Int. J. Mach. Learn. & Cyber. (2017) 8:793–805
DOI 10.1007/s13042-015-0380-3

ORIGINAL ARTICLE

# Adaptive semi-supervised dimensionality reduction based on pairwise constraints weighting and graph optimizing

**Meng Meng[1] · Jia Wei[1] · Jiabing Wang[1] · Qianli Ma[1] · Xuan Wang[2]**

Received: 17 September 2014 / Accepted: 22 May 2015 / Published online: 2 June 2015


**Abstract** With the rapid growth of high dimensional data, dimensionality reduction is playing a more and more important role in practical data processing and analysing tasks. This paper studies semi-supervised dimensionality reduction using pairwise constraints. In this setting, domain knowledge is given in the form of pairwise constraints, which specifies whether a pair of instances belong to the same class (must-link constraint) or different classes (cannot-link constraint). In this paper, a novel semi-supervised dimensionality reduction method called adaptive semi-supervised dimensionality reduction (ASSDR) is proposed, which can get the optimized low dimensional representation of the original data by adaptively adjusting the weights of the pairwise constraints and simultaneously optimizing the graph construction. Experiments on UCI classification and image recognition show that ASSDR is superior to many existing dimensionality reduction methods.

**Keywords** Adaptive dimensionality reduction · Semi-supervised learning · Pairwise constraints weighting · Graph construction optimizing

✉ Jia Wei
csjwei@scut.edu.cn

[1] School of Computer Science and Engineering, South China University of Technology, Guangzhou, China

[2] Computer Application Research Center, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China

## 1 Introduction

In many practical applications, such as face recognition, information retrieval and bioinformatics, etc, one is often confronted with high dimensional data. However, high dimensionality is a major cause of the practical limitations of many pattern recognition technologies. Specifically, it has been observed that a large number of features may actually degrade the performance of classifiers if the number of training samples is small relative to the number of the features. This is called the "Curse of Dimensionality" [1]. Fortunately, there might be reason to suspect that the naturally generated high dimensional data probably reside on a lower dimensional manifold. This leads one to consider methods of dimensionality reduction that allow one to represent the data in a lower dimensional subspace.

The goal of dimensionality reduction is to reduce the complexity of the input data with some desired intrinsic information of the data preserved. Two of the most popular methods for dimensionality reduction are principal component analysis (PCA) [2, 3] and linear discriminant analysis (LDA) [1, 4], which are unsupervised and supervised respectively. PCA tries to preserve the global covariance structure of the data in a low dimensional projection subspace without knowing the class labels of the data; while LDA aims to minimize the within-class similarity and maximize the between-class similarity simultaneously in a low dimensional projection subspace when the class labels of the data are available.

In recent years, dimensionality reduction in semi-supervised situation has attracted more and more attention [5–7]. In many practical applications such as image segmentation, web page classification and gene-expression clustering [8], a labeling process is costly and time-consuming; in contrast, unlabeled examples can be easily

obtained. Therefore, in such situations, it can be beneficial to incorporate the information which is contained in unlabeled examples into a learning problem, i.e., semi-supervised learning (SSL), instead of supervised learning, should be applied.

However, in many cases, people cannot tell which category an instance belongs to, that is, we do not know the exact label of an instance; what we know is the constraint information of whether a pair of instances belong to the same class (must-link constraint) or different classes (cannot-link constraint) [9, 10]. The above pairwise constraint information is called "Side Information". It can be seen that side information is more general than label information, because we can get side information form label information but it cannot work contrariwise [11]. So learning with side information is becoming an important area in the field of machine learning.

Recently, some related works have been proposed, which make use of the pairwise constraints to extract low dimensional structure in high dimensional data. Bar-Hillel et al. proposed relevant component analysis (RCA) which can make use of the must-link constraints for semi-supervised dimensionality reduction [12]. Xing et al. [13], Tang et al. [14], Yeung et al. [15] and An et al. [16] proposed different constraints based semi-supervised dimensionality reduction methods, which can make use of both the must-link constraints and cannot-link constraints. Zhang et al. proposed semi-supervised dimensionality reduction (SSDR) [17] and Chen et al. used SSDR in hyperspectral image classification recently [18]. SSDR can use the pairwise constraints as well as preserve the global covariance structure of the unlabeled data in the projected low dimensional subspace. Cevikalp et al. proposed constrained locality preserving projections (CLPP) [19] which is the semi-supervised version of LPP [20]. The method can make use of the information provided by the pairwise constraints and can also use the unlabelled data by preserving the local structure used in LPP. Wei et al. proposed neighborhood preserving based semi-supervised dimensionality reduction (NPSSDR) [21] by using the pairwise constraints and preserving the neighborhood structure used in LLE [22]. Baghshah et al. used the idea of NPSSDR in metric learning and used a heuristic search algorithm to solve the proposed constrained trace ratio problem [23]. Davidson proposed a graph driven constrained dimensionality reduction approach GCDR-LP for clustering [24]. In this approach, a constraint graph is firstly created by propagating the constraints due to transitivity and entailment in the graph, and then the dimensionality reduction can be conducted by the constraint graph. Yan et al. proposed a method named dual subspace projections (DSP) [25]. The method first integrates the must-link constraints in the kernel space to get kernel null space and then integrates the cannot-link constraints and the nearby/far-away data structure by using the pairwise distances in the kernel null space to get the transformation matrix of the original input space.

However, a common problem of the aforementioned methods is that the pairwise constraints are equally treated in the algorithms which ignore the fact of unequally amount of information owned by different pairwise constraints. For example, consider a binary-class case in Fig. 1a–d are two must-link constraints of class 1, (e, f) and (g, h) are two cannot-link constraints between class 1 and class 2. It is sound to say that the must-link constraint (c, d) has more information than (a, b), because the distance of (c, d) is larger than that of (a, b), which indicates c and d are more likely to be located on the margin of class 1. On the contrary, the cannot-link constraint (e, f) has more information than (g, h), because the distance of (e, f) is smaller than that of (g, h), which indicates e and f are more likely to be located on the margin between class 1 and class 2. So, it is sensible to handle different pairwise constraints according to different importance.

On the other hand, in order to utilize unlabeled data, most graph-based semi-supervised dimensionality reduction methods (e.g., CLPP and NPSSDR) generally construct a neighborhood graph using the available data. However, such graph is constructed using the nearest neighbor criterion in advance which tends to work poorly due to the high dimensions of the original space, and it is hard to compute appropriate values for the neighborhood size and the adjacency weight matrix involved in graph construction. To solve the problem, one should integrate graph construction with specific semi-supervised

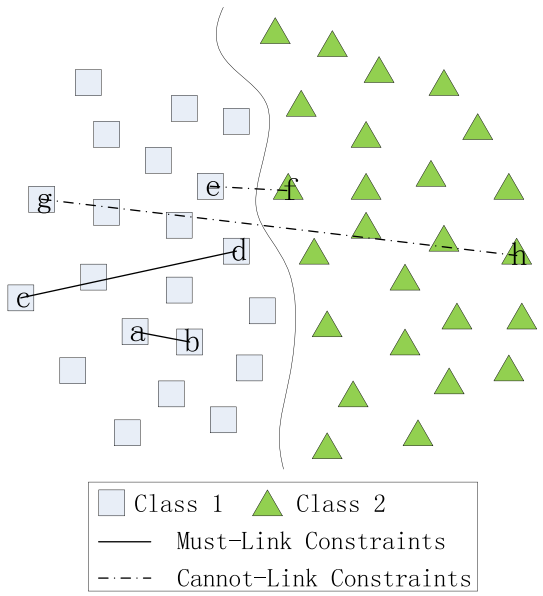


**Fig. 1** Illustration of pairwise constraints

dimensionality reduction process into a unified framework, which results in an optimized graph rather than a predefined one.

In this paper, we first propose a semi-supervised dimensionality reduction method called weighted pairwise constraints based semi-supervised dimensionality reduction (WPCSSDR). Then, a novel semi-supervised dimensionality reduction method called adaptive semi-supervised dimensionality reduction (ASSDR) is proposed which uses WPCSSDR as a subprogram. ASSDR first initialize all the pairwise constraints with equal weights and construct a neighborhood graph with initial adjacency weight matrix, and then the following procedure is repeated until the stop condition is satisfied: (1) reducing the dimensionality of the original space with the current weighted pairwise constraints and the current adjacency weight matrix using WPCSSDR; (2) clustering in the reduced subspace; (3) updating the weights of the pairwise constraints according to the clustering result; (4) updating the adjacency weight matrix. As a result, we can get the optimized weights of the pairwise constraints and the optimized adjacency weight matrix of the neighborhood graph, as well as the projection matrix.

