Digital Signal Processing 90 (2019) 110–124

Contents lists available at ScienceDirect

Digital Signal Processing

www.elsevier.com/locate/dsp

Face recognition based on dictionary learning and subspace learning

Mengmeng Liao, Xiaodong Gu *

Department of Electronic Engineering, Fudan University, Shanghai 200433, China

ARTICLE INFO

Article history:
Available online 15 April 2019

Keywords:
Face recognition
Dictionary learning
Subspace learning
Label relaxation model

ABSTRACT

Dictionary learning plays an important role in sparse representation based face recognition. Many dictionary learning algorithms have been successfully applied to face recognition. However, for corrupted data because of noise or face variations (e.g. occlusion and large pose variation), their performances decline due to the disparity between domains. In this paper, we propose a face recognition algorithm based on dictionary learning and subspace learning (DLSL). In DLSL, a new subspace learning algorithm (SL) is proposed by using sparse constraint, low-rank technology and our label relaxation model to reduce the disparity between domains. Meanwhile, we propose a high-performance dictionary learning algorithm (HPDL) by constructing the embedding term, non-local self-similarity term, and time complexity drop term. In the obtained subspace, we use HPDL to classify these mapped test samples. DLSL is compared with other 28 algorithms on FRGC, LFW, CVL, Yale B and AR face databases. Experimental results show that DLSL achieves better performance than those 28 algorithms, including many state-of-the-art algorithms, such as recurrent regression neural network (RRNN), multimodal deep face recognition (MDFR) and projective low-rank representation (PLR).

1. Introduction

Face recognition has aroused broad interest in pattern recognition and computer vision areas during the past 20 years [1]. Many face recognition methods have been proposed. Generally, each method can only solve some specific face recognition problems. For example, Yang et al. [2] proposed a kernel Fisher discriminant framework for feature extraction and recognition; Zafeiriou et al. [3] proposed a robust approach to discriminant kernel-based feature extraction for face recognition. Those methods only consider the holistic feature of face images so they are usually sensitive to the variations of misalignment, pose, and occlusion [4]. Thus they are not suitable for dealing with face recognition problems involving misalignment or pose variations. Wright et al. [5] proposed a sparse representation based classifier (SRC) which can obtain a good classification result compared with many well-known face recognition methods. However, SRC cannot deal with situations with large contiguous occlusions effectively [5]. Because the contiguous occlusion violates the assumption that the occlusion has sparse representation with respect to the identity matrix dictionary. There are many improved methods based on SRC, such as correntropy-based sparse representation [6], structured sparse error coding (SSEC) [7] and regularized robust coding (RRC) [8]. These methods can achieve good results for face recognition problems involving occlusion. However, they may overfit probe samples, i.e., they may treat the non-occluded samples as the occluded samples, which would degrade classification performance. Deng et al. [9] proposed the extended sparse representation-based classifier (ESRC) by introducing the intra-class variant dictionary. For face recognition problems either involving occlusion or non-occlusion, it can achieve better performance than SRC. Deng et al. proposed the superposed SRC (SSRC) [10]. SSRC also introduces the auxiliary dictionary but it constructs the basic dictionary by using the class centroids instead of the training samples in ESRC.

Research has demonstrated that learning a desired dictionary from training data instead of using off-the-shelf bases can lead to a great performance in many practical applications, such as face recognition [11]. Thus, a large number of dictionary learning algorithms have been proposed. Aharon et al. proposed a K-SVD algorithm [12], which is a classic dictionary learning method. It can learn a complete dictionary. Jiang proposed an LC-KSVD algorithm [13] by using the coding error term which is constructed by the label information. Zhang et al. proposed a D-KSVD algorithm [14], which can simultaneously train the discriminant dictionary and linear classifier. The goal of these dictionary learning methods is to learn a shared dictionary. However, the consistency between the dictionary elements will be lost. Yang et al. proposed an MFL algorithm [15]. It learns a series of dictionary bases from original samples, which are more expressive than the original training samples. Yang proposed a dictionary learning al-

* Corresponding author.
E-mail address: xdgu@fudan.edu.cn (X. Gu).

