Two dimensional Large Margin Nearest Neighbor for Matrix Classification

Kun Song[1], **Feiping Nie**[2], **Junwei Han**[1]
[1]School of Automation, Northwestern Polytechnical University, Xi'an, 710072, Shaanxi, P. R. China
[2]School of Computer Science, Northwestern Polytechnical University, Xi'an, 710072, P. R. China
{songkun123000, feipingnie, junweihan2010 }@gmail.com

Abstract

Matrices are a common form of data encountered in a wide range of real applications. How to efficiently classify this kind of data is an important research topic. In this paper, we propose a novel distance metric learning method named two dimensional large margin nearest neighbor (2DLMNN), for improving the performance of k-nearest neighbor (KNN) classifier in matrix classification. Different from traditional metric learning algorithms, our method employs a left projection matrix U and a right projection matrix V to define the matrix-based Mahalanobis distance, for constructing the objective aimed at separating points in different classes by a large margin. Since the parameters in those two projection matrices are much less than that in its vector-based counterpart, 2DLMNN can reduce computational complexity and the risks of overfitting. We also introduce a framework for solving the proposed 2DLMNN. The convergence behavior, computational complexity are also analyzed. At last, promising experimental results on several data sets are provided to show the effectiveness of our method.

1 Introduction

Matrix data is a common form of data that appears in a wide range of real applications. Examples of the data in matrix form include the gray-level images [Kong *et al.*, 2005; Yang *et al.*, 2013; You *et al.*, 2004; Lu *et al.*, 2016a; Lu *et al.*, 2016b] in computer vision, multichannel EEG signals in biomedical engineering [Tomioka *et al.*, 2006], and so on. In the fields of pattern recognition and computer vision, matrix data classification has essentially become one of the most important topics.

Many classification techniques have been proposed in past decades. Among those classification methods, k-nearest neighbor classifier (KNN) is one of the most popular ones since it has following advantages: the non-linearity of decision boundaries leads to good recognition performance and the computational complexity is independent of the number of classes. In addition, KNN can be applied in scenarios where not all categories are given at the time of training, such as, in face verification applications where the subjects to be recognized are unknown in advance.

The classification accuracy of KNN depends significantly on the distance metric which is used for determining k nearest neighbors[Xiang *et al.*, 2008; Song *et al.*, 2017; Shakhnarovich *et al.*, 2006; Weinberger *et al.*, 2005]. Usually, Euclidean distance is adopted in many cases. This is logical when the statistics of data is unknown or when all dimensions of samples are equally relevant. However, those assumptions are too strict. In most cases, distance analysis along some special directions of the feature space is more informative. In those cases, performing distance metric learning algorithm on data set can improve the performance of KNN greatly.

Traditional distance metric learning methods are designed for dealing with vectors. When they are applied to process matrix data, the matrix data should be previously collapsed into vectors. Normally, this is done by cascading each column of a matrix one by one. However, the spatial correlations of matrix are lost inevitably in this procedure. Moreover, the dimensionality of resulting vector may be very high. In this scenarios, traditional metric learning algorithm maybe can not be carried out due to the overfitting and high computational complexity.

To handle those problems, one feasible way is to use matrix-based feature reduction algorithms to transform the matrix data to low-dimensional vectors, and then to learn a distance metric on the low-dimensional resulting feature space by implementing the vector-based metric learning method. However, the feature reduction may lost some discriminate information of the data set which is crucial for classification. And there is literature revealing that performance of combining feature reduction and metric learning together would be better than that of the two-step solution [Brockmeier *et al.*, 2013; Parameswaran and Weinberger, 2010; Song *et al.*, 2017]. In addition, the two-step solution may involve too much computational cost. Recently, many tensor-based methods for matrix classification are proposed [Chang *et al.*, 2016; Hou *et al.*, 2014; Zhang and Chow, 2012; Nie and Xianga, 2009]. Those literatures prove that matrix-based methods are more efficient than vector-based methods for dealing with matrix data. This inspires us to develop matrix-based metric learning method.

In this paper, we propose a novel metric learning algorithm named two dimensional large margin nearest neighbor (2DLMNN) which can deal with matrix data directly. The 2DLMNN can be seen as the matrix extension of one dimensional large margin nearest neighbor (1DLMNN). The key difference between 2DLMNN and 1DLMNN lies in the model for data presentation. 2DLMNN works with matrix data, while 1DLMNN works with vector data. In the proposed method, we define a new kind of Mahalanobis distance special for matrix data, by using two projection matrices. Then by applying the proposed matrix-based Mahalanobis distance to the principle of 1DLMNN, we obtain the model of matrix-based large margin nearest nearest neighbor (2DLMNN). In addition, we propose a framework for solving the 2DLMNN. Convergence and computational complexity are discussed. Last, Several experiments on various data sets are conducted. The promising experimental results demonstrate the efficiency of the proposed method.

