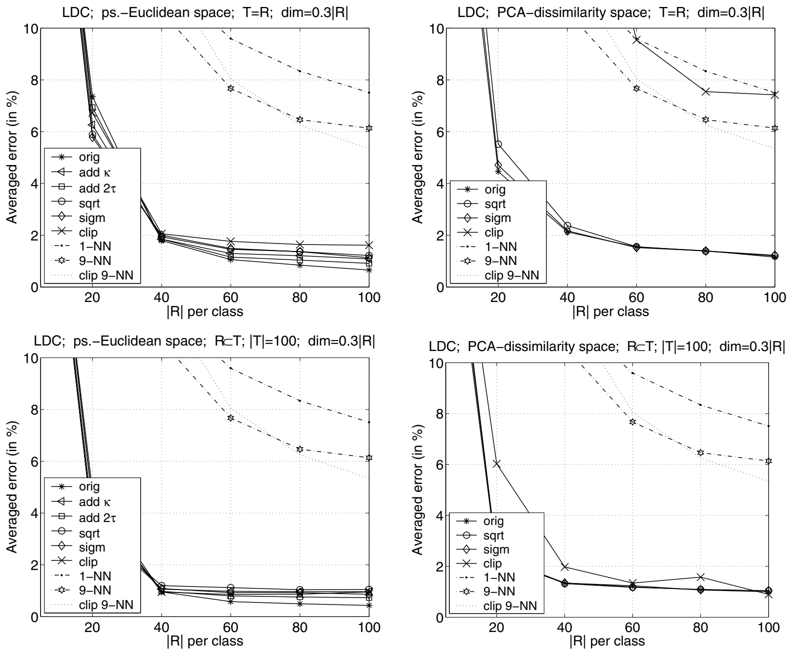


Fig. 5. The LDC performance for the Polygon-mhausd dissimilarity data.

5. The LDC/RQDC built in pseudo-Euclidean spaces derived from concave transformations of the dissimilarities may perform better than for the original dissimilarities or than the LDC/RQDC built in Euclidean spaces obtained from the corrections by clipping or by adding a constant. Still, the results reached by the LDC/RQDC in dissimilarity spaces are comparable or better.

5 Conclusions

If the k-NN is far from optimal for small representation sets, it can be significantly outperformed by linear (quadratic) classifiers built in both embedded or dissimilarity spaces. Concave transformations of dissimilarities are somewhat beneficial for classifiers in the embedded spaces, however, they may have no essential effect in dissimilarity spaces. None of the transformations considered here allows for reaching a considerably better performance than the results in original dissimilarity spaces. However, the transformations may influence the error and reject tradeoff [3]. We conclude that the potential advantages of imposed Euclideaness are doubtful. It is simply more important that the measure itself describes compact classes. This can be influenced by concave transformations which aim at diminishing the relative effect of large dissimilarities and not by making them really Euclidean or metric.

Acknowledgments

This work is supported by the Dutch Organization for Scientific Research (NWO). The authors thank prof. A. Jain and dr D. Zongker for providing the NIST dissimilarity data.