https://doi.org/10.1016/j.dsp.2019.04.006
1051-2004/

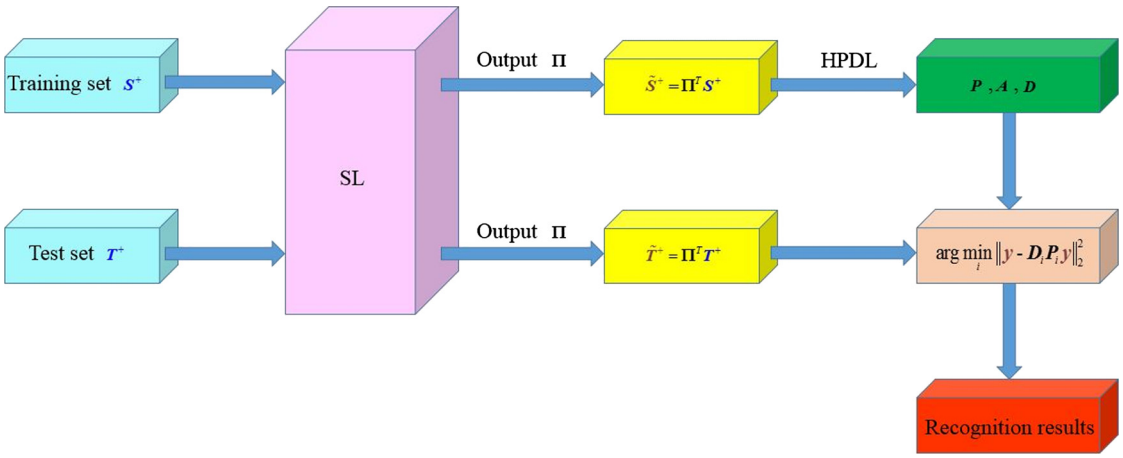


Fig. 1. The flow chart of DLSL.

gorithm based on Fisher discrimination criterion (FDDL) [16]. In FDDL, Fisher criterion is introduced to the reconstruction process of training samples, which not only makes the coded dictionaries more discriminative but also helps to obtain more accurate sparse representation coefficients. Gu et al. proposed a DPL algorithm by learning two dictionaries simultaneously [17]. Experimental results show that DPL can achieve good recognition performance. However, it ignores the label information of samples. Overall, those dictionary learning algorithms can achieve good results in many cases. However, when the data are corrupted by noise or large variations (e.g. pose variation and occlusion in face images), their performance declined heavily due to the large intra-class difference. Besides, these dictionary learning algorithms rarely consider the protection of edge details of the reconstructed samples, let alone consider the label information and edge details of the reconstructed samples at the same time. In addition, these dictionary learning algorithms take a long time to complete the classification task.

Subspace learning is widely studied and often used for image classification and face recognition. There are many subspace learning methods. Discriminant analysis [18] and metric learning [19] are local subspace learning methods, they can obtain good result generally. However, when the data come from different domains, the training set and test set have different distributions, thus the performance of these methods mentioned above is affected seriously. To solve this problem, researchers proposed many transfer subspace learning methods, such as LTSL [20], TSD [21] and TTRLSR [22]. Transfer learning aims to transfer the knowledge learned from a source domain to a target domain by exploiting the relatedness between them [23,24], which can reduce the distribution mismatch. In the scenario of speech recognition, transfer learning methods map samples into a high-dimension space where the discrepancy of distribution distances among samples is minimum [25]. Of course, transfer learning methods also have disadvantages, for example, the phenomenon of negative transfer may occur. However, in general, the advantages of transfer learning methods make them can be widely used in image classification, text classification, etc. Transductive transfer learning is a commonly used transfer learning method. It not only requires that the source and target tasks be the same but also requires that all unlabeled data in the target domain are available at training time [26–30]. However, those methods cannot capture the local features well and are sensitive to noise.

The sparse constraint has proved to be robust to noise and has shown impressive results for face recognition under noisy conditions [31]. Low-rank technology has attracted much attention recently. It can recover the underlying structure of data [32,33]. For example, Liu et al. [34] use low-rank representation to recover the corrupted data from multiple subspaces.

To overcome the shortcomings of dictionary learning methods and subspace learning methods mentioned above, we propose a face recognition algorithm based on dictionary learning and subspace learning (DLSL), which is inspired by low-rank representation and sparse technology. In DLSL, we first propose a subspace learning algorithm (SL) based on low-rank technology and binary label matrix relaxation model. SL is a transductive transfer learning method. Then, we use SL to learn a common subspace where the target domain and source domain data have similar distributions. After that, the training set (source domain data) and test set (target domain data) are mapped to the common subspace where the disparity of source and target domains is reduced. Next, we introduce a high-performance dictionary learning algorithm (HPDL). Finally, we use HPDL to classify the test samples in the common subspace. Fig. 1 shows the flow chart of DLSL. In Fig. 1, $\boldsymbol{S}^+$ represents the training set (the fully labeled source domain data) in original space. $\boldsymbol{T}^+$ is the test set (the fully unlabeled target domain data) in original space. SL is our proposed subspace learning algorithm. $\boldsymbol{\Pi}$ is the transformation matrix, which can be obtained in the process of learning common subspace by using SL. $\tilde{\boldsymbol{S}}^+$ is the training set in the common subspace, which can be obtained by mapping $\boldsymbol{S}^+$ into a common subspace using the transformation matrix $\boldsymbol{\Pi}$. $\tilde{\boldsymbol{T}}^+$ is the test set in the common subspace, which can be obtained by mapping $\boldsymbol{T}^+$ into a common subspace using $\boldsymbol{\Pi}$. HPDL is our proposed high-performance dictionary learning algorithm. $\boldsymbol{P}$ is the analysis dictionary. $\boldsymbol{A}$ is the coding coefficient matrix. $\boldsymbol{D}$ is the dictionary matrix. $\boldsymbol{y} \in \tilde{\boldsymbol{T}}^+$ is an arbitrary test sample in the common subspace.

