Semi-Supervised Clustering with Metric Learning: An Adaptive Kernel Method

Xuesong Yin[1,2] Songcan Chen[1*] Enliang Hu[1] Daoqiang Zhang[1]

[1]Department of Computer Science & Engineering, Nanjing University of Aeronautics & Astronautics, China

[2]Department of Computer Science & Technology, Zhejiang Radio & TV University, China

Abstract

Most existing representative works in semi-supervised clustering do not sufficiently solve the violation problem of pairwise constraints. On the other hand, traditional kernel methods for semi-supervised clustering not only face the problem of manually tuning the kernel parameters due to the fact that no sufficient supervision is provided, but also lack a measure that achieves better effectiveness of clustering**.** In this paper, we propose an adaptive **S**emi-supervised **C**lustering **K**ernel **M**ethod based on **M**etric learning (SCKMM) to mitigate the above problems. Specifically, we first construct an objective function from pairwise constraints to automatically estimate the parameter of the Gaussian kernel. Then, we use pairwise constraint-based K-means approach to solve the violation issue of constraints and to cluster the data. Furthermore, we introduce metric learning into nonlinear semi-supervised clustering to improve separability of the data for clustering. Finally, we perform clustering and metric learning simultaneously. Experimental results on a number of real-world data sets validate the effectiveness of the proposed method.

Keywords: Metric Learning; Pairwise Constraint; Closure Centroid；Semi-Supervised Clustering

1 Introduction

Semi-supervised clustering, which employs both supervised and unsupervised data for clustering, has received significant amount of attention in recent studies on data mining and machine learning communities. Compared with unsupervised clustering, semi-supervised clustering is aimed to mine and better understand the structure of unlabeled data and to more closely conform to the user’s preferences through those supervised data. As we know, in many application domains such as information retrieval, computational biology and image processing, labeling data is both

*Corresponding author: S.C. Chen, s.chen@nuaa.edu.cn; yinxs@nuaa.edu.cn expensive and difficult. Consequently, supervised data can just be obtained with a few labeled data or pairwise constraints which are used to aid unsupervised clustering learning [1-8].

Generally, existing methods for semi-supervised clustering can be grouped into two categories. The first category is the linear method including both metric-based and constraint-based approaches, which either aims to guide clustering process towards a more appropriate data partitioning by making use of pairwise instance constraints [2,6,16,17] or initializes cluster centroids by those labeled instances [4]. Specifically, the idea behind the linear constraint-based approach is to modify the objective function of existing unsupervised clustering so as to satisfy pairwise constraints. The metric-based approach learns a distance metric from pairwise constraints, and then utilizes an existing clustering algorithm to learn the similarity between data by using the learned distance metric [1,3,5,7,9,21]. In practice, many real-world applications may involve data along with nonlinear patterns, which may not be effectively dealt with by those linear methods. The second one is the nonlinear method or kernel method, which is recently presented and proved powerful. These methods map the data into the feature space implicitly through an mapping induced by a kernel function such that a cluster assignment is performed with the help of the nonlinear boundary in the original space [10,11,12].

In general, the nonlinear semi-supervised clustering approaches have been shown to outperform their linear competitors [10,11] for real world tasks. However, existing nonlinear approaches have the following two disadvantages: 1) they do not necessarily improve the separability of the data for clustering; 2) they cannot effectively solve the violation issue of pairwise constraints. In addition, the selection of the kernel parameters is left to manually tuning due to the fact that no sufficient supervision is provided [10,12,14]. In practice, it is well-known that the chosen values of the kernel parameters can largely affect the quality of the clustering results. Therefore, it is necessary to overcome the difficulties described above for both conforming to the user's preferences and improving the performance of semi-supervised clustering. The key challenge in improving separability of the data is how we can solve the violation issue of pairwise constraints such that the unlabeled data can still capture the available cluster information.

To this end, we present a new adaptive semi-supervised clustering kernel method based on metric learning, called SCKMM, which simultaneously performs clustering and metric learning by studying several important issues:

(1) What do we really gain by performing a metric learning for nonlinear semi-supervised clustering?
(2) How to overcome pairwise constraints violated in the clustering process?
(3) Is the performance of SCKMM comparable with linear and nonlinear semi-supervised clustering approaches in its iterative process alternating between metric learning and clustering?
(4) How to estimate the kernel parameter involved in the algorithm?

The main contributions of this paper are summarized as follows: (1) We introduce metric learning into the nonlinear semi-supervised clustering to improve the separability of the data for clustering. (2) For improving the quality of clustering, we make effective use of the cluster membership as the bridge connecting the nonlinear semi-supervised clustering and the metric learning. We perform kernel learning as a dynamic process that is adaptively adjusted by the clustering process. (3) We introduce pairwise constraint based K-means approach (PCBKM) to assign instances in the *must-link* constraints to one cluster and those in the *cannot-link* constraints to different clusters. (4) We construct an objective function in terms of the pairwise constraints to automatically estimate the parameter of Gaussian kernel which has been successfully applied in kernel learning tasks [11,12,14,15]. Finally, we evaluate the proposed theories and approaches through experiments on a collection of UCI data sets to compare several relevant algorithms under a unified experimental setup.

As is common for most semi-supervised clustering algorithms, in this paper, we consider supervision provided in the form of *must-link* constraints and *cannot-link* constraints which are defined as follows [7]:

Must-link constraints specify that two instances must belong to the same cluster.

Cannot-link constraints specify that two instances must belong to different clusters or cannot belong to the same cluster.

Indeed, such kinds of constraints have been wisely used in many fields of machine learning [20]. For instance, in the context of clustering GPS data for lane-finding [2] or grouping different actors in movie segmentation [1], the complete class information may not be available in these cases, but pairwise constraints can be extracted automatically or with minimal effort [8].

The rest of the paper is organized as follows. In Section 2, we briefly review some related works. Section 3 presents theoretical analysis and introduces the SCKMM algorithm. Section 4 presents the experimental results on synthetic data and real-world data sets. Finally, we conclude this paper in Section 5.

2 Related Work

For the linear method for semi-supervised clustering, Wagstaff et al. [2] proposed the constrained K-means algorithm by adjusting the cluster memberships to be consistent with the pairwise constraints. In [6,16,17], the authors presented probabilistic models for semi-supervised clustering where the pairwise constraints are incorporated into the clustering algorithms through the Bayesian priors. Basu et al. [4] proposed a seeded K-means which tries to get better initial cluster centroids from the labeled instances and restricts the clustering process to be consistent with the constraints. Xing et al. [5] combined the gradient descent method and the iterative projections together as a convex optimization to learn a Mahalanobis distance for the K-means clustering. Bar-Hillel et al. [3] proposed the relevant component analysis algorithm which learned a Mahalanobis distance by making use of the *must-link* constraints only. Yeung et al. [9] extended Bar-Hillel's algorithm by using both the *must-link* and the *cannot-link* constraints simultaneously. Bilenko et al. [7] integrated the constraints and metric learning in semi-supervised algorithm (MPCKM). This method is able to learn individual metrics for each cluster, which permits of different shapes. However, the violation of pairwise constraints is not effectively solved in the clustering process. Tang et al. [8] provided a way to improve the semi-supervised clustering for high-dimensional data by the constraint-guided feature projection (SCREEN) instead of the metric learning.

Another approach to the semi-supervised clustering is to cluster the data in terms of the kernel function. Kulis et al. [10] extended Basu's method [6] to a kernel-based semi-supervised clustering. Instead of adding penalty term for pairwise constraints violated, a reward was given for the satisfaction of the constraints in this method. Analogously, Yan et al. [11] presented an adaptive kernel learning method for semi-supervised clustering (ASSKKM) which kernelizes the objective function of Basu's method [6] in the input space. Different from Kulis' method that the setting of the kernel parameter was left to manually tuning, Yan's method optimized the parameter of a Gaussian Kernel iteratively during the clustering process. The above two nonlinear methods, however, do not sufficiently effectively solve the violation issue of pairwise constraints, which will be demonstrated via the experiments later. Tsang et al. [13] proposed the kernel relevant component analysis method, which generalized Bar-Hillel's method to nonlinear problems by a suitable kernel function. Yeung et al. [12] proposed a kernel approach for semi-supervised metric learning (KASML) and presented in detail two special cases of this kernel approach.

More recently, Chen et al. [19] presented a nonlinear adaptive distance metric learning for unsupervised clustering. Their method first maps the data to a high-dimensional space through a kernel function; then applies a linear projection to find a low-dimensional manifold; and finally performs clustering in the projected low-dimensional space. Kim et al. [22] proposed a distance preserving method which can exactly preserve Euclidean distances and cosine similarities between any pair of data points in the original dimensional space after reducing dimension, however, this method does not involve clustering learning and is more similar to principal component analysis.

3 The SCKMM Method

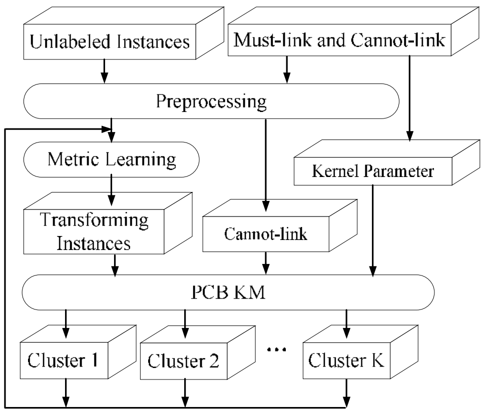


Fig. 1 The framework of the SCKMM method

The framework of the SCKMM algorithm is shown in Fig. 1. Given a set of instances and a set of supervision information in the form of *must-link* and *cannot-link* pairwise constraints, the SCKMM is composed of three steps. In the first step, a pre-processing method is exploited to estimate the optimal parameter of Gaussian kernel. Here, we present the PCBKM algorithm to cluster the data in the input space. In the second step, a metric learned by using the cluster memberships is applied to simultaneously minimize the distances between the same-cluster objects and maximize the distances between the different-cluster objects. Finally, we apply the PCBKM to generate the clustering results.