References

1. C.L. Blake and C.J. Merz. UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences, 1998.
2. H. Bunke, S. Günter, and X. Jiang. Towards bridging the gap between statistical and structural pattern recognition: Two new concepts in graph matching. In *Conf. on Advances in Pattern Recognition; LNCS 2013*, pages 1–11, 2001.
3. C.K. Chow. On optimum recognition error and reject tradeoff. *IEEE Trans. on Information Theory*, IT-16(1):41–46, 1970.
4. P. Courrieu. Straight monotonic embedding of data sets in Euclidean spaces. *Neural Networks*, 15:1185–1196, 2002.
5. L. Goldfarb. A new approach to pattern recognition. In *Progress in Pattern Recognition*, volume 2, pages 241–402. Elsevier Science Publishers B.V., 1985.
6. J.C. Gower. Metric and Euclidean Properties of Dissimilarity Coefficients. *Journal of Classification*, 3:5–48, 1986.
7. D.W. Jacobs, D. Weinshall, and Y. Gdalyahu. Classification with Non-Metric Distances: Image Retrieval and Class Representation. *IEEE Trans. on PAMI*, 22(6):583–600, 2000.
8. A.K. Jain and D. Zongker. Representation and recognition of handwritten digits using deformable templates. *IEEE Trans. on PAMI*, 19(12):1386–1391, 1997.
9. F. Moreno-Seco, L. Micó, and J. Oncina. A modification of the LAESA algorithm for approximated k-nn clasification. *Pattern Recogn. Letters*, 24(1-3):47–53, 2003.
10. Dubuisson M. P. and Jain A. K. Modified Hausdorff distance for object matching. In *12th Int. Conf. on Pattern Recognition*, volume 1, pages 566–568, 1994.
11. E. Pękalska. *working title: Dissimilarity-based pattern recognition.* PhD thesis, Delft University of Technology, The Netherlands, to appear, 2004.
12. E. Pękalska and R.P.W. Duin. Dissimilarity representations allow for building good classifiers. *Pattern Recogn. Letters*, 23(8):943–956, 2002.
13. E. Pękalska, P. Paclík, and R.P.W. Duin. A Generalized Kernel Approach to Dissimilarity Based Classification. *J. of Mach. Learn. Research*, 2:175–211, 2001.
14. V. Roth, J. Laub, J.M. Buhmann, and K.-R. Müller. Going metric: Denoising pairwise data. In *Advances in NIPS 15*, pages 841–856. MIT Press, 2003.
15. C.L. Wilson and M.D. Garris. Handprinted character database 3. Technical report, National Institute of Standards and Technology, February 1992.

Content Structure Discovery in Educational Videos Using Shared Structures in the Hierarchical Hidden Markov Models

Dinh Q. Phung[1], Hung H. Bui[2], and Svetha Venkatesh[1]

[1] Department of Computing, Curtin University of Technology,
GPO Box U 1987, Perth, Western Australia
{phungquo,svetha}@computing.edu.au
[2] Artificial Intelligence Center, SRI International
333 Ravenswood Ave Menlo Park, CA 94025, USA
bui@ai.sri.com

Abstract. In this paper, we present an application of the hierarchical HMM for structure discovery in educational videos. The HHMM has recently been extended to accommodate the concept of shared structure, ie: a state might multiply inherit from more than one parents. Utilising the expressiveness of this model, we concentrate on a specific class of video – educational videos – in which the hierarchy of semantic units is simpler and clearly defined in terms of topics and its subunits. We model the hierarchy of topical structures by an HHMM and demonstrate the usefulness of the model in detecting topic transitions.

1 Introduction

The discovery of structure in video data is an important problem. Solution to this problem will form the core of multimedia indexing and browsing systems. The discovery of structure is important as it enables the partitioning of the information into meaningful sub-units and to build a hierarchy of such units in increasing levels of detail. Such hierarchies are naturally used in other media, for example, the table of contents in a book. In the case of video, construction of such a hierarchy is equally meaningful and will allow users to browse the media using a table-of-contents style. The difficult question includes not only the construction of the hierarchy, but also the understanding of the sub-units used in the hierarchy.

In this paper, we concentrate on a specific class of video, the educational video, in which the hierarchy of units is simpler and clearly defined in terms of topics and subtopics. We propose the use of the HHMM to segment this class of video. We first modify the parameter estimation to allow multiple inheritance in hierarchic structures. This is because a video has shared sub-structures and the model needs to accommodate this fundamental aspect of topic organisation. For example, all topics generally start with some introduction shots. The novelty of this work is in the content structure discovery of educational videos where the shared 'concepts' are utilised and incorporated into the model. This is important because without the ability to model shared structures, the shared units will have to be repeated, increasing the state space and thus make the process computationally inefficient. This is particularly relevant when dealing with very long observation sequences such as a full video.

