Color Image Segmentation: Kernel Do the Feature Space

Jianguo Lee, Jingdong Wang, and Changshui Zhang

State Key Laboratory of Intelligent Technology and Systems
Department of Automation, Tsinghua University
Beijing, 100084, P. R. China
{lijg01,wangjd01}@mails.tsinghua.edu.cn
zcs@mail.tsinghua.edu.cn

Abstract. In this paper, we try to apply kernel methods to solve the problem of color image segmentation, which is attracting more and more attention recently as color images provide more information than gray level images do. One natural way for color image segmentation is to do pixels clustering in color space. GMM has been applied for this task. However, practice has shown that GMM doesn't perform this task well in original color space. Our basic idea is to solve the segmentation in a nonlinear feature space obtained by kernel methods. The scheme is that we propose an extension of EM algorithm for GMM by involving one kernel feature extraction step, which is called K-EM. With the technique based on Monte Carlo sampling and mapping, K-EM not only speeds up kernel step, but also automatically extracts good features for clustering in a nonlinear way. Experiments show that the proposed algorithm has satisfactory performance. The contribution of this paper could be summarized into two points: one is that we introduced kernel methods to solve real computer vision problem, the other is that we proposed an efficient scheme for kernel methods applied in large scale problems.

1 Introduction

Image segmentation [1] is an essential and critical topic in image processing and computer vision. Recently, color image segmentation attracts more and more attention mainly due to the following reasons: (1) Color images provide more information than gray level images do. (2) The power of computers is increasing rapidly, and PCs can deal with color images more easily.

Color image segmentation [2] can be viewed as an extension of gray level image segmentation, and there are various methods, which can be roughly categorized as three typical ways: (1) color space clustering based segmentation, (2) edge or contour detection based segmentation and (3) region or area extraction based segmentation. The basic idea of clustering based approach [3] is to directly cluster the pixels in color space by employing clustering algorithm such as k-means, Gaussian Mixture Models (GMM), etc. Edge based approach is a more global method. The basic idea is to firstly extract the edges using edges detectors such as Canny edge detector [4], secondly link the edges through edge

N. Lavrač et al. (Eds.): ECML 2003, LNAI 2837, pp. 253–264, 2003.

linking and tracing methods, and consequently obtain the segmentation by using the linked closed edges. Region based approach [5], including region growing, region split and merge, attempts to group pixels into homogeneous regions.

In this paper, we address the problem of clustering based color image segmentation. In fact, image segmentation can be viewed as a hidden variable problem, i.e., segmenting an image into clusters involves determining which source cluster generates the image pixels. Based on this fact, a global mixture probabilistic model can be built up. For instance, GMM based on the basic Expectation-Maximization (EM) [6] algorithm is applied for image segmentation.

One serious problem of GMM based segmentation is how to choose a proper color representation for segmentation. The frequently used color spaces, including linear space such as RGB and YIQ color space, nonlinear space such as HSV and LUV color space [2], have been verified in practice to be not suitable for clustering [3,7]. Thus two typical methods have been proposed towards the improvement. One is that Data-Driven Markov Chain Monte Carlo (DD-MCMC) [8] is employed to improve the performance of mixture probabilistic model for segmentation. The other is that Principal Components Analysis (PCA) is applied to find a characteristic color features space from mono- or hybrid- color spaces for detecting clusters [2]. However, two critical problems arose. One is that PCA is a linear method, but color image segmentation is more likely to be nonlinear. The other is that most clustering based approaches only utilize the color information, but do not utilize the geometrical or spatial information. All these motivate us to use kernel feature extraction method [9] instead of linear PCA to extract nonlinear features from both color and spatial information for clustering.

The basic idea of the proposed method is to extend EM algorithm to embed one kernel feature extraction step (K-Step), which does feature extraction and mapping to transform data from input space to nonlinear feature space (It should be emphasized that any kernel feature analysis methods can be utilized in K-Step, where in this paper Kernel PCA is adopted). The benefit of K-EM for GMM is that we could not only avoid the extremely large computational cost of kernel feature analysis, but also extend GMM to be capable of dealing with data sets with complex structures. The experiments show that the proposed method has satisfactory performance.

The rest of this paper is organized as follows. Section 2 presents the computational cost problem of kernel methods applied to large scale data sets, and proposes the K-EM algorithm for GMM to solve this problem. Section 3 demonstrates experimental results of the proposed algorithm on a synthetic data set and real world color image segmentation problems. Section 4 concludes.