3.1 Nonlinear Adaptive Metric Learning

Let X denote a data set with n instances, $\{x_i\}_{i=1}^n \in \Re^d$. $X = [x_1, x_2, \cdots, x_n]$ denotes the data matrix. Let ML be a set of *must-link* constraints and CL be a set of *cannot-link* constraints.

Let $\phi : \Re^d \to \Re^D$ be a nonlinear mapping function from the input space (d dimensional) to a feature space (D dimensional) with $d < D$. $\phi(X) = [\phi(x_1), \phi(x_2), \cdots, \phi(x_n)]$ is the image matrix of the input space. $m_c^\phi = \sum_{i=1}^{n_c} \phi(x_i) / n_c$ represents the centroid of cluster L_c in the feature space. Let 1_c be n-dimensional vector: $1_{ci} = 1$, if x_i belongs to c-th cluster, and $1_{ci} = 0$ otherwise. I_c is the $n \times n$ diagonal matrix through diagonalizing 1_c. Motivated by [3,13], we define the covariance matrix of the instances in the feature space as follows:

$$\begin{aligned}\hat{S} &= \frac{1}{n}\sum_{c=1}^{C}\sum_{i=1}^{n_c}(\phi(x_i) - m_c^\phi)(\phi(x_i) - m_c^\phi)^T \\ &= \frac{1}{n}\sum_{c=1}^{C}\sum_{i=1}^{n_c}(\phi(x_i) - \frac{1}{n_c}\phi(X)1_c)(\phi(x_i) - \frac{1}{n_c}\phi(X)1_c)^T \\ &= \frac{1}{n}\sum_{c=1}^{C}\phi(X)(I_c - \frac{1}{n_c}1_c1_c{}^T)\phi(X)^T \\ &= \frac{1}{n}\phi(X)H\phi(X)^T \qquad (1)\end{aligned}$$

where $H = \sum_{c=1}^{C}(I_c - \frac{1}{n_c}1_c1_c{}^T)$. Clearly, H is symmetric, block diagonal and positive semi-definite.

Note that $\hat{S}$ will be singular when $n \le D$. Thus the regularization technique should be applied to improve the estimation in terms of

$$S = \lambda I + \frac{1}{n}\phi(X)H\phi(X)^T \qquad (2)$$

where I is the identity matrix of size D and $\lambda > 0$ is a regularization parameter.

By the Woodbury formula $(A + BC)^{-1} = A^{-1} - A^{-1}B(I + CA^{-1}B)^{-1}CA^{-1}$, we can obtain the inverse of S as follows:

$$S^{-1} = (\lambda I + \frac{1}{n}\phi(X)H\phi(X)^T)^{-1}$$

$$=\frac{1}{\lambda}I-\frac{1}{n\lambda^2}\phi(X)H(I+\frac{1}{n\lambda}\phi(X)^T\phi(X)H)^{-1}\phi(X)^T \tag{3}$$

With this new measure, we compute the whitening transformation in the feature space. Thus the instance and the clustering centroid can be respectively expressed as

$$\phi(y_i)=S^{-\frac{1}{2}}\phi(x_i)$$
$$u_c^{\phi}=\frac{1}{n_c}\sum_{i=1}^{n_c}S^{-\frac{1}{2}}\phi(x_i) \tag{4}$$

The kernel K-means (KKMNS) can be applied to assign the data $\{\phi(y_i)\}_{i=1}^{n}$ into the C disjoint clusters, which minimizes the distance between each instance and its corresponding clustering centroid:

$$\begin{aligned}d_M&=\left\|\phi(y_i)-u_c^{\phi}\right\|^2\\&=(S^{-\frac{1}{2}}\phi(x_i)-\frac{1}{n_c}\sum_{j=1}^{n_c}S^{-\frac{1}{2}}\phi(x_j))^T(S^{-\frac{1}{2}}\phi(x_i)-\frac{1}{n_c}\sum_{j=1}^{n_c}S^{-\frac{1}{2}}\phi(x_j))\\&=(\phi(x_i)-\frac{1}{n_c}\sum_{j=1}^{n_c}\phi(x_j))^TS^{-1}(\phi(x_i)-\frac{1}{n_c}\sum_{j=1}^{n_c}\phi(x_j))\\&=E_{ii}+F_{cc}-G_{ic}\end{aligned} \tag{5}$$

where

$$\begin{aligned}E_{ii}&=\phi(x_i)^TS^{-1}\phi(x_i)\\&=\phi(x_i)^T(\frac{1}{\lambda}I-\frac{1}{n\lambda^2}\phi(X)H(I+\frac{1}{n\lambda}\phi(X)^T\phi(X)H)^{-1}\phi(X)^T)\phi(x_i)\\F_{cc}&=\frac{1}{n_c^2}\sum_{j=1}^{n_c}\phi(x_j)^TS^{-1}\sum_{k=1}^{n_c}\phi(x_k)\\&=\frac{1}{n_c^2}\sum_{j=1}^{n_c}\phi(x_j)^T(\frac{1}{\lambda}I-\frac{1}{n\lambda^2}\phi(X)H(I+\frac{1}{n\lambda}\phi(X)^T\phi(X)H)^{-1}\phi(X)^T)\sum_{k=1}^{n_c}\phi(x_k)\\G_{ic}&=\frac{2}{n_c}\phi(x_i)^TS^{-1}\sum_{k=1}^{n_c}\phi(x_k)\\&=\frac{2}{n_c}\phi(x_i)^T(\frac{1}{\lambda}I-\frac{1}{n\lambda^2}\phi(X)H(I+\frac{1}{n\lambda}\phi(X)^T\phi(X)H)^{-1}\phi(X)^T)\sum_{k=1}^{n_c}\phi(x_k)\end{aligned} \tag{6}$$

Obviously, d_M in Eq. (5) defines a squared Mahalanobis distance to replace the traditional

Euclidean distance in the mapped space:

$$d_M = \left\| \phi(x_i) - m_c^{\phi} \right\|_{S^{-1}}^2 \tag{7}$$

Since the data representation in the feature space may not specify a space where the clusters are sufficiently separated, modifying the distance metric warps the space to keep the data points in each cluster close together, while ensuring that those from different clusters remain far apart.

The nonlinear mapping can be implicitly specified by a kernel function *k* through computing the inner product of the images of each data pair in the feature space, that is

$$k(x_i, x_j) = \phi(x_i)^T \cdot \phi(x_j)$$

Consequently, (6) can be rewritten as follows:

$$E_{ii} = \frac{1}{\lambda} k(x_i, x_i) - k_i^T \left(\frac{1}{n\lambda^2} H (I + \frac{1}{n\lambda} KH)^{-1} \right) k_i$$

$$F_{cc} = \frac{1}{n_c^2} \left(\frac{1}{\lambda} \sum_{i,j=1}^{n_c} k(x_i, x_j) - (\sum_{i=1}^{n_c} k_i^T)(\frac{1}{n\lambda^2} H(I + \frac{1}{n\lambda} KH)^{-1})(\sum_{j=1}^{n_c} k_j) \right)$$

$$G_{ic} = \frac{2}{n_c} \left(\frac{1}{\lambda} \sum_{j=1}^{n_c} k(x_i, x_j) - k_i^T (\frac{1}{n\lambda^2} H(I + \frac{1}{n\lambda} KH)^{-1}) \sum_{l=1}^{n_c} k_l \right) \tag{8}$$

where $k_i = [k(x_1, x_i), \cdots, k(x_n, x_i)]^T$ is a vector of size n by 1, and $K = [k(x_i, x_j)]_{i,j=1}^n$ is an *n*-by-*n* Gram matrix.

3.2 The Estimation of Gaussian Kernel Parameter

Given *ML* and *CL*, Yan et al. [18] utilize the information to get a convex set of kernel functions. However, this method fails to obtain the parameters of the hybrid kernel. The objective in this paper is to try to find such optimal parameter value for the kernel function to guide the clustering process by making effective use of both *ML* and *CL*. Hence, similar to [18], we define the following objective function for optimizing the parameter:

$$F = \sum_{(x_i, x_j) \in CL} \left\| \phi(x_i) - \phi(x_j) \right\|^2 - \sum_{(x_i, x_j) \in ML} \left\| \phi(x_i) - \phi(x_j) \right\|^2 \tag{9}$$

Since the Gaussian kernel function $k(x_i, x_j) = \exp(-\left\| x_i - x_j \right\|^2 / (2\sigma^2))$ has excellent learning properties [10,11,14,15], we utilize it to derive our algorithm. Now, using the kernel trick and throwing off the constant terms which do not affect the final value, we obtain the following objective:

$$F_k = \sum_{(x_i,x_j)\in ML} k(x_i,x_j) - \sum_{(x_i,x_j)\in CL} k(x_i,x_j) \tag{10}$$

We utilize the gradient ascent algorithm to solve the parameter σ of the Gaussian kernel. To this end, we maximize F_k with respect to the kernel parameter. The partial derivative of F_k is computed as follows:

$$\frac{\partial F_k}{\partial \sigma} = \sum_{(x_i,x_j)\in ML} \frac{\partial k(x_i,x_j)}{\partial \sigma} - \sum_{(x_i,x_j)\in CL} \frac{\partial k(x_i,x_j)}{\partial \sigma} \tag{11}$$

where $\frac{\partial k(x_i,x_j)}{\partial \sigma} = \exp(-\frac{\left\| x_i - x_j \right\|^2}{2\sigma^2}) \frac{\left\| x_i - x_j \right\|^2}{\sigma^3}$.