2 Related Background

Discovering structure of videos has been a rapidly growing area, in particular as a subfield of multimedia content management. In these systems, the central task is effectively building units of indexation, possibly at different levels of abstractions, to simplify the process of retrieving and browsing, and also to enrich the viewing experience from the end-users. There have been many systems proposed for specific video genres. Partition and classification of broadcast videos into meaningful sections have attracted significant attention [1–8]. In [7], Liu *et al.* segment news reports from other categories based on both audio and visual information. Low-level features are combined with the concept of shot syntax in [2] to identify and label different narrative structures such as anchor shots, voice-over segments and interview sections found in news programs. Unsupervised groupings of news stories according topical content with only audio information was studied in [8]. Research into the domain of lecture videos has also been found in [9]. In their work, visual events are detected from the visual stream and then incorporated with audio information in a probabilistic framework to detect topic transitions. The domain of entertainment film has also been targeted lately [10–13]. In [10], for example, Adams *et al.* formulate an algorithmic solution for the computation of movie *tempo*, a high-level construct, and later utilise this function to segment a movie into story units. Wang *et al.* [12] attempt to detect *scenes* in film using the similarity in visual information and further improve the results with guidance from cinematic grammar.

The HHMM is a powerful stochastic model, first introduced in [14], in which the HHMM is viewed as a form of probabilistic context free grammar (PCFG), and the inference algorithm and parameter learning procedures are constructed based on the inside-outside algorithm. In [15], the HHMM is converted to a DBN, and applies general DBN inference to the model to achieve complexity linear in time T, but exponential in the depth D of the model. The same analysis applied is [16], ie: the HHMM is 'flatened' into regular HMM with a very large state space for inference purpose. Their work [17, 16] aims to detect structures of soccer videos in an unsupervised manner. The model selection is first carried out using the MCMC to determine the structure parameters for the model, followed by a feature selection procedure. Finally, the HHMM is used to detect two semantic concepts, namely *play* and *break* in soccer videos. As there is little hierarchy at this level, the power of the hierarchic probabilistic model is not used. The HHMM is also applied in other domains other than multimedia such as in hand-written recognition [14], robot navigation [18], behaviour recognition [19] and information retrieval [20].

3 Model Definition and EM for the HHMM

The discrete HHMM and its extension to accommodate shared structured has been addressed in our previous work [21]. Here we refocus our attention to elucidate the idea of the shared structures and briefly discuss the EM algorithm for parameter estimation. We then discuss the case when the emission probability is modeled as mixture of Gaussian in the context of the hierarchical HMM.

A HHMM is formally defined by a topological structure ζ and a set of parameter θ attached to the topology. The general form of DBN representation is shown in Fig. 1(a). The depth D and the number of states available at each level Q^d, for $d = 1, .., D$, are specified by ζ. Level 1 is the root level and is always fixed to have only a single state. Furthermore, the topological structure reveals the 'parent-children' relationship of states between two consecutive levels[1]. A state p at level d is assigned to a set of children, $\mathrm{ch}(p)$, at level $d+1$. A state i at level $d+1$ therefore *might multiply inherit* from more than one parents at level d. For example, the HHMM defined in Fig. 2 has a depth $D = 3$ with $1, 3, 4$ are the number of states respectively at level $d = 1, 2, 3$. The set of children for state 3 at level 2 is $\{2, 3, 4\}$ (at lower level 3). The set of parents for state 2 at level 3 is $\{1, 2, 3\}$, which, in this case, is 'shared' by all states at level 2.

Given such a topological structure ζ, the parameter θ of the HHMM is specified in the following way. For each level $d \in \{1..D-1\}$, $p \in Q^d$, $i, j \in \mathrm{ch}(p)$, where $\mathrm{ch}(p)$ denote the children set of p:

- $\pi_i^{d,p} \triangleq \Pr(q_t^{d+1} = i \mid \cdot q_t^d = p)$: is the initial probability of the child i given the parent is p at level d.
- $\mathrm{A}_{i,j}^{d,p} \triangleq \Pr(q_{t+1}^{d+1} = j \mid e_t^d = 0, q_t^{d+1}\cdot = i, q_t^d = p)$: is the transition probability from child i to child j given that both are children of p.
- $\mathrm{A}_{i,\mathsf{end}}^{d,p} \triangleq \Pr(e_t^d = 1 \mid q_t^d = p, q_t^{d+1}\cdot = i)$: is the probability that state p terminates at level d given its current child is i.