2 K-EM Algorithm for GMM

As is known, EM algorithm often fails in data sets with complex structures (nonlinear and non-Gaussian). One possible way is to find a nonlinear map so that the data is projected and well clustered in the mapped feature space. That leads to the intuitive idea of our K-EM algorithm. In this section, we first

introduce kernel feature analysis method and present great computational cost problem in kernel methods for large scale data sets. Secondly, we propose the speedup technique for Kernel PCA based on Monte Carlo sampling and mapping. Thirdly, we provide the whole K-EM algorithm for GMM. Finally, we give some discussions of the proposed algorithm and related work.

2.1 Kernel Feature Analysis

Kernel trick [10] is an efficient method for nonlinear data analysis early used in Support Vector Machine (SVM). The idea is that we could implicitly map input data into a high dimension feature space via a nonlinear function:

$$\begin{aligned} \Phi : X &\to H \\ x &\mapsto \phi(x) \end{aligned} \tag{1}$$

And a similarity measure is defined from the dot product in H as follows:

$$k\left(x, x'\right) \triangleq \langle \phi(x), \phi(x') \rangle \tag{2}$$

where the kernel function $k(\cdot, \cdot)$ should satisfy *Mercer's condition* [9,10].

Besides being successfully used in SVM, kernel trick has been transported to many other kinds of algorithms. Among which, Kernel Feature Analysis (KFA), including Kernel Principal Component Analysis (Kernel PCA) [11] and Kernel Fisher Discriminant (KFD) [9] etc, is a class of methods with the most influence.

KFA aims to extract features in a nonlinear way. In this paper, we apply Kernel PCA as our feature extractor. We must stress that any other kernel feature extraction method could be employed in our final framework. Given a set of $\{x_i\}_{i=1}^m \in R^d$, with zero means, using the nonlinear mapping and kernel trick defined as in (1) and (2), Kernel PCA mainly depends on the eigen-decomposition problem on a Gram matrix.

$$m\lambda\alpha = \mathbf{K}\alpha \tag{3}$$

where $\alpha = \left(\alpha^1, \cdots, \alpha^m\right)^T$ is the expansion coefficient, λ is the eigen-value and $\mathbf{K}$ is the $m \times m$ Gram matrix with element $K_{ij} = \langle \phi(x_i), \phi(x_j) \rangle$.

To extract nonlinear features from a test point x, Kernel PCA computes dot product between $\phi(x)$ and the n^{th} eigenvector v^n in feature space to obtain the projection $y = (y^1, \cdots, y^p)$.

$$y^n = \langle v^n, \phi(x) \rangle = \sum\nolimits_{i=1}^{m} \alpha_i^n k\left(x_i, x\right) \quad n = 1, \cdots, p \tag{4}$$

where p ($p < m$) is the dimension of extracted features space F, and p is automatically chosen according to the leading eigenvalues λ.

Kernel PCA is an efficient nonlinear method for feature extraction. The advantage of Kernel PCA is that data with complex structure could be well clustered in feature space. However, the computational cost problem arises when Kernel PCA is applied to large scale data sets.

As is known, through kernel trick, the computational cost of Kernel PCA is mainly determined by the eigen-decomposition problem, such as (3) for Kernel

PCA. Thus the computational cost directly depends on the size of Gram matrix $\mathbf{K}$, i.e. the size of data sets. With the increase of size m, it is liable to meet with the curse of dimension. As is known, if m is large, e.g. larger than 5,000, currently it is impossible to finish the eigen-decomposition within hours even on the fastest PCs. Unfortunately, the size m is often very large in many cases, especially for data mining problems and pixels clustering based image segmentation (For example, m will be 16,384 for a not very large image with size 128×128). That is to say, the application of Kernel PCA method is seriously restricted by the size of data sets. Any attempt in this point is significative for the application of kernel methods. In the following subsection, we will focus on this point.

2.2 Speed up Kernel PCA by Monte Carlo Sampling and Mapping

Some techniques are proposed to handle the great computational cost problem in Kernel PCA. Most assume that the symmetry Gram matrix has a low rank. There are three typical techniques based on that assumption. The first technique is based on traditional Orthogonal Iteration or Lanzcos Iteration. The second is to make the Gram matrix sparse by sampling techniques [12]. The third is to apply Nyström method to speedup kernel machine [13]. However, all these methods still can't efficiently solve the vast-data problem such as pixel clustering based image segmentation. Moreover, none of these techniques has been successful in a practical problem.

