*Journal of **AI** and **D**ata **M**ining*
Vol 5, No 2, 2017, 259-273

Composite Kernel Optimization in Semi-Supervised Metric

T. Zare[1*], M. T. Sadeghi[1], H. R. Abutalebi[1] and J. Kittler[2]

1. Signal Processing Research Group, Electrical Engineering Department, Yazd University, Yazd, Iran.
2. Centre for Vision, Speech and Signal Processing (CVSSP), Faculty of Engineering and Physical Sciences, University of Surrey, Guildford, Surrey.

Received 20 February 2016; Revised 10 October 2016; Accepted 14 February 2017
**Corresponding author: t.zare@stu.yazd.ac.ir (T. Zare).*

Abstract

Machine-learning solutions to classification, clustering, and matching problems critically depend on the adopted metric, which in the past was selected heuristically. In the last decade, it has been demonstrated that an appropriate metric can be learnt from data, resulting in a superior performance as compared with the traditional metrics. This has recently stimulated a considerable interest in the topic of metric learning, especially using the kernel functions, which map the input data to feature spaces with enhanced class separability, and implicitly define a new metric in the original feature space. The formulation of the problem of metric learning depends on the supervisory information available for the task. In this work, we focus on semi-supervised kernel-based distance metric learning, where the training dataset is unlabelled, with the exception of a small subset of pairs of points labelled as belonging to the same class (cluster) or different classes (clusters). The proposed method involves creating a pool of kernel functions. The corresponding kernel matrices are first clustered to remove redundancy in the representation. A composite kernel constructed from the kernel clustering result is then expanded into an orthogonal set of base functions. The mixing parameters of this expansion are then optimised using point similarity and dissimilarity information conveyed by the labels. The proposed method is evaluated on the synthetic and real datasets. The results obtained show the merit of using similarity and dissimilarity information jointly as compared to using just the similarity information, and the superiority of the proposed method over all the recently introduced metric learning approaches.

Keywords: *Distance Metric Learning, Semi-supervised Clustering, Composite Kernels, Pairwise Similarity and Dissimilarity Constraints, Optimization Problem.*

1. Introduction

Distance metrics play a key role in many supervised and unsupervised learning algorithms. The k-nearest neighbour classifier and the k-means clustering algorithm are examples of such supervised and unsupervised algorithms. Selecting an appropriate metric for these algorithms is an important issue. A promising alternative approach is to learn the optimal distance metric from a collection of training data. Distance metric learning has recently received considerable attention [1, 2]. The advocated algorithms in the literature can be divided into three main categories: supervised, unsupervised, and semi-supervised algorithms. A training dataset with explicit class labels or some other supervisory information is required for supervised metric learning [3-7]. Although unsupervised learning such as clustering is known to be highly influenced by the choice of the distance metric, the lack of supervisory information makes the problem of choosing a suitable metric very challenging. In unsupervised distance metric learning, the main idea is to learn a metric that preserves the geometric relationships between most of the observed samples [8]. There is a deep connection between unsupervised distance metric learning methods and unsupervised dimensionality reduction approaches. The main advantage of these algorithms is that they do not require laborious labelling of the training data to provide supervisory information, although their performance is usually lower. These issues have recently motivated the development of a compound solution known as semi-supervised learning. In semi-supervised approaches, a large quantity of unlabelled data and a limited amount of labelled data or some other supervisory knowledge are used to learn a distance metric. One kind of supervisory information for metric learning is expressed in the form of pairwise similarity and/or dissimilarity relationships [9]. The use of these constraints for distance metric learning, and especially semi-supervised distance metric learning, has lately become very popular [10-12]. Compared to the explicitly labelled data, pairwise constraints are weaker but more natural than the clustering concepts. Xing *et al.* [13] have proposed an iterative approach to learn a Mahalanobis metric using pairwise constraints for the clustering task. Bar-Hillel *et al.* [14] have proposed a non-iterative algorithm using only pairwise similarity constraints called relevant component analysis (RCA). The RCA algorithm has been extended to a generalized form that makes use of both the pairwise similarity and the dissimilarity constraints in [15]. In the above-mentioned methods, the distance metric learning procedure can be summarized as a process of mapping the associated data to a new feature space using a linear learned transformation and then applying the Euclidean distance in the resultant feature space. However, a linear transformation may not necessarily be a desirable transformation for a given problem or a complicated structure of the training data. In these cases, the kernelized version of the metric learning algorithms can be seen as offering a more general alternative [16, 18].

A kernel function can be considered as a function that implicitly transfers two data points to a new feature space using a non-linear mapping function. It measures their associated similarity by computing the inner product of the projected samples. An important issue in kernel-based approaches is how to find an appropriate kernel and/or how to set the kernel parameter(s). The type of kernel function and the value(s) of kernel parameter(s) play a key role in these algorithms. A typical solution to this problem is to apply cross-validation in order to select the best kernel function among a set of candidates. However, this procedure is time-consuming. Moreover, there is no guarantee that the best possible solution will be found. The other solution to this problem is to use an appropriate combination of different kernel functions. Within this framework, several different algorithms have been proposed for computing composite kernels [19, 20]. It has been shown that the performance of some pattern recognition algorithms such as Support Vector Machine (SVM) classifier and kernel-based feature extraction approaches can be improved by applying composite kernel techniques.

Kernelized versions of metric learning algorithms have already been proposed in [16-18]. However, the methods are not sufficiently flexible because they are based on a fixed kernel function without any generalisation property. In other works, a distance metric learning framework is used for determining the elements of the kernel matrix [21, 22]. However, a common problem of these methods is that a large number of variables are required to be learned. In fact, as the size of a kernel matrix is proportional to the number of training samples, the number of unknowns is usually very large. Soleymani *et al.* have proposed an iterative kernel-based metric learning algorithm that reduces the number of variables to the number of the related constraints [23].

To date, the problem of metric learning using composite kernel functions has not been studied very extensively. In [24] and [25], the idea of weighted summation of different kernels has been considered within the framework of supervised distance metric learning. In [24], the authors have propose a method that learns a set of Mahalanobis metrics, one for each feature space induced by the respective kernels. The kernel weights and the Mahalanobis metrics are learned using an iterative optimization procedure that is computationally complex. In [25], assuming a linear combination of a set of kernels, several distance metric learning objectives have been defined in order to learn the kernel weights. In our previous work [26], we used composite kernels in semi-supervised metric learning, given a set of pairwise similarity constraints and a limited number of kernels.

As a main contribution, in this paper, we show how to construct a composite kernel matrix for a set of similarity and dissimilarity constraints. For this purpose, first an initial composite kernel matrix is produced by combining a set of kernel matrices. The eigen-decomposition of the matrix is then performed. The resulting eigen-vectors are linearly combined with weights obtained using a semi-supervised distance metric learning objective. Effectively, we rescale the axes of the new feature space (eigen-vectors) induced by the composite kernel so that the pairwise similarity and dissimilarity constraints are satisfied. We finally use the learned kernel matrix in a kernel-based k-means clustering algorithm.

The rest of the paper is organized as what follows In Section 2, the related works are briefly reviewed. In Section 3, our proposed method of composite kernel-based metric learning, which uses both pairwise similarity and dissimilarity constraints, is presented. In the method, a composite kernel is used as the base kernel, and a set of mixing variables are determined. These variables are computed by analytically solving an optimization problem. Our experimental studies are reported in Section 4, where the performance of the method is compared to some other states of the art approaches. Finally, Section 5 offers the concluding remarks.

2. Related works and proposed method

Recently, kernel-based semi-supervised metric learning has attracted the attention of many researchers [16, 18, 21, 22]. In [21], Chang and Yeung have proposed two kernel-based metric learning methods, which are called the kernel-**A** and kernel-**β** methods. They use a set of similarity constraints to guide the learning process. These methods use a pre-specified kernel such as a specific RBF kernel to form the kernel matrix $\mathbf{K}_{N\times N}$. In the kernel-**A** method, the target kernel matrix is defined as $\tilde{\mathbf{K}} = \mathbf{AKA}^T$, where $\mathbf{A}$ is an $N\times N$ adaptation matrix. The elements of $\mathbf{A}$ are determined using a criterion expressed in terms of pairwise similarity information penalized by a regularization term. They are calculated by applying an iterative algorithm. In the kernel-β method, the kernel matrix $\mathbf{K}$ is first decomposed into a set of base kernels by applying the eigen-decomposition operation. The weighted sum of the base kernels is then used as the final kernel matrix, where the weights are determined analytically using a constraint imposed by identically labelled pairs. For handling large datasets, they extend the kernel-β method to a scalable method by applying low-rank approximation to the kernel matrix [18]. The main limitation of the kernel-β method and its extension is that the target kernel is computed using a linear combination of eigen-matrices, derived from an RBF kernel with a specific width. Thus the diversity of the basic kernel matrix is limited. Therefore, the performance of the method heavily depends on the adopted kernel function (i.e. RBF kernel) and its parameter. Moreover, only pairwise similarity constraints are considered in the learning process.

In the kernel-**A** method and in [22], the learning process exploits similarity and/or dissimilarity measures. In these methods, an optimization procedure is used to determine the elements of the base kernel. The authors show that here the choice of the base kernel is not as critical as in the case of the kernel-β method. However, the number of the variables that have to be adjusted is very large.