Thus, we have

$$\sigma^{(new)} = \sigma^{(old)} + \rho \frac{\partial F_k}{\partial \sigma} \tag{12}$$

where ρ is a scalar step length parameter optimized via a line-search method.

Using the gradient ascent algorithm can ensure that a locally optimal parameter σ of the kernel function can be obtained [11,18].

3.3 The PCBKM Algorithm

The MPCKM algorithm tries to find the solution that is as consistent with the given pairwise constraints as possible. It combines both the cost incurred by any violating pairwise constraint and metric learning into the objective function to solve the violation issue of constraints. The objective function of the MPCKM is defined as

$$J = \sum_{x_i \in X} (\left\| x_i - u_{l_i} \right\|^2_{A_{l_i}} - \log(\det(A_{l_i})) + \sum_{(x_i,x_j)\in ML} \omega_{ij} 1[l_i \neq l_j] + \sum_{(x_i,x_j)\in CL} \varpi_{ij} 1[l_i = l_j] \tag{13}$$

where the penalty factors ω_{ij} for violating *must-link* constraints and ϖ_{ij} for violating *cannot-link* constraints are respectively expressed as follows:

$$\omega_{ij} = \frac{1}{2} \left\| x_i - x_j \right\|^2_{A_{l_i}} + \frac{1}{2} \left\| x_i - x_j \right\|^2_{A_{l_j}} \tag{14}$$

$$\varpi_{ij} = \left\| x_{l_i}{'} - x_{l_i}{''} \right\|^2_{A_{l_i}} - \left\| x_i - x_j \right\|^2_{A_{l_i}} \tag{15}$$

The MPCKM indicates that the penalty cost for violating a *must-link* constraint between the distant points should be higher than that between nearby points, and also the penalty cost for violating a cannot-link constraint between two points that are nearby according to the current metric should be higher than that for two distant points. Unfortunately, the above method can not enough effectively solve the violation issue of pairwise constraints.

Similar to the MPCKM, the ASSKKM adds the cost incurred by any violating pairwise constraint to the objective in the feature space:

$$J=\sum_{c=1}^{C}\sum_{i=1}^{n_c}\left\|\phi(x_i)-m_c^{\phi}\right\|^2+\sum_{(x_i,x_j)\in ML}\omega_{ij}1[l_i\neq l_j]+\sum_{(x_i,x_j)\in CL}\varpi_{ij}1[l_i=l_j] \tag{16}$$

Analogously, this method can not completely solve the violation issue of pairwise constraints. The similar case is shown in [5,6,10]. To solve the difficulty described above, we introduce the PCBKM algorithm. In order to arrive at the goal, we incorporate the *must-link* constraints into the corresponding closures to form the new *cannot-link* constraints as the initialization motivated by [8].

Definition 1. Closure. Given three instances a_1 , a_2 and a_3 . If $(a_1,a_2)\in ML$ and $(a_1,a_3)\in ML$, then $(a_2,a_3)\in ML$ as well in terms of transitive property of the *must-link* constraints. The set $\{a_1,a_2,a_3\}$ is said to be a closure.

It is well-known that all the instances in a closure should be assigned to the same cluster according to the definition of the *must-link* constraint. Specifically, as shown in Fig. 2, the *must-link* constraint is represented in the solid line and the *cannot-link* constraint is described in the dashed line. Also, the white nodes represent the original instances, and the closure centroids consisting of the mean of the instances in each closure are shown in the black nodes. Sets $\{a_1,a_2,a_3\}$ and $\{b_1,b_2,b_3,b_4,b_5\}$ represent different closures constructed by incorporating *must-link* constraints. We make use of the closure centroids to replace the closures such that the new *cannot-link* constraint is set up as shown in Fig. 2, where a and b replace closures $\{a_1,a_2,a_3\}$ and $\{b_1,b_2,b_3,b_4,b_5\}$ respectively and form a new *cannot-link* constraint. Especially, one instance that is not in any constraint forms a closure. After the corresponding closures are replaced by the closure centroids and the new *cannot-link* constraints are constructed, the original instances X are reduced to $X'=[x_1',x_2',\cdots,x_{n'}']$ ($n'<n$, where n' is the number of the instances after *must-link* constraints are merged) and the new *cannot-link*

constraints are changed to $CL' = \{(x_i', x_j')\}$.

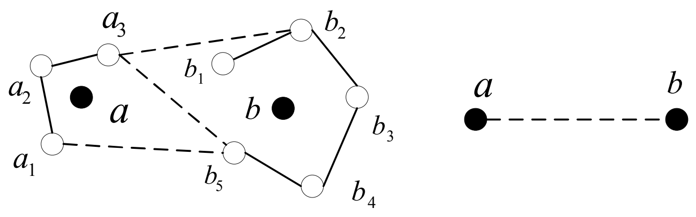


Fig. 2 An illustration of incorporating *must-link* constraints into a new *cannot-link* constraint

One noticeable issue is that all the constraints above are supposed to be consistent in any closures. In other words, they all comply with the transitivity of *must-link* constraints. However, in practice, such an ideal case does not always hold when pairwise constraints provided are inconsistent, and then some conflicts or contradictions among them will occur. For example, if an expert provided the following three pairs of constraints (*a1*, *a2*), (*a1*, *a3*) and (*a2*, *a3*), in which (*a1*, *a2*) and (*a1*, *a3*) both belong to *must-link* while (*a2*, *a3*) belongs to *cannot-link*, then they three yield a conflict because according to the transitivity of *must-link* constraints, *a2* and *a3* should belong to the same cluster and while in terms of the *cannot-link* property, they should *not* be in the same cluster. As a result, such conflicts naturally mislead the clustering process so that the clustering result is (heavily) biased. In order to avoid such unfavorable effects, before clustering, we can first easily perform a checking step to examine whether the transitivity is violated and then remove them directly if true so as to not only guarantee no conflicts among constraints but improve the final quality of clustering as well.

Based on the above analysis, the PCBKM algorithm is presented in the following Algorithm 1.

```
Algorithm 1: PCBKM
Input: $X, K, CL'$
Output: The $C$ partitions of instances
  1. Initialize the $C$ centroids of clusters;
  2. Check the conflicts, i.e., examine whether a cannot-link constraint is in any closure.
     If this occurs, then those constraints involved are removed to guarantee no conflicts;
  3. Repeat the following steps until convergence
     (a) For each instance $x_i'$ which does not involve in any cannot-link constraint, find
         the closest centroid $l_c = \arg\min_c \|\phi(x_i) - m_c^{\phi}\|_{S^{-1}}^2$;
     (b) For each $(x_i', x_j') \in CL'$, find two different centroids $m_i^{\phi}$ and $m_j^{\phi}$ which
```

```
        minimize $\left\|\phi(x_i') - m_i^\phi\right\|_{S^{-1}}^2 + \left\|\phi(x_j') - m_j^\phi\right\|_{S^{-1}}^2$;
    (c) Update all the clusters;
```

Similar to most methods for the distance metric learning, the computational time of the PCBKM algorithm mainly spends on solving the inverse of the matrix. Thus the computational complexity of PCBKM is $O(Cn'tn^3)$, where C is the clustering number, and t the iterative time of clustering. Due to less clustering number and iterative time, the computational time of the PCBKM algorithm approximates $O(n'n^3)$.

Proposition 1. The violation issue of pairwise constraints can be solved in the PCBKM algorithm.

Proof. First, we prove that *must-link* constraints are not violated in the clustering process using the PCBKM algorithm.

Suppose $A_i = \{x_1, x_2, \cdots, x_p\}$ be the *i*-th closure which is composed of incorporating *must-link* constraints, and $\bar{x}_i$ is the closure centroid of A_i. Let $L = \{L_1, L_2, \cdots, L_C\}$ be a set of the C clusters, and $M = \left\{m_1^\phi, m_2^\phi, \cdots, m_C^\phi\right\}$ a set of the respective cluster centroids.

In terms of definition of the closure, $\exists\ m_i^\phi \in M$, such that $m_i^\phi = \arg\min\limits_{m_s^\phi \in M}(\left\|\phi(\bar{x}_i) - m_s^\phi\right\|_{S^{-1}}^2)$. Thus we obtain $\bar{x}_i \in L_i$.

Since now using $\bar{x}_i$ to replaces A_i, and $\bar{x}_i \in L_i$, thus we get $A_i \subset L_i$.

Further, $\forall x_i \in A_i$, having $x_i \in L_i \ \ , i = 1, 2, \cdots, p$.

The above results show all the instances in any closure composed of *must-link* constraints are assigned to the same cluster. Thus, the *must-link* constraints are not violated in the PCBKM.

Second, we prove that the violation issue of *cannot-link* constraints is solved in the clustering process using the PCBKM.

Let $A_i = \{x_1, x_2, \cdots, x_p\}$ and $A_j = \left\{x_1, x_2, \cdots, x_q\right\}$ be arbitrary two closures whose centroids are $\bar{x}_i$ and $\bar{x}_j$ respectively, and $(\bar{x}_i, \bar{x}_j) \in CL'$. By their definitions, there are $m_i^\phi \in M$ and $m_j^\phi \in M$ ($i \neq j$) such that $(m_i^\phi, m_j^\phi) = \arg\min\limits_{m_s^\phi, m_t^\phi \in M}(\left\|\phi(\bar{x}_i) - m_s^\phi\right\|_{S^{-1}}^2 + \left\|\phi(\bar{x}_j) - m_t^\phi\right\|_{S^{-1}}^2)$

As a result, we have $\overline{x}_i \in L_i$ and $\overline{x}_j \in L_j$. Further, we obtain $A_i \subset L_i$, $A_j \subset L_j$.