2 Related Works

2.1 One Dimensional Large Margin Nearest Neighbor

One dimensional large margin nearest neighbor (1DLMNN) was first proposed in literature [Weinberger *et al.*, 2005] to learn the distance metric for improving the performance of KNN. Then in literature [Torresani and Lee, 2006], the algorithm of 1DLMNN was extended to learn the low rank projection matrix for feature reduction. In this paper, we consider both of the two tasks of 1DLMNN.

Given l labeled examples $\{(\vec{x_i}, y_i)\}_{i=1}^{l}$, where $\vec{x_i} \in R^d$ and $y_i \in \{1, 2, \cdots, c\}$ is the corresponding label, c is the class number. For an inquiry input $\vec{x_i}$, we call its k-nearest neighbors which have the same labels with $\vec{x_i}$ as "**target neighbors**", and the rest samples whose labels are different from y_i in the dataset as "**imposters**". The goal of 1DLMNN is to learn a linear transformation $\mathcal{L} : R^d \rightarrow R^{d_1} (d_1 \leq d)$, which can be used to define the Mahalanobis distance as:

$$\mathcal{D}(\vec{x_i},\vec{x_j})=\|L(\vec{x_i}-\vec{x_j})\|_2=\sqrt{(\vec{x_i}-\vec{x_j})^T M(\vec{x_i}-\vec{x_j})} \quad (1)$$

where L is the projection matrix of linear transformation $\mathcal{L}$, and $M = L^T L$.

In the projection space determined by the projection $\mathcal{L}$, each sample $L\vec{x_i}$ is closer to its target neighbors than the imposters by one distance unit, i.e. large margin. The relationship of $\vec{x_i}$'s target neighbor $\vec{x_j}$ and imposter $\vec{x_l}$ can be expressed as linear inequality constraint with respect to the $\mathcal{D}(\cdot, \cdot)$:

$$\mathcal{D}^2(\vec{x_i},\vec{x_l}) - \mathcal{D}^2(\vec{x_i},\vec{x_j}) \geq 1 \quad (2)$$

To achieve this goal, the loss function of 1DLMNN is defined as

$$\begin{aligned} \epsilon(M) = \sum_{ij} \eta_{ij}\mathcal{D}^2(\vec{x_i},\vec{x_j}) + \\ \lambda \sum_{i,j,l} \eta_{ij}(1-y_{il})[1 + \mathcal{D}^2(\vec{x_i},\vec{x_j}) - \mathcal{D}^2(\vec{x_i},\vec{x_l})]_+ \end{aligned} \quad (3)$$

where $\eta_{ij} = 1$ if the example $\vec{x_j}$ is one of the k target neighbors of $\vec{x_i}$, otherwise, $\eta_{ij} = 0$. $y_{il} \in \{0, 1\}$ is 1 if and only if $y_i = y_l$, $[z]_+ = max(z, 0)$ denotes the standard hinge loss and λ is a positive constant playing the role in balancing the effect of the two terms in Eq.(3). The optimization problem for minimizing objective in Eq.(3) can be solved by standard semi-definite programming solver, or the sub-gradient descent method.

2.2 Two Dimensional Linear Discriminant Analysis

2DLDA is one of the most important two dimensional subspace learning methods. It can be seen as the two dimensional extension of the vector-based linear discriminant analysis (1DLDA). For a matrix data set $\{X_i\}_{i=1}^{l}$ with c classes, 2DLDA aims at finding two projection matrices L and R, to project X_i to its low dimensional embedding, i.e. $Y = L^T X_i R$. Let M_j denote the j-th class, which has l_j samples. So, $\overline{X}_j = (1/l_j)\sum_{X_i \in M_j} X_i$ is the mean of samples in the j-th class, and $\overline{X} = 1/l \sum X_i$ is the mean of the whole training set. The between-class distance D_b and within-class distance D_w in the original space are defined as follows.

$$\begin{aligned} D_w &= \sum_{j=1}^{c} \sum_{X_i \in M_j} \|X_i - \bar{X}_j\|_F^2 \\ D_b &= \sum_{j=1}^{c} l_j \|\bar{X}_j - \bar{X}\|_F^2 \end{aligned} \quad (4)$$

Meanwhile, in the projection space, we can also define the within-class distance $\tilde{D}_w$ and between-class distance $\tilde{D}_b$.

$$\begin{aligned} \tilde{D}_w &= Tr(\sum_{j=1}^{c} \sum_{X_i \in M_j} L^T(X_i - \bar{X}_j)RR^T(X_i - \bar{X}_j)L) \\ \tilde{D}_b &= Tr(\sum_{j=1}^{c} L^T(\bar{X}_j - \bar{X})RR^T(\bar{X}_j - \bar{X})^T L) \end{aligned} \quad (5)$$

Thus, the optimal transformation matrices L and R can be solved by maximizing $\tilde{D}_w$ and minimizing $\tilde{D}_b$, simultaneously. As it is difficult to derive the optimal L and R simultaneously, 2DLDA solves L and R in an alternative way. In each alternative step, L and R are calculated by performing eigen-decomposition, respectively.