Here, we also adopt this basic assumption, but reconsider the problem in a different way. Consider that samples forming the Gram matrix are drawn from a probabilistic distribution $p(x)$, thus the eigen-problem could be written down as a continuous form.

$$\int k(x,y)p(x)V_i(x)dx = \lambda_i V_i(y) \tag{5}$$

where λ_i, $V_i(y)$ are eigenvalue and eigenvector corresponding with the Gram matrix, and $k(\cdot,\cdot)$ is a given kernel function. Note that the eigenvalues are ordered so that $\lambda_1 > \lambda_2 \cdots$.

The integral on the left of Equation (5) could be approximated by using Monte Carlo method to draw a subset of samples $\{x_i\}_{i=1}^{N}$ according to $p(x)$.

$$\int k(x,y)p(x)V_i(x)dx \approx \frac{1}{N}\sum_{j=1}^{N} k(x_j,y)V_i(x_j) \tag{6}$$

Then plugging in $y = x_k$ for $j = 1, \cdots, N$, we obtain a matrix eigen-problem.

$$\frac{1}{N}\sum_{j=1}^{N} k(x_j,x_k)V_i(x_j) = \hat{\lambda}_i V_i(x_k) \tag{7}$$

where $\hat{\lambda}_i$ is the approximation of eigenvalue λ_i.

Fortunately, this approximation has been proved feasible and has bounded error performance [14]. This result is extremely beautiful. It indicates that we can approximate the eigenvalues of large size Gram matrix by using only a subset of samples.

Our proposed technique is based on this result. Considering data $x_i \in X$ with dimension d, Kernel PCA projects data x_i into feature space F to be y_i with dimension p, where $p \ll m$ holds true under the basic low rank assumption. Since eigenvalues are associated with eigenvectors, the corresponding eigenvectors of the approximated eigenvalues form a representative space D, which could be viewed as an approximation of feature space F. Our work will utilize this property to construct a representative space D to approximate feature space F in an iterative procedure.

Suppose sample x_i is drawn from known distribution $p(x)$, in other words, each sample $x_i \in X$ is associated with a known sampling weight $w(x_i)$. In most cases, especially in our color segmentation problem, $p(x)$ is not known. However it is possible using all samples to estimate a distribution $\hat{p}(x)$ to approximate $p(x)$. We just sample data set X according to $\hat{p}(x)$ independently T times to obtain T subset S_i $(1 \leqslant i \leqslant T)$, where S_i is a subset with N elements (with the low rank assumption, $N \ll m, N \geqslant p$) which are drawn from X according to $\hat{p}(x)$. Thus the sampling procedure could be viewed as Sampling Important Resampling (SIR) [15], which is one kind of Monte Carlo sampling method. Afterward, perform Kernel PCA on each S_i, and obtain representative space D_i. We apply Support Vector Regression (SVR) [10] to construct the mapping from S_i to D_i.

$$y = f_i(x),\ 1 \leqslant i \leqslant T \tag{8}$$

And T maps are combined together to be a nonlinear map from X to D as follows.

$$y = f(x) = \frac{1}{T} \sum\nolimits_{i=1}^{T} f_i(x) \tag{9}$$

When a new sample x comes, it could be projected to the representative space D by (9). We should emphasize here that using SVR to learn the mapping is due to its capability of generalization, and the reason of using T subsets is that the ensemble of T subsets can make result with less variance and more robust.

The whole sampling, resampling and combining procedure could be viewed as development of bagging ensemble [16] technique.

However, there is still a sticking point in the speedup scheme. That is how to obtain the distribution $p(x)$ or how to provide sampling weight information. We can't provide the information in one step, but we adopt the idea from Sequential Monte Carlo method that we provide the information in an iterative procedure. That we first project all the samples into the representative space, then we estimate a distribution in the space and update the sampling weight according to the estimated distribution, and then we can draw samples according to the new sampling weight information. Iteratively running the procedure till convergence, we can obtain the expected result. This is achieved by efficiently combining Kernel PCA with GMM in the next subsection.