## 2 Adaptive semi-supervised dimensionality reduction algorithm (ASSDR)

### 2.1 The problem

Here we define the weighted pairwise constraints based semi-supervised dimensionality reduction problem as follows: Suppose we have a set of $D$-dimensional data samples $X = \{x_1, x_2, ..., x_n\} \subset R^D$ together with some pairwise must-link constraints ($M$) and cannot-link constraints ($C$) as domain knowledge: $(x_i, x_j) \in M$, if $x_i$ and $x_j$ belong to the same class; $(x_i, x_j) \in C$, if $x_i$ and $x_j$ belong to the different classes. In addition, each pairwise constraint $(x_i, x_j)$ has a weight $S_{ij}$ to indicate the importance of information owned by itself, which means one should be paid more attention to the pairwise constraint $(x_i, x_j)$ if $S_{ij}$ is large. In this case, what we want to do is to find a set of linear projection vectors $W = [w_1, w_2, ..., w_d] \in R^{D\times d}$, where $d << D$, such that the transformed low dimensional projections $Y = \{y_1, y_2, ..., y_n\} \subset R^d$, where $y_i = W^T x_i$, can preserve some properties of the original dataset as well as the pairwise constraints in $M$ and $C$. For the convenience of discussion, one dimensional case is discussed below, namely $y_i = w^T x_i$, which is easy to be extended to the high dimensional case.

### 2.2 Weighted pairwise constraints based semi-supervised dimensionality reduction (WPCSSDR)

To make use of the pairwise constraints, the pairwise points in $M$ should end up close to each other while the pairwise points in $C$ should end up far from each other. This means the instances belonging to the same class in the original space should be close to each other in the reduced subspace, and the instances belonging to different classes in the original space should be far from each other in the reduced subspace. In addition, if $(x_i, x_j) \in M$ and $S_{ij}$ is large, it means the Euclidean distance of $x_i$ and $x_j$ in the low dimension should be smaller to each other than with small weight; if $(x_i, x_j) \in C$ and $S_{ij}$ is large, it means the Euclidean distance of $x_i$ and $x_j$ in the low dimension should be larger from each other than with small weight.

As for the weighted must-link constraints $M$, the intraclass compactness is characterized by the term as follows:

$$\begin{aligned} Q^M(w) &= \sum_{(x_i,x_j)\in M \text{ or } (x_j,x_i)\in M} \left(w^T x_i - w^T x_j\right)^2 S_{ij} \\ &= 2\sum_i \left(w^T x_i D^M_{ii} x_i^T w\right) - 2\sum_{i,j}\left(w^T x_i S^M_{ij} x_j^T w\right) \\ &= 2w^T X(D^M - S^M)X^T w \\ &= 2w^T X L^M X^T w \end{aligned} \tag{1}$$

$$S^M_{ij} = \begin{cases} S_{ij} & (x_i, x_j) \in M \text{ or } (x_j, x_i) \in M \\ 0 & else \end{cases} \tag{2}$$

where $D^M$ is a diagonal matrix whose entries are column sums of $S^M$ (or row sums, since $S^M$ is symmetric), $D^M_{ii} = \sum_j S^M_{ij}$, $L^M = D^M - S^M$ is the Laplacian matrix [26].

$Q^M(w)$ should be as small as possible, which means the weighted distance sum in the transformed low dimensional subspace between instances involved in the must-link constraints $M$ should be small.

On the other hand, the interclass separability of the weighted cannot-link constraints $C$ can be characterized by the term:

$$\begin{aligned} Q^C(w) &= \sum_{(x_i,x_j)\in C \text{ or } (x_j,x_i)\in C} \left(w^T x_i - w^T x_j\right)^2 S_{ij} \\ &= 2\sum_i \left(w^T x_i D^C_{ii} x_i^T w\right) - 2\sum_{i,j}\left(w^T x_i S^C_{ij} x_j^T w\right) \\ &= 2w^T X(D^C - S^C)X^T w \\ &= 2w^T X L^C X^T w \end{aligned} \tag{3}$$

$$S_{ij}^C = \begin{cases} S_{ij} & (x_i, x_j) \in Cor \ (x_j, x_i) \in C \\ 0 & else \end{cases} \tag{4}$$

where $D^C$ is a diagonal matrix, $D_{ii}^C = \sum_j S_{ij}^C$, $L^C = D^C - S^C$.

$Q^C(w)$ should be as large as possible, which means the weighted distance sum in the transformed low dimensional subspace between instances involved in the cannot-link constraints $C$ should be large.

With the above preparation, we can define the objective function of the dimensionality reduction as maximizing the following equation:

$$J(w) = \frac{1}{2}\left(\frac{1}{n_C} Q^C(w) - \frac{\alpha}{n_M} Q^M(w)\right) \tag{5}$$

where $n_C$ and $n_M$ are the number of the cannot-link constraints and the must-link constraints, respectively. $\alpha$ is the scaling parameter to balance the contribution of the must-link constraints.

However, Equation (5) considers only the pairwise constraints. When there are abundant unlabeled samples in the semi-supervised case, Equation (5) should be extended such that both the pairwise constraints and the unlabeled samples can be used. Here, we use the idea proposed in LPP [20] to utilize the unlabeled samples in the semi-supervised case. Given the data samples $X = \{x_1, x_2, ..., x_n\}$, the geometric structure of the data can be modeled by a $k$-nearest neighbor graph $G = \{X, P\}$ with the vertex set $X$ and the affinity weight matrix $P$. More specifically, nodes $x_i$ and $x_j$ are linked by an edge if $x_i$ is among the $k$-nearest neighbors of $x_j$ or $x_j$ is among the $k$-nearest neighbors of $x_i$. Then, the weights of these edges are assigned by the heat kernel function $P_{ij} = \exp(-\|x_i - x_j\|^2/2\sigma^2)$ or assigned by $P_{ij} = 1$ simple-minded which avoids the necessity of choosing $\sigma$. According to [20], the object of LPP is to minimize the following function:

$$\begin{aligned} Q^L(w) &= \sum_{i,j} \left(w^T x_i - w^T x_j\right)^2 P_{ij} \\ &= 2w^T X(D^L - P)X^T w \\ &= 2w^T X L^L X^T w \end{aligned} \tag{6}$$

where $D^L$ is a diagonal matrix, $D_{ii}^L = \sum_j P_{ij}$, $L^L = D^L - P$.

So, the extended objective function is defined as maximizing $J(w)$, where

$$\begin{aligned} lJ(w) &= \frac{1}{2}\left(\frac{1}{n_C} Q^C(w) - \frac{\alpha}{n_M} Q^M(w) - \frac{\beta}{n^2} Q^L(w)\right) \\ &= w^T X\left(\frac{1}{n_C} L^C - \frac{\alpha}{n_M} L^M - \frac{\beta}{n^2} L^L\right) X^T w \\ &= w^T X L^J X^T w \end{aligned} \tag{7}$$

where $L^J = (\frac{1}{n_C} L^C - \frac{\alpha}{n_M} L^M - \frac{\beta}{n^2} L^L)$, $\beta$ is the scaling parameter to balance the contribution of the unlabeled data, $n$ is the number of all the data samples.

Obviously, the problem expressed by Eq. (7) is a typical eigen-problem, which can be easily and efficiently solved by computing the eigenvectors of $XL^J X^T$ corresponding to the largest eigenvalues. Let $W = [w_1, w_2, ..., w_d]$, where $w_i$ $(i = 1, ..., d)$ are the eigenvectors corresponding to the maximum eigenvalues of $XL^J X^T$. The linear dimensionality reduction is shown as followed:

$$x \to y = W^T x \tag{8}$$

The time complexities of calculating the $k$-nearest neighbors and calculating the eigen-problem are $O(Dn^2 logk)$ and $O(D^3)$ respectively. So, the time complexity of WPCSSDR is $O(Dn^2 logk) + O(D^3)$. The WPCSSDR algorithm is given in Table 1.