where the dot in front of q_t^d represents the event $q_{t-1}^d\cdot{=}1$ (ie: q_t^d is started at t), and the dot after q_t^d represents the event $q_t^d\cdot{=}1$ (ie: q_t^d is ended at t). The constraints of stochastic processes requires that $\sum_i \pi_i^{d,p} = 1$, $\sum_j \mathrm{A}_{i,j}^{d,p} = 1$, and $\mathrm{A}_{i,\mathsf{end}}^{d,p} \leq 1$. Finally, at the lowest level D, an observation probability matrix B is specified in the discrete observation case, or a set of $\{\mu_{im}, \Sigma_{im}\}$ are given when the observation values are continous and modeled as a mixture of Gaussians.

Given an observed data set $\mathcal{O}$ and some initial parameters, the EM algorithm iteratively re-estimates a new parameter $\hat{\theta}$, hill climbing in the parameter space which is guaranteed to converge to a local maxima. As shown in [21], doing EM parameter re-estimation reduces to first calculating the expected sufficient statistics (ESS) $\bar{\tau} = \mathrm{E}_{\mathcal{V}\setminus\mathcal{O}}\, \tau$, and then set the re-estimated parameter $\hat{\theta}$ to the normalized value of $\bar{\tau}$. The ESS for parameter $\left\{\mathrm{A}_{i,j}^{d,p}\right\}$, for example, is calculated as:

$$\bar{\tau}(\mathrm{A})_{i,j}^{d,p} = \underset{\mathcal{V}\setminus\mathcal{O}}{\mathrm{E}}\, \tau(\mathrm{A})_{i,j}^{d,p} = \sum_{t=1}^{T-1} \xi_t^{d,p}(i,j) \Big/ \Pr(\mathcal{O}) \qquad (1)$$

where the auxiliary variable $\xi_t^{d,p}(i,j)$ is defined as the probability $\Pr(q_{t+1}^{d+1} = j, q_t^{d+1} = i, q_{t+1}^d = p, e_t^{d:d+1} = 01, \mathcal{O})$. Readers are referred to [21] for further details on computation of the auxiliary variables and other expected sufficient statistics. In the rest of this section we will discuss the case when the emission probability is modeled as a mixture of Gaussians.

[1] Note that the original HHMM[14] assumes that a state has a only a single parent and therefore the topology reduces strictly to a tree.

In general, modeling the observation probability as a mixture of Gaussians for the hierarchical HMM is similar to the regular HMM. The DBN structure at level D is modified as in Fig 1(b), where a mixture variable z_t is added. For simplicity, thereafter in this section we will drop the index D. Let M be the number of mixtures and N be the num-

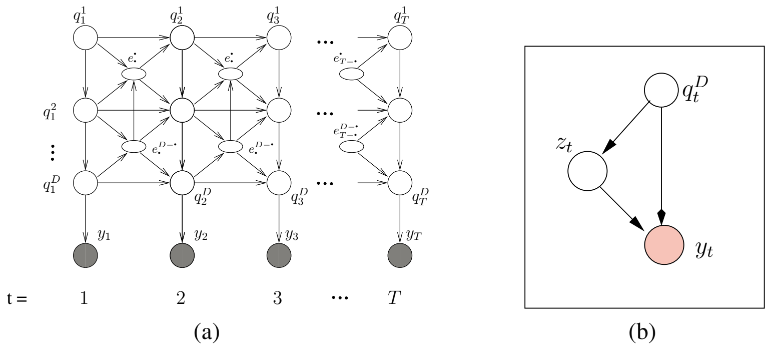


Fig. 1. (a) DBN representation for the discrete HHMM; (b) Mixture component z_t at level D.

