Scaling up Kernel Grower Clustering Method for Large Data Sets via Core-sets

CHANG Liang[1] DENG Xiao-Ming[2,3] ZHENG Sui-Wu[1] WANG Yong-Qing[1]

Abstract Kernel grower is a novel kernel clustering method proposed recently by Camastra and Verri. It shows good performance for various data sets and compares favorably with respect to popular clustering algorithms. However, the main drawback of the method is the weak scaling ability in dealing with large data sets, which restricts its application greatly. In this paper, we propose a scaled-up kernel grower method using core-sets, which is significantly faster than the original method for large data clustering. Meanwhile, it can deal with very large data sets. Numerical experiments on benchmark data sets as well as synthetic data sets show the efficiency of the proposed method. The method is also applied to real image segmentation to illustrate its performance.

Key words Kernel clustering, core-set, large data sets, image segmentation, pattern recognition

Clustering algorithms, which partition a data set into groups based on the similarity between them, are very useful in exploring data structure. They are widely used in many applications such as data mining, document retrieval, image segmentation, pattern classification, and so on[1−2]. Generally speaking, clustering methods include hierarchical methods, graph theoretic methods, decomposing a density function, and minimizing an objective function.

Kernel clustering methods have drawn much attention in recent years[2−5]. The basic idea of kernel methods is to transform the low dimensional input space $\mathcal{X}$ into a high dimensional kernel deduced feature space $\mathcal{F}$ in which the patterns are more likely to be linearly separable[6]. Comparing with classical clustering methods, kernel clustering methods have advantages in dealing with more complex, especially nonlinear separable data sets. Existing kernel clustering methods either kernelize the original methods such as kernel K-means, kernel self-organizing maps (SOM), kernel neural gas, or adopt the support vectors description such as kernel grower and support vector clustering.

Kernel grower[3] is a recently proposed unsupervised learning method. It is based on the classical K-means and one-class (SVM). It maps the data space to the kernel-deduced feature space and then adopts an iteration strategy similar to that of K-means. In each iteration, instead of computing the mean of points as centers, it computes the smallest sphere which encloses the closest data. The main advantages of the kernel grower method over other clustering algorithms are its ability to generate naturally nonlinear clustering boundaries and its robustness to outliers by using soft margin one-class SVM, in which a slack variable is used to allow some points to be outside the sphere.

However, a major drawback of the algorithm is its weak scalability. The algorithm becomes very slow when dealing with large data sets. It is even impossible to deal with extremely large data sets (with thousands or millions of data points). The time complexity of the algorithm is $O(N^3)$, where N is the cardinality of the data set. The weak scalability restricts the application of this algorithm greatly. To our knowledge, no successful practical application of this algorithm has been reported.

In this paper, we propose a scaled-up kernel grower algorithm using core-sets. A core-set is a subset of points by which the $(1+\epsilon)$-approximate minimum enclosing ball (MEB) can be obtained efficiently[7]. The idea of core-sets has been widely applied to fast smallest-enclosing ball computation[8−9], SVM classification[10], large data regression[11], and so on. The results in this paper show that the idea of core-sets can also be applied to develop effective clustering algorithms. Compared with the time complexity of $O(N^3)$ of the original kernel grower method, the scaled-up algorithm is of a much lower time complexity, which is only linear with N. Therefore, it is significantly faster in large data clustering than the original method. Furthermore, the new algorithm can deal with extremely large data sets, which makes the algorithm suitable for use in many practical applications.

An important application of clustering methods is image segmentation, which plays an important role in computer vision, object recognition, preprocessing of medical images, and so on. Among existing segmentation methods is the well-known normalized cut method[12], which treats image segmentation as a graph partitioning problem. Kernel-induced distance measure is also proposed to give a robust image segmentation method[13]. In this paper, we also apply the new scaled-up kernel grower algorithm to practical image segmentation. Experimental results show the validation and effectiveness of the method.

The paper is organized as follows. Section 1 introduces the kernel grower algorithm. Section 2 presents the scaled-up kernel grower algorithm. In Section 3, numerical experiments on benchmark data sets as well as synthetic data sets are presented. Real data experiments on image segmentation are also conducted in this section. Some concluding remarks are given in Section 4.

1 The kernel grower clustering algorithm

In this section, we briefly discuss the kernel clustering algorithm proposed by Camastra and Verri[3] and Cama

Received June 29, 2007; in revised form September 6, 2007
Supported by National Natural Science Foundation of China (60675039), National High Technology Research and Development Program of China (863 Program) (2006AA04Z217), and Hundred Talents Program of Chinese Academy of Sciences
1. The Key Laboratory of Complex System and Intelligence Science, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, P. R. China 2. Virtual Reality Laboratory, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, P. R. China 3. National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, P. R. China
DOI: 10.3724/SP.J.1004.2008.00376

-stra[4], which is called the kernel grower (KG) algorithm in this paper. The notion of kernel grower first appears in the Ph. D. dissertation of Camastra[4].

