On the Training Patterns Pruning for Optimum-Path Forest

João P. Papa and Alexandre X. Falcão

University of Campinas, Campinas SP 13083-970, Brazil
Institute of Computing, Visual Informatics Laboratory
{papa.joaopaulo, afalcao}@gmail.com

Abstract. The Optimum-Path Forest (OPF) classifier is a novel graph-based supervised pattern recognition technique that has been demonstrated to be superior to Artificial Neural Networks and similar to Support Vector Machines, but much faster. The OPF classifier reduces the problem of pattern recognition to a computation of an optimum-path forest in the feature space induced by a graph, creating discrete optimal partitions, which are optimum-path trees rooted by prototypes, i.e., key samples that will compete among themselves trying to conquer the remaining samples. Some applications, such that medical specialist systems for image-based diseases identification, need to be constantly re-trained with new instances (diagnostics) to achieve a better generalization of the problem, which requires large storage devices, due to the high number of generated data (millions of voxels). In that way, we present here a pruning algorithm for the OPF classifier that learns the most irrelevant samples and eliminate them from the training set, without compromising the classifier's accuracy.

1 Introduction

Recently, a novel approach for graph-based classifiers that reduces the pattern recognition problem as an optimum-path forest computation (OPF) in the feature space induced by a graph was presented [1]. The OPF does not interpret the classification task as a hyperplanes optimisation problem, but as a combinatorial optimum-path computation from some key samples (prototypes) to the remaining nodes. Each prototype becomes a root from its optimum-path tree and each node is classified according to its strongly connected prototype, that defines a discrete optimal partition (influence region) of the feature space. The OPF classifier has some advantages with respect to Artificial Neural Networks using Multilayer Perceptron (ANN-MLP) [2] and Support Vector Machines (SVM) [3]: (i) one of them is free of parameters, (ii) they do not assume any shape/separability of the feature space and (iii) run training phase faster. Notice that the OPF classifier has been extensively used for several purposes, including laryngeal pathology [4] and oropharyngeal dysphagia detection [5], fingerprint classification [6] and satellite-based rainfall estimation [7].

Some specialist systems, such that image-based medical diagnosis systems, need to be constantly re-trained, allowing a better generalization performance. Parasitological systems, for instance, need to be regularly evaluated due to some diseases that occur in some periods of the year, in which we have a small amount of available information of such pathologies. The main problem of these systems is the limited storage capacity, which are extremely important in 3D datasets, such that in brain MRI (Magnetic Resonance Imaging) images. These images have millions of voxels to be stored, increasing the training set size exponentially, and have two main impacts: the large amount of information to be stored and the high computational effort for training classifiers. Despite of SVM good generalization capacity, its have a major drawback regarding its use in real applications: they are time consuming and demand a high computational burden. Artificial Neural Networks (ANN) are unstable classifiers, leading us to use collections of ANN [8] trying to improve their performance up to some unknown limit of classifiers, increasing their whole complexity.

The OPF classifier has been demonstrated to be similar and much faster than the aforementioned classifiers [1], which can be a very interesting approach in situations with large datasets. Based on this assumption, we propose here an OPF-based approach for training set pruning without compromising the classifier accuracy. A third evaluation set is used to identify the irrelevant training samples, which are further removed and the classifier re-trained with this reduced training set. The remainder of this paper is organized as follows: Sections 2 and 3 present, respectively, the OPF theory and its training set pruning algorithm. The experimental results are further discussed in 4 and the conclusions are stated in 5.

2 Optimum-Path Forest Classifier

Let Z_1 , Z_2 , and Z_3 be training, evaluation, and test sets with $|Z_1|$, $|Z_2|$, and $|Z_3|$ samples of a given dataset. We use samples as points, images, voxels, and contours in this paper. As already explained, this division of the dataset is necessary to validate the classifier and evaluate its learning capacity from the errors. Z_1 is used to project the classifier and Z_3 is used to measure its accuracy, being the labels of Z_3 kept unseen during the project. A pseudo-test on Z_2 is used to teach the classifier to identify the irrelevant samples of Z_1 .