### 2.3 The ASSDR algorithm

Obviously, it is hard to determine the weights of the pairwise constraints in practice and it is also hard to compute appropriate values for the neighborhood size and the adjacency weight matrix as described in Introduction. Here, we propose a heuristic iteration scheme to solve these problems.

At the beginning, initialize the weights of the pairwise constraints with a weight matrix $S$ and initialize an affinity weight matrix $P$ of a $k$-nearest neighbor graph. Then, the

**Table 1** WPCSSDR algorithm

| |
|---|
| Input: |
| $X$—data matrix |
| $M$—must-link constraints |
| $C$—cannot-link constraints |
| $S$—pairwise constraints weight matrix |
| $k$—nearest neighborhood size |
| $\sigma$—parameter of the heat kernel function |
| $\alpha$—scaling parameter of must-link constraints |
| $\beta$—scaling parameter of unlabeled data |
| $d$—dimensions of the reduced subspace |
| Output: |
| $W$—projection matrix |
| Procedure |
| Calculate the affinity weight matrix $P$ of the $k$-nearest neighbor graph |
| Calculate $L^J$ according to Eq. (7) |
| Calculate the projection matrix $W$ by the eigen-decomposition of $XL^J X^T$ |

iterative procedure consists of the following four main steps:

Step 1: Calling WPCSSDR procedure to get the projection matrix $W$ from the original space to the reduced subspace with current $S$ and $P$.

Step 2: Clustering in the current reduced subspace. For simplicity, we use $K$-Means procedure in our experiments, where $K$ can be provided by the user or estimated as the number of the chunklets [12] derived from the pairwise constraints if there are many must-link constraints, that is, small subsets of points that are known to belong to the same although unknown class.

Step 3: Updating the weights of the pairwise constraints with the following rules: (1) For must-link constraint $(x_i, x_j) \in M$, let $S_{ij} = S_{ij}e^{\Delta S_M}$, if $x_i$ and $x_j$ are mis-clustered into two different clusters; otherwise, let $S_{ij} = S_{ij}e^{-\Delta S_M}$, if $x_i$ and $x_j$ are clustered into the same cluster. Here, $\Delta S_M > 0$ is a predefined updating parameter for the must-link constraints. (2) For cannot-link constraint $(x_i, x_j) \in C$, let $S_{ij} = S_{ij}e^{\Delta S_C}$, if $x_i$ and $x_j$ are mis-clustered into the same cluster; otherwise, let $S_{ij} = S_{ij}e^{-\Delta S_C}$, if $x_i$ and $x_j$ are clustered into two different clusters. Here, $\Delta S_C$ is a predefined updating parameter for the cannot-link constraints.

Step 4: Updating the $k$-nearest adjacency weight matrix $P$ in the current reduced subspace with the following equation:

$$P_{ij} = \exp\left(-\left\|W^T x_i - W^T x_j\right\|^2 / 2{\sigma_W}^2\right) \tag{9}$$

where $\sigma_W$ is the parameter of the heat kernel function for the current subspace with $W$.

In the iterative procedure, cluster assumption [27] is implied in step 2, which means that the reduced subspace should be well clustered. The weights of the pairwise constraints are updated in step 3, which states that the mis-clustered pairwise constraints should be paid more attention, and the correctly clustered pairwise constraints should be paid less attention. The adjacency weight matrix $P$ is updated in step 4, which means that the neighborhood graph constructed in the transformed space includes more discriminative information than the one constructed in the original space. The updating of $S$ and $P$ will result in the updating of $W$ in step 1, which will influence $S$ and $P$ in turn. The procedures will continue until getting a stable $W$ or reaching the maximum number of iterations.

After the iterative procedure, the final pairwise constraints weight matrix can be calculated as follows:

$$S = \frac{\sum_{i=1}^{i_{max}} \theta_i S_i}{\sum_{i=1}^{i_{max}} \theta_i} \tag{10}$$

where $i_{max}$ is the maximum number of iterations, $S_i$ is the pairwise constraints weight matrix of the $i$-th iteration, $\theta_i = \ln[(1 - E_i)/E_i]$, $E_i$ is the unsatisfied pairwise constraints error of the clustering in the reduced subspace of the $i$-th iteration as followed:

$$E_i = \frac{numc + nucc}{n_M + n_C} \tag{11}$$

where $numc$ is the number of unsatisfied must-link constraints, $nucc$ is the number of unsatisfied cannot-link constraints.

In the same way, the final adjacency weight matrix can be calculated as followed:

$$P = \frac{\sum_{i=1}^{i_{max}} \theta_i P_i}{\sum_{i=1}^{i_{max}} \theta_i} \tag{12}$$

where $P_i$ is the adjacency weight matrix of the $i$-th iteration.

Finally, we can Call WPCSSDR procedure to get the final projection matrix $W$ with the final $S$ and $P$.

The time complexity of Step 1 is $O(Dn^2logk) + O(D^3)$. The time complexity of step 2 is $O(IKnd)$, where $I$ is the fixed number of $K$-Means iterations. The time complexity of step 3 is $O(n^2)$. The time complexity of step 4 is $O(dn^2)$. So, the time complexity of ASSDR is $O(i_{max}D^3) + O(i_{max}IKnd) + O(i_{max}Dn^2logk)$. The ASSDR algorithm is given in Table 2.

**Table 2** ASSDR algorithm

Input:
- $X$, $M$, $C$, $k$, $\sigma$, $\alpha$, $\beta$, $d$—the same as WPCSSDR
- $K$—number of clusters
- $\Delta S_M$—paramater to update the weights of the must-link constraints
- $\Delta S_C$—paramater to update the weights of the cannot-link constraints
- $i_{max}$—maximum number of iterations

Output:
- $S_{final}$—final pairwise constraints weight matrix
- $P_{final}$—final adjacency weight matrix
- $W_{final}$—final projection matrix

Procedure:
- Initialize $S$ and $P$
- For $i$ = 1, 2, ..., $i_{max}$
  - Calculating the projection matrix $W$ as in step 1
  - Clustering in the reduced subspace as in step 2
  - Updating the matrix $S$ as in step 3
  - Updating the matrix $P$ as in step 4
- EndFor
- Calculate $S_{final}$ according to Eq. (10)
- Calculate $P_{final}$ according to Eq. (12)
- Calculate $W_{final}$ by calling WPCSSDR with $S_{final}$ and $P_{final}$

### 2.4 A variation of ASSDR

ASSDR has a variation which uses pairwise constraints only. We call it ASSDR-CM, which uses Eq. (5) as the objective function, so the procedure is almost the same as ASSDR except setting matrix $P = 0$ without updating. ASSDR-CM is also a semi-supervised dimensionality reduction method, because of using unlabeled samples in the clustering step implicitly, though unlabeled samples through adjacency weight matrix explicitly, like ASSDR, is not used.

## 3 Experiments

In this section, the performance of ASSDR and ASSDR-CM are evaluated on the classification tasks and compared with SSDR and CLPP, which are semi-supervised dimensionality reduction methods, by preserving global structure and local structure, respectively. Parameter analysis for ASSDR is also discussed in this section.

### 3.1 Classification in the UCI datasets

In what follows, we first perform classification experiments on four datasets from UCI machine learning repository[1] which are widely used in machine learning field. The four datasets include Iris, Wine, Soybean and Ionosphere. The detailed descriptions are shown in Table 3.

In the experiments, the pairwise constraints are obtained by randomly selecting pairs of instances from the training samples (50 % of the samples are for training and the rest samples are for testing) and creating must-link or cannot-link constraints depending on whether the underlying classes of the two instances are the same or not. The number of must-link constraints is equal to the number of cannot-link constraints in the experiments. After obtaining the constraints, we firstly carry out these algorithms on the training samples and learn the projection matrix; second, each test sample is mapped into a low-dimensional subspace via the projection matrix; finally, we classify the test samples by the nearest neighbor classifier using the ground truth class labels of all the training data to evaluate the performances of the dimensionality reduction methods. Although the class labels are unavailable in our semi-supervised scenario, the evaluation method is commonly used in many pairwise constraints based dimensionality reduction methods [12, 17, 18, 21] due to its simplicity and effectiveness.