Suppose there are m input data $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m\} \subseteq \mathcal{X}$, where $\boldsymbol{x}_i \in \mathbf{R}^n$. First, a nonlinear transformation $\phi : \mathcal{X} \to \mathcal{F}$ is used to map the data point $\boldsymbol{x}_i$ to $\phi(\boldsymbol{x}_i)$. It holds that $\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j) = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, where $K(\cdot, \cdot)$ is a kernel function such as the Gaussian kernel. Then, a K-means like iterative strategy is adopted in the feature space. Two important concepts, Voronoi region and Voronoi set, will be introduced in the following.

Suppose $\{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_K\} \subset \mathcal{F}$ is the set of clustering centers, and K is the number of cluster centers. Set $\boldsymbol{z} = \phi(\boldsymbol{x})$ and $\boldsymbol{z}_i = \phi(\boldsymbol{x}_i)$. Then, the Voronoi region R_k of $\boldsymbol{p}_k (1 \leq k \leq K)$ in the feature space is defined as[3]

$$R_k = \{\boldsymbol{z} \in \mathcal{F} | k = \arg \min_{j=1,\cdots,K} \|\boldsymbol{z} - \boldsymbol{p}_j\|\} \tag{1}$$

The Voronoi set V_k of $\boldsymbol{p}_k$ is defined by

$$V_k = \{\boldsymbol{z}_i \in \mathcal{F} | k = \arg \min_{j=1,\cdots,K} \|\boldsymbol{z}_i - \boldsymbol{p}_j\| \text{ and } \|\boldsymbol{z}_i - \boldsymbol{p}_j\| < \rho\} \tag{2}$$

where the parameter ρ is a constant, which can be chosen by the model selection technique[14].

For a fixed kernel function, the kernel grower method consists of the following steps:

1) Initialize K Voronoi sets $V_k(\rho)$, $k = 1, \cdots, K$ by randomly choosing a small subset of the training points;

2) Train a one-class SVM for each $V_k(\rho)$, $k = 1, \cdots, K$ to get new cluster centers in the feature space;

3) Update each $V_k(\rho)$, $k = 1, \cdots, K$ according to the centers obtained in Step 2;

4) If no Voronoi set changes in Step 3, stop; otherwise, go to Step 2.

Instead of getting the clustering centers by calculating the mean of the Voronoi sets in K-means, the kernel clustering method uses the one-class SVM.

1.1 The one-class SVM

The one-class SVM proposed by Tax and Duin was named as support vector data description (SVDD)[15]. SVDD seeks the minimum enclosing hypersphere of the target data and excludes the space of outliers. On the other hand, the one-class SVM based on hyperplanes was also proposed[16].

Suppose $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_l\}$ is the set of training patterns, where $\boldsymbol{x}_i \in \mathbf{R}^n$. The SVDD is formulated as

$$\begin{array}{ll} \min\limits_{R, \boldsymbol{a}, \xi_i} & R^2 + \frac{1}{\mu l} \sum\limits_{i=1}^{l} \xi_i \\ \text{s.t.} & ||\phi(\boldsymbol{x}_i) - \boldsymbol{a}||^2 \leq R^2 + \xi_i,\ i = 1, 2, \cdots, l \\ & 0 \leq \xi_i,\ i = 1, 2, \cdots, l \end{array} \tag{3}$$

where R and $\boldsymbol{a}$ denote the radius and the center of the hypersphere in the feature space, respectively, ϕ is the corresponding feature mapping, $\boldsymbol{\xi} = [\xi_1, \cdots, \xi_l]^{\mathrm{T}}$ denotes the slack variable, and $\mu \in (0, 1)$ is the user-defined parameter specifying the upper bound on the fraction of outliers.

Problem (3) is often solved by its dual problem:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & -\sum\limits_{i=1}^{l} \sum\limits_{j=1}^{l} \alpha_i \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum\limits_{i=1}^{l} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}_i) \\ \text{s.t.} & \sum\limits_{i=1}^{l} \alpha_i = 1,\ 0 \leq \alpha_i \leq \frac{1}{\mu l},\ i = 1, \cdots, l \end{array} \tag{4}$$

where $\boldsymbol{\alpha} = \{\alpha_1, \cdots, \alpha_l\}^{\mathrm{T}}$ is the dual variable, and $K_{l \times l} = [k(\boldsymbol{x}_i, \boldsymbol{x}_j]_{l \times l} = [\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j]_{l \times l}$ is the kernel matrix.