3 Two Dimensional Large Margin Nearest Neighbor(2DLMNN)

3.1 Useful Conceptions

Before giving the formulation of 2DLMNN, several useful conceptions are introduced.

Definition 1: Suppose $R^{m \times n}$ is a matrix linear space, the inner product operation between two matrices $Y, X \in R^{m \times n}$ is defined as $< X, Y >= Tr(X^T Y)$, where $Tr(\cdot)$ is the trace of matrix $\cdot$.

Definition 2: Given a matrix linear space $R^{m \times n}$ with inner product defined by **Definition 1**, the distance between X_1, $X_2 \in R^{m \times n}$ is defined as $D(X_1, X_2) = \sqrt{< X_1 - X_2, X_1 - X_2 >}$.

As 2DLDA, we consider the projection $Y = U^T XV$, where $U \in R^{l_1 \times m}$ and $V \in R^{l_2 \times n}$ are projection matrices to project matrix $X \in R^{m \times n}$ from $R^{m \times n}$ to $R^{l_1 \times l_2}$. In the projected space, the distance between X_1 and X_2 is presented as follows.

$$\begin{aligned} D_{U,V}(X_1, X_1) &= \sqrt{<U^T(X_1 - X_2)V, U^T(X_1 - X_2)V>} \\ &= \sqrt{Tr(M_1(X_1 - X_2)M_2(X_1 - X_2)^T)} \end{aligned} \tag{6}$$

where $M_1 = UU^T$ and $M_2 = VV^T$. When Eq.(6) is seen from original space, it is called **the matrix-based Mahalanobis distance**. When U and V are identity matrices, E-q.(6) is called as matrix-based Euclidean distance.

It is easily to find that the matrix-based Mahalanobis distance $D_{U,V}(X, Y)$ subjected to following properties.

1. $D_{U,V}(X, Y) = D_{U,V}(Y, X)$;
2. $D_{U,V}(X, Y) \geq 0$;
3. $D_{U,V}(X, Y) + D_{U,V}(Y, Z) \geq D_{U,V}(X, Z)$;

3.2 Problem Formulation

Given a matrix data set $\{(X_i, y_i)\}_i^n$, $X_i \in R^{m \times n}$ and $y_i \in \{1, \cdots, c\}$ is the label of X_i. $U \in R^{m \times l_1} (l_1 \leq m)$ and $V \in R^{n \times l_2} (l_2 \leq n)$ are left projection matrix and right projection matrix, respectively. The optimization problem of 2DLMNN is defined as follows.

$$\begin{aligned} \min_{U,V} \lambda \sum_{i,j,l} \eta_{ij}(1 - y_{il})[1 + D^2_{U,V}(X_i, X_j) - D^2_{U,V}(X_i, X_l)]_+ \\ + \sum_{i,j} \eta_{ij} D^2_{U,V}(X_i, X_j) \end{aligned} \tag{7}$$

where $\eta_{ij} = 1$ if example X_j is one of the k target neighbors of X_i, otherwise, $\eta_{ij} = 0$. λ is a positive constant used to balance the two terms in the objective. $y_{il} \in \{0, 1\}$ is 1 if and only if $y_i = y_l$, and $[s]_+ = max(s, 0)$ is the hinge function. It should be noticed that, the "target neighbors" and the "imposters" of each X_i are determined under the matrix-based Euclidean distance in advance.

The objective of the optimization problem (7) consists of two contrasting terms. The first term aims at pushing each sample X_i apart from other points with labels different from y_i by an amount equal to 1 plus the distance from X_i to any of its k similarly-labeled closest points in the projected space. The second term encourages pulling each example X_i and its k target neighbors determined in the original input space closer. The first term corresponds to a margin condition similar to that of SVMs and it is used to improve generalization. The constant λ controls the relative importance of these two competing terms and it can be chosen via cross validation.

Considering the projection $Y = U^T XV$, each row of U can be seen as one coefficient vector of columns in matrix X, therefore, the left projection matrix U capturing the distance information in rows of X. Similarly, right projection matrix V can be seen as the coefficient matrix of columns in matrix X, and it captures the distance information of columns of X. Since in the projections, the places of elements in the matrix X are not changed, it means that the spatial correlations of elements in the matrix X is preserved.

Upon optimization of the problem, test example X_t is classified according to the KNN rule applied to its projection $X_t = U^T X_t V$, by using the matrix-based Euclidean distance.

3.3 Optimization

In this section, we give the framework for solving the proposed method. Since it is difficult to solve the two projection matrices U and V in the objective (7) simultaneously, we derive an iterative algorithm in the following. More specially, for a fixed V, we can compute the optimal U by solving an optimization problem similar to the model of 1DLMNN. With the computed V fixed, we then update U by solving another optimization problem also similar to 1DLMNN. This procedure is repeated with a certain number of times as discussed in Algorithm 1.