Let $\lambda(s)$ be the function that assigns the correct label i , $i = 1, 2, \dots, c$, of class i to any sample $s \in Z_1 \cup Z_2 \cup Z_3$, $S \subset Z_1$ be a set of prototypes from all classes, and v be an algorithm which extracts n features (color, shape, texture properties) from any sample $s \in Z_1 \cup Z_2 \cup Z_3$ and returns a vector $\mathbf{v}(s)$. The distance $d(s, t) \geq 0$ between two samples, s and t , is the one between their feature vectors $\mathbf{v}(s)$ and $\mathbf{v}(t)$. One can use any distance function suitable for the extracted features. The most common is the Euclidean norm $\|\mathbf{v}(t) - \mathbf{v}(s)\|$, but some image features require special distance algorithms [9]. A pair (v, d) then describes how the samples of a dataset are distributed in the feature space. Therefore, we call (v, d) a *descriptor* and the experiments in Section 4 use shape [10], texture [6] and color [11] descriptors based on this definition.

Our problem consists of projecting a classifier which can predict the correct label $\lambda(s)$ of any sample $s \in Z_3$. Training consists of finding a special set $S^* \subset Z_1$ of prototypes and a discrete optimal partition of Z_1 in the feature space (i.e., an optimum-path forest rooted in S^*). The classification of a sample $s \in Z_3$ (or $s \in Z_2$) is done by evaluating the optimum paths incrementally, as though it were part of the forest, and assigning to it the label of the most strongly connected prototype.

2.1 Training

Let (Z_1, A) be a complete graph whose nodes are the training samples and any pair of samples defines an arc in $A = Z_1 \times Z_1$. The arcs do not need to be stored and so the graph does not need to be explicitly represented. A path is a sequence of distinct samples $\pi_t = \langle s_1, s_2, \dots, t \rangle$ with terminus at a sample t . A path is said *trivial* if $\pi_t = \langle t \rangle$. We assign to each path π_t a cost $f(\pi_t)$ given by a connectivity function f . A path π_t is said optimum if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t . We also denote by $\pi_s \cdot \langle s, t \rangle$ the concatenation of a path π_s and an arc (s, t) . We will address the connectivity function $f_{\max}$.

$$\begin{aligned} f_{\max}(\langle s \rangle) &= \begin{cases} 0 & \text{if } s \in S, \\ +\infty & \text{otherwise} \end{cases} \\ f_{\max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{\max}(\pi_s), d(s, t)\} \end{aligned} \quad (1)$$

such that $f_{\max}(\pi_s \cdot \langle s, t \rangle)$ computes the maximum distance between adjacent samples along the path $\pi_s \cdot \langle s, t \rangle$. The minimization of $f_{\max}$ assigns to every sample $t \in Z_1$ an optimum path $P^*(t)$ from the set $S \subset Z_1$ of prototypes, whose minimum cost $C(t)$ is

$$C(t) = \min_{\forall \pi_t \in (Z_1, A)} \{f_{\max}(\pi_t)\}. \quad (2)$$

The minimization of $f_{\max}$ is computed by Algorithm 1, called OPF algorithm, which is an extension of the general image foresting transform (IFT) algorithm [12] from the image domain to the feature space, here specialized for $f_{\max}$. As explained in Section 1, this process assigns one optimum path from S to each training sample t in a non-decreasing order of minimum cost, such that the graph is partitioned into an optimum-path forest P (a function with no cycles which assigns to each $t \in Z_1 \setminus S$ its predecessor $P(t)$ in $P^*(t)$ or a marker *nil* when $t \in S$). The root $R(t) \in S$ of $P^*(t)$ can be obtained from $P(t)$ by following the predecessors backwards along the path, but its label is propagated during the algorithm by setting $L(t) \leftarrow \lambda(R(t))$.

Algorithm 1 – OPF ALGORITHM

INPUT: A training set Z_1 , λ -labeled prototypes $S \subset Z_1$ and the pair (v, d) for feature vector and distance computations.
 OUTPUT: Optimum-path forest P , cost map C and label map L .
 AUXILIARY: Priority queue Q and cost variable cst .

1. For each $s \in Z_1 \setminus S$, set $C(s) \leftarrow +\infty$.
2. For each $s \in S$, do
3. $\perp$ $C(s) \leftarrow 0$, $P(s) \leftarrow \text{nil}$, $L(s) \leftarrow \lambda(s)$, and insert s in Q .
4. While Q is not empty, do
5. $\perp$ Remove from Q a sample s such that $C(s)$ is minimum.
6. $\perp$ For each $t \in Z_1$ such that $t \neq s$ and $C(t) > C(s)$, do
7. $\perp$ Compute $cst \leftarrow \max\{C(s), d(s, t)\}$.
8. $\perp$ If $cst < C(t)$, then
9. $\perp$ If $C(t) \neq +\infty$, then remove t from Q .
10. $\perp$ $P(t) \leftarrow s$, $L(t) \leftarrow L(s)$ and $C(t) \leftarrow cst$.
11. $\perp$ Insert t in Q .