As for the parameters of ASSDR, $k$ is set to 3, $\sigma$ and $\sigma_W$ are set to the average pairwise Euclidean distance of the original space and the reduced subspace with $W$ respectively, $\alpha$ is always set to 1 because ASSDR can adjust the weights of the pairwise constraints adaptively, $\beta$ is searched from $\{1, 10, 10^2, 10^3\}$. For simplicity, we set $\Delta S_M = \Delta S_C = \Delta S$ which is searched from $\{0.1, 0.2, 0.3, \ldots, 1.0\}$. In addition, $K$ is set to the number of the ground truth categories of the corresponding dataset, $i_{max}$ is set to 40 for iris and 20 for other datasets. The parameters of other algorithms are set by the methods of the corresponding papers. All the experimental results are the average over 20 random splits of training and testing samples. Figure 2 shows the results where NOC means the number of constraints (in our experiments, we set $n_M = n_C =$NOC).

**Table 3** The four UCI datasets

| Dataset | Sample size | Dimension | Class |
|---|---|---|---|
| Iris | 150 | 4 | 3 |
| Wine | 178 | 13 | 3 |
| Soybean | 47 | 35 | 4 |
| Ionosphere | 351 | 34 | 2 |

It can be seen from Fig. 2: (1) With the increasing of NOC, the performance of ASSDR is getting better. (2) ASSDR is nearly always one of the best two methods, although it cannot get the best results in all cases. (3) ASSDR is the most stable method in all the methods, because ASSDR-CM get the worst results at Iris, SSDR get the worst results at Wine and Soybean, and CLPP get the worst results at Ionosphere.

### 3.2 Image recognition

An image recognition task can be viewed as a multi-class classification problem in high dimensional spaces. In this section, the performance of ASSDR and ASSDR-CM are evaluated on the PIE [28], extended Yale-B [29], MNIST [30], and PolyU Palmprint [31] image databases. In the experiments, we use the preprocessed versions of the PIE, extended Yale-B, and MNIST database which are publicly available from the web page of Cai,[2] PolyU Palmprint database which are publicly available from the web page of Biometrics Research Centre.[3]

#### *3.2.1 Database description*

The PIE face database contains 41,368 images of 68 people, each person under 13 different poses, 43 different illumination conditions, and with 4 different expressions. In this experiment, our dataset only contains five near frontal

[1] http://archive.ics.uci.edu/ml/.

[2] http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html.

[3] http://www4.comp.polyu.edu.hk/~biometrics/.

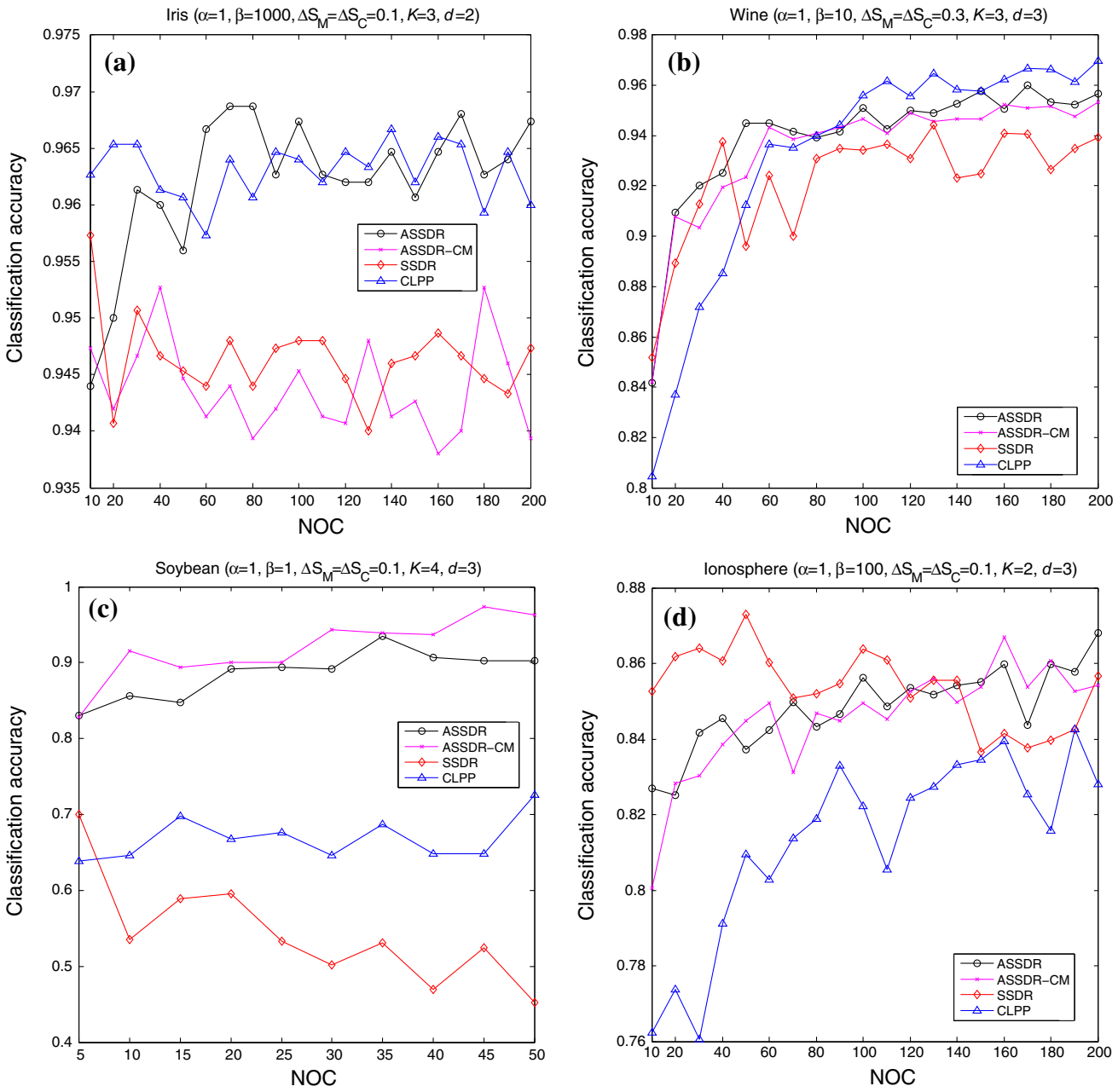


**Fig. 2** Classification results of ASSDR on UCI datasets

poses (C05, C07, C09, C27, C29) and all the images under different illuminations and expressions. As a result, there are 170 images for each individual. All the face images are aligned and cropped. The cropped images are $32 \times 32$ pixels, with 256 gray levels per pixel. In the following experiments, 20 images of each individual are for training and the rest 150 images are for testing.

The extended Yale-B face database contains 21,888 images of 38 human subjects under 9 poses and 64 illumination conditions. In this experiment, we choose the frontal pose and use all the images under different illumination, thus we get 64 images for each person. All the face images are aligned and cropped. The cropped images are $32 \times 32$ pixels, with 256 gray levels per pixel. In the following experiments, 30 images of each individual are for training and the rest 34 images are for testing.

The MNIST database contains 70,000 handwritten digit images. In this experiment, we choose 4000 images from the original database, thus we get 400 images for each number. All the face images are aligned and cropped. The cropped images are $20 \times 20$ pixels, with 256 gray levels per pixel. In the following experiments, 100 images of each

individual are for training and the rest 300 images are for testing.

The PolyU Palmprint database contains 7,752 grayscale images corresponding to 386 different palms in BMP image format. Around 20 samples from each of these palms were collected in two sessions, where around 10 samples were captured in the first session and the second session, respectively. The average interval between the first and the second collection was two months. All the images are aligned and cropped. The cropped images are $32 \times 24$ pixels, with 256 gray levels per pixel. In the following experiments, 5 images of each individual are for training and the rest images are for testing.

### 3.2.2 *Experimental results and discussions*

The experimental settings is the same as that of the UCI datasets, except we first project the face images into a PCA subspace by retaining 99 % of the principal components to deal with small sample size problem.