Thus, $\forall x_i \in A_i$ and $\forall x_j \in A_j$, we have $x_i \in L_i$ ($i = 1, 2, \cdots, p$) and $x_j \in L_j$ ($j = 1, 2, \cdots, q$) respectively. ■

It is known from Proposition 1 that the PCBKM algorithm does not cluster the instances of each closure, but clusters the centroid of each closure. After the closure centroid is confirmed, all the instances inside the closure are assigned to the corresponding clusters. Consequently, the PCBKM ensures that the instances in the *must-link* constraints are allocated to the same cluster, and those in the *cannot-link* constraints are assigned to different clusters. Therefore, it can overcome the violation issue of pairwise constraints, which entangled most existing methods for semi-supervised clustering.

3.4 The SCKMM Method

Based on the discussion above, we develop an iterative algorithm, called SCKMM, for nonlinear semi-supervised clustering. The corresponding algorithm of the SCKMM is presented in Algorithm 2.

Algorithm 2: SCKMM

```
Input: $X$, $C$, $ML$, $CL$
Output: The $C$ disjoint partitions
  1. Preprocess
     (a) Estimate $\sigma$ as in Section 3.2;
     (b) Compute $C$ clusters by applying the PCBKM on the original instance space;
  2. Repeat until convergence
     (a) Update $H$ as in Section 3.1;
     (b) Apply the PCBKM algorithm to compute $C$ clusters;
  3. Return the $C$ partitions.
```

The convergence of the SCKMM can be guaranteed like most metric learning algorithms [5,7,8]. In its implementation, we observe from our experiments that the SCKMM typically converges within 3 to 4 iterations.

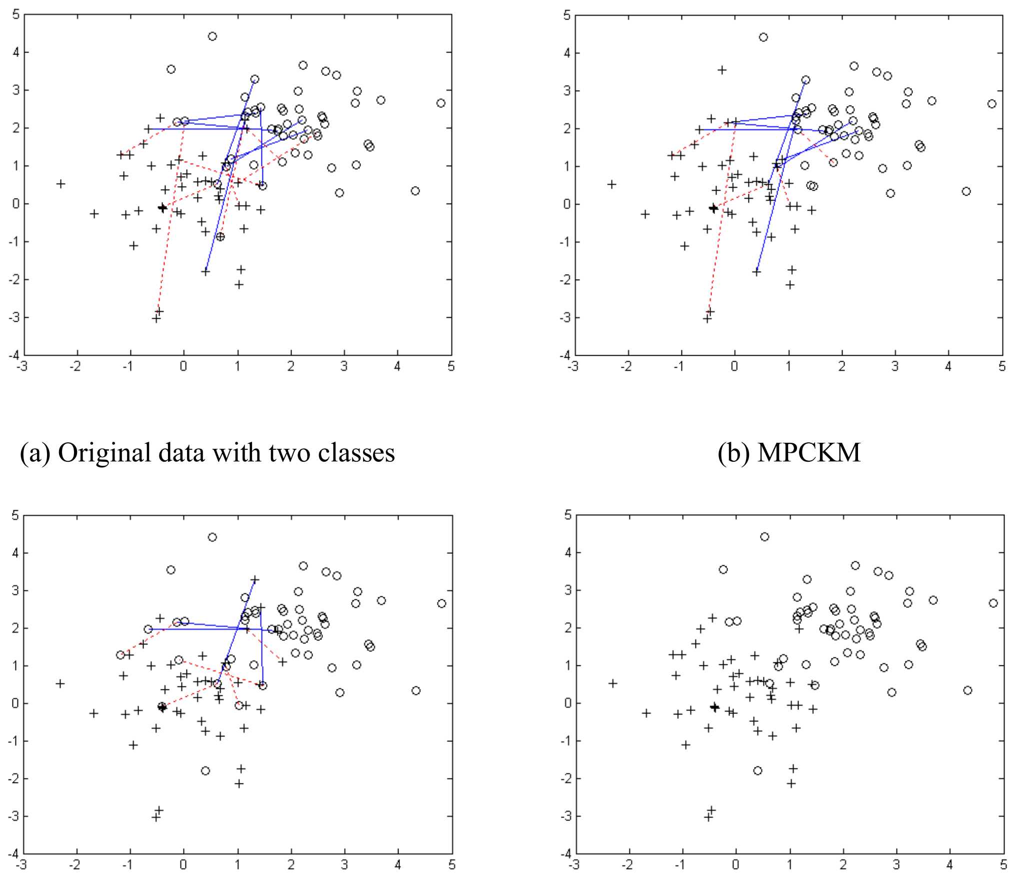
4 Experiments

We are now in a position to empirically evaluate the performance of the SCKMM in comparison with the representative algorithms and verify its effectiveness in solving the violation issue of pairwise constraints. Also, we conduct a sensitivity study on the number of constraints and on noisy pairwise constraints. The study will help us better understand the proposed algorithm, and delineate new challenge and research issues.

4.1 The Violation Issue of Pairwise Constraints

For verifying the effectiveness in solving the violation issue of pairwise constraints, in this section, we describe some experiments respectively based on both synthetic and real-world data to compare our PCBKM with the MPCKM and ASSKKM.

4.1.1 Experiments on Synthetic Data



(a) Original data with two classes

(b) MPCKM

(c) ASSKKM

(d) PCBKM

Fig. 3 Pairwise constraints violated on synthetic data

We perform some experiments on the synthetic data as shown in Fig. 3(a), where the point pairs in *ML* are connected by the solid lines, while the point pairs in *CL* are connected by the dashed lines. The clustering results using MPCKM and ASSKKM are respectively displayed in Figs. 3(b) and (c), where the point pairs connected by the solid lines are assigned to different clusters, which violate the *must-link* constraints, and those connected by the dashed lines are allocated to the same cluster, which violate the *cannot-link* constraints.

Obviously, MPCKM, a linear semi-supervised clustering algorithm, cannot give satisfactory result due to many pairwise constraints violated. ASSKKM, a nonlinear semi-supervised clustering algorithm, has less number of constraints violated than MPCKM, but it does not solve the issue completely yet. The cluster memberships obtained by the PCBKM are shown in Fig. 3(d), where no pairwise constraints are violated.

4.1.2 Experiments on UCI Data Sets

We further perform experiments on three real-world data sets, i.e. Glass, Protein and Digits. The comparison results are summarized in Table 1. For the glass data set, we randomly select 93 and 97 pairs as *must-link* and *cannot-link* constraints respectively. After the data are clustered by the MPCKM algorithm, the numbers of violating *must-link* and *cannot-link* constraints are 51 and 16 respectively. In the ASSKKM, the violated numbers are 42 and 14 respectively. However, no constraints are violated in the PCBKM. Similar observations can be obtained from the experiments on protein and digits. Thus, the PCBKM algorithm effectively solves the violation issue of constraints occurred in both MPCKM and ASSKKM.

Table 1: The number of pairwise constraints violated on UCI data sets

Dataset	*ML*	*CL*	*MPCKM*		*ASSKKM*		*PCBKM*	
			ML	*CL*	*ML*	*CL*	*ML*	*CL*
glass	93	97	51	16	42	14	0	0
protein	87	81	22	5	3	0	0	0
digits	99	101	11	5	27	9	0	0

4.2 Comparison with Representative Algorithms

In this section, we compare SCKMM with some representative algorithms on real-world data. Directly, we measure the effectiveness of the SCKMM by how much it can improve the clustering results in semi-supervised clustering tasks with the *must-link* and *cannot-link* constraints.

4.2.1 Experiment Setup

To evaluate the performance of the SCKMM, we compare the proposed algorithm with five representative clustering algorithms as follows:

(1) KKMNS, a kernel K-means algorithm;

(2) MPCKM [8], a linear semi-supervised clustering algorithm which involves both constraints and metric learning;

(3) ASSKKM [12], a nonlinear semi-supervised clustering algorithm which integrates the constraints into the KKMNS and generally outperforms the algorithms presented in [10,14];

(4) KASML [13], a nonlinear semi-supervised metric learning algorithm;

(5) SCREEN [9], a semi-supervised clustering algorithm which integrates constraint-guided feature projection into semi-supervised clustering.

Previous studies [9,12,13] have showed that the above algorithms delivered the state-of-the-art performance compared with other semi-supervised clustering such as the constrained K-means.

We test the semi-supervised clustering algorithms on twelve UCI data sets: glass, iris, vowel, cmc, wine, zoo, wdbc, segmentation, protein, digits, letter (a-d) and pendigits. The descriptions of these data sets are summarized in Table 2.

Table 2: Summary of UCI data sets

Data set	Dimension	Instance	Class
Glass	9	214	6
Iris	4	150	3
Vowel	10	360	4
CMC	9	1473	3
Wine	13	178	3
Zoo	18	101	7
WDBC	30	569	2
Segmentation	20	420	7
Protein	20	115	6
Digits	16	332	6
Letter(a-d)	16	3096	4
Pendigits	16	7000	10

We use two standard measures to evaluate the clustering performance. The first measure is Rand Index [3,5,11,12], which estimates the quality of the clustering results with respect to the ground truth of the data. More formally, the accuracy measure can be written as:

$$Accuracy = (N_S + N_D) / N_P \tag{17}$$

where N_S is the number of pattern pairs that are assigned to the same cluster in both the resultant partition and the ground truth, and N_D is the number of pattern pairs that are assigned to different clusters in both the resultant partition and the ground truth. The Rand Index is defined as the ratio of $(N_S + N_D)$ to N_P, where $N_P = n(n-1)/2$, the total number of pattern pairs.