The kernel-based metric learning method introduced in [23] reduces the adjustable variables to the number of constraints. However, the iterative algorithm used to determine the unknown parameters is time-consuming.

As mentioned in our previous work [26], we used composite kernels in semi-supervised metric learning, given a set of pairwise similarity constraints. The composite kernel is constructed by combining a limited number of kernels through either averaging or augmenting the associated kernel matrices. A set of orthogonal matrices are then generated by eigen-decomposition of the resulting Gram matrix. The final kernel matrix is created by weighted averaging of the orthogonal matrices, while the weights are determined via a learning process. In fact, through the learning process, the axes of the new feature space (which is the result of the eigen-decomposition process) are rescaled so that the pairwise similarity constraints are satisfied.

In this paper, we present a new metric learning algorithm using composite kernels to generalize the method proposed in [26]. We reformulate the optimization problem so that the effect of the supervisory information, which is expressed in the form of both the pairwise similarity and dissimilarity constraints, is taken into account. During optimization, a limited number of variables are learned such that the resulting distance between similar pairs from the pairwise similarity set becomes as low as possible while the distance between dissimilar pairs from the pairwise dissimilarity set becomes as large as possible. An important characteristic of the proposed method is that the optimization process does not require any iterative algorithm to find the solution.

The proposed approach is inspired by the kernel-β method introduced in [21], and the optimization algorithm has been proposed in [26]. However, in contrast to [21] and [26], in this work, we considered pairwise dissimilarity as well as similarity constraints in the optimization problem. Moreover, compared to [21], instead of a pre-determined kernel, we made use of composite kernels to improve the flexibility of the method and to avoid the problem of choosing an inappropriate kernel function. Furthermore, throughout the eigen-decomposition operation, the number of optimization variables is reduced by keeping those that preserve a pre-specified level of total variation of the data. Also noting that the matrix augmentation process generates a high dimensional kernel matrix when a large set of candidate kernel matrices is available, we avoided the computational complexity problem by grouping the base kernels using a kernel alignment measure. A representative kernel was then used for each group. In fact, in our previous work [26], we were not able to increase the number of base kernels within the framework of the matrix augmentation method. This problem was here removed by the proposed grouping process. In the next section, the proposed method is detailed.

3. Proposed metric learning method

In this section, our formulation of the underlying optimization problem is presented. The proposed method falls within the framework of kernel-based distance metric learning using pairwise similarity and dissimilarity constraints. Since composite kernels are used as the kernel function, the adopted approaches for producing composite kernels are subsequently reviewed. We used the following notations in this section: **A** represents a matrix, ***a*** denotes a vector, ***a**^j*... denotes the *j*-th column of matrix **A**, and a^{ij} is its *i*-th element (i.e. . $a^{ij} = \mathbf{A}(i,j)$.).

3.1. Problem formulation

Denote by $X = \{\mathbf{x}_i\}_{i=1}^{N}$ a set of data points, where $\mathbf{x}_i \in R^{r_x}$, $i = 1,\ldots,N$, and *N* is the total number of samples. Also suppose that ϕ is a feature space induced by a non-linear transformation $\phi : R^{r_x} \to H$. For a positive semi-definite kernel function k, a Mercer kernel is computed as the inner product of samples in *H*, and $k(\mathbf{x},\mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$. Also suppose that from the data points in *X*, we can construct two sets of training data *S* and *D* satisfying $S = \{(\mathbf{x}_i, \mathbf{x}_j) \mid \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ belong to the same class}\}$ and $D = \{(\mathbf{x}_i, \mathbf{x}_j) \mid \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ belong to different classes}\}$

As we focus on the semi-supervised setting of the distance metric learning problem, we assume that the size of the *S* and *D* datasets is very limited and a large number of unlabelled data points are available in *X*. Our goal is that by virtue of learning, distances between similar pairs are reduced, while distances between dissimilar pairs are increased. Accordingly, the objective function can be defined as:

$$J = \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S} \left\| \phi(\mathbf{x}_i) - \phi(\mathbf{x}_j) \right\|_2^2 - \beta \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in D} \left\| \phi(\mathbf{x}_i) - \phi(\mathbf{x}_j) \right\|_2^2 \quad (1)$$

The above criterion function is, in fact, a weighted sum of squares of the Euclidean distances between the similar and dissimilar pairs of points measured in the feature space. This function should be minimized using the learned distance metric so that the pairs of the samples that belong to the same class will be as close as possible, and vice versa, for the dissimilar pairs. The role of parameter β is to balance the contributions of the similarity and dissimilarity pairs. In practice, β can be estimated using the cross-validation procedure or it can simply be set to $\frac{|D|}{|S|}$, where $|.|$ denotes the cardinality of a set. Using the kernel representation for the Euclidean distances in (1), we can express *J* as:

$$J = \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S} \left[k^{ii} + k^{jj} - 2k^{ij} \right] - \beta \sum_{(\mathbf{x}_{i'},\mathbf{x}_{j'})\in D} \left[k^{i'i'} + k^{j'j'} - 2k^{i'j'} \right] \quad (2)$$

where, k^{ab}'s are elements of the kernel or Gram matrix, i.e. $k^{ab} = \mathbf{K}(a,b)$. Elements of $\mathbf{K}$ are actually the values of the kernel function corresponding to the respective pairs of data samples. This objective function is used in order to find the optimum kernel function, i.e. the optimum feature space ϕ . Since the corresponding kernel matrix contains N^2 elements, the number of variables that should be optimized is very large. To avoid this computational burden, we suppose that an initial Gram matrix is available a priori. This matrix is not necessarily the optimal one but it highly likely contains the relevant information. Then, new variables facilitating adaptation are introduced in the objective function to achieve a better performance. In particular, we assume that by decomposing the Gram matrix into orthogonal components, it is written as:

$$\mathbf{K} = \sum_{n=1}^{N} \alpha_n \mathbf{u}_n \mathbf{u}_n^T = \sum_{n=1}^{N} \alpha_n \mathbf{U}_n \quad (3)$$

where, α_n are positive eigen-values of $\mathbf{K}$ and $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N$ are the corresponding normalized eigen-vectors. Now by remixing the bases with different weighting coefficients, λ_n^2 , we have:

$$\tilde{\mathbf{K}} = \sum_{n=1}^{N} \lambda_n^2 \mathbf{U}_n \quad (4)$$

The remixing idea is based upon the well-known fact that while coefficients α_n are ideal for good reconstruction of matrix $\mathbf{K}$, they are not necessarily optimal for good discrimination. By replacing the eigen-values by a set of positive coefficients λ_1^2, λ_2^2,…, and λ_N^2, we reformulate the distance metric learning problem as one determining the optimum value of λ_n^2's. Thus instead of matrix $\mathbf{K}$, which is commonly used in kernel-based metric learning approaches, the matrix $\tilde{\mathbf{K}}$ is used, and the learning process optimizes these mixing parameters (λ_n^2).

The exponent of λ_n^2 emphasises that the coefficients are positive. It will also simplify the subsequent equations.

An important issue in problem formulation is selecting the initial kernel matrix, $\mathbf{K}$, so that (5) provides a scope for finding a good solution to our problem. The objective function described in (1) or (2) can then be used for learning the variables λ_n^2. Using this technique, the number of variables reduces from the order of N^2 to N. The number of variables can be reduced even more by preserving a specific level (for example, 99%) of the total variance of the data during the eigen-decomposition operation. Hence, (4) can be re-written as:

$$\tilde{\mathbf{K}} = \sum_{n=1}^{P} \lambda_n^2 \mathbf{U}_n \tag{5}$$

where, $P < N$ is the number of the eigen-values α_n, $n = 1,2,...,P$ capturing the specified amount of variance of the data in the original kernel space. By substituting the kernel function of (5) in (2), we have:

$$J = \sum_{n=1}^{P} \lambda_n^2 \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S} \left[u_n^{ii} + u_n^{jj} - 2u_n^{ij}\right] - \beta \sum_{n=1}^{P} \lambda_n^2 \sum_{(\mathbf{x}_{i'},\mathbf{x}_{j'})\in D} \left[u_n^{i'i'} + u_n^{j'j'} - 2u_n^{i'j'}\right] \tag{6}$$