Figure 3 displays the experimental results on PIE database.It can be seen from Fig. 3a: (1) When NOC is small (i.e., 100), ASSDR is better than ASSDR-CM, which means preserving local structure is useful and the updating strategy of the adjacency weight matrix is effective in this case. (2) When NOC is small (i.e., 100), ASSDR is a little worse than SSDR, which means preserving global structure

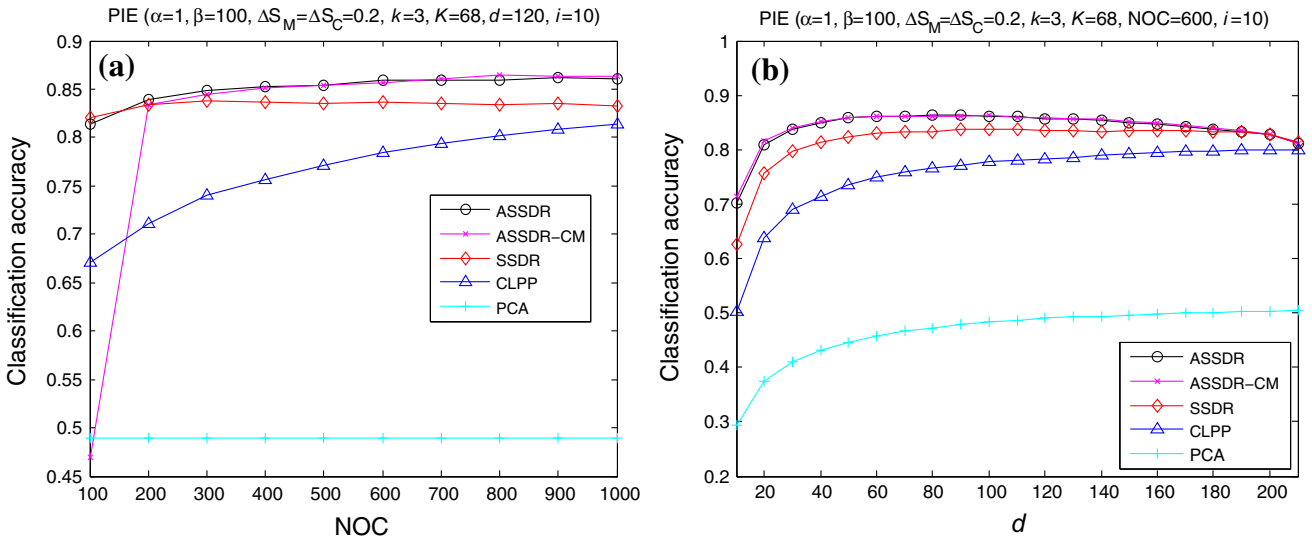


**Fig. 3** Experimental results on PIE

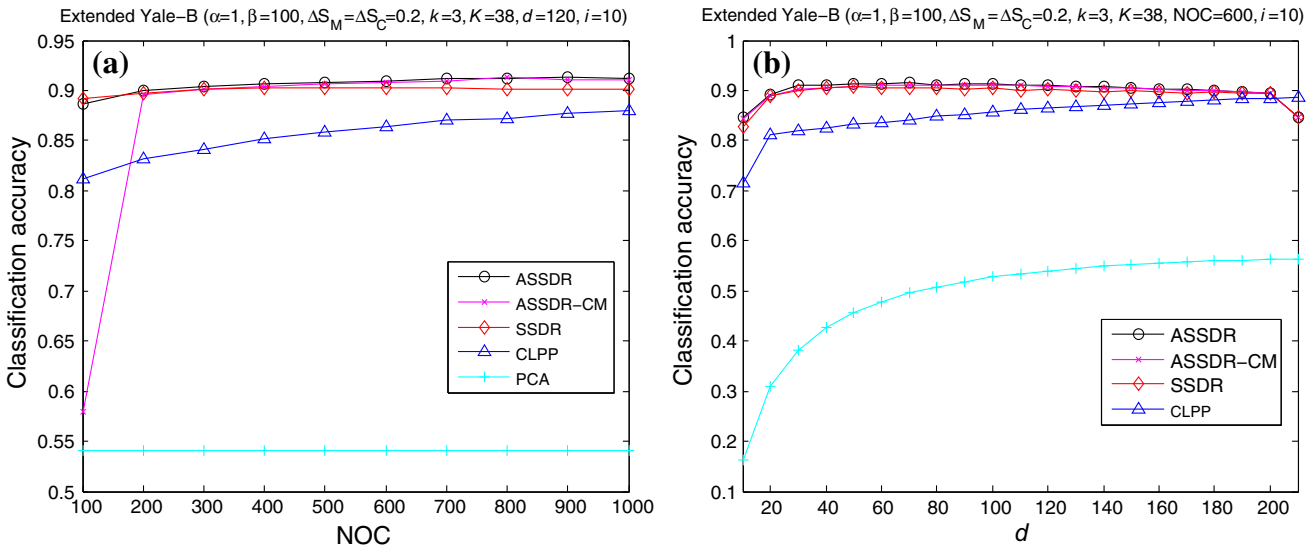


**Fig. 4** Experimental results on extended Yale-B

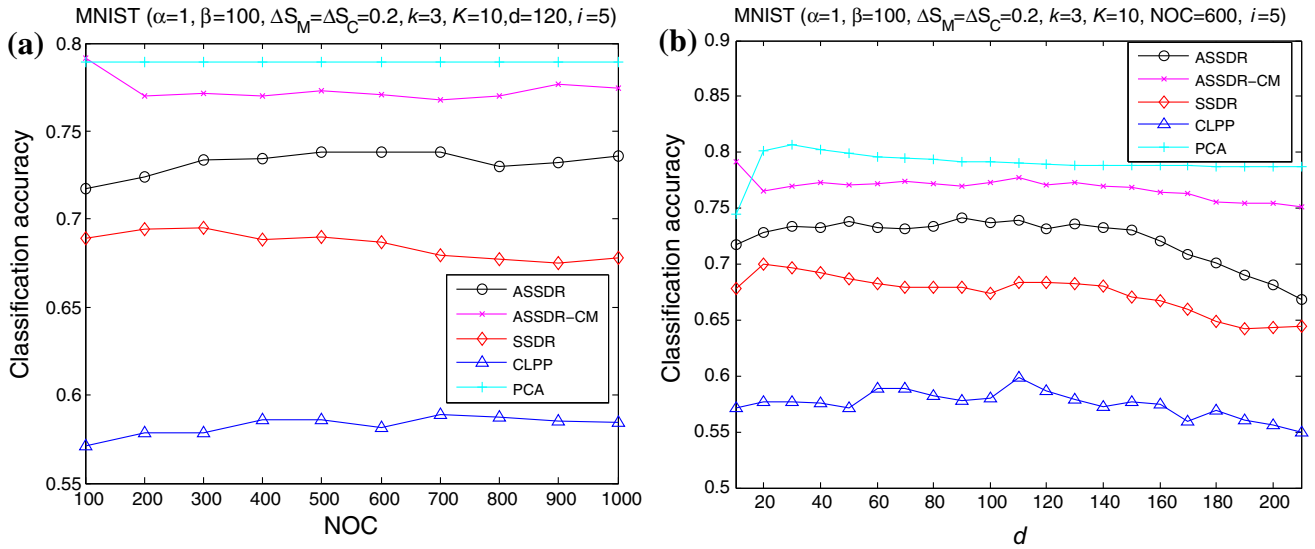


**Fig. 5** Experimental results on MNIST

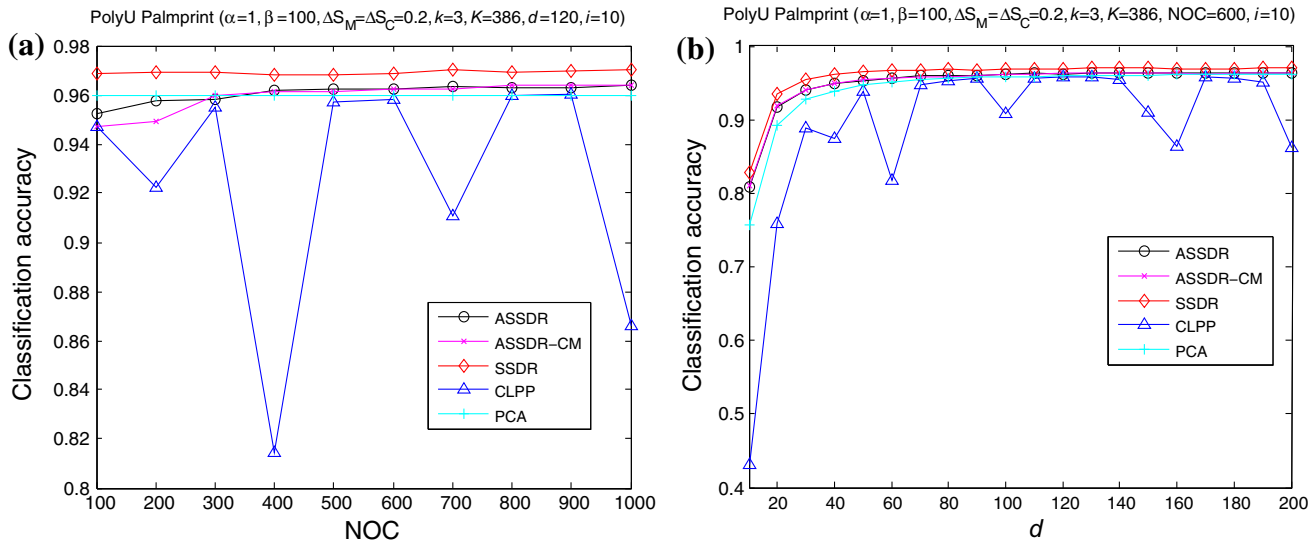


**Fig. 6** Experimental results on PolyU Palmprint

is more effective than preserving local structure in this case. (3) When NOC is large (i.e., $\geqslant$300), ASSDR is almost the same as ASSDR-CM, which means preserving local structure is less helpful for ASSDR in this case. (4) When NOC is large (i.e., $\geqslant$300), ASSDR and ASSDR-CM are better than SSDR, CLPP and PCA, which means the updating strategy of pairwise constraints weights of ASSDR is effective in this case.

It can be seen from Fig. 3b: When NOC is large (i.e., 600), with the varying of reduced dimensions, ASSDR and ASSDR-CM are almost the same with each other and always the best methods. However, if $d$ is too small or too large, the performance of ASSDR and ASSDR-CM will be descending.

Figure 4 displays the experimental results on extended Yale-B database. From Fig. 4, we can get the similar conclusions as Fig. 3.