Thus, the SVDD is transformed into solving a dual quadratic programming (QP), which is commonly solved by Newton or quasi-Newton method in optimization.

1.2 Drawbacks of KG

Although KG compares favorably against popular clustering algorithms on many benchmark databases[3], the main drawbacks of KG lie in

1) KG requires heavy computation time, since QP problems are solved in each iteration.

2) Moreover, KG is nearly impossible to run for very huge data sets due to its slowness.

In order to improve the KG algorithm, we propose a scaled-up KG method for large data set in the next section.

2 Scaling up kernel grower clustering for large data sets

In this section, we improve the KG algorithm by using the idea of core-sets. The improved KG method is called the scaled-up KG algorithm.

2.1 Core-sets

The notion of core-sets appears in solving the approximate minimum enclosing ball (MEB) problem in computational geometry[7−9]. A core-set is a subset of indices such that the $(1 + \epsilon)-$approximate MEB can be obtained by solving the optimization directly. An encouraging property of core-sets is that the number of elements in it is independent of the dimension of the space and the size of the data set[7]. Fig. 1 demonstrates the idea of core-sets in solving the $(1 + \epsilon)-$approximate MEB in computational geometry.

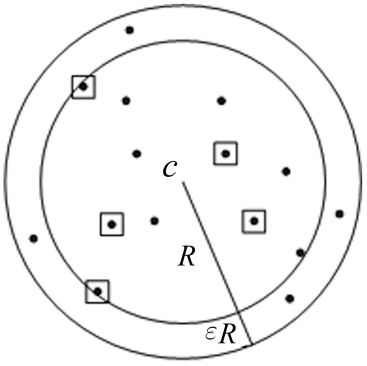


Fig. 1 Denote S to be the set of training examples, the inner circle is the MEB of points in squares, and the outer circle is the $(1 + \epsilon)$-approximate of $MEB(S)$. The set of points in squares is thus a core set

Tsang et al.[10] transformed the traditional SVM problem into solving the MEB in the feature space using core-sets and proposed an approximate method to solve SVM, called core vector machine (CVM). CVM is considerably fast in large data classification. The idea of core-sets has also been used widely in fast smallest-enclosing ball computation[8−9], large data regression[11], and support vector data description (SVDD)[17] for massive data problems and so on.

2.2 Scaled-up support vector data description

Denote by S_t, $\boldsymbol{c}_t$, and R_t, the core-set, the center of the ball, and the radius at the t-th iteration, respectively $(t = 1, 2, \cdots)$. Suppose $MEB(S)$ is the MEB of the set S and $B(\boldsymbol{c}, R)$ is a ball with the center of $\boldsymbol{c}$ and the radius of R. Let the parameter μ be an upper bound on the fraction of outliers. SVDD gets the minimum enclosing hypersphere in the kernel-deduced feature space. For a given $\epsilon > 0$, the scaled up SVDD works as follows[17]:

1) Initialize $S_1, \boldsymbol{c}_1$, and R_1.

2) Terminate if the number of points $\varphi(\boldsymbol{z})$ which falls outside $B(\boldsymbol{c}_t, (1+\epsilon)R_t)$ is smaller than μN, where N is the number of patterns.

3) Otherwise, find $\boldsymbol{z}$ such that $\boldsymbol{z}$ is the closest point outside $B(\boldsymbol{c}_t, (1+\epsilon)R_t)$. Set $S_{t+1} = S_t \bigcup \{\boldsymbol{z}\}$.

4) Find the new $MEB(S_{t+1})$ and set $\boldsymbol{c}_{t+1}$ to be the center of $MEB(S_{t+1})$ and R_{t+1} to be the radius of $MEB(S_{t+1})$.

5) Increase t by 1 and go back to Step 2.

2.3 The scaled-up KG algorithm

The scaled-up KG algorithm consists of the following steps:

1) Initialize K Voronoi sets $V_k(\rho), k = 1, \cdots, K$ using a small subset of the training points.

2) Instead of using the popular dd-tool for one-class problem, the one-class SVM for each $V_k(\rho)$ is trained using the scaled-up SVDD.

3) Update each $V_k(\rho)$.

4) If no Voronoi set changes in Step 3 or the iteration number is larger than a given threshold, stop; otherwise, go to Step 2.