Let $\mathbf{e}^i (i = 1,\dots,N)$ be the *i*-th column of an $N \times N$ identity matrix. We can rewrite the objective function in (6) as:

$$J = \sum_{n=1}^{P} \lambda_n^2 \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S} (\mathbf{e}^i - \mathbf{e}^j)^T \mathbf{U}_n (\mathbf{e}^i - \mathbf{e}^j) - \beta \sum_{n=1}^{P} \lambda_n^2 \sum_{(\mathbf{x}_{i'},\mathbf{x}_{j'})\in D} (\mathbf{e}^{i'} - \mathbf{e}^{j'})^T \mathbf{U}_n (\mathbf{e}^{i'} - \mathbf{e}^{j'}) = \sum_{n=1}^{P} \lambda_n^2 a_n - \beta \sum_{n=1}^{P} \lambda_n^2 b_n \tag{7}$$

where, $a_n = \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in S} (\mathbf{e}^i - \mathbf{e}^j)^T \mathbf{U}_n (\mathbf{e}^i - \mathbf{e}^j)$ and $b_n = \sum_{(\mathbf{x}_{i'},\mathbf{x}_{j'})\in D} (\mathbf{e}^{i'} - \mathbf{e}^{j'})^T \mathbf{U}_n (\mathbf{e}^{i'} - \mathbf{e}^{j'})$. Now, if $\mathbf{A}_S$ and $\mathbf{B}_D$ matrices are defined as $\mathbf{A}_S = diag(a_1, a_2, ..., a_P)$ and $\mathbf{B}_D = diag(b_1, b_2, ..., b_P)$, then (7) can be re-written as:

$$J = \boldsymbol{\lambda}^T \mathbf{A}_S \boldsymbol{\lambda} - \beta \boldsymbol{\lambda}^T \mathbf{B}_D \boldsymbol{\lambda} = \boldsymbol{\lambda}^T (\mathbf{A}_s - \beta \mathbf{B}_D) \boldsymbol{\lambda} \tag{8}$$

where $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, ..., \lambda_P]$. To prevent $\boldsymbol{\lambda}$ from being a vector of zero values, we consider an extra constraint on $\boldsymbol{\lambda}$ such that $\mathbf{1}^T \boldsymbol{\lambda} = c$ for some constant value of $c > 0$. In order to preserve the total variance of the data, we set this value as: $c = \sum_{n=1}^{P} \sqrt{\alpha_n}$. The constraint optimization problem in (8) with this equality constraint can be solved using the method of Lagrange multipliers, leading to the following constraint objective function:

$$L = \boldsymbol{\lambda}^T (\mathbf{A}_s - \beta \mathbf{B}_D) \boldsymbol{\lambda} + \mu (c - \mathbf{1}^T \boldsymbol{\lambda}) \tag{9}$$

where, μ is a Lagrange multiplier. The partial derivatives of the above expression are as:

$$\frac{\partial L}{\partial \boldsymbol{\lambda}} = 2(\mathbf{A}_s - \beta \mathbf{B}_D)\boldsymbol{\lambda} - \mu \mathbf{1} \qquad \frac{\partial L}{\partial \mu} = c - \mathbf{1}^T \boldsymbol{\lambda} \tag{10}$$

The optimal vector of $\boldsymbol{\lambda}$ is obtained by setting the partial derivatives to zero as:

$$\boldsymbol{\lambda} = \frac{c(\mathbf{A}_s - \beta \mathbf{B}_D)^{-1} \mathbf{1}}{\mathbf{1}^T (\mathbf{A}_s - \beta \mathbf{B}_D)^{-1} \mathbf{1}} \tag{11}$$

As mentioned earlier, we assume that the initial kernel matrix, $\mathbf{K}$, is available at the beginning. The optimization process modifies the associated mapping process by replacing the eigen-values of the kernel with the values obtained via the learning process. It should be noted that the structure of the initial kernel matrix, $\mathbf{K}$, is important, as it should contain the "optimal" kernel. It means that we cannot expect to achieve the best possible results if a totally inappropriate kernel function is adopted. Thus the most important issue now is how to choose the initial kernel function. The usual approaches such as the cross-validation procedure that try to find the best kernel are not useful for this purpose. Therefore, we focus on the composite kernels instead of choosing a specific kernel function.

The general optimization goal in the proposed optimization process is somehow similar to that of the optimization method in [28], where the authors have proposed a semi-supervised metric learning technique for learning a projection matrix that is used for the dimensionality reduction purpose. Both optimization processes try to minimize the average distance between similarity pairs and maximize the average distance between dissimilarity pairs. However, their final objective and the adopted methodologies are different. In our proposed method, within the clustering framework, the optimization process is performed in the kernel space considering a weighted sum of a set of kernel functions and the main purpose is to find the optimal weights, while in [28], the optimization process is done in the feature space, and the main purpose is to learn a projection matrix for the dimensionality reduction purpose. Moreover, in their learning process, all the data points including the unlabeled ones are used. However, we utilize only the similarity and/or dissimilarity pairs in order to learn the weights of the adopted kernels.

3.2. Composite kernels

Let $\{\mathbf{K}_1,\mathbf{K}_2,\ldots,\mathbf{K}_M\}$ be M different kernel matrices that map the data points to M Hilbert spaces as:

$$\begin{aligned}\mathbf{K}_m &= \left[k_m(\mathbf{x}_i,\mathbf{x}_j)\right]_{N\times N} \\ &= \left[\left\langle \phi_m(\mathbf{x}_i),\phi_m(\mathbf{x}_j)\right\rangle\right]_{N\times N}\end{aligned} \tag{12}$$

where, $\phi_m \in \mathbb{R}^{r_{\phi_m}}$ is the feature space corresponding to the *m*-th Hilbert space, and r_{ϕ_m} is the number of dimensions of ϕ_m. These different kernels correspond to different feature spaces and they contain different sources of information. This suggests that complimentary information which leads to a better performance can be extracted by a suitable combination of the kernel functions. In what follows, we briefly review two popular combining methods, namely the unweighted sum [19] and matrix augmentation methods [25].

3.2.1 Unweighted sum method

The unweighted sum of a set of base kernels is denoted by:

$$\mathbf{K}_{un} = \sum_{m=1}^{M}\mathbf{K}_m \tag{13}$$

Since the kernel matrices are positive semi-definite, the summation of them is also a positive semi-definite matrix. Thus the new kernel satisfies the mercer's condition, and it is a valid kernel. It can be shown that the new kernel corresponds to a new feature space that is obtained by unweighted concatenation of the base feature vectors, i.e.:

$$\mathbf{\Phi}_{un} = \left[\phi_1^T,\phi_2^T,\ldots,\phi_M^T\right]^T \tag{14}$$

One of the advantages of this composite kernel is that there is no limit on the number of base kernels, and a large number of base kernels can be combined without increasing the computational complexity. Moreover, the reported results confirm the effectiveness of the method [19].

3.2.2 Augmenting kernel matrices

In [25], Yan *et al.* have proposed a novel method of creating composite kernels that involves augmenting kernel matrices. An augmented kernel matrix (AKM) is defined as:

$$\mathbf{K}_{aug} = \begin{bmatrix}\mathbf{K}_1 & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \ddots & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{K}_M\end{bmatrix}_{(M\times N)(M\times N)} \tag{15}$$

It can be shown that this kernel is also a valid Mercer's kernel [25]. This formulation indicates that the discriminative importance of the data points in different feature spaces is preserved by augmenting the kernel matrices. One of the limitations of AKM is that by increasing the number of base kernels, a large amount of memory space is needed, resulting in a high computational complexity. This makes AKM inapplicable to large datasets, especially when the number of base kernels, M, is large.

We deal with this problem by initial grouping of the base kernels according to their similarity. We use the Kernel Alignment (KA) method proposed in [27] as a measure of similarity between kernels in order to group them in an unsupervised manner. Considering two kernel matrices as $\mathbf{K}_k$ and $\mathbf{K}_l$, the Frobenius inner product between the matrices is defined as: $\langle\mathbf{K}_k,\mathbf{K}_l\rangle_F = \sum_{i,j=1}^{N}\mathbf{K}_k(\mathbf{x}_i,\mathbf{x}_j)\mathbf{K}_l(\mathbf{x}_i,\mathbf{x}_j)$. The empirical alignment between these two kernel matrices can be measured by:

$$\mathrm{KA}(\mathbf{K}_k,\mathbf{K}_l) = \frac{\langle\mathbf{K}_k,\mathbf{K}_l\rangle_F}{\sqrt{\langle\mathbf{K}_k,\mathbf{K}_k\rangle_F\langle\mathbf{K}_l,\mathbf{K}_l\rangle_F}} \tag{16}$$

We group the kernels in an agglomerative manner using the kernel alignment measure. For this purpose, first, we initialize all kernels as separate clusters and merge two most similar clusters at a time. Similarity between two clusters is defined as the largest distance between all possible pairs of cluster members. The merging process is then continued until *G* groups are obtained. After determining the members of each group, we find a representative kernel for each of them. In this study, the representative kernels are determined by unweighted summation of the kernels of each group. In fact, we use the beneficial effects of unweighted sum within each group and the discriminative effect of the augmenting method over the representative kernels. The proposed algorithm is summarized in figure 1.

```
Algorithm 1. The proposed algorithm
Input: Input data, $X = \{x_i\}_{i=1}^{N}$ a set of similarity pairs S and a set of dissimilarity pairs D.
Output: Clustering the input data.
Step 1: Compute the basis kernels $\{\mathbf{K}_m\}_{m=1}^{M}$ .
Step 2: Compute composite kernel using unweighted or augmenting method, $\{\mathbf{K}_{Com}\} = \mathbf{K}_{un}$ or $\{\mathbf{K}_{Com}\} = \mathbf{K}_{aug}$ .
Step 3: Eigen-decomposition of composite kernels in previous the step, $\mathbf{K}_{Com} = \sum_{n=1}^{N} \alpha_n \mathbf{u}_n \mathbf{u}_n^T = \sum_{n=1}^{N} \alpha_n \mathbf{U}_n$ .
Step 4: Specify the eigen-values that preserve a specific level (for example, 99%) of the total variance of the data.
Step 5: Replace the eigen-values in the previous step with new variables and remove the rest, $\tilde{\mathbf{K}}_{Com} = \sum_{n=1}^{P} \lambda_n^2 \mathbf{U}_n$ .
Step 6: Compute the optimal coefficient vector as $\boldsymbol{\lambda} = \frac{c(\mathbf{A}_S - \beta \mathbf{B}_D)^{-1}\mathbf{1}}{\mathbf{1}^T(\mathbf{A}_S - \beta \mathbf{B}_D)^{-1}\mathbf{1}}$ .
Step 7: Compute new kernel matrix using the coefficients in the previous step, $\tilde{\mathbf{K}}_{Com} = \sum_{n=1}^{P} \lambda_n^2 \mathbf{U}_n$ .
Step 8: Use $\tilde{\mathbf{K}}_{Com}$ in kernelized k-means algorithm to cluster data, and repeat the k-means algorithm until a stopping criterion is met.
```

Figure 1. Proposed algorithm.