Computation of U

Let $V \in R^{n \times l_2}$ fixed, we denote $Q_i = X_i V$, $i = 1, \cdots, l$. The matrix-based Mahalanobis distance between X_i and X_j is $\sqrt{Tr(U^T(Q_i - Q_j)(Q_i - Q_j)^T U)}$. The formulation in Eq.(7) is rewritten as

$$\begin{aligned} \min_{U} \lambda \sum_{i,j,l} \eta_{ij}(1 - y_{il})[1 + D^2_U(Q_i, Q_j) - D^2_U(Q_i, Q_l)]_+ \\ + \sum_{j,i} \eta_{ij} D^2_U(Q_i, Q_j) \end{aligned} \tag{8}$$

where $D^2_U(Q_i, Q_j) = Tr(M_1(Q_i - Q_j)(Q_i - Q_j)^T)$, and $M_1 = UU^T$.

If the vector distance in Eq.(6) is transformed into matrix form as

$$D(\vec{x_i}, \vec{x_j}) = \sqrt{Tr(M(\vec{x_i} - \vec{x_j})(\vec{x_i} - \vec{x_j})^T)} \tag{9}$$

, we can find that $D^2_U(Q_i, Q_j)$ and $D^2(\vec{x_i}, \vec{x_j})$ share the same formula. That means the optimization problem in the Eq.(8) is same with the one in Eq.(3) in shape. Therefore the solution method of Eq.(3) can be applied to that of Eq.(8).

In Eq.(3), optimization problem of the parameter M can be solved as a standard semi-definite programming model [Weinberger and Saul, 2009]. In Eq.(8), when $l_1 = m$, i.e., M_1 is full rank, the optimization problem in Eq.(8) is a convex problem, and it can also be solved by the method of Eq.(3). However, in our method, we do not constrain the projection matrices U and V to be full rank, that is to say, $M_1 = UU^T$ and $M_2 = VV^T$ might be low rank. Since low-rank constraint problem is awkward to solve, in our paper, we adopt sub-gradient descend method [Torresani and Lee, 2006] to learn the low rank projection matrices.

Let $\Gamma(U)$ denote the objective function in Eq.(8), the differentiating $\Gamma(U)$ with resect to the projection matrix U^T

gives the following gradient for the update rule:

$$
\begin{aligned}
\frac{\partial\Gamma(U)}{\partial U^T} = & 2U^T\sum_{j,i}\eta_{ij}(Q_i – Q_j)(Q_i – Q_j)^T + \\
& 2\lambda U^T\sum_{i,j,l}\eta_{ij}(1-y_{il})[(Q_i – Q_j)(Q_i – Q_j)^T \\
& -(Q_i-Q_l)(Q_i-Q_l)^T]h'(t) \\
& t = D_U^2(Q_i,Q_j)-D_U^2(Q_i,Q_l)+1
\end{aligned} \tag{10}
$$

where $h(s) = max\{s, 0\}$, and at $s = 0$. We let $h'(s) = 0$ by adopting a smooth hinge function as in literature [Rennie and Srebro, 2005].

Computation of *V*

Similarly, when $U \in R^{m\times l_1}$ is fixed, we can change the formula of our optimization problem in Eq. (7), and derive its V by solving another model similar to 1DLMNN. More concretely, by denoting $R_i = U^T X_i$, $i = 1, \cdots, l$, Eq.(7) is transformed into Eq.(11).

$$
\begin{aligned}
\min_V \lambda\sum_{i,j,l}\eta_{ij}(1-y_{il})[1+D_V^2(R_i,R_j)-D_V^2(R_i,R_l)]_+ \\
+\sum_{j,i}\eta_{ij}D_V^2(R_i,R_j)
\end{aligned} \tag{11}
$$

where $D_V^2(R_i,R_j) = Tr(V^T(R_i-R_j)^T(R_i-R_j)V)$. Thus, the computation of V is transformed into solving another optimization problem similar to 1DLMNN. Eq.(11) is also solved by sub-gradient descent method.

It should be noticed that, when U and V are not full rank, the solution of Eq.(8) or solution of Eq.(11) is closely related to the initial searching point. To make the alternative method converge, the initial searching point of U or V should be selected as the result solved in previous iteration step. The details are described in Algorithm. 1.

3.4 Convergence Analysis

As mentioned above, the proposed method is solved in an alternative way. Namely, we fix one variable and compute the other. The following proposition shows the convergence of the proposed method.

Proposition 1: In Algorithm 1, we use $\{U^{(s)}, V^{(s)}\}$ to denote the solution at the s-th iteration, the objective function of the optimization problem defined in Eq.(7) at$\{U^{(s)}, V^{(s)}\}$ is $\Gamma(U^{(s)}, V^{(s)})$. So $\Gamma(U^{(s)}, V^{(s)})$ monotonically decreases with growth of iteration number s.