Lines 1–3 initialize maps and insert prototypes in Q . The main loop computes an optimum path from S to every sample s in a non-decreasing order of minimum cost (Lines 4–11). At each iteration, a path of minimum cost $C(s)$ is obtained in P when we remove its last node s from Q (Line 5). Ties are broken in Q using first-in-first-out policy. That is, when two optimum paths reach an ambiguous sample s with the same minimum cost, s is assigned to the first path that reached it. Note that $C(t) > C(s)$ in Line 6 is false when t has been removed from Q and, therefore, $C(t) \neq +\infty$ in Line 9 is true only when $t \in Q$. Lines 8–11 evaluate if the path that reaches an adjacent node t through s is cheaper than the current path with terminus t and update the position of t in Q , $C(t)$, $L(t)$ and $P(t)$ accordingly. One can use other *smooth* connectivity functions, as long as they group samples with similar properties [12].

We say that S^* is an optimum set of prototypes when Algorithm 1 minimizes the classification errors in Z_1 . S^* can be found by exploiting the theoretical relation between minimum-spanning tree (MST) [13] and optimum-path tree for $f_{\max}$ [14]. By computing a MST in the complete graph (Z_1, A) , we obtain a connected acyclic graph whose nodes are all samples of Z_1 and the arcs are undirected and weighted by the distances d between adjacent samples. The spanning tree is optimum in the sense that the sum of its arc weights is minimum as compared to any other spanning tree in the complete graph. In the MST, every pair of samples is connected by a single path which is optimum according to $f_{\max}$. That is, the minimum-spanning tree contains one optimum-path tree for any selected root node.

The optimum prototypes are the closest elements of the MST with different labels in Z_1 . By removing the arcs between different classes, their adjacent samples become prototypes in S^* and Algorithm 1 can compute an optimum-path forest in Z_1 . Note that, a given class may be represented by multiple prototypes (i.e., optimum-path trees) and there must exist at least one prototype per class.

It is not difficult to see that the optimum paths between classes tend to pass through the same removed arcs of the minimum-spanning tree. The choice of prototypes as described above aims to block these passages, reducing the chances of samples in any given class be reached by optimum paths from prototypes of other classes.

2.2 Classification

For any sample $t \in Z_3$, we consider all arcs connecting t with samples $s \in Z_1$, as though t were part of the training graph. Considering all possible paths from S^* to t , we find the optimum path $P^*(t)$ from S^* and label t with the class $\lambda(R(t))$ of its most strongly connected prototype $R(t) \in S^*$. This path can be identified incrementally, by evaluating the optimum cost $C(t)$ as

$$C(t) = \min\{\max\{C(s), d(s, t)\}\}, \quad \forall s \in Z_1. \quad (3)$$

Let the node $s^* \in Z_1$ be the one that satisfies Equation 3 (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*) = \lambda(R(t))$, the classification simply assigns $L(s^*)$ as the class of t . An error occurs when $L(s^*) \neq \lambda(t)$. Similar procedure is applied for samples in the evaluation set Z_2 .

2.3 Accuracy Computation

The accuracies $\mathcal{L}(I)$, $I = 1, 2, \dots, T$, are measured by taking into account that the classes may have different sizes in Z_3 (similar definition is applied for Z_2). If there are two classes, for example, with very different sizes and a classifier always assigns the label of the largest class, its accuracy will fall drastically due to the high error rate on the smallest class.

Let $NZ_2(i)$, $i = 1, 2, \dots, c$, be the number of samples in Z_3 (similar procedure is applied to Z_2) from each class i . We define

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|} \quad \text{and} \quad e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, \quad i = 1, \dots, c \quad (4)$$

where $FP(i)$ and $FN(i)$ are the false positives and false negatives, respectively. That is, $FP(i)$ is the number of samples from other classes that were classified as being from the class i in Z_3 , and $FN(i)$ is the number of samples from the class i that were incorrectly classified as being from other classes in Z_3 . The errors $e_{i,1}$ and $e_{i,2}$ are used to define

$$E(i) = e_{i,1} + e_{i,2}, \quad (5)$$

where $E(i)$ is the partial sum error of class i . Finally, the accuracies $\mathcal{L}(I)$, $I = 1, 2, \dots, T$, are written as

$$\mathcal{L}(I) = \frac{2c - \sum_{i=1}^c E(i)}{2c} = 1 - \frac{\sum_{i=1}^c E(i)}{2c}. \quad (6)$$

2.4 Complexity Analysis

The OPF algorithm, as aforementioned, can be divided in two phases: training and classification. In the training phase, essentially, we just need to compute the optimum prototypes (the closest samples in the MST) and after run the OPF algorithm (Algorithm 1). Let $|V_1|$ and $|E_1|$ be the number of samples (vertex) and

the number of edges, respectively, in the training set represented by the complete graph (Z_1, A) . We can compute the MST using the Prim's algorithm [13] with complexity $O(|E_1| \log |V_1|)$ and to find the prototypes in $O(|V_1|)$.