Another evaluation measure is Normalized Mutual Information (NMI) [14,19] defined as follows:

$$NMI = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} \tag{18}$$

where *X* and *Y* denote the random variables of cluster memberships from the ground truth and the output of clustering algorithm, respectively. $I(X,Y)$ is the mutual information between *X* and *Y*. $H(X)$ and $H(Y)$ are the entropies of *X* and *Y* respectively.

Finally, all experiments were conducted on a PENTIUM DUAL 1.6G PC with 1.0 GB RAM. For each data set, we repeated experiments for 20 trials. In order to make a thorough comparison, we randomly generated different pairwise constraints from 50 to 500 for each data set and conducted the six algorithms using the same constraints in each comparison. We tested the performance on the whole dataset. Also, the final average result is reported from the 20 trials.

4.2.2 Experimental Results

We use the KKMNS algorithm as the baseline clustering. The regularization parameter λ is set to 0.0001. Like [8,9,12], the cluster number *C* is simply assigned to the actual number of classes in the data for all the six algorithms. Table 3 and Fig. 4 show the clustering results based on our algorithm and the five popular algorithms mentioned in Section 4.2.1 and evaluated by NMI and Rand Index respectively. Obviously, we can observe from Table 3 that SCKMM outperforms all the other five clustering methods on most data sets. The similar observation can also be obtained from Fig. 4. Moreover, it is clear that on most data sets, the clustering performances of all the algorithms are constantly improved with the increase of the number of pairwise constraints. However, as seen from Fig. 4, the learning curves of different algorithms display different performance behaviors. The detailed analysis of the experiments is listed as follows.

Table 3: NMI results with 500 pairwise constraints

Dataset	KKMNS	MPCKM	ASSKKM	KASML	SCREEN	SCKMM
CMC	0.2441	0. 3639	0.3965	0.4140	0.3701	**0.5182**
Digits	0.6835	0.8409	0.8223	0.7504	0.7797	**0.8652**
Glass	0.4708	0.5147	0.6292	0.6687	0.5329	**0.7036**
Iris	0.6874	0.8642	0.9071	0.8539	0.8365	**0.9702**
Protein	0.6952	0.8550	0.9036	0.8764	0.8342	**0.9921**
Segmentation	0.6647	0.8990	0.8630	0.8559	0.9163	**0.9785**

Vowel	0.6489	0.8098	**0.8752**	0.8496	0.8336	0.8431
WDBC	0.3204	0.4147	0.4878	0.4426	**0.5294**	0.4706
Wine	0.6331	0.7679	0.7839	0.7937	**0.8265**	0.8183
Zoo	0.6709	0.8903	0.8543	0.8730	0.8849	**0.9542**
Letter(a-d)	0.4493	0.5277	0.5263	0.5171	0.4968	**0.5390**
Pendigits	0.6005	0.7374	0.7426	0.7055	0.7269	**0.7781**

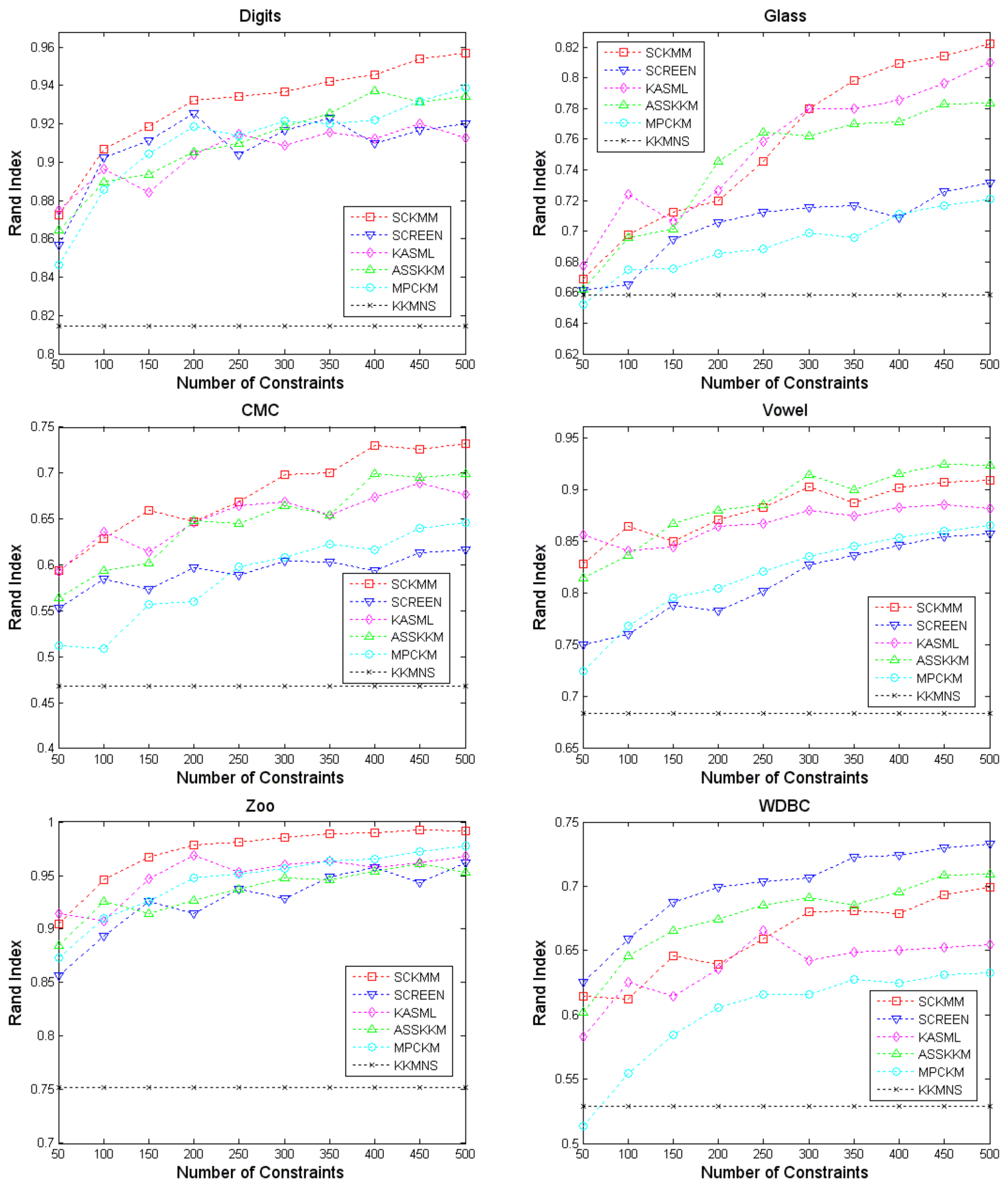
Firstly, we observe that the SCKMM achieves the best performance on the nine data sets in Table 2 and also comparable performance to both ASSKKM and SCREEN on the rest of twelve data sets, such as vowel, wdbc and wine. This can be attributed to the fact that the proposed method is considerably effective in utilizing the pairwise constraints to improve clustering performance. Thus, the SCKMM algorithm provides an appealing clustering performance.

Secondly, as can be seen from results, the SCREEN algorithm achieves the best performance on wdbc and wine data sets. This observation can be explained by the fact that both the wdbc and wine respectively have an attribute with much larger values than other attributes. The performance of clustering is sufficiently improved, when such data are normalized and projected to the low dimensional space, which is also exactly consistent with what the SCREEN performs. However, the experimental results on the other data sets show that the SCREEN performs relatively worse than the ASSKKM.

Thirdly, because the ASSKKM algorithm neither performs metric learning nor completely solves the violation issue of pairwise constraints, its performance gain is lower than SCKMM's. The KASML only uses the *must-link* constraints to learn a metric for clustering such that it does not considerably improve the performance of clustering compared with both SCKMM and ASSKKM as observed in the experiments. In practice, the *cannot-link* constraints are more important than the *must-link* constraints for clustering the data [8].

Fourthly, we can observe from Fig. 4 that the performance of the SCKMM is always comparable to, or better than those of the other algorithms when there are just a small number of constraints in the data sets. Also, the performance of the SCKMM is significantly improved as the number of available constraints increases. This is mainly due to the fact that the learned metric can easily enforce points from different clusters far apart and those from the same cluster closely.

Finally, the learning curves of the KASML and SCREEN in Fig. 4 do not always display the ascending trend, when the initial constraints are provided (but via many starts, the ascent trend can be increased). In other words, they do not always improve the clustering performance. The similar behavior is also found in the ASSKKM, although its learning curves display well. In contrast, the clustering performance improvement brought by the SCKMM is substantially more stable and consistent, across different data sets. This suggests that the SCKMM can be adaptive to both clustering and metric learning: it takes the feedback from the result of clustering to learn a metric, and then clusters the data using the learned metric, finally alternating such a process is till convergence.