2.4 Time complexity analysis

Suppose N denotes the cardinality of the data set, K denotes the number of clustering centers and T_1 and T_2 denote the iteration numbers to guarantee convergence of KG and scaled-up KG, respectively. If the number of cluster indices belonging to each cluster set at the t-th iteration is $\{n_1^{(t)}, \cdots, n_K^{(t)}\}$, then, KG has a time complexity of

$$C_1 = \sum_{i=1}^{T_1} O((n_1^{(t)})^3) + \cdots + O((n_K^{(t)})^3) = O(T_1 N^3) = O(N^3)$$

In comparison, the scaled-up KG has a time complexity of

$$C_2 = \sum_{i=1}^{T_2} O(\frac{1}{\epsilon^2} n_1^{(t)} + \frac{1}{\epsilon^4}) + \cdots + O(\frac{1}{\epsilon^2} n_K^{(t)} + \frac{1}{\epsilon^4}) = O(\frac{T_2}{\epsilon^2} N + \frac{T_2}{\epsilon^4}) = O(\frac{1}{\epsilon^2} N + \frac{1}{\epsilon^4})$$

Here, T_1 and T_2 are the given numbers. They can be chosen using a model selection criteria such as cross validation. Thus, for any fixed $\epsilon > 0$, the time complexity of the scaled-up KG is linear with the size of the data set,whereas the time complexity of KG is cubic with the size of the data set.

3 Experiments and comparisons

We compare KG and scaled-up KG on benchmark data sets as well as synthetic data sets in both CPU time and the correct ratio. An application of scaled-up KG to real image segmentation is also given. Experiments were run on a 3.20GHz Pentium-4 machine, with 512MB RAM. The program was written with Matlab. We adopted the data description toolbox dd-tools 1.5.5[18] in the implementation of KG.

3.1 Comparison of clustering performance with benchmark databases

Example 1. Iris data

The Iris data set contains 150 instances, and each is composed of four measurements. The instances belong to three classes, and each class is represented by 50 instances. One class is linearly separable from the other two, whereas the other two classes are nonlinearly separable.

We compared KG and scaled-up KG on iris data in 20 runs with respect to different initializations. The parameters of the Gaussian kernel in KG and scaled-up KG were set to be $\sigma_1 = \sigma_2 = 1.1$, respectively. The parameter μ was set to be $\mu_1 = 10^{-10}$ in KG and $\mu_2 = 10^{-10}$ in scaled-up KG. We considered $\epsilon = 0.05$. In each run, 20 iterations were taken to guarantee convergence for both methods.

Scaled-up KG took longer CPU time than KG in this experiment, as shown in Table 1. It is understandable since the size of iris data is relatively small and the scaled-up KG has to run the QP multiple times, which takes more time than solving the QP directly. As an approximation algorithm, scaled-up KG has a correct ratio that is slightly lower than that of KG. However, it is much higher than other clustering algorithms and is acceptable, as shown in Table 2.

The results show that KG is more suitable for the clustering of small data sets (e. g., the number of patterns are smaller than 200). The performance of KG and scaled-up KG on large-scale data clustering was tested in the following.

Example 2. Wisconsin breast cancer database

The Wisconsin breast cancer database consists of 699 cases. It has 9 dimensions, and each dimension has a value between 1 to 10. It contains 16 cases which have at least one unknown attribute value. Thus, the database contains 683 available patterns. All the patterns belong to two classes.

We compared the KG and scaled-up KG in 20 runs with respect to different initializations. The parameters of the

Gaussian kernels were set as $\sigma_1 = 0.9$ and $\sigma_2 = 0.5$, respectively. In each run, we took 20 iterations in KG to guarantee convergence and 10 iterations in scaled-up KG for convergence. The parameter $\mu_1 = \mu_2 = 10^{-6}$ in both algorithms, and we considered $\epsilon = 0.1$ in scaled-up KG. It turns that scaled-up KG has a much shorter CPU time than KG, as shown in Table 1. The accuracy of scaled-up KG is slightly lower than that of KG but much higher than that of other methods, as shown in Table 2.

The results show that the scaled-up KG can speed up the original KG significantly, and its accuracy is validated to be close to that of KG.

Example 3. Spam database

The Spam email database contains 4 601 patterns, which belong to two classes of spam and non-spam data. Each pattern has 58 dimensions. Among the entire patterns, 1 813 patterns belong to the class of spam (39.4 %), whereas the remaining patterns belong to the class of nonspam (60.6 %).

The experiment was performed on 1 534 patterns from the Delta set, which is the same as it is in [3]. 20 runs with different initializations were tested for scaled-up KG. In each run, 24 iterations were taken to guarantee convergence. The parameter of Gaussian kernel was set as $\sigma = 8.0$. We set $\mu = 10^{-6}$ and $\epsilon = 0.2$. The result shows that KG becomes very slow, but scaled-up KG is very fast, as is shown in Table 1. Although the accuracy of scaled-up KG is slightly lower than that of KG, it outperforms other commonly used clustering algorithms, as shown in Table 2. Thus, it follows that the scaled-up KG is efficient in large-scale data clustering. Its average accuracy is higher than those of other popular clustering algorithms and slightly lower than that of KG.