ber of states at level D. The observation matrix B in the discrete case is replaced by the mixing weight matrix $\{\varepsilon_{im}\}$ and a set of means and covariance matrices $\{\mu_{mi}, \Sigma_{mi}\}$ for $i = 1, \ldots, N$ and $m = 1, \ldots, M$. Given observed data $\mathcal{O}$, expressing the expected complete log-likelihood and discarding terms irrelevant to z_t and y_t we have:

$$\langle \ell(\theta; \mathcal{O}) \rangle = \sum_{\substack{1 \le i \le N \\ 1 \le m \le M}} \left[\sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t, q_t} \right\rangle \log \mathcal{N}(y_t, \mu_{mi}, \Sigma_{mi}) + \sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t, q_t} \right\rangle \log \varepsilon_{mi} \right]$$

where $\mathrm{I}_{m,i}^{z_t,q_t}$ is the identity function and $\triangleq 1$ if $\{z_t = m\} \cup \{q_t = i\}$; $\triangleq 0$ otherwise; and its expected value is calculated as:

$$\begin{aligned}
\left\langle \mathrm{I}_{m,i}^{z_t,q_t} \right\rangle &= \Pr(z_t{=}m, q_t{=}i \mid \mathcal{O}) = \Pr(z_t{=}m \mid q_t{=}i, y_t) \Pr(q_t{=}i \mid \mathcal{O}) \\
&= \frac{\Pr(y_t \mid z_t{=}m, q_t{=}i) \Pr(z_t{=}m \mid q_t{=}i)}{\Pr(y_t \mid q_t{=}i)} \times \frac{\Pr(q_t{=}i, \mathcal{O})}{\Pr(\mathcal{O})} \\
&= \frac{\varepsilon_{mi} \mathcal{N}(y_t, \mu_{mi}, \Sigma_{mi})}{\sum_{m=1}^{M} \varepsilon_{mi} \mathcal{N}(y_t, \mu_{mi}, \Sigma_{mi})} \times \frac{\gamma_t^D(i)}{\Pr(\mathcal{O})}
\end{aligned}$$

where[2] the auxiliary variable $\gamma_t^D(i)$ is defined as the probability $\Pr(q_t^D{=}i, \mathcal{O})$ and can be computed directly from horizontal transition probability $\xi_t^{d,p}(i,j)$ and vertical transition probability $\chi_t^{d,p}(i)$ (*see* [21]). Finally, maximising the expected complete log-likelihood $\langle \ell(\theta; \mathcal{O}) \rangle$ with respect ε_{im} and μ_{mi}, Σ_{mi} respectively. Introducing the Lagrange multipliers for ε_{im}; and setting derivatives to zero for the case of μ_{mi}, Σ_{mi}. The set of re-estimated parameters is given as:

[2] We put back hierarchic index D for clarity here.

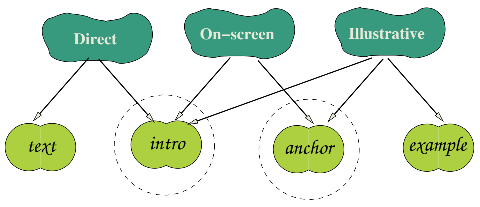
$$\hat{\varepsilon}_{mi} = \frac{\sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t,q_t} \right\rangle}{\sum_{m=1}^{M} \sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t,q_t} \right\rangle}, \qquad \hat{\mu}_{mi} = \frac{\sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t,q_t} \right\rangle y_t}{\sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t,q_t} \right\rangle}$$

$$\hat{\Sigma}_{mi} = \frac{\sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t,q_t} \right\rangle (y_t - \mu_{mi})(y_t - \mu_{mi})^{\mathsf{T}}}{\sum_{t=1}^{T} \left\langle \mathrm{I}_{m,i}^{z_t,q_t} \right\rangle}$$

When multiple observation sequences are given, the set of above equations can be adjusted by simply adding a summation over the number of sequences. This corresponds to ‘counting’ over all sequences.

4 Elucidating Structures in Educational Videos

An intrinsic functionality of educational videos[3] is to ‘teach’ [22], and therefore structuralizing the content and building meaningful indices are important to improve the learning experience. Materials delivered in an educational video might vary widely to suit different purposes; however, when restricted to instructional and safety videos, the content organization is relatively simple. In this paper, we are interested in this particular type of video and observe that linear presentation is generally chosen to present the content. Subjects are arranged into a sequence of topics started with a few introduction shots. Literature in this field [22] offers further insight into how a topic is constructed. Generally, there are three presentational styles: (1) *direct* instruction, (2) *on-screen* instruction, and (3) *illustrative* instruction (*see* Fig. 2). In *direct* instruction, the video-maker choose to present a topic by means of text captions and voice over. In *on-screen* instruction, s/he decides to directly appear on the camera to talk directly to the viewers. Lastly in *illustrative* instruction, illustrative examples are the major mode of presentation to convey the subject with possible appearance of the anchors. These presentational styles shares certain similar semantic concepts at the lower levels such as introduction shot, or direct appearance of anchor(s) (Fig. 2).