Figure 5 displays the experimental results on MNIST database. It can be seen from Fig. 5a: (1) ASSDR-CM is better than ASSDR, which means preserving local structure is less useful and the updating strategy of the adjacency weight matrix is less effective in this case. (2) ASSDR and ASSDR-CM are a little worse than PCA, but better than SSDR and CLPP, which means the updating strategy of pairwise constraints weights of ASSDR is effective, while unsupervised learning might be more suitable in this case. (3) As NOC varies, the performance of ASSDR and ASSDR-CM vary little.

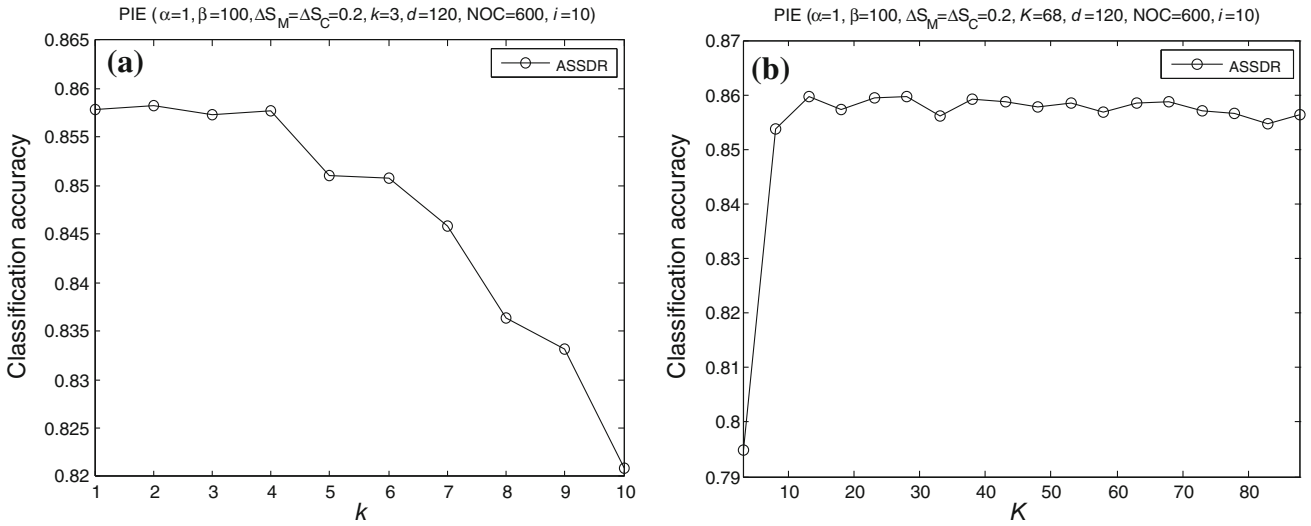


**Fig. 7** Influences of $k$ and $K$ to ASSDR on PIE

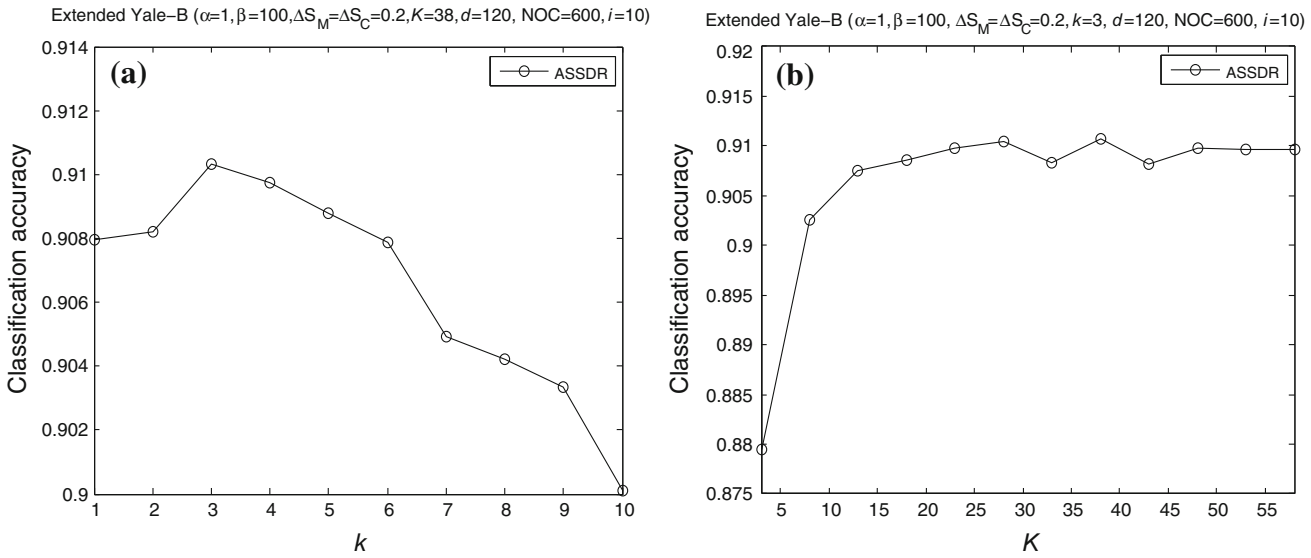


**Fig. 8** Influences of $k$ and $K$ to ASSDR on extended Yale-B

It can be seen from Fig. 5b : When NOC is large (i.e., 600), with the varying of reduced dimensions, ASSDR and ASSDR-CM are still no better than PCA. When $d$ is small, the performance of ASSDR and ASSDR-CM will be ascending on a small scale; however, if $d$ is too large, the performance of ASSDR and ASSDR-CM will be descending.

Figure 6 displays the experimental results on PolyU Palmprint database. From Fig. 6, we can get the similar conclusions with Fig. 5, except (1) SSDR, instead of PCA, performances better than ASSDR and ASSDR-CM. (2) When NOC is small (i.e., 200), ASSDR is better than ASSDR-CM, which means preserving local structure is useful and the updating strategy of the adjacency weight matrix is effective in this case. (3) When NOC is large (i.e., 600), with the varying of reduced dimensions, if $d$ is too small, the performance of ASSDR and ASSDR-CM will be descending.