To study the computational complexity of the above solution, we exploit the complexity of different parts of it. Without considering any structure for the kernel matrix $\mathbf{K}$, the eigen-decomposition of $\mathbf{K}$ to express it in the form of (3) takes $O(N^3)$ time. The diagonal matrices $\mathbf{A}_S$ and $\mathbf{B}_D$ take $O(P|S|)$ and $O(P|D|)$ times, respectively. The optimal value for $\boldsymbol{\lambda}$ can be computed according to (11) in $O(P)$ time. Note that $P < N$ and typically $|S|,|D| \ll N$, so the overall complexity of the proposed algorithm is $O(N^3)$, which is dominated by the complexity of the eigen-decomposition step. The kernel alignment can also be computed in $O(N^2)$. The computational complexity of kernel-based algorithms is summarized in table 1. It can be seen that the computational complexity of all kernel-based algorithms is approximately $O(N^3)$.

Table 1. Computational complexity of the proposed algorithm.

Kernel-β	Unweighted sum	Augmenting
$O(N^3)$	$O(N^3 + N^2)$	$O(GN^3 + N^2)$

4. Experimental results

In this section, our experimental results on both the synthetic and real-world data are reported, and the performance of the proposed method for semi-supervised metric clustering is evaluated.

4.1. Experimental setup

We compare our kernel-based metric learning method with some other benchmark approaches. The Euclidean distance without metric learning is used as the baseline in our comparative study. The RCA method proposed in [14] is one of the benchmark methods. The RCA algorithm assigns lower weights to the irrelevant directions in the input space by applying whitening transformation on the dataset. The Xing *et al.*'s method in [13] is the other adopted method. This method considers both the pairwise similarity and dissimilarity constraints in contrast to the RCA method that makes use of only the similarity constraint. As one of the recently proposed kernel-based methods, we also report the results using the Kernel-β method [21]. This method uses the pairwise similarity constraint as well. Thus overall, we compare the performance of our kernel-based metric learning algorithm with the following algorithms (the short names inside the brackets will be used subsequently):

1) k-means without metric learning (Euclidean);
2) k-means with the RCA metric learning method (RCA) [14];
3) k-means with Xing *et al.*'s metric learning method (Xing's) [13];
4) Kernelized k-means with the kernel obtained by the Kernel-β method (Kernel-β) [21];
5) Kernelized k-means with the unweighted composite kernel obtained by the proposed method using only the similarity constraints (unweighted-*S*) [26];
6) Kernelized k-means with the unweighted composite kernel obtained by the proposed method using both the similarity and dissimilarity constraints (unweighted-*SD*);
7) Kernelized k-means with the augmented composite kernel obtained by the proposed method using only the similarity constraints (augmented-*S*) [26];
8) Kernelized k-means with the augmented composite kernel obtained by the proposed method using both the similarity and dissimilarity constraints (augmented-*SD*);

We utilize the RBF kernels as the primary non-linearity inducing function for all the kernel-based metric learning methods. The RBF kernel is given as:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{\left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2}{2\sigma^2} \right) \tag{17}$$

This kernel is widely used in kernel-based methods. An appropriate selection of the kernel parameter, σ, is an important issue. The value for σ should be selected to reflect the distribution of data points. The distribution is usually unknown, especially in the case of high dimensional data. Instead of choosing a specific value for σ, we combine Gaussian kernels with different widths. The composite kernel encloses information for all the kernels. It is expected to automatically extract the most useful information via the learning process.

In this study, the performance of the metric learning clustering algorithms is measured using the Rand Index (*RI*) value. *RI* is a measure of agreement between the clustering result and the ground truth. Let n_s be the number of pairs assigned to the same partitions by both the clustering and the ground truth annotation, and n_d be the number of pairs assigned to different partitions by them. The *RI* is defined as $RI = 2(n_s + n_d)/(N(N-1))$, i.e. it is the ratio of correctly assigned pairs to the total number of pairs. When there are more than two clusters, the standard Rand index, as defined above, will tend to assign data pairs to different clusters. We use the modified Rand index as in [13, 18, 21, 23]. In the modified Rand index, the equal chance of occurrence (0.5) is considered for both the similarity and dissimilarity pairs, and *RI* is defined as:

$$RI(C,\hat{C}) = \frac{0.5 \times \sum_{i>j} \delta(c_i = c_j \wedge \hat{c}_i = \hat{c}_j)}{\sum_{i>j} \delta(\hat{c}_i = \hat{c}_j)} + \frac{0.5 \times \sum_{i>j} \delta(c_i \neq c_j \wedge \hat{c}_i \neq \hat{c}_j)}{\sum_{i>j} \delta(\hat{c}_i \neq \hat{c}_j)} \tag{18}$$

where, $\delta(.)$ is an indicator function (i.e. $\delta(True) = 1$ and $\delta(False) = 0$), $\hat{c}_i$ is the cluster to which $\mathbf{x}_i$ is assigned by the clustering algorithm, and c_i is the correct cluster assignment. The indicator $\wedge$ is used as "*and*" operator.

Each dataset is normalized to zero mean and unit standard deviation before applying the clustering algorithm. As described in Section 3, the proposed metric learning approach uses the similarity and dissimilarity pairwise constraints in the learning process. This data plays an important role. Therefore, for each dataset, we randomly generate 20 different similarity (*S*) and dissimilarity (*D*) sets. We also perform 20 runs of k-means with random initialization for each pairwise constraint (*SD*) set. Thus each clustering experiment is repeated for 400 times and the statistical characteristics of the results are reported.

As mentioned in Section 3.2, the main aim in creating the augmented structure is to retain the discriminative importance of the data points in different feature spaces. After clustering the base kernels into *G* groups using kernel alignment as the similarity measure, we will have G representations of a data point at the same time. Thus we define the distance between two data points in the augmented feature space as the mean of the distances between these two points in the different feature spaces induced by the base kernels.

In the kernel-β method, the kernel parameter, σ, is set using the following equation [21]:

$$\sigma^2 = (\theta / N(N-1)) \sum_{i,j=1}^{N} \left\| \phi(\mathbf{x}_i) - \phi(\mathbf{x}_j) \right\|^2 \tag{19}$$

where, similar to the associated reference, $\theta = 5$ is used.

4.2. Experiments on Synthetic Dataset

We first perform some experiments on a synthetic XOR dataset. Figure 2(a) shows a scatter plot of the dataset. Two different signs with different colours are used to show the data points with their labels. The solid green and pink lines are used to show, respectively, the adopted similarity (*S*) and dissimilarity (*D*) pairs. We randomly pick 10 similarity pairs (i.e. |*S*| = 10) and 5 dissimilarity pairs (i.e. |*D*| = 5). Based on the experiments performed on the UCI datasets, which will be reported later, we set $G = 2$. The parameter σ_i varies from 1 to 8, which is in steps of 0.7 to obtain 11 base kernels.

The scatter plots of the data points projected into the new feature spaces induced by the different distance metric-based clustering approaches are shown in figure 2 (b-h). The RCA and Xing's method learn global Mahalanobis metric in the original input space. For the kernel-based methods, in order to represent the transformed data in the learned kernel space, we apply the kernelized PCA to the data points using the learned kernel matrix. The resulting two-dimensional (2-D) feature space is then displayed.

Figure 2 (b) and (c) show the transformed data points using the RCA and Xing's methods, respectively. Note that the linear transformation using the RCA and Xing's methods cannot yield significant clustering results for non-linearly distributed data points such as the XOR data. As shown in figure 2(d), the non-linear kernel-β method yields a good clustering result for this type of dataset but the main problem with this method is that it is very sensitive to the choice of the RBF kernel parameter, σ. Slight changes can highly degrade the performance of the method. Moreover, although the data points of different classes are separated, the scattering of the data points in each class is quite high. Figures 2 (e) and (g) show the mapped data points using the proposed method with the unweighted averaging and augmenting strategies, respectively, in a scenario when just similarity constraints are available. Compared to the kernel-β method, combining the base kernels, i.e. combining the information for different feature spaces induced by different kernels, reduces the scattering of the data points in each class. As shown in figures 2 (f) and (h), using additional information in the form of dissimilarity pairwise constraints, the performance of the proposed kernel-based method is meaningfully improved. It can be seen that by adding this information, more compacted and separated clusters are created. It has to be noted that in the eigen-decomposition step of the experiments, 99% of the variance of the overall kernel matrix, $\mathbf{K}$, is preserved. As a result, the number of variables that have to be learned is significantly reduced (for instance, from N = 200 to P = 7).