Remark 1. In Table 1, T_1 is the CPU time of scaled-up KG, T_2 is the CPU time of KG, $*$ means that the algorithm takes too long a time or is nearly impossible to run. It can be seen that KG achieves a better performance for small data clustering. On the other hand, scaled-up KG is significantly faster than KG for large-scale data clustering.

Remark 2. Table 2 shows the comparison results on average clustering accuracy using different clustering methods. The results for SOM, K-means, neural gas, Ng-Jordan, and kernel grower are from [3]. It can be seen that the accuracy of scaled-up KG outperforms SOM, K-means, neural gas, and Ng-Jordan on the three benchmark datasets. The accuracy of scaled-up KG is slightly lower than that of KG since it is an approximate algorithm.

Table 1 Comparison of scaled-up KG and KG in terms of CPU time

Data set	Data size	T_1 (s)	T_2 (s)
Iris	150	47.9523	**12.9477**
Delta	424	**9.3985**	226.2800
Wisconsin	683	**22.8453**	807.1695
Spam	1534	**44.8258**	$*$

Table 2 Comparison of average correct ratios with respect to different clustering methods

Algorithm	Iris(%)	Wisconsin(%)	Spam(%)
SOM	81.0	96.7	78.9
K-means	89.0	96.1	70.6
Neural gas	91.7	96.1	68.4
Ng-Jordan	84.3	95.5	60.6
Kernel grower	94.7	97.0	81.3
Our method	93.4	96.8	80.2

In summary, scaled-up KG is significantly faster in large-scale data clustering; moreover, it provides an effective approximation algorithm with preferable correct ratios compared with the original KG.

3.2 Comparison of clustering performance with synthetic data sets

Example 4. Delta set

We used the synthetic delta set to test the KG algorithm and the scaled-up KG algorithm. Delta set is publicly available[1]. It is formed by 424 training samples in $\mathbf{R}^2$. The points are randomly distributed on two semicircles with

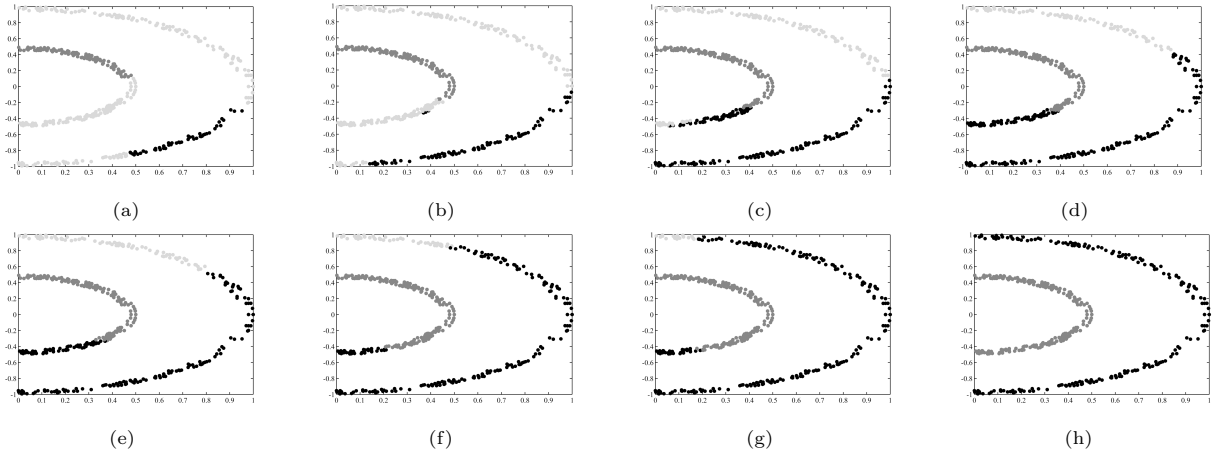
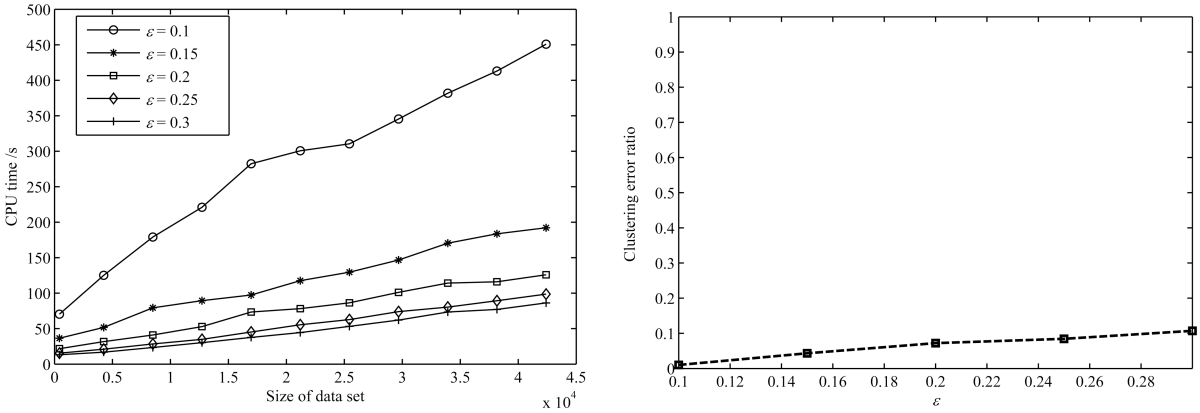