2.3 K-EM Algorithm for GMM

GMM is a kind of mixture density models, which assumes that each component of the probabilistic model is a Gaussian density. That is to say:

$$p(y|\Theta) = \sum_{i=1}^{M} \alpha_i G_i(y|\theta_i) \tag{10}$$

where parameters $\Theta = (\alpha_1, \cdots, \alpha_M; \theta_1, \cdots, \theta_M)$ satisfy $\sum_{i=1}^{M} \alpha_i = 1$, $\alpha_i \geqslant 0$ and $G_i(y|\theta_i)$ is a Gaussian probability density function with parameter $\theta_i = (\mu_i, \Sigma_i)$.

EM [17] algorithm has been successfully used in generative models such as GMM. However, traditional EM for GMM often fails when structure of the data does not conform to mixture Gaussian assumption, especially for image pixels. Thus EM has been extended in many ways to overcome this problem.

D-EM [18] is one of the extensions of EM. It focuses mainly on two categorized semi-supervised learning problem, which has two categories with the minority samples labeled and the majority unlabeled. D-EM solves the problem by embedding a linear transformation step, which finds good fit of the data distributions as well as automatically selects good features. However, the linear transformation and two categorized learning framework limit its applications. Kernel trick is also introduced into D-EM [19], but the existed kernel version of D-EM did not start from the idea of speedup kernel feature extraction method in a probabilistic framework.

The proposed K-EM for GMM aims to extend D-EM in at least three aspects. The linear transformation will be replaced by Kernel PCA (In fact, any kernel feature analysis can be ultilized), the two categorized semi-supervised learning problem will be extended to multi-component clustering problem, and GMM provide sampling weight information so that Kernel PCA will be sped up greatly by Monte Carlo sampling and mapping technique.

The basic idea of K-EM for GMM is that it will efficiently combine two operations of Kernel PCA and parameters estimation of GMM. Kernel PCA extracts good features for GMM, whereas GMM provides sampling weight information needed by the proposed speedup technique of Kernel PCA. That is to say, in iteration step $(t-1)$, we estimate a distribution $p(y|\Theta^{(t-1)})$ in representative space D using GMM. Then we update the sampling weight of sample x_i by the following equation since each x_i corresponds to one y_i.

$$w^{(t-1)}(x_i) = \frac{p(y_i|\Theta^{(t-1)})}{\sum_{j=1}^{m} p(y_j|\Theta^{(t-1)})} \tag{11}$$

Consequently, we do the sampling important resampling and mapping step according to the updated weight information and the scheme described in previous subsection.

To summarize, for a given data set X, the proposed K-EM algorithm for GMM iterates over three steps. That is "Expectation-Kernel feature extraction-Maximization". The detail algorithm of K-EM for GMM is shown in Table 1. The algorithm initializes all the samples with the same sampling weight $1/m$. In

Table 1. The detailed K-EM algorithm for GMM.

Input: Data set X

Output: Clustering parameters Θ of GMM

S1: Initialize all samples with the sampling same weight $1/m$, number of clusters C, largest iteration steps L, iterating counter $t = 0$, number of subsets T and size of subsets N.

S2: Iteration counter $t = t + 1$

- **E-Step** does sampling in set X according to sampling weight information by M-Step to obtain T subsets S_i.
- **K-Step** performs Kernel PCA on each subset S_i, projects data in S_i into the representative space D_i, then learns the mapping function (8) between the input space and the representative space of set S_i. The T learned maps are combined to obtain the final map by (9), thus all the samples in X are projected into the representative space D by (9).
- **M-Step** performs parameters Θ estimation of GMM and updates sampling weight of each sample by (11).

S3: Test whether convergence is reached or $t > L$. If not, loop back to $S2$, otherwise return parameters Θ and exit.

other words, the first E-Step performs a uniform sampling. The algorithm could not be terminated until the parameters converge or the presetting iteration steps are reached.

2.4 Discussion of the Algorithm

As we have mentioned in previous section, the problem we addressed is to apply eigen-decomposition based kernel methods to large scale data sets. If the data set size m is large enough, e.g. larger than 5,000, the eigen-decomposition is intractable. However, the time complexity of the proposed method depends upon the subset size N instead of whole data set size m, and N could be much less than m ($N \ll m$), thus the proposed method can solve the problem indirect but much more efficiently. That could be viewed as our motivation.