Figure 3 contains schematic plots of the initial and learned kernel matrices computed over the XOR dataset using the kernel-β and unweighted-*SD* methods. We arrange the data points according to their class label. The kernel matrix for these arranged data points is then computed. The top row of the figure shows the initial kernel that has been obtained from a single Gaussian kernel for the kernel-β method and a combination of the base kernels for the unweighted-*SD* method.
It can be seen that the initial kernel matrix of both methods consists of four parts, meaning that the samples are cast into four clusters, which is due to the geometrical distribution of the data. Although the data points in both initial kernels are incorrectly partitioned, the kernel values for each part of the data in the composite methods are much higher than those of the single kernel. The kernel matrices after the learning process are shown in the next row of the figure. The learning process leads to a better association of kernel values and data point labels (i.e. the matrix can be divided into two parts). Using the composite kernels, the kernel matrix is perfectly divided into two distinct parts that correspond to the ideal kernel for this dataset.

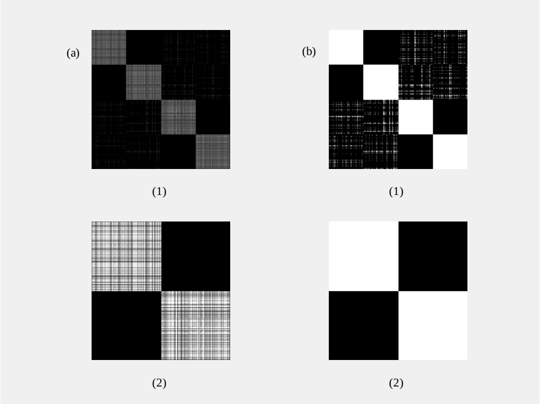

Figure 3. Kernel matrices for XOR dataset before and after kernel-based metric learning algorithms. (1) initial kernel matrix used in metric learning, (2) learned kernel matrix in metric learning process. (a) kernel-β, (b) unweighted-*SD*.

The semi-supervised clustering results for the XOR dataset are displayed as the box-plots in figure 4. All the kernel-based methods achieve remarkably good semi-supervised clustering. The kernel-based methods always lead to perfect results ($RI = 1$), and the variance of the results is zero.

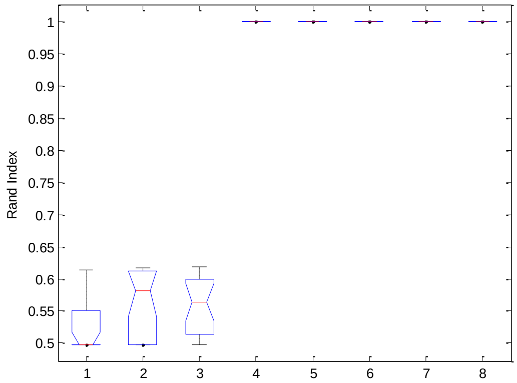

Figure 4. Clustering results of different algorithms for XOR dataset. Eight algorithms (numbered in Section 4.1) are as follow: (1) Euclidean, (2) RCA, (3) Xing's, (4) kernel-β, (5) unweighted-*S*, (6) unweighted-*SD*, (7) augmented-*S*, and (8) augmented-*SD*.

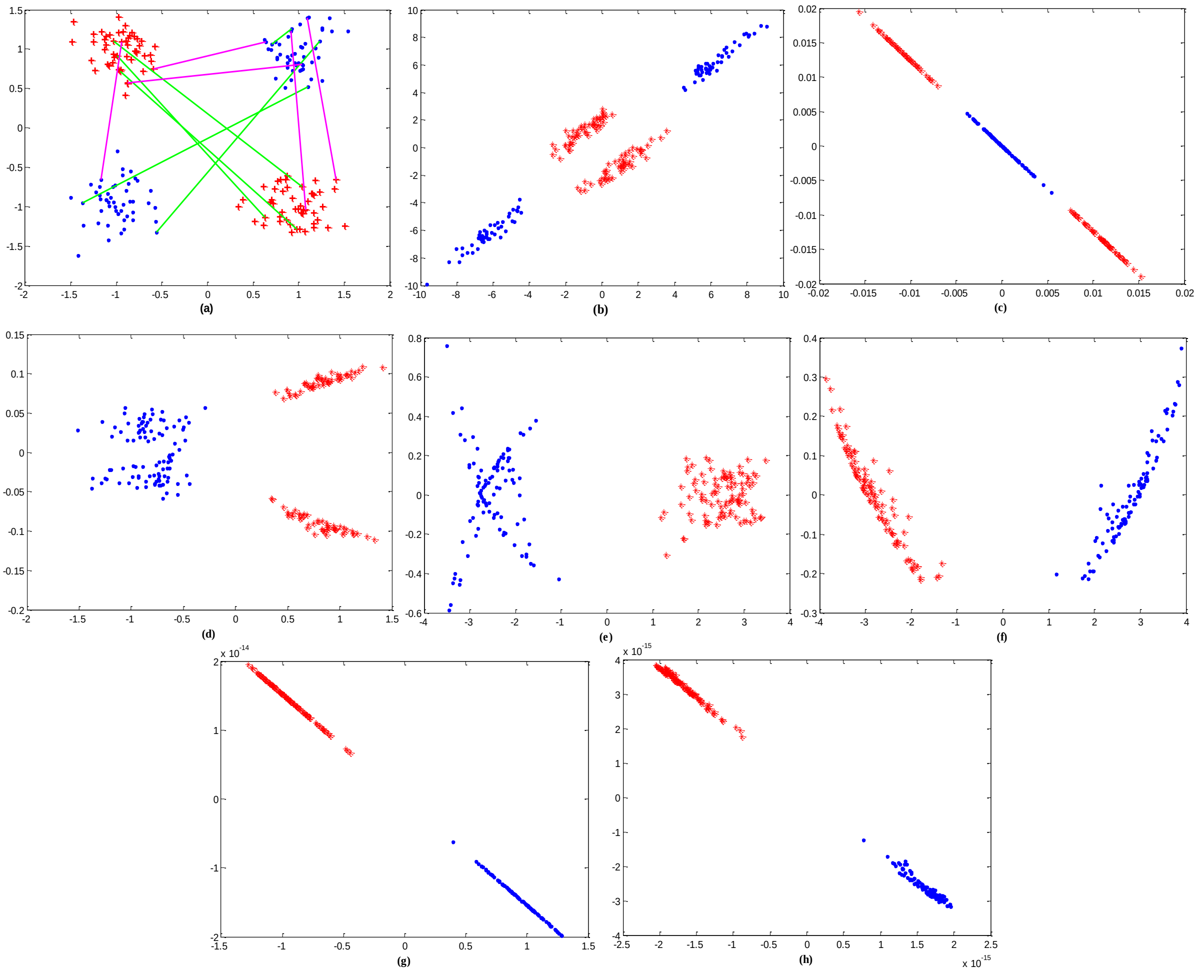

Figure 2. Comparison of different metric learning methods on XOR dataset. (a) Original dataset with two classes, and dataset after applying (b) RCA, (c) Xing's, (d) kernel-β, (e) unweighted-S, (f) unweighted-SD, (g) augmented-S, and (h) augmented-SD.

We also repeat the experiments for the XOR with overlapping clusters. Parameters are set similar to the previous experiment. Figure 5 shows new XOR data and the corresponding clustering results of different algorithms. It can be seen that the proposed methods outperform the others. Figure 6 (b-h) shows the new XOR data points projected into the feature spaces induced by the different distance metric-based clustering approaches.

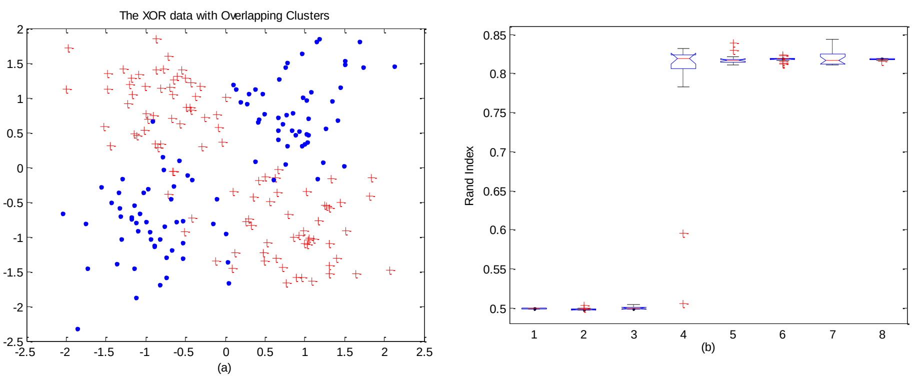

Figure 5. Clustering results of different algorithms for XOR data with overlapping clusters. (a) original data, (b) eight algorithms (numbered in Section 4.1) are as follow: (1) Euclidean, (2) RCA, (3) Xing's, (4) kernel-β, (5) unweighted-*S*, (6) unweighted-*SD*, (7) augmented-*S*, and (8) augmented-*SD*.