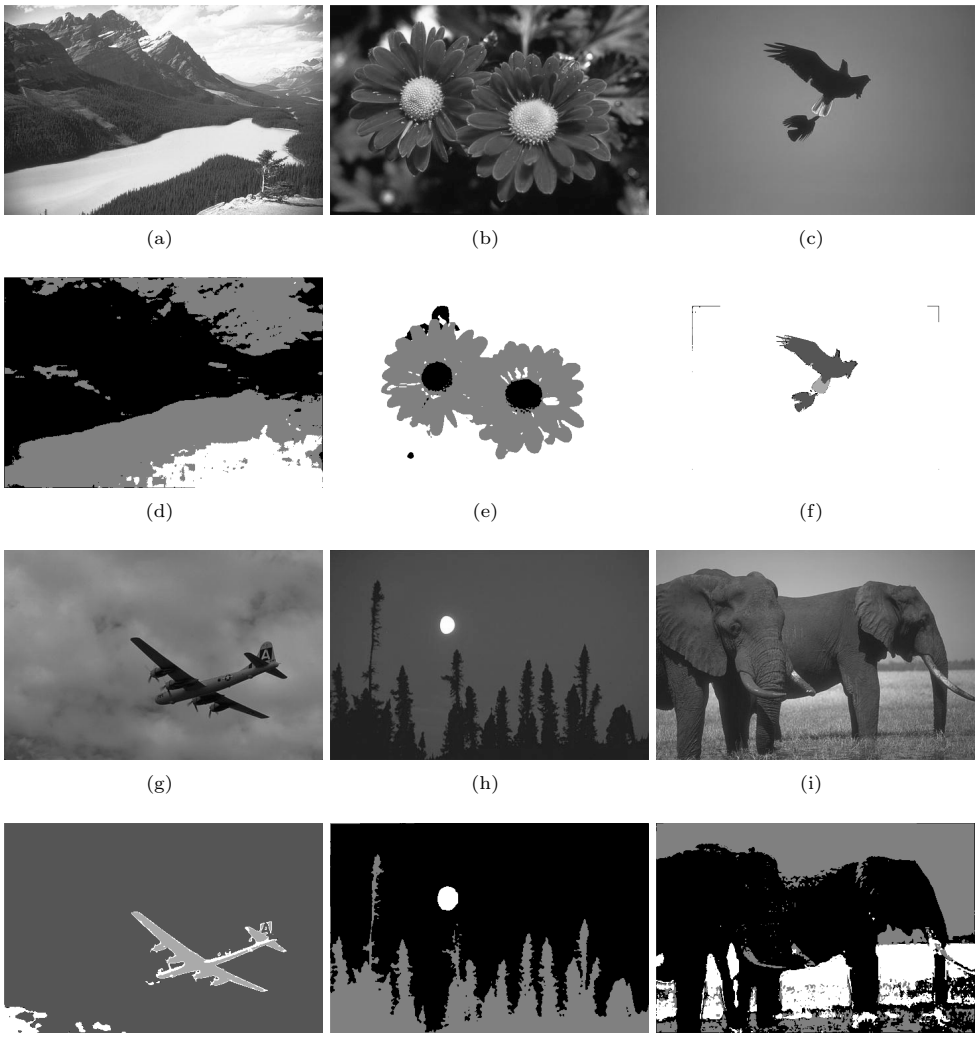
Fig. 2 An example of the convergence procedure of the scaled-up KG on Delta set. It takes eight iterations to converge. The data which belong to the first and second clusters, and data which do not belong to any cluster are denoted with different gray levels. It can be seen that the points are clustered into two classes successfully at the last iteration