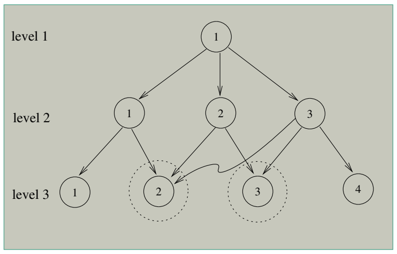


Fig. 2. Structure of topic generating process with assumed hidden ‘styles’; and its mapping to a topology for the HHMM. Shared structures are identified with extra dotted circle.

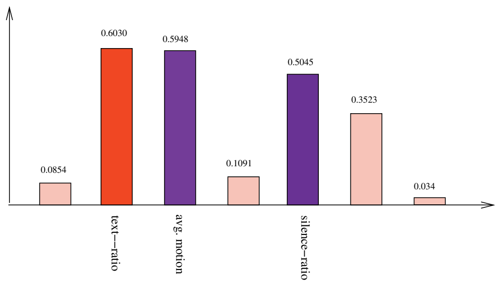
[3] The class of educational videos discussed in this work is of professional productions, excluding hand-held recorded videos such that lecture videos recorded in the classroom.

5 Experimental Results

Our aim is to apply the HHMM model, taking advantage of the shared units, to segment an educational video into high-levels of abstraction – ie: detection of topic transitions in this case. We construct a 3-level HHMM as follows. The root level represents the entire video, followed by three states at the next level, each of which corresponds to one topic presentational style. Our assumption here is that the topic content organisation strictly follows the styles outlined in Sec. 4. The production level includes four states, corresponding to four semantic concepts at the shot level: (1) the introduction, (2) instruction delivered by mean of captioned texts, (3) instruction delivered directly by the presenter and (4) illustrative example (Fig.2). Given a set of training N videos, we extract features from each video and use them as input observation sequences to the EM parameter learning algorithm to estimate a new model parameter. This new parameter set will be used in the second phase to segment a video based on results from the generalised Viterbi decoding algorithm. The data set includes eight instructional and safety videos, whose topics span a variety of subjects such as how to exercise safety at home, in office, or at workplace. Shot indices are assumed to be available, which is first detected by a commercial software and errors are manually corrected. In the training phase, each video yields an observation sequence with each shot-based feature vector $\mathbf{o}$ is a column vector of seven elements $\mathbf{o} = [o_1, o_2, o_3, o_4, o_5, o_6, o_7]^t$. From the visual stream, we extract three features including the face-content-ratio, text-content-ratio and average motion based on camera pan and tilt (o_1, o_2, o_3). The other four features namely music-ratio, speech-ratio, silence-ratio and non-literal sound ratio (o_4, o_5, o_6, o_7) are from the audio track. Feature music-ratio, for example, is calculated as the ratio of number of clips classified as music to the total number of audio clips in the shot. Readers are referred to [23] for further details on the computation of these features.

At the production level of the trained model, the estimated matrices $\hat{\mu}$ and $\hat{\Sigma}$ can be examined to get an idea about the semantics . Fig. 3(a), for instance, shows the estimated mean value for different features with respect to state 2, which is intended to model the 'style' of shots that used to introduce a new topic. As can be seen, this state is 'sensitive', ie: will yield a high probability, to shots with displayed captioned texts and no audio. When compared with the ground-truth, we observe that this is indeed a major kind of shots that demarcate topics.