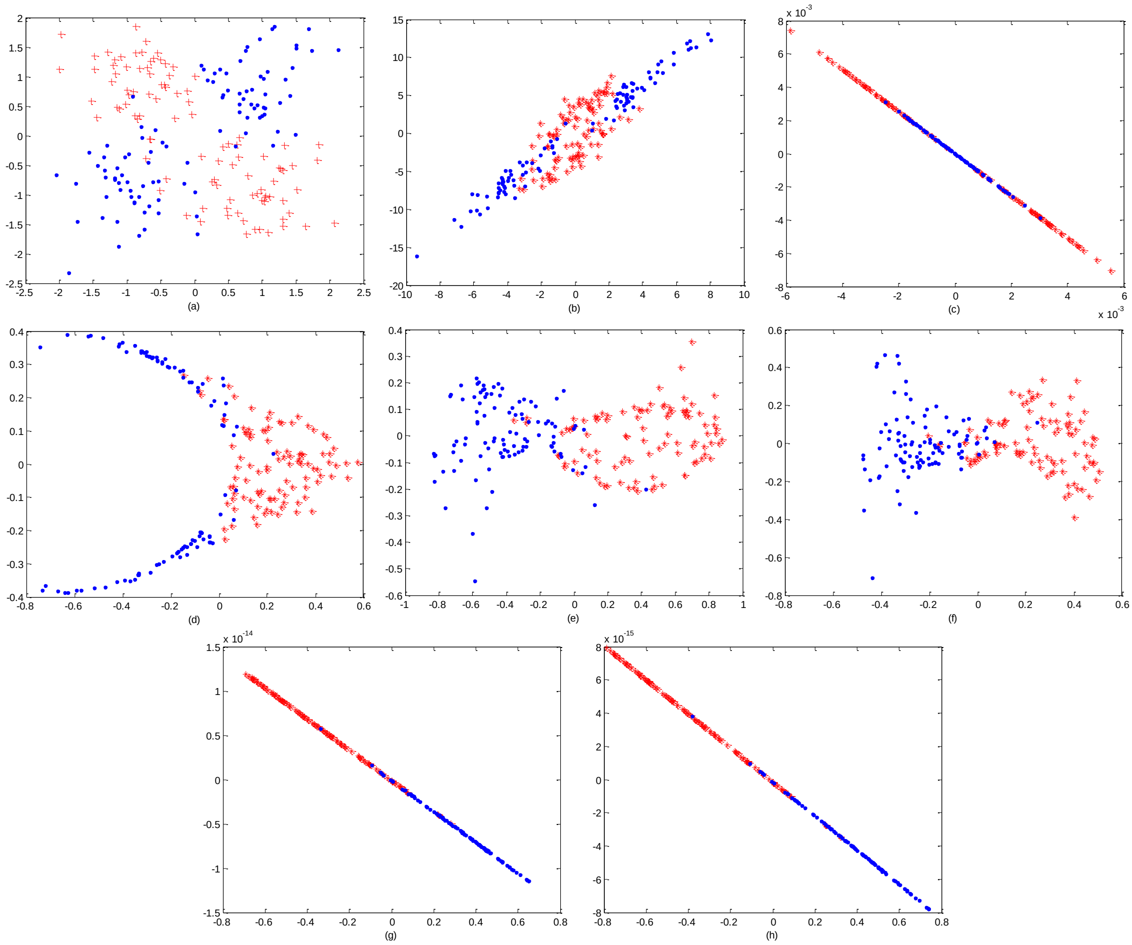

Figure 6. Comparison of different metric learning methods on XOR dataset with overlapping clusters. (a) original dataset with two classes, and dataset after applying (b) RCA, (c) Xing's, (d) kernel-β, (e) unweighted-S, (f) unweighted-SD, (g) augmented-S, and (h) augmented-SD.

4.3. Experiments on UCI Data

We repeated our experiments on six real-world datasets from the University of California at Irvine (UCI) Machine Learning Repository. Some details about the main characteristics of these datasets and our experimental settings are shown in table 2. These details include the size of dataset (*N*), number of features (r_x), number of clusters (*C*), and number of randomly selected similarity and dissimilarity pairs (|*S*| and |*D*|). |*S*| was selected to be the same as that in [21] and [23].

In the case of multiple kernel learning, Gaussian kernels with different parameter values, σ , were utilized. The number of Gaussian kernels (*M*) and the adopted range of this parameter (σ_i) have also been reported in the table. For each dataset, the lower bound of variance value, σ , has been chosen by computing the variance of the associated data points in the original feature space. For each dataset, *M* Gaussian kernels are computed by extracting *M* parameter σ_i from the interval $\left[\frac{1}{2}\sigma, 2\sigma\right]$. In the experiments, the number of Gaussian kernels has been chosen as $M = 11$ for all datasets.

M is randomly selected. The datasets are normalized before applying the clustering algorithms.

The performance of the proposed method may be affected by the value of a few parameters. The main parameters are the number of kernel groups, *G*, and number of the similarity/dissimilarity pairs. Some discussions about the parameter settings are in the following order:

(i) *G*: As mentioned, the number of base kernels is reduced to *G* groups using a kernel alignment procedure. The grouping process reduces the computational complexity of the method, especially in the case of the kernel augmentation method. A number of experiments were performed using different values of *G*. Figure 7 contains samples of the results, where the average Rand Index is plotted versus the number of groups, *G*, for different datasets using the unweighted-*SD* method. These plots demonstrate that the performance of the proposed method is not highly affected by the number of the groups. Therefore, we set $G = 2$ in the experiments.

(ii) |*S*| and |*D*|: The plots in figure 8 show the relationship between the average Rand Index and the number of constraints. These are the results obtained using the unweighted-*SD* method on different datasets. As expected, the performance of the proposed method normally improves when the number of the constraints is increased. Since this study focuses on semi-supervised learning, a limited amount of supervisory information has to be provided. Hence, in the rest of the paper, we report the results obtained using the number of superiority constraints, as specified in table 2.

Figure 9 contains the clustering results on the UCI datasets using different algorithms (numbered as discussed in Section 4.1). From these results, the findings of the paper can be summarized as follow:

1) As expected, the kernel-based methods (4 to 8 in Section 4.1) outperform the simple Euclidean distance measure or the linear metric learning approaches. This is thanks to the benefits obtained by the non-linear mapping induced by the kernel functions.

2) It can be seen that the proposed kernel learning methods (5 to 8 in Section 4.1) generally lead to better results compared to the kernel-β and the other approaches. The main exception is the Wine dataset. As mentioned earlier, in this study, a set of RBF kernels is considered as the primary kernels. Perhaps, there still is a need for including a greater variety of kernels as the base kernels in order to capture the inherent clustering characteristics of different datasets with better efficiency. This is a matter of interest in the future studies. Also in some cases, although the average Rand Index of the proposed methods is higher, a low confidence interval (i.e. high variance) has been observed. As mentioned earlier, for any sets of pairwise similarity/dissimilarity constraints, the k-means clustering process is repeated for 20 times. The high variance problem could be due to the problem of local optima in some runs of the algorithm. This problem can be reduced by automatically selecting the best clustering results using the relevant solutions such as using the silhouette measure [29].

3) Among the advocated approaches (5 to 8 in Section 4.1), the kernel augmentation process usually leads to slightly better results. Also adding the dissimilarity pairs as a part of the supervisory information usually improves the clustering quality.

Table 2. Main characteristics of UCI datasets and associated experimental settings.

Dataset	*N*	r_x	*C*	|*S*|	|**D**|	σ_i
Soybean	47	35	4	10	5	[5-20]
Heart	270	13	2	30	10	[5-20]
Ionosphere	351	34	2	30	10	[5-20]
Wine	178	13	3	20	10	[5-20]
Sonar	208	60	2	30	10	[10-40]
Iris	150	4	3	20	10	[3-12]

4.4. Experiments on MNIST digits dataset

To evaluate the performance of the proposed methods on real datasets, we also performed some experiments on the hand-written digits from the MNIST database[1]. This database contained 60000 hand-written numerical characters as the training set and another 10000 as the test set. All the relevant images were centred and normalized to 28×28 gray level images. In our experiments, the clustering process was performed on three subsets of the numerical characters.

These subsets contained {0,1}, {1,5}, and {1,9} digits, respectively. We randomly picked 200

[1] http://yann.lecun.com/exdb/mnist/

samples for each digit. Also 10 different constraint sets were randomly generated, where each set contained 20 similarity pairs (|*S*| = 20) and 5 dissimilarity pairs (|*D*| = 5). For each pairwise constraint set (*S* and *D*), 20 runs of the k-means clustering algorithm were performed by different random initializations.

The number of groups was set to $G = 2$, and the kernel parameter σ_i varied from 10 to 60 in steps of 5. Thus overall, each clustering experiment was repeated for 200 times, and the mean and standard deviation of the Rand Index were calculated. Table 3 contains the results obtained using different approaches. It can be seen that the proposed optimization process leads to better results.