The OPF algorithm (Algorithm 1) main (Line 4) and inner (Line 6) loops run in $O(|V_1|)$ times each one, because we have a complete graph. In this sense, the OPF algorithm runs in $O(|V_1|^2)$. The overall OPF training step complexity can be executed in $O(|E_1| \log |V_1|) + O(|V_1|) + O(|V_1|^2)$, which is dominated by $O(|V_1|^2)$.

The classification step can be done in $O(|Z_2| |V_1|)$, in which $|Z_2|$ is the test set size. As aforementioned, a sample $t \in Z_2$ to be classified is connected to all samples in Z_1 , and its optimum-path cost is evaluated. Since OPF classifier can be understood as a dynamic programming algorithm [13], we do not need to execute the OPF algorithm again in the test phase, because each node $s_i \in Z_1$, $i = 1, 2, \dots, |Z_1|$ already has its optimum-path value stored. What we just need to do is to evaluate the paths from each node s_i until t . Finally, the overall estimated OPF complexity is $O(|V_1|^2) + O(|Z_2| |V_1|)$.

3 Pruning Algorithm

There are many situations that limit the size of Z_1 : large datasets, limited computational resources, and high computational time as required by some approaches. Mainly in applications with large datasets, it would be interesting to select for Z_1 the most informative samples, such that the accuracy of the classifier is little affected by this size limitation. This section presents a pruning algorithm which uses a third evaluation set Z_2 to improve the composition of samples in Z_1 and reducing its size.

From an initial choice of Z_1 and Z_2 , the algorithm projects an instance I of the OPF classifier from Z_1 and evaluates it on Z_2 . As aforementioned in Section 2.2, we can identify the node $s^* \in Z_1$ that satisfies Equation 3. In that way, for each node $s^* \in Z_1$ that participated from some $t \in Z_2$ classification process, we mark them as relevant nodes and at the final of evaluating Z_2 , we remove the unmarked (irrelevant) samples and we re-trained the classifier. To ensure that this new instance of the trained classifier will be as most similar as possible to the initial instance (before pruning), we also mark the samples in $P^*(t)$, i.e., all samples from $P(t)$ until $R(t)$ (Section 2.2). The pruning algorithm is described below.

Algorithm 2 – OPF TRAINING PATTERNS PRUNING ALGORITHM

INPUT: Training and evaluation sets, Z_1 and Z_2 , labeled by λ , and the pair (v, d) for feature vector and distance computations.
 OUTPUT: The OPF pruned training set.
 AUXILIARY: Arrays FP and FN of sizes c for false positives and false negatives and list LM of misclassified samples.

1. *Train OPF with Z_1 .*
2. *For each sample $t \in Z_2$, do*

3. | *Use the classifier obtained in Line 1 to classify t with a label $L(t)$.*
4. | *While $P(t) \neq R(t)$*
5. | | *Mark $P(t)$ as relevant*
6. | | | $t \leftarrow P(t)$
7. | | *Mark $R(t)$ as relevant*
8. | *Remove the unmarked samples from Z_1*
9. | *Return the pruned training set Z_1*

Line 1 trains the OPF classifier using Algorithm 1. The main loop (Lines 2 – 7) performs the classification step and the inner loop (Lines 4 – 6) and Line 7 marks all nodes in $P^*(t)$. Lines 8 – 9 remove the irrelevant samples and return the pruned training set Z_1 .

Afterwards, by comparing the accuracies of the classifier on Z_3 , before and after the pruning process, we can evaluate its reducing capacity without compromising its accuracy.