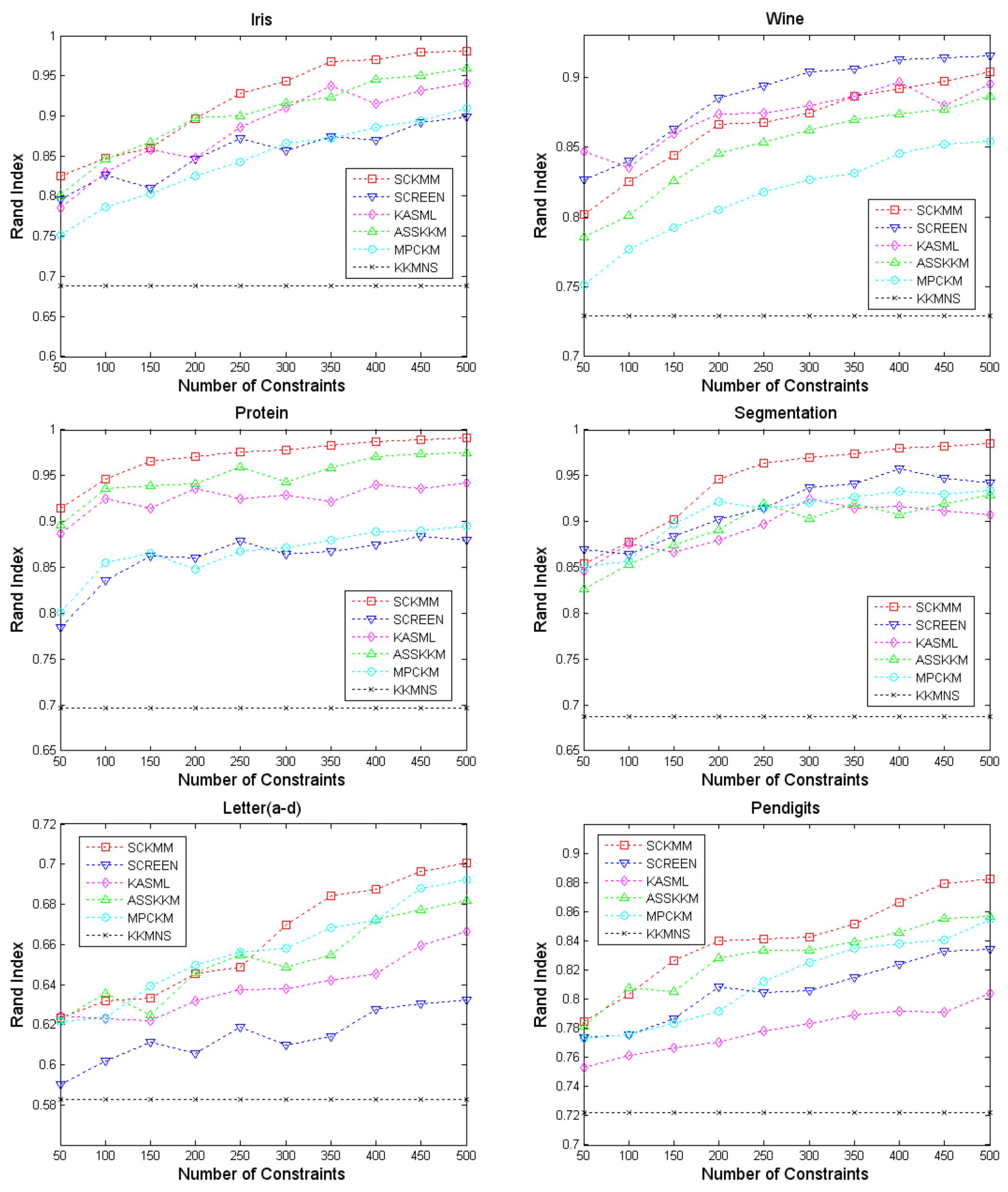


Fig. 4 Clustering performance on twelve UCI data sets with different numbers of constraints

4.2.3 Regularization Parameter λ

As discussed in Section 3.1, the addition of a regularization parameter λ to the covariance matrix is for improving its estimation as in [19]. In the subsection, we study its effect on the clustering performance of the SCKMM with the following experiment. In the experiment, we try a series of different λ values from 10^{-8} to 10^{4} with multiplicity of 10, and obtain their corresponding results as shown in Fig. 5. We can observe that the SCKMM is not much so sensitive to the choice of λ in the range. On most data sets, the performance of the SCKMM varies slightly with the changing of λ value. In particular, on the Protein dataset, its performance is almost unchanged with respect to λ value in [$10^{-8}, 10^{4}$]. Thus, the experimental results imply that the performance of the SCKMM is substantially relatively stable. Furthermore, the relatively better result can be obtained when the λ value lies in the range of [$10^{-6}, 10^{-1}$] in most cases.

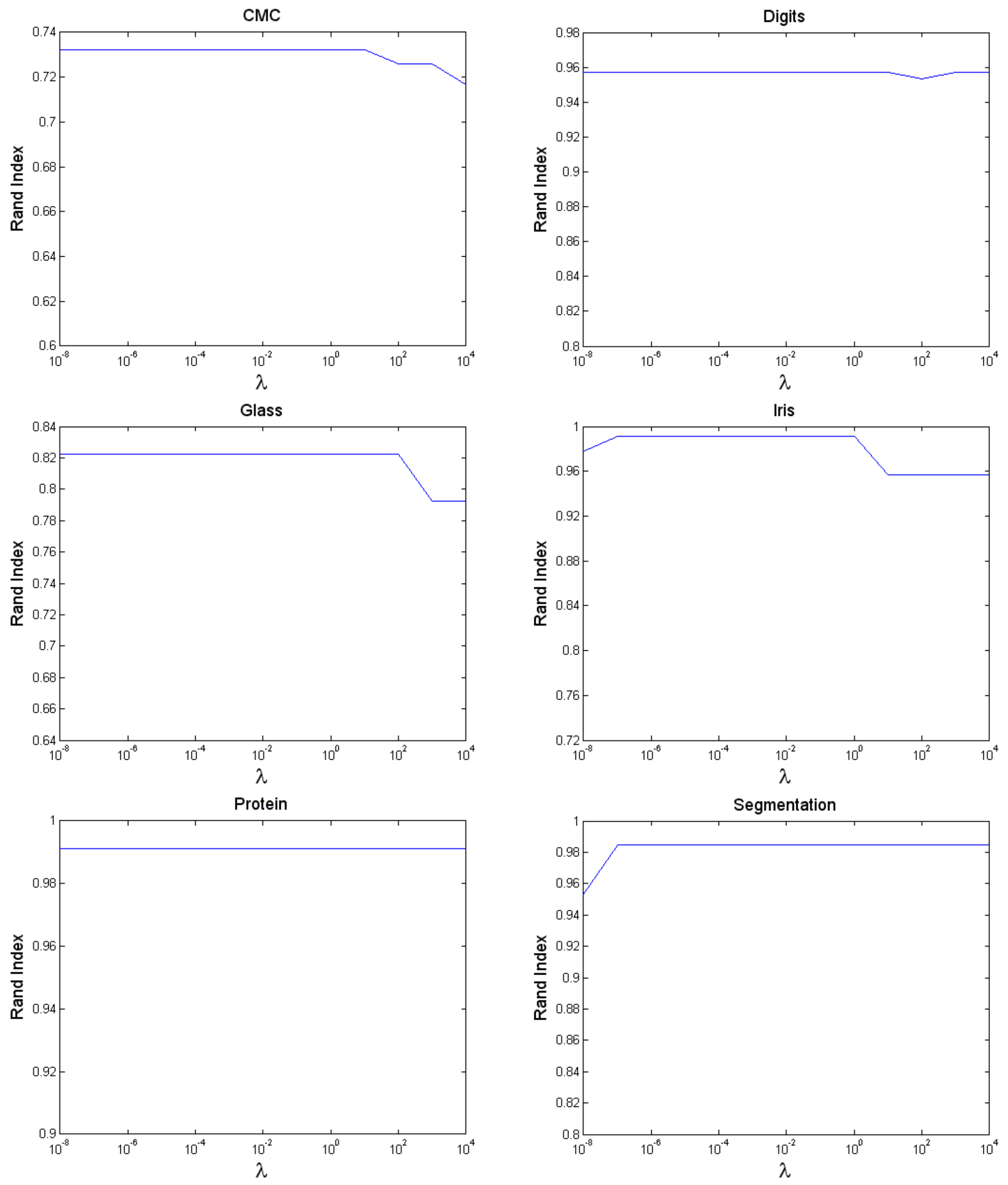
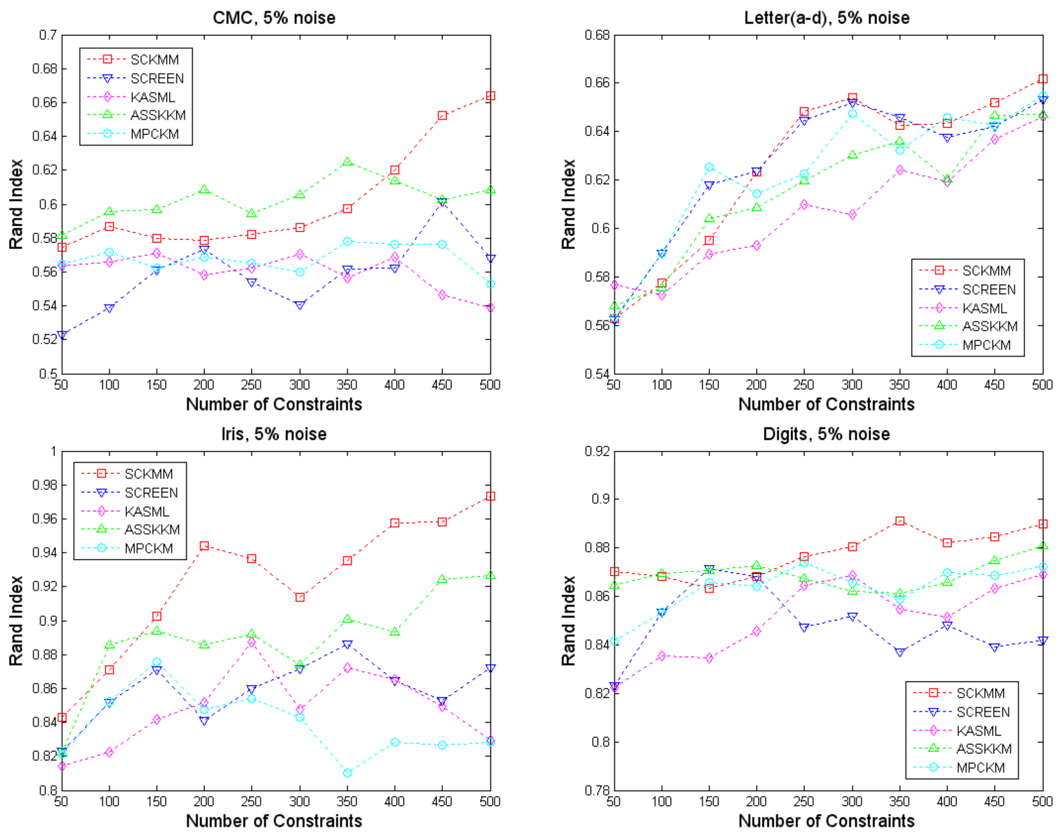