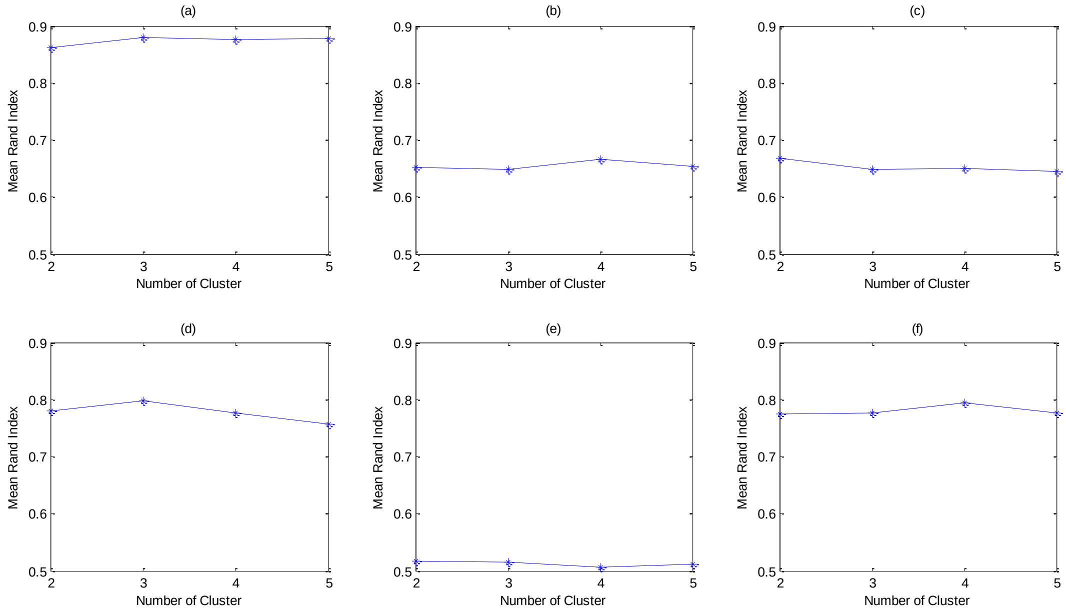

Figure 7. Clustering results of unweighted-*SD* method versus number of groups, *G*, for UCI datasets: (a) Soybean, (b) Heart, (c) Ionosphere, (d) Wine, (e) Sonar, and (f) Iris.

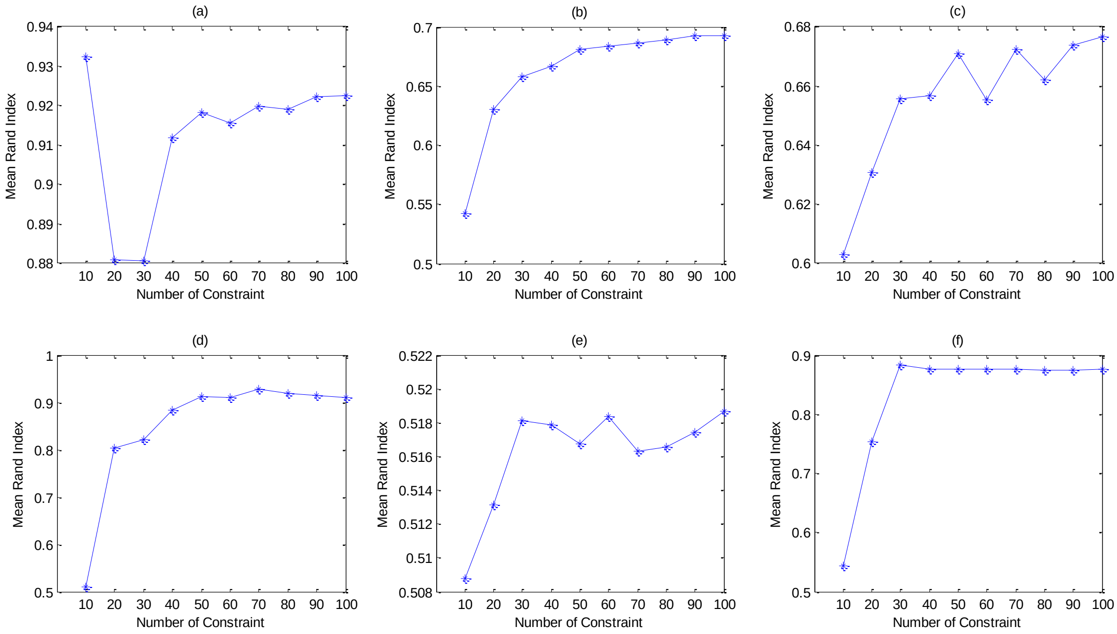

Figure 8. Clustering results of unweighted-*SD* method versus number of constraints for UCI datasets: (a) Soybean, (b) Heart, (c) Ionosphere, (d) Wine, (e) Sonar, and (f) Iris.

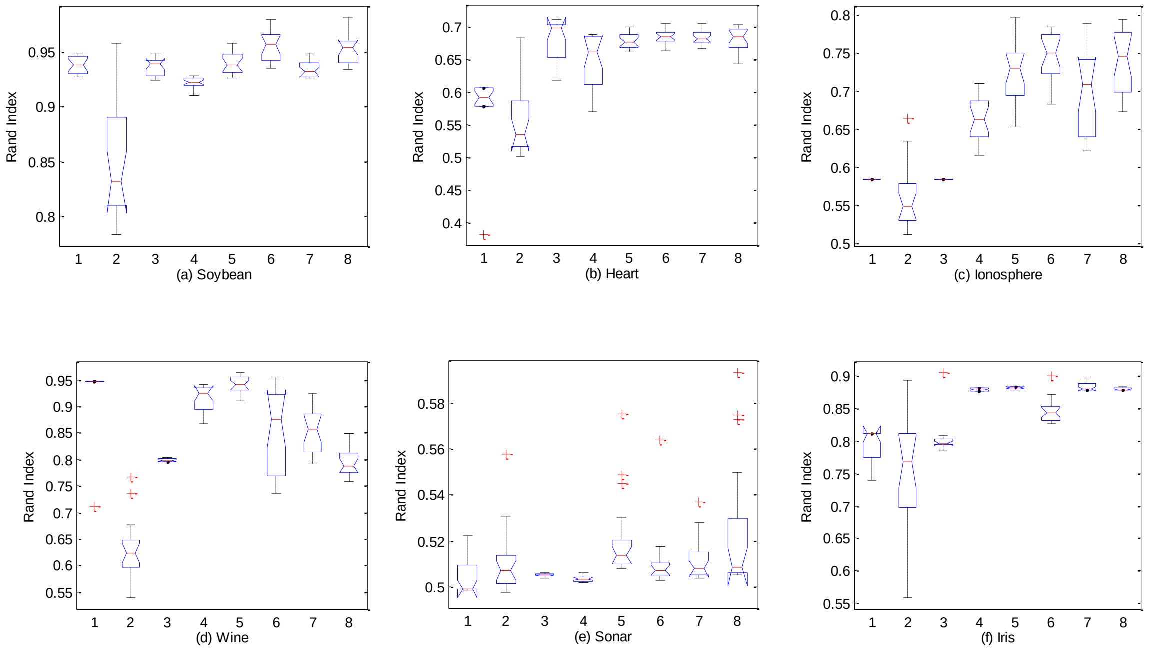

Figure 9. Clustering results of different algorithms for UCI datasets. Algorithms (numbered in Section 4.1) are as follow: (1) Euclidean, (2) RCA, (3) Xing's, (4) kernel-β, (5) unweighted-*S*, (6) unweighted-*SD*, (7) augmented-*S*, and (8) augmented-*SD*.

5. Conclusion

In this paper, we proposed a new metric learning method that makes use of the concept of composite kernels. We formulated a semi-supervised metric learning approach, which utilizes both the similarity and dissimilarity pairwise constraints. Within the framework of the proposed method, the merits of two groups of composite kernels, namely unweighted-average and augmented kernels, were investigated. The learning process concentrates on determining the weights of the eigen-matrices obtained by the eigen-decomposition of the associated composite kernel matrix. In the learning process, a set of similarity and/or dissimilarity constraints have to be jointly satisfied. Our experimental results on the synthetic and real world datasets confirm that overall, the proposed methods are superior to the existing approaches. The proposed methods are suitable for the case that all data points are available in advance. However, extending the proposed methods for dealing with stream data (on-line clustering) could be a matter of interest in the future studies.

Table 3. Clustering results on three subsets of the MNIST dataset.

Subset	Euclidean	RCA	Xing's	kernel-β	unweighted-*S*	unweighted-*SD*	augmented-*S*	augmented-*SD*
{0,1}	0.9781 ±0.0000	0.9812 ±0.0125	0.9850 ±0.0031	0.9851 ±0.0000	0.9900 ±0.0000	0.9950 ±0.0000	0.9888 ±0.0146	0.9851 ±0.0120
{1,5}	0.8249 ±0.0012	0.8314 ±0.0311	0.8401 ±0.0170	0.8528 ±0.0336	0.9003 ±0.0015	0.9184 ±0.0023	0.8731 ±0.0037	0.8839 ±0.0031
{1,9}	0.9524 ±0.0035	0.9531 ±0.0412	0.9549 ±0.0359	0.9366 ±0.0059	0.9611 ±0.0034	0.9760 ±0.0031	0.9550 ±0.0261	0.9588 ±0.0192

6. Acknowledgment

The partial supports of the Ministry of Science, Research, and Technology of Iran, EPSRC/dstl project EP/K014307/1, and the EPSRC project EP/N007743/1 are gratefully acknowledged.