Proof: Since $U^{(s)}$,$V^{(s)}$ are the solution at the s-th iteration calculated by an alternative way. $U^{(s)} = argmin_U(\Gamma(U, V^{(s-1)}))$ with initial searching point $U^{(s-1)}$. $V^{(s)} = argmin_V(\Gamma(U^{(s)}, V))$ with initial searching point $V^{(s-1)}$. So there are $\Gamma(U^{(s)}, V^{(s-1)}) < \Gamma(U^{(s-1)}, V^{(s-1)})$ and $\Gamma(U^{(s)}, V^{(s)}) < \Gamma(U^{(s)}, V^{(s-1)})$. Therefore, $\Gamma(U^{(s)}, V^{(s)}) < \Gamma(U^{(s-1)}, V^{(s-1)})$.■

Since $\Gamma(U^{(s)}, V^{(s)}) > 0$, the series $\Gamma(U^{(s)}, V^{(s)})$ would converge to a constant with s increasing. It means Algorithm 1 converges.

Algorithm 1 2DLMNN

```
Input:
  1) Matrix data $X_i|, i = 1, 2, \cdots, l\}$
  2) Initial matrix $V^{(0)}$,$U^{(0)}$,
  3) Parameter $\lambda$, $l_1$ and $l_2$, maximum iteration number $s_{max}$
Output:
  1) Left projection matrix $U$,
  2) Right projection matrix $V$
Processing:
Initialize $V$ by $V^{(0)}$
Repeat
  a) Update $U$ by $U^{(i)}$: Calculate $Q_i = X_i V^{(i-1)}, i = 1, \cdots, n$,and $U^{(i)}$ by solving optimization problem in
     Eq.(8) by sub-gradient gradient descent method with initial searching point $U^{(i-1)}$.
  b) Update $V$ by $V^{(i)}$: Calculate $R_i = {U^{(i-1)}}^T X_i, i = 1, \cdots, n$, and $V^{(i)}$ by solving optimization problem in
     (11) by sub-gradient gradient descent method with initial searching point $V^{(i-1)}$.
Until Convergence or the maximum number of iterations $s_{max}$ is reached
Return $U$, $V$.
```

3.5 Computational Complexity

One of the motivations to develop the matrix-based method is to reduce the computational complexity. In this section, we consider the computation complexities of 1DLMNN and 2DLMNN. Since different methods would lead to different computational complexity, we suppose both of 2DLMNN and 1DLMNN are solved by sub-gradient descent method without using any accelerating technique. The majority of the computational complexity in sub-gradient descent method is to calculate the sub-gradient of the objective function. Let l_a denote the amount of the iterations in sub-gradient descent. Eq.(8) involves $l_a l_1 nml_2(m+n)kl^2 + lmnl_2$ times of multiplications. So for s_{max} times upgrading U and V, 2DLMNN have $2s_{max}l_a(l_1 nml_2(m+n)kl^2 + l^2 mnl_2)$ times multiplications. Since s_{max}, k are far less than other parameters, $l_1 \approx l_2$ and $m \approx n$, the complexity of 2DLMNN is $O(l_a l_1^2 n^3 l^2)$. Similarly, we can find that the computational complexity of 1DLMNN is $O(l_a d_1 m^3 n^3 l^2)$. Since $l_1^2 \ll d_1 m^3$, the computational complexity of 2DLMNN is much less than that 1DLMNN.

4 Experimental Results

We evaluate our proposed method in four different aspects. The first is about the convergence behavior. The second is the comparison of classification accuracy. The third is to explore the ability of feature reduction of 2DLMNN. The last is to compare the computational consumption of the proposed 2D method with 1DLMNN.

Table 1: Characters of Different Data Sets

Data set	#Size ($l + t$)	#Scale ($m \times n$)	#Class
UMIST	575	56×48	20
USPS	2000	16×16	10
AR	2600	64×64	100
POLLEN	630	25×25	7
Extended Yale-B	1368	48×42	38
Coil-100	1440	32×32	100

4.1 Dataset description

We utilize six typical image datasets to evaluate the performance of the proposed method. They are Extended Yale-B database [Belhumeur *et al.*, 1997], AR face database[1], Coil-100[2], USPS handwritten digital database[3], UMIST face database[4] and POLLEN database[5]. The scale of these image databases ranges from 16×16 to 64×64, and the feature dimensions range from 256 to 4096. The details of those datasets are described in Table 1.

4.2 Convergence behavior

We select four datasets to demonstrate the convergence of the proposed iterative algorithm, they are POLLEN, UMIST, AR, Coil-100, respectively. In the experiments, we empirically set $\lambda = 0.5$ [Parameswaran and Weinberger, 2010; Weinberger and Saul, 2009] and initial points $V^{(0)}$ and $U^{(0)}$ are given by performing 2DPCA [Yang *et al.*, 2004]. We conduct the experiments with different dimensions, and the function value results obtained from 1 to 25 iterations are recorded. The curves of objective function are presented in Figure 1.