4 Experimental Results

We conducted two series of experiments: in the first one (Section 4.1) we evaluated the OPF algorithm against SVM and ANN-MLP in several public datasets using 50% for training and 50% for testing. These samples were randomly selected and each experiment was repeated 10 times with different sets Z_1 and Z_3 to compute mean (robustness) and standard deviation (precision) of the accuracy values. In the second round of experiments (Section 4.2), we evaluated the pruning algorithm capacity in reducing the training set size. The same datasets and similar procedure used as before was adopted: a training set Z_1 with 30% of the samples, an evaluation set Z_2 with 20% of the samples, and a test set Z_3 with 50% of the samples. These samples were randomly selected and each experiment was repeated 10 times with different sets Z_1 , Z_2 and Z_3 to compute mean accuracy (robustness) and the mean pruning rate.

For SVM implementation, we use the latest version of the LibSVM package [15] with Radial Basis Function (RBF) kernel, parameter optimization and the one-versus-one strategy for the multi-class problem. The cross validation technique is used for selecting parameters that minimize the error rate in the training set. We use the Fast Artificial Neural Network Library (FANN) [16] to implement the ANN-MLP with learning by backpropagation algorithm. The network configuration is $x:y:z$, where $x = n$ (number of features), $y = |Z_1| - 1$ and $z = c$ (number of classes) are the number of neurons in the input, hidden and output layers, respectively. For OPF implementation we used the LibOPF package [17].

We used here six datasets with diverse types of samples. The dataset MPEG-7 [18] uses shape images and COREL [19] uses color images. These datasets allow to evaluate the performance of the classifiers using shape and color descriptors, respectively. The remaining datasets use the (x, y) coordinates of 2D points as features: Cone-Torus, Petals and Saturn [20]. For MPEG-7 dataset, we used two

descriptors: FC (Fourier Coefficients) [21] and BAS (Beam Angle Statistics) [10]. For COREL dataset we used BIC descriptor [11].

4.1 Effectiveness of the Optimum-Path Forest Algorithm

We evaluated here the robustness the OPF classifier with respect to the SVM and ANN-MLP. Table 1 displays the results. We can see that OPF classifier outperformed SVM and ANN-MLP and all datasets, except for COREL-BIC. Some results show that OPF and SVM are similar if we consider the standard deviation.

Table 1. Mean accuracy results for different randomly generated training and test sets

Database	OPF	SVM	ANN-MLP
Cone-Torus	86.33%±0.1	78.41%±0.24	85.33%±2.21
Petals	100%±0.0	100%±0.0	100%±0.0
Saturn	82.88%±0.03	86.90%±0.05	83.60%±0.54
MPEG7-FC	71.71%±0.01	70.07%±0.01	57.28%±4.41
MPEG7-BAS	87.37%±1.21	87.05%±1.11	77.99%±3.41
COREL-BIC	86.88%±0.11%	90.65%±0.18	83.07%±1.41

4.2 Effectiveness of the Pruning Algorithm

We evaluated here the pruning algorithm capacity in reducing the training set size. Table 2 shows the results. We can see that we achieved good results with respect to the training set pruning rate. The MPEG-7 dataset was reduced up to 45% with its accuracy decreased up to 5%. The remaining datasets got lower compression rates, but their accuracies was decreased by only 1%-2%.

Table 2. Average pruning results for different randomly generated training, evaluating and test sets

Database	pruned Average	Accuracy before pruning	Accuracy after pruning
Cone-Torus	27.03%	86.33%	85.77%
Petals	17.06%	100%	98.88%
Saturn	32.00%	82.88%	76.22%
MPEG7-FC	48.09%	71.71%	65.66%
MPEG7-BAS	45.90%	87.37%	81.33%
COREL-BIC	39.12%	86.88%	85.11%

The most impressive results relie on the MPEG-7 dataset, which has 1400 samples distributed in 70 classes. The Corel dataset has 1603 colored images and its pruning rate was 39.12%. The lower pruning rate was achieved in the smallest dataset, Petals, which has only 100 samples.

5 Conclusions

We present here a novel training set pruning algorithm for OPF classifier, which is a new supervised approach for pattern recognition purposes that models the samples as nodes of a complete graph and tries to compute an collection of optimum-path trees (OPT) in this structure. Each OPT is rooted by its prototype, which offers optimum-path costs to the remaining nodes starting a competition process among all prototypes.

Some applications, such that automatic image-based diagnosis systems, need to be re-trained after some period to allow a better generalization performance. These kind of systems also suffer from the large amount of available data (3D MRI images), and the limited storage capacity of some devices is a problem that need to be considered. Based on this assumption, we proposed here an OPF pruning algorithm that can identify the training irrelevant samples using a third evaluation set, and further remove them from the training set. The results showed high pruning rates in some datasets (MPEG7-FC, MPEG77-BAS, COREL-BIC and Saturn). Currently, we are evaluating these results with respect to the computational effort (running time).