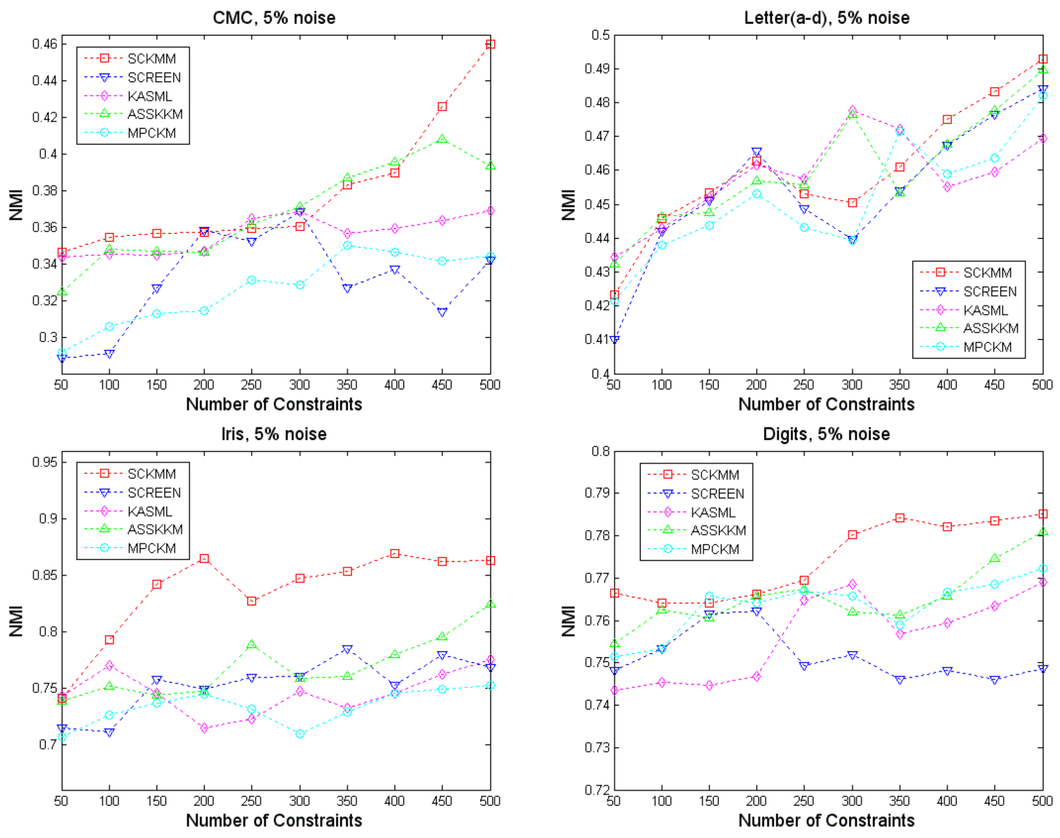
Fig. 5 The clustering performance of SCKMM with different λ values

4.2.4 Influence of Noisy Pairwise Constraints on Clustering Performance

In the previous works [1-5,7-13,18], most of semi-supervised clustering algorithms base on the same assumption that given pairwise constraints are clean or not noisy, implying that the constraints the experts provide are completely correct. However, in real world, such a case does not always hold, i.e., the pairwise constraints provided are sometimes partially noisy. Hence, in this subsection, we aim to examine influence of noisy constraints on clustering performance (in fact, to our best knowledge, in semi-supervised clustering, such a problem has hardly been mentioned). For this purpose, we conduct an experiment respectively using 5% and 10% of all the given constraints as noisy pairwise constraints. In order to generate the noisy pairwise constraints, we randomly flip 5% (10%) of the pairwise constraints to the opposite constraints to form corresponding noisy constraints. For fair comparison with our algorithm, before performing the other algorithms mentioned above, we can remove those conflicting noisy constraints by checking if they violate the transitive property in *must-link* set as described in Section 3.3. In fact, in real world, pre-checking such conflicts is not only feasible but necessary as well. However, it must be pointed out that we generally have no feasible way to check those noisy paired constraints not violating the transitivity. For example, (*a1*, *a2*) must originally belong to *must-link* but now is mis-designated as a *cannot-link* by an expert and while it does not give rise to any conflict with all other constraints, as a result, it is they that become one of main sources for degrading clustering performance in such noisy scenario. Therefore, our experiment is based on the above process and the corresponding results are respectively demonstrated in Figs. 6 and 7.