There are still some other related work besides D-EM and its kernel version. One of the most influential methods is the particle filter or bootstrap filter [20]. Particle filter also iteratively uses weight updating and important sampling, which is called Sequential Important Sampling (SIS). And in our algorithm, the samples in subset could be viewed as particles in particle filter, and size of them is also fixed. But there are still two obvious differences. First, particle filter only uses particles to approximate the data distribution, but our method projects all data into feature space and approximates the data distribution by GMM. Second, particle filter only performs in the input data space, whereas our method performs in a feature space by Kernel PCA.

The other one is spectral clustering [21]. Spectral clustering could be regarded as first using RBF (only using RBF) based kernel methods to extract features,

and then performing clustering by k-means. It is not an iterative procedure. Moreover, it still can not deal with problems with large scale data set.

Our proposed method combines the idea of particle filter and that of spectral clustering, and could be regarded as a kernel extension of particle filter and a sequential version of spectral clustering in some sense.

3 The Experimental Results

To provide an intuitive illustration of the proposed algorithm, we firstly demonstrate the K-EM for GMM on a synthetic 2-D data set, and then on real world color image segmentation problem.

3.1 Synthetic 2-D Data Clustering

The synthetic data set with 2,000 samples is depicted in Fig 2. Traditional GMM based on basic EM, using coordinates of sample points as features, partitions the data set into two clusters as shown in Fig 2(b). The result is obviously not satisfying. As a comparison, the proposed K-EM for GMM is also employed this task using a polynomial kernel

$$k(x, x') = (x \cdot x')^d \tag{12}$$

with degree $d = 2$, and setting the representative dimension $p = 4$, subset S_i with size $N = 15$. In order to observe the sampling procedure easily, we just set number of subset $T = 1$. With these paramters, we achieve the promising results shown in Fig 2(a). Fig 2(c) shows the 15 samples drawn at the beginning of the algorithm, and Fig 2(d) shows the 15 samples drawn at the end of the algorithm. It is obvious that samples in Fig 2(d) provide more information about the structure of the data set than that of Fig 2(c).

What we still want to emphasize here is the running time compared with spectral clustering. On a Pentium4 2.0GHZ PC, the proposed algorithm could finish the whole clustering problem within 10 seconds, but spectral clustering need half an hour to achieve the same result using RBF kernel function

$$k(x, x') = \exp(-\gamma \left\| x - x' \right\|^2) \tag{13}$$

where sets $\gamma = 10$. We could see that the proposed algorithm achieves satisfied result as well as reduces the computational cost noticeably. All these demonstrate the power of the proposed algorithm.

3.2 Color Image Segmentation

In this part, we present some experiments on color image segmentation.

As Kernel PCA will project original color space to nonlinear space, the original color space we adopt is linear color space, that the RGB color space. As mentioned in the first section, the spatial information is also utilized for the

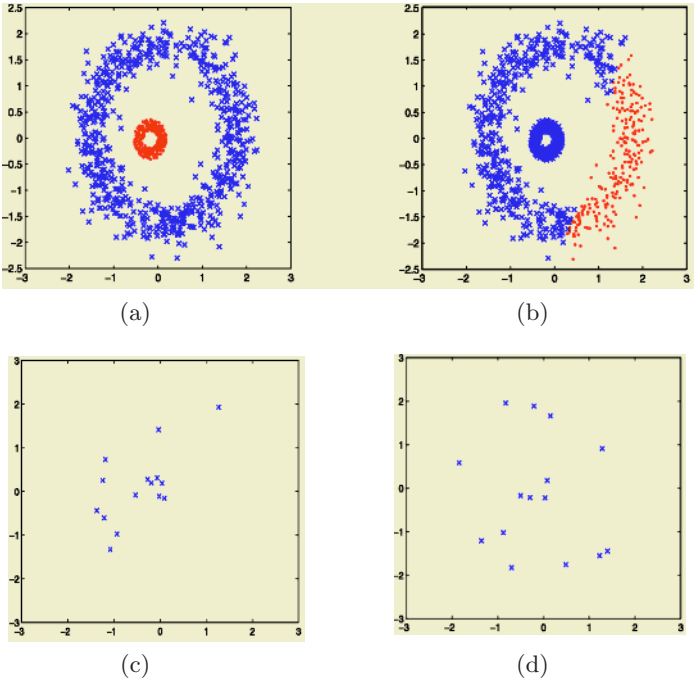

Fig. 1. Intuitive illustration of K-EM for GMM. (a) Clustering result by the proposed method. (b) Clustering result by GMM based on basic EM. (c) 15 Samples drawn at the beginning of the K-EM algorithm (marked by '×'). (d) 15 Samples drawn at the end of the K-EM algorithm.

clustering. The input features for our algorithm are $(r, g, b, x, y)^T$, where (x, y) is coordinate and (r, g, b) is RGB value of the corresponding pixel. We choose Kernel PCA as the kernel feature extraction method, and RBF kernel as in Equation (13) is used in all segmentation experiments. Some parameters could almost be fixed as dimension of subsets number $T = 10$, representative space $p = 7$, and samples in each subset $N = 400$ for most segmentation problem. Number of clusters C and parameter of RBF kernel γ need to be determined according to practice.