To evaluate the detection performance, we manually watch and segment each video into topics. In some cases, this information is available directly from the video manuals. This results in a total of 75 indices. We use two well-known metrics, namely, recall and precision to measure the performance of the detection. To perform segmentation, we first run the Viterbi algorithm to get the time indices for which a state at topic level (ie: level 2) make the transition. Let τ be such an index, we then examine the state x_τ^3, which is the corresponding state at the production level being called. If this state coincides with the introduction shot (ie: $= 2$), then τ is recorded as a topic transition. The entire segmentation results are reported in Fig. 3. Calculation yields a recall of 77.3% and a precision of 70.7%. Given that the segmentation has been done in an completely unsupervised manner, ie: there is no hints in the training data as to what is a topic boundary, the result demonstrate the validity of the HHMM-based detection scheme.



(a) $\hat{\mu}_{\cdot\cdot}$

Video	GT	TP	Errors	
			FN	FP
1.	10	8	2	4
2.	9	6	3	3
3.	8	7	1	0
4.	5	3	2	4
5.	7	6	1	10
6.	19	15	4	2
7.	10	7	3	0
8.	7	6	1	1
Total	**75**	**58**	**17**	**24**

(b) detection results

Fig. 3. (a) estimated μ vector for state 2 (introduction shot), (b) detection results for 8 videos – GT : number of ground-truth indices, TP : number of correct detection, FN : number of miss, FP: number of over segmented indices.

This result is comparative with the probabilistic detection framework developed in [24] with a slight degradation in performance.

6 Discussion

Fig. 3 reveals that over segmentation is the major source of error causing a degradation in precision; and the high number of 'miss' (false negatives) causing a low recall rate. The resulting false negatives is not surprising since a topic is introduced in numerous ways, but the estimated model has learned only a subset of these methods of introduction. To overcome this, we obviously need a more complex model structure, which will be considered in our future work. The fact that the detector usually over segments a video (eg: video 5) is worth further discussion. A close analysis discloses that while these (over-segmenting) indices do not match the ground-truth, they frequently map to the lower level of sub-structures within a topic such as segments emphasising a safety message (for example, this happens many times in video 5). Fig. 4 draws an insight into the structure of an educational video and illustrate this problem. The vertical solid lines have been the target of our detection, while dashed-lines correspond to the over-segmented indices from the detector. This fact suggests that the model might be utilised to exploit further structure in topics, which will also be considered in our future work.

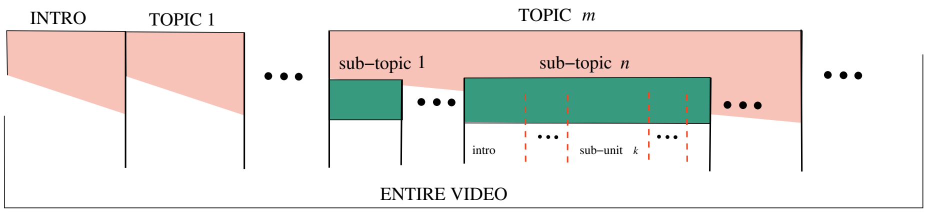


Fig. 4. Structure of a Video.

7 Conclusion

We have presented a framework for topic structure discovery using the HHMM in this paper. An important aspect of video data when considering its content organisation is the shared structures embedded in the data. This suggests a natural mapping to the hierarchical HMM, which we have utilised to model the topic structures in educational videos. We have briefly addressed the issue of parameter learning in the HHMM, in particular when the emission probability is modeled as a mixture of Gaussians. Finally, the experimental results have demonstrated the usefulness of the detection scheme.