[1] ftp.disi.unige.it/person/CamastraF/delta.dat



(a) CPU time vs the size of data sets (b) Average error ratio increases as a function of ϵ

Fig. 3 Experiment results on scaled-up delta sets using the scaled-up KG



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Fig. 4 The segmentation results for images from Berkeley image segmentation database with scaled-up KG. The segmentations are based on color information. Odd row: original images. Even row: segmentation results. The original images are color images, while the segmentation results are shown by intensity images. Each segmented image is shown by k discrete levels of gray values, where k is the number of clusters. Pixels with the same level of gray values belong to the same cluster. It can be seen that the scaled-up KG can deal with image segmentation problems efficiently

the same center and are linearly inseparable.

Commonly used algorithms, which are based on distance criteria in the original input space such as K-means, cannot separate delta set with just two exemplars well[3]. On the other hand, similar to KG, scaled-up KG can separate the two clusters successfully.

We compared KG and scaled-up KG in 20 runs with different initializations. The parameters of the Gaussian kernel $K(x,y)=\exp(\frac{-\|x-y\|^2}{\sigma^2})$ were set as $\sigma=\sigma_1=\sigma_2=0.4$. We took $\epsilon=0.1$ in scaled-up KG. In each run, we took 20 iterations in both algorithms to guarantee convergence. The result was that the correct ratio was equal to 100 percent for both algorithms. However, scaled-up KG took a much shorter CPU time than KG, as shown in Table 1.

Therefore, scaled-up KG can speed up the clustering on Delta data significantly with the same correct ratio as KG.

Example 5. Scaled-up delta sets

In this experiment, we tested scaled-up KG on a series of scaled-up delta data sets. The points were randomly drawn around 2 semi-circles in $\mathbf{R}^2$ with the same center. The radius of the semi-circles were set as 0.5 and 1.0, respectively. The size of the data sets ranged from 4 240 to 42 400. Based on the synthetic data sets, we compared the scalability of KG and scaled-up KG. The parameter of Gaussian kernel was chosen as $\sigma=0.4$. The iteration number of scaled-up KG was set to be 100. Here, we took $\epsilon=0.15$.

KG was no longer applicable to the scaled-up delta data due to its slowness, but the scaled-up KG gave us the clustering results accurately and efficiently.

Fig. 3 shows the performance of scaled-up KG on scaled-up Delta sets in terms of both CPU time and the average error ratio. It can be seen in Fig. 3 (a) that the CPU time increases with the increasing of the size of the data set. For fixed degree of approximation, that is, fixed ϵ, the CPU time increases linearly in the size of the data set. Moreover, a larger ϵ corresponds to a shorter time. Fig. 3 (b) shows that the average clustering error increases slowly as a function of ϵ.

In summary, the above experiments on benchmark data sets as well as synthetic data sets show that in dealing with large-scale data clustering, scaled-up KG is significantly faster than the original KG, with the correct ratio close to that of KG. Furthermore, scaled-up KG can deal with much larger data sets which KG finds it difficult to handle.

3.3 Application with real data

Many problems in pattern recognition and computer vision require fast and effective clustering methods for large data sets. In this experiment, we used the scaled-up KG in color image segmentation. In each picture, we took the pixels as samples. Since the number of pixels is huge, KG becomes unavailable due to its slowness. Note that the segmentation in this paper is basically based on color.

The Berkeley image segmentation database[19] contains 300 images of natural scenes with true hand segmentations of each image. The original images are available online[2]. We used 9 color images from the Berkeley image segmentation database for segmentation. Each image is an $481\times321\times3$ array of color pixels, where each color pixel is a triplet corresponding to red, green, and blue components. For each picture, after converting from RGB to HSV color space, we obtained a matrix composed of 154 401 column vectors in the $\mathbf{R}^2$ space. We explored the performance of our scaled-up KG. The segmentation results were demonstrated in Fig. 4. Images (a), (b), (c), and (h) in Fig. 4 used 3 classes for segmentation, whereas images (g) and (i) used 4 classes. We took 24 iterations in each run for convergence. Each image took less than 2 minutes for segmentation using the scaled-up KG, with program written in Matlab. Compared with the unavailability of KG, scaled-up KG can deal with the image segmentation problems fast and effectively.

4 Conclusion

In summary, this paper makes three contributions. First, we propose a scaled-up kernel grower clustering method using core-sets. It is significantly faster than the original kernel grower in dealing with large data sets. Second, the scaled-up kernel grower can handle much larger data sets than the original method can. Numerical experiments on benchmark data sets as well as synthetic data sets show the priority of the method. Third, a real application of scaled-up kernel grower on image segmentation is also given to show the efficiency and validation of the method. The future work will be focused on the convergence and model selection methods for better parameter settings of the method.