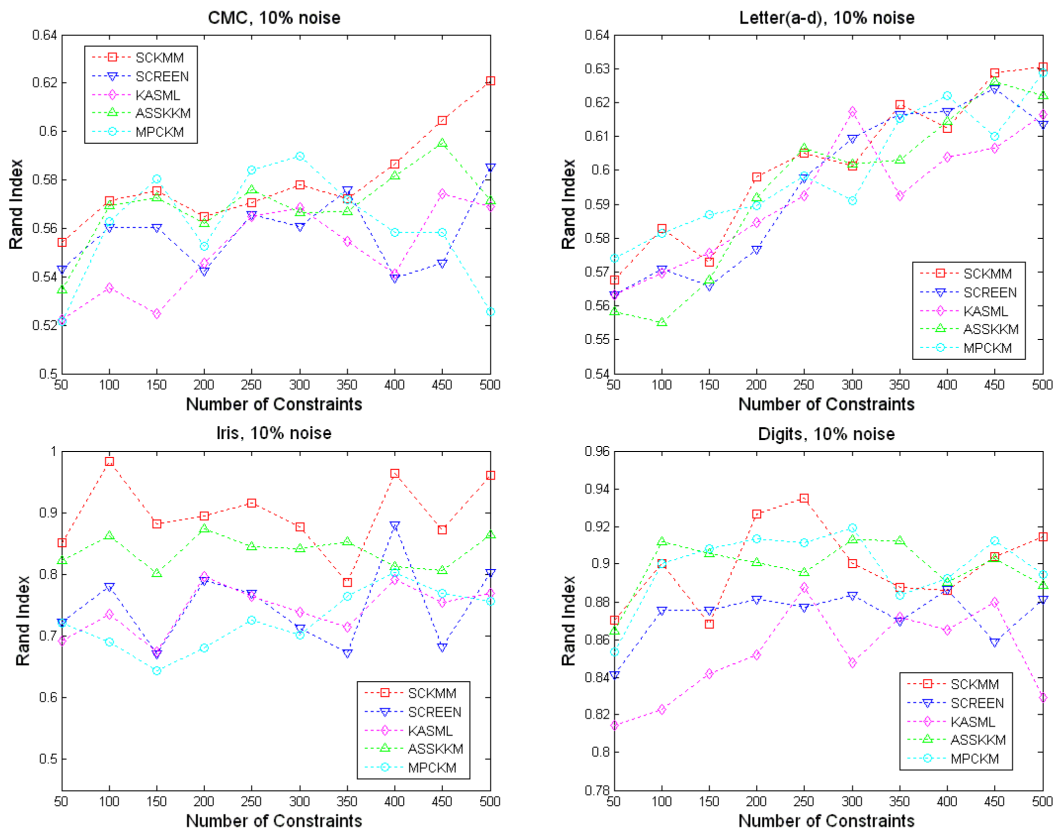


(a) Comparison of clustering performance evaluated by Rand Index

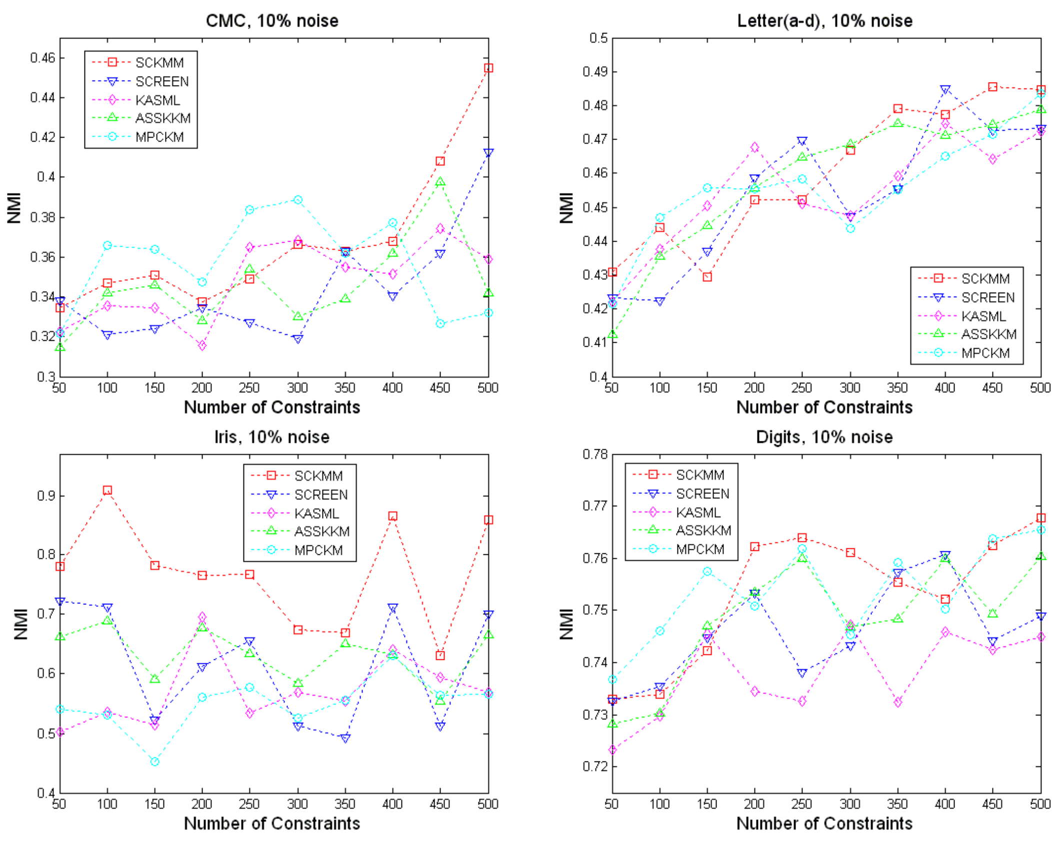


(b) Comparison of clustering performance evaluated by NMI

Fig. 6 Comparison of clustering performance with 5% noise in the pairwise constraints



(a) Comparison of clustering performance evaluated by Rand Index



(b) Comparison of clustering performance evaluated by NMI

Fig. 7 Comparison of clustering performance with 10% noise in the pairwise constraints

Figs. 6 and 7 show, respectively, the 20-run average performances of all the algorithms on CMC, Letter(a-d), Iris and Digits data sets[+] with respect to the noisy pairwise constraints of 5% and 10%. From both the figures, we can find preliminarily that first for 5% noisy pairwise constraints, the displayed performances of all the algorithms are degraded to different extents in comparison with the case having no noisy constraints (as shown in Fig. 4). Overall, both SCKMM and ASSKKM algorithms still yield the same trend as that of the case without noisy constraints, which indicates that such two algorithms have resistant ability to noisy constraints to some extent, relatively more distinct for small amount of noisy constraints, while SCREEN, MPCKM and KASML do not exhibit such a trend and do produce more fluctuation, which indicates that their performances are not stable and thus relatively sensitive to noisy constraints. Secondly, for 10% noisy pairwise constraints, all the algorithms, including ours and ASSKKM, exhibit unstable and relatively inferior clustering performance as shown in Fig. 7. Intuitively, it is not difficult to understand because all the algorithms will be more misled in clustering process as the number of noisy pairwise constraints increases. Obviously, this has almost become a common flaw of such a

[+] Similar observations can be obtained for the rest of the data sets used in this paper and thus are omitted here.

class of semi-supervised clustering algorithms and thus it deserves a deep investigation. Thirdly, jointly from Figs. 4, 6 and 7, we can also observe that our SCKMM algorithm can relatively effectively improve the clustering performance of the current algorithms in the clean and the slightly noisy (less than 5%) cases. Though ASSKKM can also be adaptable to the same situation, its improved extent in performance is still limited due to lack of metric learning. As for the MPCKM algorithm, since its starting point is also to require the cluster memberships to be as consistent with the pairwise constraints as possible [7, 8], thus likewise it does not guarantee good performance in noisy case as shown in Figs. 6 and 7.

Finally, we have to point out that if the noisy constraints an expert provides are relatively many, especially semantically, undoubtedly, his expertise is questionable.

5 Conclusions and Future Work

In this paper, we have proposed a new adaptive semi-supervised clustering kernel method (SCKMM) which unifies the pairwise constraint based K-means (PCBKM) and metric learning to a nonlinear framework. Specifically, the SCKMM algorithm first uses the PCBKM to cluster the data in the original space and then learns a metric from the results of clustering. Second, the PCBKM is applied to cluster the data using both the learned metric and the learned parameter of Gaussian kernel. Thus, the SCKMM performs data clustering and metric learning simultaneously. The experimental results on real-world data sets indicate that the SCKMM can achieve a better clustering performance in comparison with existing semi-supervised clustering methods.

In the development of the SCKMM algorithm, we introduce the PCBKM to effectively solve the violation issue of pairwise constraints, which enable it to be applied in much wider application domains. In addition, we formulate the problem of parameter choice of Gaussian kernel as a gradient ascent problem. The goal is to find an optimal parameter of kernel function based on the pairwise instance constraints.

As in [1,3,7,9,13], owing to the fact that the computational complexity of solving metric matrix is expensive, we will discover a method to solve it such that our method can be applied to the large-scale problems. On the other hand, we will explore a way to handle noisy constraints.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments and suggestions to significantly improve the quality of this paper. The research reported in this paper has been partially supported by National Science Foundations of China under Grant Nos. 60875030 and 60773061.

References

[1] A. Bar-Hillel, T. Hertz, N. Shental, D. Weinshall. Learning a Mahalanobis Metric from Equivalence Constraints. Journal of Machine Learning Research, 2005, 6: 937-965.

[2] K. Wagstaff, C. Cardie, S. Rogers, S. Schroedl. Constrained K-means Clustering with Background Knowledge. In: Proceedings of the 18th International Conference on Machine Learning. San Francisco, 2001. 577-584.

[3] A. Bar-Hillel, T. Hertz, N. Shental, D. Weinshall. Learning Distance Functions Using Equivalence Relations. In: Proceedings of the 20th International Conference on Machine Learning, Washington, DC USA, 2003, 11-18.

[4] S. Basu, A. Banerjee, and R.J. Mooney. Semi-Supervised Clustering by Seeding. In: Proceedings of the 19th International Conference on Machine Learning, Sydney, Australia, 2002, 19-26.

[5] E.P. Xing, A.Y. Ng, M.I. Jordan and S. Russell. Distance Metric Learning, with application to Clustering with side-information, Advances in Neural Information Processing Systems 15, Cambridge, MA, 2002, 505-512.

[6] S. Basu, M. Bilenko, and R.J. Mooney. A Probabilistic Framework for Semi-Supervised Clustering. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, 2004, 59-68.

[7] M. Bilenko, S. Basu, R.J. Mooney. Integrating Constraints and Metric Learning in Semi-Supervised Clustering. In: Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004, 81-88.

[8] W. Tang, H. Xiong, S. Zhong, and J. Wu. Enhancing Semi-Supervised Clustering: A Feature Projection Perspective. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, 2007, 707-716.

[9] D.Y. Yeung, H. Chang. Extending the Relevant Component Analysis Algorithm for Metric Learning Using both Positive and Negative Equivalence Constraints. Pattern Recognition. 2006, 39(5):1007-1010.

[10] B. Kulis, S. Basu, Dhillon, R.J. Mooney. Semi-supervised Graph Clustering: A Kernel Approach. In: Proceedings of the 22st International Conference on Machine Learning, 2005.

[11] B. Yan, C. Domeniconi. An Adaptive Kernel Method for Semi-supervised Clustering. In: Proceedings of the 17th European Conference on Machine Learning, Berlin, Germany, 2006, 18-22.

[12] D.Y. Yeung, H. Chang. A Kernel Approach for Semi-supervised Metric Learning. IEEE Transactions on Neural Networks, 2007, 18(1):141-149.

[13] I.W. Tsang, P.M. Cheung, J.T. Kwok. Kernel Relevant Component Analysis for Distance Metric Learning. Proceedings of the International Joint Conference on Neural Networks, Montreal, Canada, 2005, 954-959.

[14] M. Wu and B. Schölkopf. A Local Learning Approach for Clustering. Advances in Neural Information Processing Systems 19, Mass. USA, 2007, 1529-1536.

[15] S.J. Kim, A. Magnani, and S. Boyd. Optimal Kernel Selection in Kernel Fisher Discriminant Analysis. In Proceedings of the International Conference on Machine Learning, 2006, 465–472.

[16] Z. Lu and T. Leen. Semi-supervised Learning with Penalized Probabilistic Clustering. Advances in Neural Information Processing Systems 18, 2005.

[17] B. Nelson, I. Cohen. Revisiting Probabilistic Models for Clustering with pair-wise Constraints. In: Proceedings of the 19th International Conference on Machine Learning, Corvalis, Oregon, USA, 2007, 673-680.

[18] B. Yan and C. Domeniconi, Kernel Optimization using Pairwise Constraints for Semi-supervised Clustering, Technical Report ISE-TR-06-09, George Mason Univ., 2006.

[19] J.H. Chen, Z. Zhao, J.P. Ye, and H. Liu. Nonlinear Adaptive Distance Metric Learning for Clustering. In: Proceedings of the The Thirteenth ACM SIGKDD International Conference On Knowledge Discovery and Data Mining, San Jose, California, USA ,2007,123-132.

[20] D.Q. Zhang, Z.H. Zhou, and S.C. Chen. Semi-supervised Dimensionality Reduction. In: Proceedings of the 7th SIAM International Conference on Data Mining, Minneapolis, 2007.

[21] X.S. Yin, E.L. Hu and S.C. Chen. Discriminative Semi-Supervised Clustering Analysis with Pairwise Constraints. Journal of Software (in Chinese) 19(11) (2008) 2791-2802.

[22] H. Kim, H. Park, H.Y. Zha. Distance Preserving Dimension Reduction Using the QR Factorization or the Cholesky Factorization. In: Proceedings of the 7th IEEE International on Conference on Bioinformatics and Bioengineering, 2007, 263-269.