Acknowledgements

We would like to acknowledge the grants from projects BioCore (CNPq 550890/2007-6), CNPq 302617/2007-8 and FAPESP 07/52015-0.

References

1. Papa, J.P., Falcão, A.X., Suzuki, C.T.N., Mascarenhas, N.D.A.: A discrete approach for supervised pattern recognition. In: Brimkov, V.E., Barneva, R.P., Hauptman, H.A. (eds.) IWCIA 2008. LNCS, vol. 4958, pp. 136–147. Springer, Heidelberg (2008)
2. Haykin, S.: Neural networks: a comprehensive foundation. Prentice Hall, Englewood Cliffs (1994)
3. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the 5th Workshop on Computational Learning Theory, pp. 144–152. ACM Press, New York (1992)
4. Papa, J.P., Spadotto, A.A., Falcão, A.X., Pereira, J.C.: Optimum path forest classifier applied to laryngeal pathology detection. In: Proc. of the 15th Intl. Conf. on Systems, Signals, and Image Processing, vol. 1, pp. 249–252. IEEE, Los Alamitos (2008)
5. Spadotto, A.A., Pereira, J.C., Guido, R.C., Papa, J.P., Falcão, A.X., Gatto, A.R., Cola, P.C., Schelp, A.O.: Oropharyngeal dysphagia identification using wavelets and optimum path forest. In: Proc. of the 3rd IEEE Intl. Symp. on Communications, Control and Signal Processing, pp. 735–740 (2008)
6. Montoya-Zegarra, J.A., Papa, J.P., Leite, N.J., Torres, R.S., Falcão, A.X.: Learning how to extract rotation-invariant and scale-invariant features from texture images. EURASIP Journal on Advances in Signal Processing 2008, 1–16 (2008)

7. Freitas, G.M., Ávila, A.M.H., Pinto, H.S., Papa, J.P., Falcão, A.X.: Optimum-path forest-based models for rainfall estimation. In: Proceedings of the 14th IEEE Symposium on Computers and Communications (submitted, 2009)
8. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience, Hoboken (2004)
9. Wang, Y.P., Pavlidis, T.: Optimal correspondence of string subsequences. IEEE Transactions on Pattern Analysis and Machine Intelligence 12(11), 1080–1087 (1990)
10. Arica, N., Vural, F.T.Y.: BAS: A Perceptual Shape Descriptor Based on the Beam Angle Statistics. Pattern Recognition Letters 24(9-10), 1627–1639 (2003)
11. Stehling, R.O., Nascimento, M.A., Falcão, A.X.: A compact and efficient image retrieval approach based on border/interior pixel classification. In: Proceedings of the 11th International Conference on Information and Knowledge Management, pp. 102–109. ACM Press, New York (2002)
12. Falcão, A.X., Stolfi, J., Lotufo, R.A.: The image foresting transform: Theory, algorithms, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(1), 19–29 (2004)
13. Cormen, T., Leiserson, C., Rivest, R.: Introduction to Algorithms. MIT, Cambridge (1990)
14. Rocha, A., Miranda, P.A.V., Falcão, A.X., Bergo, F.P.G.: Object delineation by κ -connected components. EURASIP Journal on Advances in Signal Processing 2008, 1–5 (2008)
15. Chang, C.C., Lin, C.J.: LIBSVM: A Library for Support Vector Machines (2001), <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
16. Nissen, S.: Implementation of a Fast Artificial Neural Network Library (FANN), Department of Computer Science University of Copenhagen, DIKU (2003), <http://leenissen.dk/fann/>
17. Papa, J.P., Suzuki, C.T.N., Falcão, A.X.: LibOPF: A library for the design of optimum-path forest classifiers (2008), Software version 1.0, <http://www.ic.unicamp.br/~afalcao/LibOPF>
18. MPEG-7. Mpeg-7: The generic multimedia content description standard, part 1. IEEE MultiMedia 09(2), 78–87 (2002)
19. Corel Corporation. Corel stock photo images, <http://www.corel.com>
20. Kuncheva, L.: Artificial data. School of Informatics, University of Wales, Bangor (1996), <http://www.informatics.bangor.ac.uk/~mas00a>
21. Persoon, E., Fu, K.: Shape Discrimination Using Fourier Descriptors. IEEE Transactions on Systems, Man, and Cybernetics 7(3), 170–178 (1977)