As a comparison, GMM based on basic EM also performs the same task. And it adopts the same clusters number as the proposed method, but only uses the RGB color space as the feature space for clustering.

Results are shown in Fig 2. First column depicts the original images, second column depicts the corresponding results by our K-EM for GMM, and the third column depicts the result by GMM based on basic EM. Both examples set the clusters number C to be three ($C = 3$). Parameter of RBF kernel is set to be $\gamma = 0.15$ and $\gamma = 1$ for Fig 2(b) and Fig 2(e) respectively. It is obvious that

the results achieved by our method are satisfactory and much better than GMM based on basic EM.

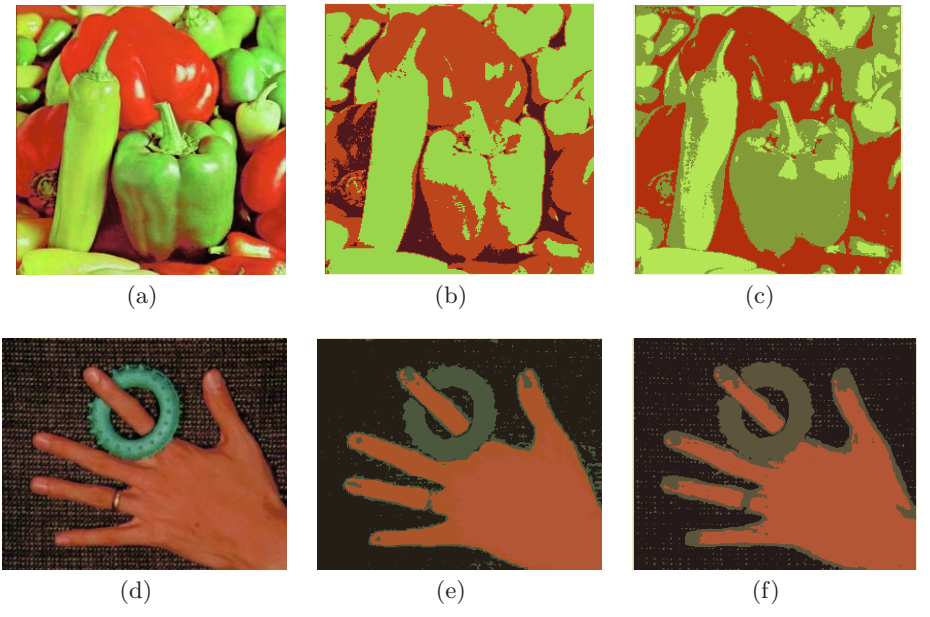

Fig. 2. Color Image segmentation results achieved by the proposed algorithm and basic GMM. Original images are depicted in the first column, the corresponding results by our K-EM for GMM are depicted in second column, and results by GMM based on basic EM are depicted in the third column. Both examples set the clusters number C to be three (C=3).

4 Conclusion

In this paper, we extend EM algorithm to K-EM by involving one kernel feature extraction (in this paper, kernel PCA) step. The proposed K-EM for GMM efficiently combines Kernel PCA and GMM, where GMM provides sampling weight information needed by the speedup technique of Kernel PCA, whereas Kernel PCA extracts good features for GMM. The advantage of the combination is that we could avoid the great computational cost, which could be encountered when directly performing eigen-decomposition of the Gram matrix in problems with large scale data sets. That is just the original intention of our proposed algorithm.

Experiments have been done on synthetic data set and real world color image segmentation problem by our algorithm. Results show that the algorithm has a satisfactory performance and does much better than GMM based on basic EM.

The attempts in color image segmentation is significative since we present a way to solve real computer vision problem efficiently by kernel methods.