As seen from Figure 1. the values of objective function described in Eq.(7) decrease with iterations on the four datasets and the convergence is reached within 10 iterations. It identifies the **Proposition 1** in real data and the convergence of our proposed method.

4.3 Classification results

We consider the 2DLMNN not using feature reduction, i.e the projection matrices are square matrices. We compare it with several state-of-art one dimensional supervised distance metric learning methods, i.e. one dimensional large margin nearest neighbor algorithm (1DLMNN), one dimensional sparse compositional metric learning (1DSCML) [Shi and Sha, 2014], one dimensional local distance metric learning (1DLDML) [Yang and Sukthankar, 2006], one dimensional information theory metric learning (1DITML) and one dimensional regressive virtual metric learning (1DLRVML) [Perrot and A., 2015]. In additional, we also compare 2DLMNN with one dimensional support vector machine (1DSVM) and two dimensional support vector machine classifier (2DSVM) [Hou *et al.*, 2014]. The kernel of those SVM based classifiers are adopt as linear kernel, and the rank of 2DSVM is set as 4, empirically. For all metric learning algorithms, the classification are output by KNN. The parameter k in KNN is tuned in $\{1, 2, 3, 4\}$ by cross-validation [Kohavi and others, 1995]. The parameters in 1DLMNN and 2DLMNN are tuned in grid $\{0.1, \cdots, 0.9\}$. Each dataset listed in Tabel 1 is randomly divided into training set, validate set and testing set by ratio $2 : 1 : 1$. Since our method convergence within 10 iterations, the iteration number of proposed method is set as 10. For each algorithm, the average results of 50 runs are recorded. The experimental results shown in Table 2 demonstrate the efficiency of the proposed matrix-based method.

4.4 Feature reduction results

In this section, we explore the performance of the proposed method for feature reduction. Since 2DLDA is the most famous supervised matrix-based feature reduction method. We compare the classification accuracy of 2DLMNN with 2DLDA under different dimensions. For 2DLDA, matrix data was projected into low-dimensional space, then we employ 1DLMNN to learn a full rank metric on the low-dimensional space. The results of both of 2DLDA and 2DLMNN are output by KNN. The datasets are divided into three parts as section 4.3. The parameter k in KNN are tuned in $\{1, 2, 3, 4\}$ by five-fold cross-validation. The parameters 2DLMNN are tuned in grid $\{0.1, \cdots, 0.9\}$. Since our method convergence within 10 iterations, the iteration number of proposed method is set as 10. The feature reduction results are shown as Figure 2.

4.5 Computational complexity

Another motivation for investigating matrix-based methods is reducing the computational complexity of 1DLMNN in manipulating vectorized high-dimensional data. In this section, we compare 2DLMNN with 1DLMNN on the six data sets which have different size and scales. Both of 2DLMNN and 1DLMNN learn full rank metrics. For justice, these methods are implemented in their original formulations, without using other accelerating strategies. And the algorithms are performed in Matlab on Intel(R) i5-6300HQ @ 2.30 HZ. The results are shown in Table 3. We can find that 2DLMNN runs much faster than 1DLMNN.

5 Conclusions

In this paper, we have proposed a novel two dimensional metric learning algorithm for learning matrix-based distance metric. Different from the traditional LMNN, our method solved a left projection matrix and a right projection matrix by an alternative way. The convergence behavior, classification accuracy, and computational complexity of the proposed algorithm were analyzed. At last, comprehensive experimental evaluations have been conducted to demonstrate the effectiveness of the proposed methods.

[1] http://www2.ece.ohio-state.edu/ aleix/ARdatabase.html

[2] http://www1.cs.columbia.edu/CAVE/research/softlib/coil-100.html.