Figure 7 shows the experimental results of the influences of $k$ and $K$ to ASSDR on PIE.

It can be seen from Fig. 7a: (1) When $k$ is small (i.e., 1, 2, 3, 4), the performance of ASSDR is good and stable. (2) When $k$ is relatively large (i.e., $\geqslant$5), the performance of ASSDR is descending. The observation

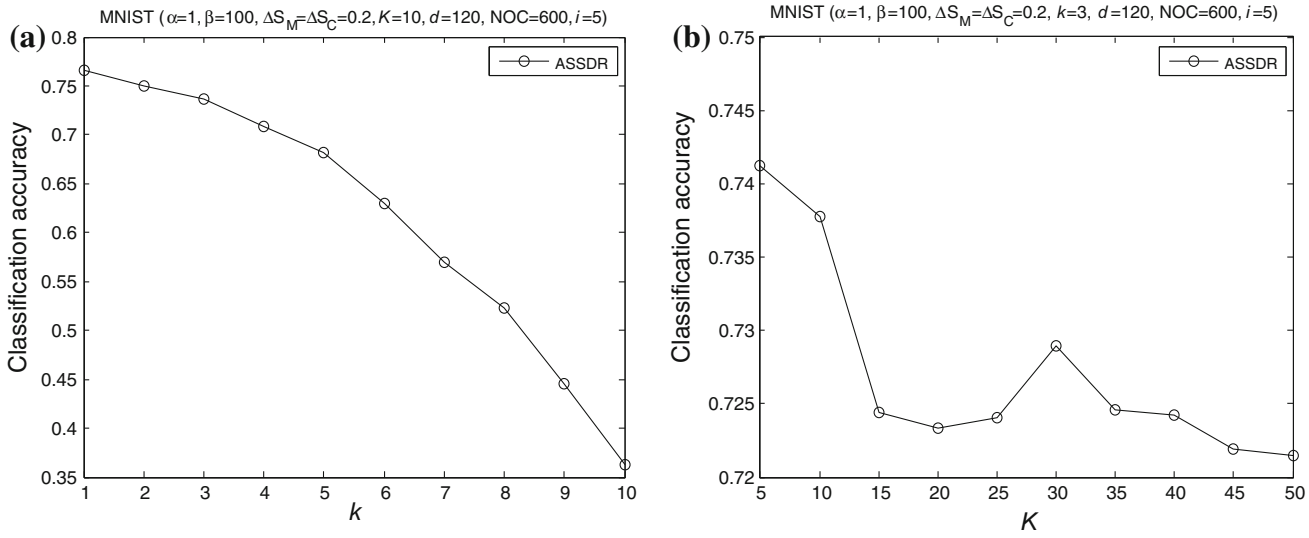


**Fig. 9** Influences of $k$ and $K$ to ASSDR on MNIST

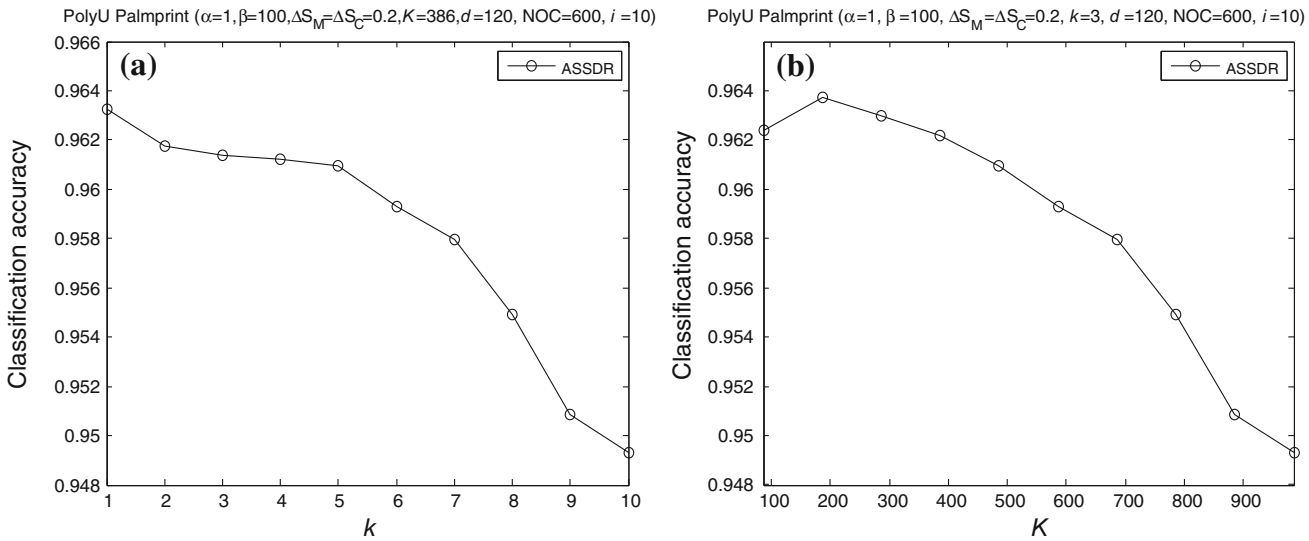


**Fig. 10** Influences of $k$ and $K$ to ASSDR on PolyU Palmprint

implies that we should select a relatively small $k$ in the experiments.

It can be seen from Fig. 7b: Although the number of the ground truth categories of PIE is 68, we can get a satisfactory result by setting the value of $K$ in a wide range (i.e., $\geqslant$13). This implies that ASSDR is not sensitive to the value of $K$, which is very useful in practice.

Figures 8, 9, 10 shows the experimental results of the influences of $k$ and $K$ to ASSDR on extended Yale-B, MNIST, PolyU Palmprint, respectively. From Figs. 8, 9, 10, we can get the similar conclusions as Fig. 7, except that $K$ is better to be set to 10 and 386 as for MNIST and Poly U Palmprint, respectively.

Figure 11 shows the experimental results of the influences of $i$ to ASSDR and ASSDR-CM on the four databases.

It can be seen from Fig. 11a, b: (1) When $i$ is small (i.e., 1, 2, 3, 4), the performance of ASSDR and ASSDR-CM will be ascending. (2) When $i$ is relatively large (i.e., $\geqslant$5), the performance of ASSDR and ASSDR-CM vary little.

It can be seen from Fig. 11c: The performance of ASSDR and ASSDR-CM will be descending as $i$ ascends, which shows $i$ in ASSDR and ASSDR-CM might not work that well on this database.

It can be seen from Fig. 11d: (1) When $i$ is small (i.e., $\leqslant$9), the performance of ASSDR and ASSDR-CM will be