References

1. Ariki, Y., Shibutani, A., Sugiyama, Y.: Classification and retrieval of TV Sports News by DCT features. In: IPSJ International Symposium on Information System and Technologies for Network Society. (1997) 269–272
2. Shearer, K., Dorai, C., Venkatesh, S.: Incorporating domain knowlege with video and voice data analysis (2000) MDM/KDD 2000, Workshop on Multimedia Data Minning, Aug 20-23, Boston, USA.
3. Bertini, M., Bimbo, A.D., Pala, P.: Content based annotation and retrieval of news videos. In: International Conference on Multimedia and Expo. (2000) 479–482
4. Eickeler, S., Müller, S.: Content-based video indexing of TV broadcast news using Hidden Markov Model. In: Proceedings of IEEE International on Acoustics Speech and Signal Processing. Volume 6., Phoenix (1999)
5. Huang, Q., Liu, Z., Rosenberg, A.: Automated semantic structure reconstruction and representation generation for broadcast news. In: Proc. IS&T/SPIE Conference on Storage and Retrieval for Image and Video Databases VII. Volume 3656. (1999) 50–62
6. Liu, Z., Huang, J., Wang, Y.: Classification of TV programs based on audio information using hidden markov model. In: IEEE Signal Processing Society 1998 Workshop on Multimedia Signal Processing. (1998) 27–32
7. Liu, Z., Huang, Q.: Detecting news reporting using audio/visual information. In: International Conference on Image Processing, Kobe, Japan (1999) 24–28
8. Walls, F., Jin, H., Sista, S., Schwartz, R.: Topic detection in broadcast news. In: Proceedings of the DARPR Broadcast News Workshop. (1999) 193–198
9. Seyeda-Mahmood, T., Srinivasan, S.: Detecting topical events in digital video. In: ACM Multimedia. (2000) 85–94
10. Adams, B., Dorai, C., Venkatesh, S.: Novel approach to determining movie tempo and dramatic story sections in motion pictures. In: 2000 International Conference on Image Processing,. Volume II., Vancouver, Canada (2000) 283–286
11. Adams, B., Dorai, C., Venkatesh, S.: Role of shot length in characterizing tempo and dramatic story sections in motion pictures. In: IEEE Pacific Rim Conference on Multimedia 2000, Sydney, Australia (2000) 54–57
12. Wang, J., Chua, T.S., Chen, L.: Cinematic-based model for scene boundary detection. In: The Eighth Conference on Multimedia Modeling, Amsterdam, Netherland (2001)
13. Sundaram, H., Chang, S.F.: Video scene segmentation using audio and video features. In: International Conference on Multimedia and Expo. (2000)
14. Fine, S., Singer, Y., Tishby, N.: The hierarchical hidden markov model: Analysis and applications. Machine Learning **32** (1998) 41–62
15. Murphy, K., Paskin, M.: Linear time inference in hierarhical hidden markov models. In: Proceedings of Neural Information Processing Systems, Vancouver, Canada (2001)

16. Xie, L., Chang, S.F., Divakaran, A., Sun, H.: Unsupervised discovery of multilevel statistical video structures using hierarhical hidden markov models. In: IEEE International on Multimedia and Expo, Baltimore, USA (2003) III.29 – III.32
17. Xie, L., Chang, S.F., Divakaran, A., Sun, H.: Learning hierarhical hidden markov models for unsupervised structure discovery from video. Technical report, Columbia University, New York (2002)
18. Theocharous, G., Mahadevan, S.: Learning the hierarchical structure of spatial environments using multiresolution statistical models. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). (2002)
19. Luhr, S., Bui, H.H., Venkatesh, S., West, G.: Recognition of human activity through hierarchical stochastic learning. In: International Conference on Pervasive Computing and Communication (PERCOM-03). (2003)
20. Skounakis, M., Craven, M., Ray, S.: Hierarchical hidden markov models for information extraction. In: Proceedings of the Eighteen International Joint Conference on Artificial Intelligence (IJCAI-03). (2003)
21. Bui, H.H., Phung, D.Q., Venkatesh, S.: Hierarchical hidden markov models with general state hierarchy. In: *to appear in* The Nineteenth National Conference on Artificial Intelligence (AAAI-04), San Jose, California USA (2004)
22. Herman, L.: Educational Films: Writing, Directing, and Producing for Classroom, Television, and Industry. Crown Publishers, INC., New York (1965)
23. Phung, D.Q., Venkatesh, S., Dorai, C.: On extraction of thematic and dramatic functions in educational films. In: IEEE International Conference on Multimedia and Expo, Baltimore, New York, USA (2003) 449 – 452
24. Phung, D.Q., Dorai, C., Venkatesh, S.: High level segmentation of instructional videos based on the content density function. In: ACM International Conference on Multimedia, Juan Les Pins, France (2002) 295–298

Author Index