[3] http://www-i6.informatik.rwth-aachen.de/ keysers/usps.html

[4] http://images.ee.umist.ac.uk/danny/database.html

[5] http://ome.grc.nia.nih.gov/iicbu2008/pollen/index.html

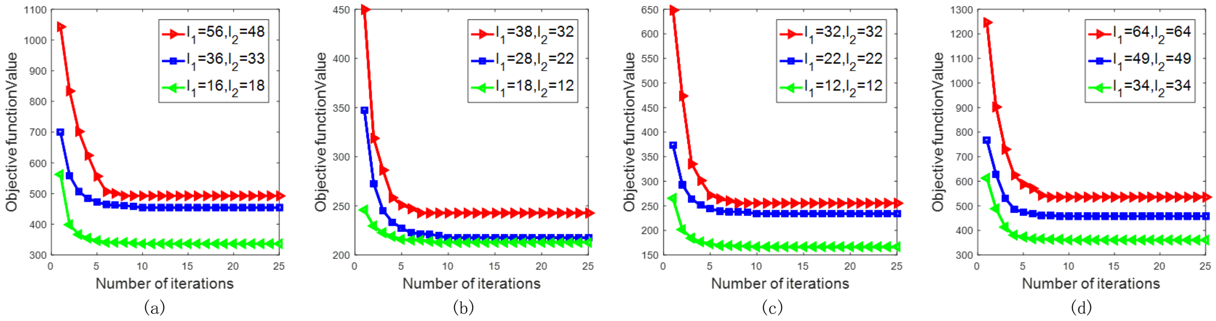

Figure 1: Objective function values of 2DLMNN for different numbers of iterations on four datasets. (a)UMIST dataset; (b)Extended Yale-B; (c)Coil-100; (d)AR face dataset.

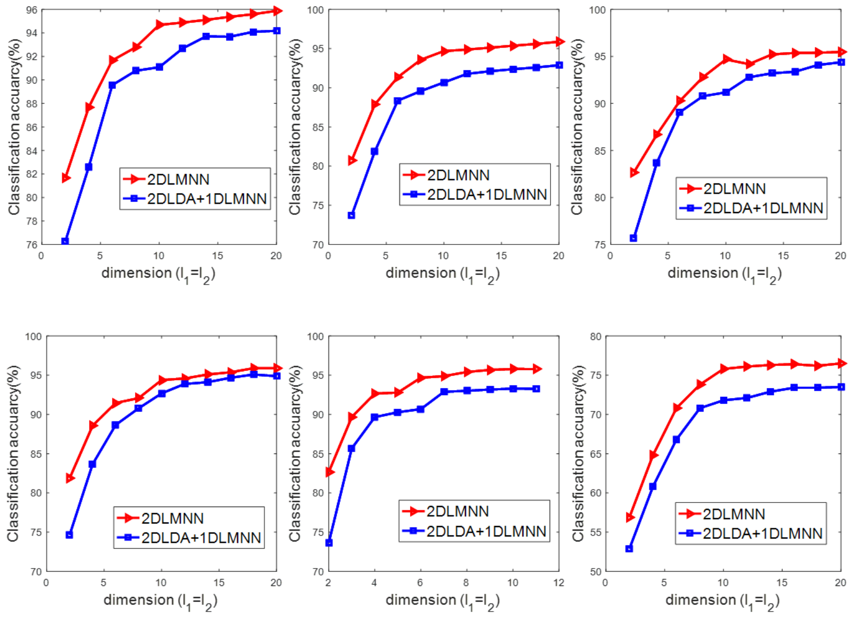

Figure 2: Classification accuracies of different feature dimensions on six data sets. (a) Coil-100; (b) Extended Yale-B; (c)POLLEN; (d) UMIST; (e) USPS; (f) AR.

Table 2: Comparison of our approach 2DLMNN with several baselines (mean ± std %).

Base	1DSVM	kNN	1DLMNN	1DITML	1DLDML	1DSCML	1DRVML	2DSVM	2DLMNN
UMIST	92.12±3.11	82.12±2.31	92.12±2.31	94.23±1.54	93.42±2.52	92.15±2.47	94.32±2.48	93.43±1.34	**96.41±1.23**
Pollen	94.42±2.21	84.12±2.31	95.97±2.41	95.83±3.21	94.83±2.67	93.42±1.49	93.61±2.37	96.40±3.41	**97.45±1.21**
AR	71.49±2.41	64.12±2.31	74.74±1.27	76.43±1.21	75.43±1.28	75.57±1.12	76.13±1.45	77.25±1.61	**77.92±1.54**
Coil-100	96.52±2.03	85.12±2.31	94.63±1.34	96.84±1.56	96.21±2.32	95.32±2.46	95.44±1.54	95.55±1.51	**98.89±1.33**
USPS	92.12±1.51	83.32±1.32	97.28±1.25	95.43±1.21	96.32±2.21	96.46±1.32	96.31±1.21	96.42±1.22	**98.21±1.21**
YB	92.12±2.31	85.12±2.31	95.13±3.22	96.81±1.41	97.12±2.21	96.45±2.11	96.52±1.63	96.42±1.34	**97.36±2.34**

Table 3: Comparison of the computational time of 2DLMNN and 1DLMNN (sec ± std)

Methods	UMIST	Pollen	AR	USPS	Extended Yale-B	Coil-100
2DLMNN	82.12±3.31	43.12±4.32	163.23±4.54	67.42±2.52	146.15±3.47	94.32±2.48
1DLMNN	178.12±6.31	89.47±5.12	324.83±7.21	134.83±2.67	312.42±2.49	212.61±2.37

Acknowledgments

This work was supported in part by the National Science Foundation of China under Grants 61522207 and 61473231.

References

[Belhumeur *et al.*, 1997] Peter N. Belhumeur, João P Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720, 1997.