Since the proposed K-EM algorithm needs to preset the number of clusters C, our future work will focus on automatically selecting C with methods such as Reversible Jump Markov Chain Monte Carlo as in [22]. We also intend to integrate multi-scale information to improve the color image segmentation results.

Acknowledgement

The author would like to thank anonymous reviewers for their helpful comments, also thank Xing Yi and Chi Zhang for their helpful work on revising this paper.

References

1. Forsyth, D. and Ponce, J.: Computer Vision: A Modern Approach, Prentice Hall Press (2002)
2. Cheng, H., Jiang, X., et al: Color image segmentation: advances and prospects, Pattern Recognition, Vol. 34, (2001) 2259-2281
3. Roberts, S J.: Parametric and Non-Parametric Unsupervised Cluster Analysis, Pattern Recognition, Vol.30, No.2, (1997) 261-272
4. Canny, J. F.: A Computational Approach to Edge Detection, IEEE Trans. on PAMI, Vol.8, No.6, (1986) 679-698
5. Adams, R. and Bischof, L.: Seeded Region Growing, IEEE Trans. on PAMI, Vol. 16, No. 6, (1994)
6. Belongie, S., Carson, C., Greeenspan, H. and Malik, J.: Color and Texture Based Image Segmentation Using EM and Its Application to Content-Based Image Re-trieval, Proceeding of the International Conferences on Computer Vision (ICCV'98) (1998)
7. Comaniciu, D. and Meer, P.: Mean Shift: A Robust Approach toward Feature Space Analysis, IEEE Trans. on PAMI, Vol.24, No.5, (2002) 603-619
8. Tu, Z. and Zhu, S.C.: Image Segmentation by Data-Driven Markov Chain Monte Carlo, IEEE Trans. on PAMI, Vol.24, No.5, (2002) 657-673
9. Schölkopf, B. and Smola, A. J.: Learning with Kernels: Support Vector Machines, Regularization and Beyond, MIT Press, Cambridge, Massachusetts(2002)
10. Vapnik, V.: The Nature of Statistical Learning Theory, Springer-Verlag, 2nd Edition, New York (1997)
11. Schölkopf, B., Smola, A.J. and Müller, K R.: Nonlinear Component Analysis as a Kernel Eigenvalue Problem, Neural Computation, Vol.10, No.5, (1998) 1299-1319
12. Achlioptas, D., McSherry, F. and Schölkopf, B.: Sampling techniques for kernel methods. Advance in Neural Information Processing Systems (NIPS) 14, (2002)
13. Williams, C. and Seeger, M.: Using the Nyström Method to Speed Up Kernel Machines. Advance in Neural Information Processing Systems (NIPS) 13, (2001)
14. Taylor, J. S., Williams, C., Cristianini, N. and Kandola J.: On the Eigenspectrum of the Gram Matrix and Its Relationship to the Operator Eigenspectrum, N. CesaBianchi et al. (Eds.): ALT 2002, LNAI 2533, Springer-Verlag, Berlin Heidelberg (2002) 23-40
15. Andrieu, C., N. de Freitas, Doucet, A. and Jordan, M. I.: An introduction to MCMC for machine learning. Machine Learning, 50 (2003) 5-43

16. Breiman, L.: Bagging Predictors, Machine Learning, Vol. 20, No.2, (1996) 123-140
17. Bilmes, J. A.: A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models, Technical Report, ICSI-TR-97-021, UC Berkeley (1997)
18. Wu, Y., Tian, Q. and Huang, T. S.: Discriminant-EM Algorithm with Application to Image Retrieval, Proceeding of the IEEE Conferences on Computer Vision and Pattern Recognition (CVPR' 00), (2000) 222-227
19. Wu, Y., Huang, T. S. and Toyama, K.: Self-Supervised Learning for Object Recognition based on Kernel Discriminant-EM Algorithm. Proceeding of the International Conferences on Computer Vision (ICCV'01), (2001) 275-280
20. Doucet, A., Nando de Freitas and Gordon, N. (ed): Sequential Monte Carlo in Practice, Springer-Verlag (2001)
21. Ng, A. Y., Jordan, M. I. and Weiss, Y., On Spectral Clustering: Analysis and an algorithm, Advance in Neural Information Processing Systems (NIPS) 14, (2002)
22. Kato, Z.: Bayesian Color Image segmentation using reversible jump Markov Chain Monte Carlo, CWI Research Report PNA-R9902, ISSN 1386-3711, (1999)