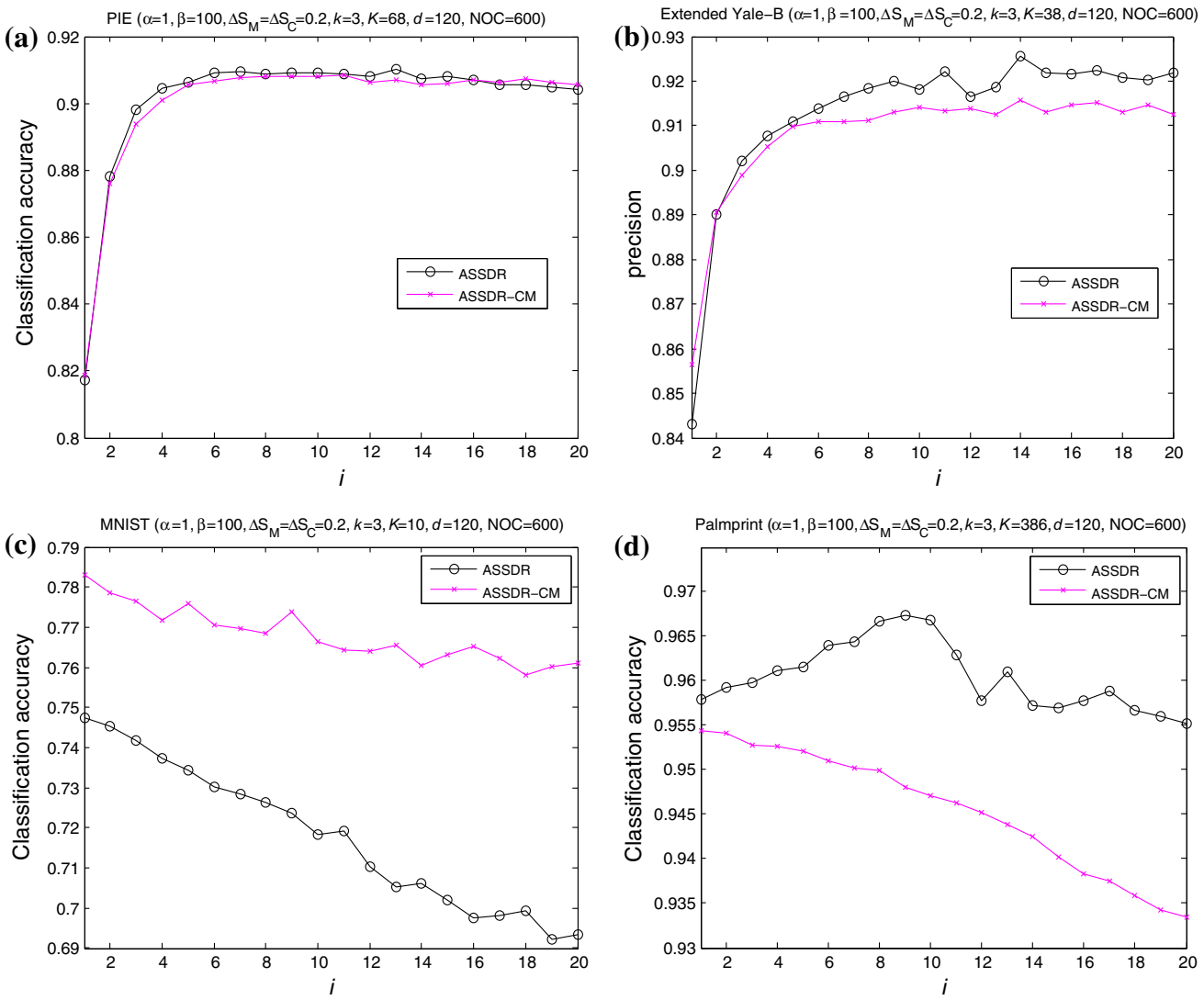


**Fig. 11** Influences of *iteration* to ASSDR and ASSDR-CM

ascending. (2) When $i$ is relatively large(i.e., $\geqslant 9$), the performance of ASSDR will be descending. (3) The performance of ASSDR-CM will be descending as $i$ ascends, which shows $i$ in ASSDR-CM might not work that well on this database.

## 4 Conclusions

In this paper, we present a novel pairwise constraints based semi-supervised dimensionality reduction method called ASSDR. Different from existing methods such as SSDR and CLPP, which treat the pairwise constraints equally, ASSDR can take into account the importance of different pairwise constraints respectively by updating the weights of the pairwise constraints. In addition, it can simultaneously optimize graph construction at each updating iteration in order to utilize unlabeled data. Although ASSDR does not turn out to be the best method all the time, when used on different databases, it turns out to perform relatively well in most cases. Experiments on classification tasks have been conducted to demonstrate the effectiveness of our method.

Future work is needed both with respect to theory and application. In particular, the convergence property for this problem is unknown yet. Furthermore, the power of the method would be increased, for example, by incorporating kernels in an adaptive way. In addition, decreasing the number of the parameters would be further studied.

**Acknowledgments** This work is supported by the National Natural Science Foundation of China (61402181, 61273363), the Science and Technology Programme of Guangzhou Municipal Government (2014J4100006), the Guangdong Natural Science Foundation (S2012040008022, S2012010009961).

## References


1. Duda RO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. Wiley, New York
2. Jolliffe IT (2002) Principal component analysis, 2nd edn. Wiley, New York
3. Sharma A, Paliwal KK, Imoto S, Miyano S (2013) Principal component analysis using QR decomposition. Int J Mach Learn Cybern (IJMLC) 4:679–683
4. Sharma A, Paliwal KK (2015) Linear discriminant analysis for the small sample size problem: an overview. Int J Mach Learn Cybern (IJMLC) 6:443–454
5. Chapelle O, Schölkopf B, Zien A (2006) Semi-supervised learning. MIT Press, Cambridge
6. Song Y, Nie F, Zhang C, Xiang S (2008) A unified framework for semi-supervised dimensionality reduction. Pattern Recognit 41(9):2789–2799
7. Chatpatanasiri R, Kijsirikul B (2010) A unified semi-supervised dimensionality reduction framework for manifold learning. Neurocomputing 73(10–12):1631–1640
8. Alok AK, Saha S, Ekbal A (2015) Semi-supervised clustering for gene-expression data in multiobjective optimization framework. Int J Mach Learn Cybern (IJMLC). doi:10.1007/s13042-015-0335-8
9. Wagstaff K, Cardie C (2000) Clustering with instance-level constraints. In: Proceedings of the 17th International Conference on Machine Learning, pp 1003–1110
10. Chen C, Zhang J, He X, Zhou ZH (2012) Non-parametric kernel learning with robust pairwise constraints. Int J Mach Learn Cybern (IJMLC) 3:83–96
11. Klein D, Kamvar SD, Manning CD (2002) From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering. In: Proceedings of the 19th International Conference on Machine Learning, pp 307–314
12. Bar-Hillel A, Hertz T, Shental N, Weinshall D (2005) Learning a mahalanobis metric from equivalence constraints. J Mach Learn Res 6:937–965
13. Xing EP, Ng AY, Jordan MI, Russell S (2003) Distance metric learning, with application to clustering with side-information. Adv Neural Inf Process Syst 15:505–512
14. Tang W, Zhong S (2006) Pairwise constraints-guided dimensionality reduction. In: Proceedings of the SDM'06 Workshop on Feature Selection for Data Mining, pp 59–66
15. Yeung DY, Chang H (2006) Extending the relevant component analysis algorithm for metric learning using both positive and negative equivalence constraints. Pattern Recognit 39(5): 1007–1010
16. An S, Liu W, Venkatesh S (2008) Exploiting side information in locality preserving projection. In: Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp 1–8
17. Zhang D, Zhou Z, Chen S (2007) Semi-supervised dimensionality reduction. In: Proceedings of the 7th SIAM International Conference on Data Mining, pp 629–634
18. Chen S, Zhang D (2011) Semisupervised dimensionality reduction with pairwise constraints for hyperspectral image classification. IEEE Geosci Remote Sens Lett 8(2):369–373
19. Cevikalp H, Verbeek J, Jurie F, Klaser A (2008) Semi-supervised dimensionality reduction using pairwise equivalence constraints. In: Proceedings of the 2008 International Conference on Computer Vision Theory and Applications, pp 489–496
20. He X, Niyogi P (2004) Locality preserving projections. Adv in Neural Inf Process Syst 16:153–160
21. Wei J, Peng H (2008) Neighborhood preserving based semi-supervised dimensionality reduction. Electron Lett 44(20):1190–1191
22. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2327
23. Baghshah MS, Shouraki SB (2009) Semi-supervised metric learning using pairwise constraints. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence, pp 1217–1222
24. Davidson I (2009) Knowledge driven dimension reduction for clustering. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence, pp 1034–1039
25. Yan S, Bouaziz S, Lee D, Barlow J (2012) Semi-supervised dimensionality reduction for analyzing high-dimensional data with constraints. Neurocomputing 76(1):114–124
26. Belkin M, Niyogi P (2003) Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput 15(6):1373–1396
27. Chapelle O, Zien A (2005) Semi-supervised classification by low density separation. In: Proceedings of the 22nd International Conference on Machine Learning, pp 57–64
28. Sim T, Barker S, Bsat M (2003) The CMU pose, illumination, and expression database. IEEE Trans Pattern Anal Mach Intell 25(12):1615–1618
29. Georghiades AS, Belhumeur PN, Kriegman DJ (2001) From few to many: illumination cone models for face recognition under variable lighting and pose. IEEE Trans Pattern Anal Mach Intell 23(6):643–660
30. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11): 2278–2324
31. Zhang D, Kong W, You J, Wong M (2003) Online palmprint identification. IEEE Trans Pattern Anal Mach Intell 25(9): 1041–1050