[Brockmeier *et al.*, 2013] Austin J Brockmeier, Luis G Sanchez Giraldo, Matthew S Emigh, Jihye Bae, John S Choi, Joseph T Francis, and Jose C Principe. Information-theoretic metric learning: 2-d linear projections of neural data for visualization. In *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, pages 5586–5589. IEEE, 2013.

[Chang *et al.*, 2016] Xiaojun Chang, Feiping Nie, Sen Wang, Yi Yang, Xiaofang Zhou, and Chengqi Zhang. Compound rank-k projections for bilinear analysis. *IEEE transactions on neural networks and learning systems*, 27(7):1502–1513, 2016.

[Hou *et al.*, 2014] Chenping Hou, Feiping Nie, Changshui Zhang, Dongyun Yi, and Yi Wu. Multiple rank multilinear svm for matrix data classification. *Pattern Recognition*, 47(1):454–469, 2014.

[Kohavi and others, 1995] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Stanford, CA, 1995.

[Kong *et al.*, 2005] Hui Kong, Lei Wang, Eam Khwang Teoh, J-G Wang, and Ronda Venkateswarlu. A framework of 2d fisher discriminant analysis: application to face recognition with small number of training samples. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 1083–1088, 2005.

[Lu *et al.*, 2016a] X Lu, Y Yuan, and X Zheng. Jointly dictionary learning for change detection in multispectral imagery. *IEEE Transactions on Cybernetics*, 2016.

[Lu *et al.*, 2016b] Xiaoqiang Lu, Xiangtao Zheng, and Xuelong Li. Latent semantic minimal hashing for image retrieval. *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society*, 26(1):355–368, 2016.

[Nie and Xianga, 2009] Feiping Nie and Shiming Xianga. Extracting the optimal dimensionality for local tensor discriminant analysis. *Pattern Recognition*, 42(1):105–114, 2009.

[Parameswaran and Weinberger, 2010] Shibin; Parameswaran and Q. Weinberger, Kilian. Large margin multi-task metric learning. In *Advances in neural information processing systems*, pages 1867–1875, 2010.

[Perrot and A., 2015] M.; Perrot and Habrard A. Regressive virtual metric learning. In *Advances in Neural Information Processing Systems*, pages 1810–1818. 2015.

[Rennie and Srebro, 2005] Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, pages 713–719. ACM, 2005.

[Shakhnarovich *et al.*, 2006] Gregory Shakhnarovich, Piotr Indyk, and Trevor Darrell. *Nearest-neighbor methods in learning and vision: theory and practice*. 2006.

[Shi and Sha, 2014] Bellet A.; Shi, Y. and F. Sha. Sparse compositional metric learning. *arXiv preprint*, page 1404.4105, 2014.

[Song *et al.*, 2017] Kun Song, Feiping Nie, Junwei Han, and Xuelong Li. Parameter free large margin nearest neighbor for distance metric learning. *AAAI*, 2017.

[Tomioka *et al.*, 2006] Ryota Tomioka, Kazuyuki Aihara, and Klaus-Robert Müller. Logistic regression for single trial eeg classification. In *Advances in neural information processing systems*, pages 1377–1384, 2006.

[Torresani and Lee, 2006] Lorenzo; Torresani and Kuang-chih Lee. Large margin component analysis. In *Advances in neural information processing systems*, pages 1385–1392, 2006.

[Weinberger and Saul, 2009] Q.; Weinberger, Kilian and K. Saul, Lawrence. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009.

[Weinberger *et al.*, 2005] Q.; Weinberger, Kilian, John; Blitzer, and K. Saul, Lawrence. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2005.

[Xiang *et al.*, 2008] Shiming Xiang, Feiping Nie, and Changshui Zhang. Learning a mahalanobis distance metric for data clustering and classification. *Pattern Recognition*, 41(12):3600–3612, 2008.

[Yang and Sukthankar, 2006] R.; Yang, L.; Jin and R. Sukthankar. An efficient algorithm for local distance metric learning. *AAAI*, pages 543–548, 2006.

[Yang *et al.*, 2004] Jian Yang, David Zhang, Alejandro F Frangi, and Jing-yu Yang. Two-dimensional pca: a new approach to appearance-based face representation and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 26(1):131–137, 2004.

[Yang *et al.*, 2013] J. Yang, D. Chu, L. Zhang, Y. Xu, and J. Yang. Sparse representation classifier steered discriminative projection with applications to face recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 24(7):1023–1035, July 2013.

[You *et al.*, 2004] Jane You, Wai-Kin Kong, David Zhang, and King Hong Cheung. On hierarchical palmprint coding with multiple features for personal identification in large databases. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(2):234–243, 2004.

[Zhang and Chow, 2012] Zhao Zhang and Tommy WS Chow. Maximum margin multisurface support tensor machines with application to image classification and segmentation. *Expert Systems with Applications*, 39(1):849–860, 2012.