Acknowledgement

The authors would like to acknowledge professor Zhang Bo and professor Qiao Hong for their valuable suggestions, and the authors thank professor Francesco Camastra for his helpful discussions.

References

1 Xu R, Wonsch II D. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 2005, **16**(3): 645−678

2 Filippone M, Camastra F, Masulli F, Rovetta S. A survey of kernel and spectral methods for clustering. *Pattern Recognition*, 2008, **41**(1): 176−190

3 Camastra F, Verri A. A novel kernel method for clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005, **27**(5): 801−805

4 Camastra F. Kernel Methods for Unsupervised Learning [Ph. D. dissertation], University of Genova, 2004

5 Ben-Hur A, Horn D, Siegelmann H T, Vapnic V. Support vector clustering. *Journal of Machine Learning Research*, 2001, **2**: 125−137

6 Schölkopf B, Smola A J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. London: The MIT Press, 2002. 25−55, 187−248, 405−468

7 B·adoiu M, Clarkson K L. Optimal core sets for balls. In: Proceedings of DIMACS Workshop on Computational Geometry. New Jersey, USA: 2002

8 Kumar P, Mitchell J S B, Yildirim A. Approximate minimum enclosing balls in high dimensions using core-sets. *Journal of Experimental Algorithmics*, 2003, **8**

[2] http://www.eecs.berkeley.edu/Research/Projects/CS/visi on/grouping/segbench

9 Nielsen F, Nock R. Approximating smallest enclosing balls. In: Proceedings of International Conference on Computational Science and Its Applications. Assisi, Italy: Springer, 2004. 147−157

10 Tsang I W, Kwok J T, Cheung P M. Core vector machines: fast SVM training on very large data sets. *Journal of Machine Learning Research*, 2005, **6**: 363−392

11 Tsang I W, Kwok J T, Lai K T. Core vector regression for very large regression problems. In: Proceedings of the 22nd International Conference on Machine Learning. Bonn, Germany: ACM, 2005. 912−919

12 Shi J B, Malik J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, **22**(8): 888−905

13 Chen S C, Zhang D Q. Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure. *IEEE Transactions on System, Man, and Cybernetics*, 2004, **34**(4): 1907−1916

14 Bishop C M. *Neural Networks for Pattern Recognition*. Oxford: Oxford University Press, 1995

15 Tax D M J, Duin R P W. Support vector domain description. *Pattern Recognition Letters*, 1999, **20**(11-13): 1191−1199

16 Schölkopf B, Platt J C, Shawe-Taylor J C, Smola A J, Williamson R C. Estimating the support of a high-dimensional distribution. *Neural Computation*, 2001, **13**(7): 1443−1471

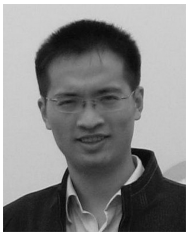
17 Chu C S, Tsang I W, Kwok J T. Scaling up support vector data description by using core-sets. In: Proceedings of International Joint Conference on Neural Networks. Budapest, Hungary: IEEE, 2004. 425−430

18 Tax D M J. Data description toolbox [Online], available: http://www-ict.ewi.tudelft.nl/˜davidt/dd_tools.html, September 6, 2007

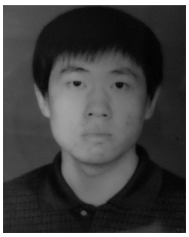
19 Martin D, Fowlkes C, Tal D M J, Malik J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measure ecological statistics. In: Proceedings of the 8th International Conference on Computer Vision. Vancouver, Canada: IEEE, 2001. 416−423

CHANG Liang Ph. D. candidate at Institute of Automation, Chinese Academy of Sciences. Her research interest covers SVM classification and clustering methods for large data sets in machine learning. Corresponding author of this paper.
E-mail: liang.chang@ia.ac.cn

DENG Xiao-Ming Ph. D. candidate at Institute of Automation, Chinese Academy of Sciences. His research interest covers calibration of omnidirectional sensors, feature extraction, and matching in omnidirectional images.
E-mail: xmdeng@nlpr.ia.ac.cn



ZHENG Sui-Wu Ph. D. candidate at Institute of Automation, Chinese Academy of Sciences. His research interest covers visual tracking, dynamic image processing, and robot vision.
E-mail: suiwu.zheng@ia.ac.cn



WANG Yong-Qing Ph. D. candidate at Institute of Automation, Chinese Academy of Sciences. His research interest covers machine learning and kernel methods. E-mail: yongqing.wang@ia.ac.cn