References

[1] Yang, L., Jin, R. (2006). Distance metric learning: a comprehensive survey. Technical Report, Michigan State University.

[2] Yang, L. (2007). An Overview of Distance Metric Learning. Technical Report, Michigan State University.

[3] Domeniconi, C., Peng, J., Gunopulos, D. (2002). Locally adaptive metric nearest neighbor classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 9, pp. 1281-1285.

[4] Domeniconi, C., Gunopulos, D. (2005). Large margin nearest neighbor classifiers. IEEE Transactions on Neural Networks, vol. 16, no.4, pp. 899-909.

[5] Wang, D., Lim, J. S., Han, M. M. & Lee, B. W. (2005) Learning similarity for semantic images classification. Neurocomputing, vol. 6, pp. 363-368.

[6] Goldberger, J., Roweis, S., Hinton, G. & Salakhutdinov, R. (2005) Neighbourhood components analysis. In Advances in Neural Information Processing Systems (NIPS).

[7] Weinberger, K., Blitzer, J. & Saul, L. (2006). Distance metric learning for large margin nearest neighbor classification. In Advances in Neural Information Processing Systems (NIPS).

[8] Saul, L. K. & Roweis, S. T. (2003). Think globally, fit locally: Unsupervised learning of low dimensional manifolds. Journal of Machine Learning Research, vol. 4, pp. 119–155.

[9] Xiang, S., Nie, F. & Zhang, C. (2008). Learning a Mahalanobis distance metric for data clustering and classification. Pattern Recognition, vol. 41, no. 12, pp. 3600-3612.

[10] Kumar, N. & Kummamuru, K. (2007). Semi-supervised clustering with metric learning using relative comparisons. IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 4, pp. 496-503.

[11] Chang, H. & Yeung, D. Y. (2006). Locally linear metric adaptation with application to semi-supervised clustering and image retrieval. Pattern Recognition, vol. 39, no. 7, pp. 1253-1264.

[12] Schultz, M. & Joachims, T. (2004). Learning a distance metric from relative comparisons. In Advances in Neural Information Processing Systems (NIPS).

[13] Xing, EP., Ng, A. Y., Jordan M. I & Russell, S. (2003). Distance metric learning, with application to clustering with side-information. In Advances in Neural Information Processing Systems (NIPS).

[14] Bar-Hillel, A., Hertz, T., Shental, N. & Weinshall. D. (2003). Learning distance functions using equivalence relations. In Proceedings of 20th International Conference on Machine Learning, Washington, DC.

[15] Chang, H. & Yeung, D. Y. (2006). Extending the relevant component analysis algorithm for metric learning using both positive and negative equivalence constraints. Pattern Recognition, vol. 39, no. 5, pp. 1007-1010.

[16] Nguyen, N. & Guo, Y. (2008). Metric Learning: A Support Vector Approach. In Proceedings of European Conference on Machine Learning /PKDD.

[17] Kwok, J. T. & Tsang, I. W. (2003). Learning with idealized kernels. In Proceedings of International Conference on Machine Learning.

[18] Yeung, D. Y. and Chang, H. & Dai, G. (2008). A scalable kernel-based semi-supervised metric learning algorithm with out-of-sample generation ability. Neural computation, vol. 20, no.11, pp. 2839-2861.

[19] Cristianini, N. & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press.

[20] Yan, F., Mikolajczyk, K., Kittler, J. & Tahir, MA. (2010). Combining Multiple Kernels by Augmenting the Kernel Matrix. In Proceedings of Multiple Classifier Systems.

[21] Yeung, D. Y. & Chang, H. (2007). A kernel approach for semisupervised metric learning. IEEE Transactions on Neural Networks, vol. 18, no. 1, pp. 141-149.

[22] Hoi, S. CH, Jin, R. & Lyu, M. R. (2007). Learning nonparametric kernel matrices from pairwise constraints. In Proceedings of 20th International Conference on Machine Learning, New York, USA.

[23] Soleymani Baghshah, M. & Bagheri Shouraki, S. (2010). Kernel-based metric learning for semi-supervised clustering. Neurocomputing, vol. 73, no. 1, pp. 1352-1361.

[24] Wang, J., Do, H., Woznica, A. & Kalousis, A. (2011). Metric learning with multiple kernels. In Advances in Neural Information Processing Systems (NIPS).

[25] Yan, F., Mikolajczyk, K. & Kittler, J. (2011). Multiple Kernel Learning via Distance Metric Learning for Interactive Image Retrieval. In Proceedings of Multiple Classifier Systems.

[26] Zare, T., Sadeghi, M. T. & Abutalebi, H. R. (2012). Semi-supervised Metric Learning Using Composite Kernels. In Proceedings of 6th International Telecommunication symposium (IST).

[27] Cristianini, N., Shawe-Taylor, J., Elisseeff, A. & Kandola, J. (2001). On Kernel-Target Alignment. In Advances in Neural Information Processing Systems (NIPS).

[28] Zhang, D., .Zhou, ZH., & Chen, S. (2007). Semi-supervised dimensionality. In Proceedings of the 7th SIAM International Conference on Data Mining (SDM'07), Minneapolis, MN.

[29] Rousseeuw, P. J. (1987). Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis. Computational and Applied Mathematics.

نشریه هوش مصنوعی و داده کاوی

بهینه‌سازی کرنل مرکب در مسئله یادگیری معیار فاصله نیمه‌نظارتی

طاهره زارع بیدکی[۱*]، محمدتقی صادقی[۱]، حمیدرضا ابوطالبی[۱] و جوزف کیتلر[۲]

[۱] گروه پردازش سیگنال، دانشکده مهندسی برق، دانشگاه یزد، یزد، ایران.

[۲] مرکز پردازش سیگنال، گفتار و بینایی ، دانشگاه ساری، گیلفورد، انگلستان.

ارسال ۲۰۱۶/۰۲/۲۰؛ بازنگری ۲۰۱۶/۱۰/۱۰؛ پذیرش ۲۰۱۷/۰۲/۱۴

چکیده:

حل مبتنی بر یادگیری ماشین در مسائلی نظیر طبقه‌بندی، خوشه بندی و انطباق داده‌ها، به شدت وابسته به معیار فاصله به‌کار گرفته شده است که در گذشته این معیار عموما به صورت ابتکاری تعیین می‌شد. در دهه گذشته، نشان داده شده است که می‌توان از روی نمونه‌های آموزشی، معیار فاصله مناسبی را آموزش داد که منجر به عملکرد بهتر الگوریتم در مقایسه با معیارهای فاصله معمول شود. از این‌رو در سال‌های اخیر، یادگیری معیار فاصله مورد توجه بسیاری از محققین قرار گرفته است که به طور خاص می‌توان به الگوریتم‌های یادگیری معیار فاصله مبتنی بر کرنل اشاره نمود. در این الگوریتم‌ها، با استفاده از تابع کرنل، نمونه‌ها به طور غیر صریح به فضای ویژگی جدیدی با ابعاد بالاتر نگاشت یافته و سپس در این فضای ویژگی جدید، معیار فاصله برای کاربرد مورد نظر آموزش داده می‌شود. ساختار مسئله بهینه‌سازی برای یادگیری معیار فاصله، به میزان اطلاعات نظارتی موجود برای آموزش وابسته است. در این پژوهش، تمرکز بر روی یادگیری معیار فاصله نیمه نظارتی مبتنی بر کرنل است که در آن، نمونه‌ی آموزشی بدون برچسب هستند به جز مجموعه کوچکی از داده‌ها که در قالب زوج نمونه‌های متعلق به کلاس (یا خوشه) مشابه و یا زوج نمونه‌های متعلق به دو کلاس (یا دو خوشه) متفاوت وجود دارند. در الگوریتم پیشنهادی، فرض می‌شود تعداد زیادی کرنل پایه وجود دارد که ابتدا این کرنل‌ها گروه‌بندی شده و با استفاده از نماینده این گروه‌ها، کرنل مرکب تشکیل می‌شود. سپس بسط ماتریس کرنل مرکب بر روی مجموعه‌ای از پایه‌های عمود بر هم محاسبه شده و پارامتر ترکیب در این بسط با استفاده از مجموعه زوج‌های مشابه و نامشابه بهینه می‌شود. الگوریتم پیشنهادی، بر روی مجموعه داده مصنوعی و واقعی مورد ارزیابی قرار گرفت. نتایج نشان می‌دهد که استفاده همزمان از زوج‌های مشابه و نامشابه سبب بهبود عملکرد الگوریتم در مقایسه با استفاده از تنها زوج‌های مشابه می‌شود. همچنین نتایج به دست آمده حاکی از برتری الگوریتم پیشنهادی در مقایسه با سایر الگوریتم‌های مورد بررسی است.

کلمات کلیدی: یادگیری معیار فاصله، خوشه‌بندی نیمه نظارتی، کرنل‌های مرکب، قیود زوج‌های مشابه و نامشابه، مسئله بهینه‌سازی.