In recent studies, deep learning-based methods have achieved good results in face recognition field. Thus in our experiments, three deep learning-based methods with good performance such as ConvNet-RBM [35], MDFR [36] and RRNN [37] are used to evaluate the performance of the proposed method.

The major contributions of this paper can be summarized as follows.

1) We propose a new subspace learning algorithm (SL). SL is a transductive transfer learning method, which is different from other ones [38–42]. First, SL is better than other methods in capturing the local features. The captured local features are used in subsequent face recognition, which can improve the recognition rate. Second, SL also has good robustness against noise by using the sparse constraint ($\ell_{2,1}$-norm constraint). This makes it easier to recover the clean data so that our method can learn more discriminative subspace where the

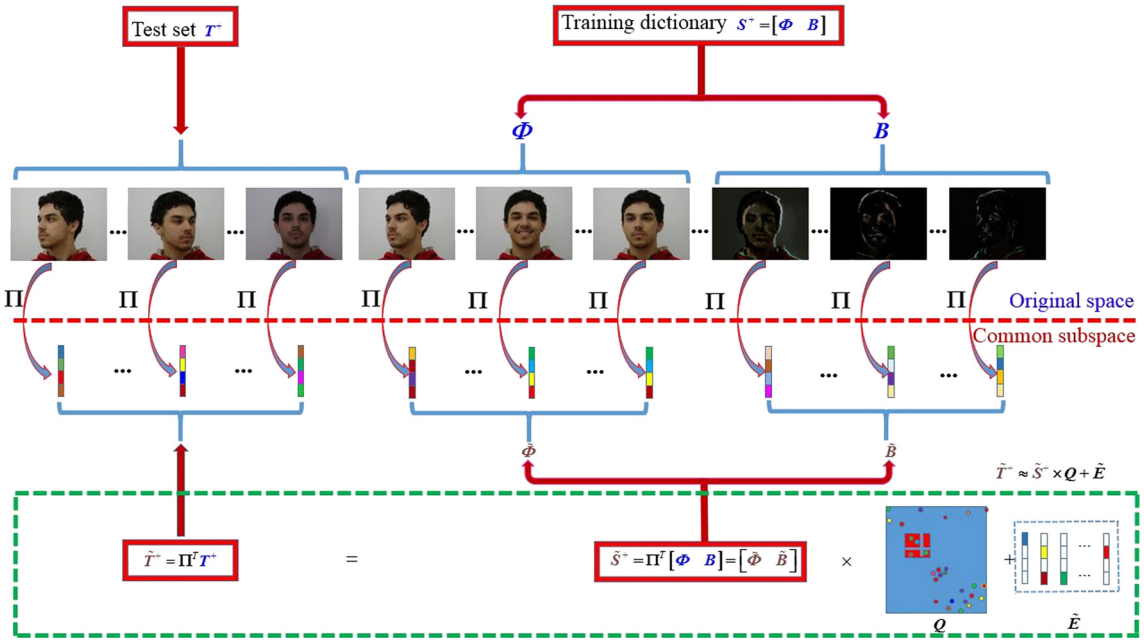


Fig. 2. The sketch map of SL. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

disparity between domains reduces. Third, SL can accurately align the source and target domains by using the low-rank constraint. Fourth, we introduce a label relaxation model. The introduced relaxation model enables easy access to a good transformation matrix. The objective function visually shows the differences between our SL and other methods.

2) A high-performance dictionary learning algorithm (HPDL) is proposed in this paper. HPDL is different from other dictionary learning methods. Firstly, a new label embedding term is constructed by using the label information of atoms instead of the classification error. The new label embedding term forces the coding coefficient matrix to be block-diagonal, which ensures atoms of each class reconstruct one class of the training samples. Secondly, the label information and the protection of edge details of reconstructed samples are considered at the same time in our HPDL, which makes the learned dictionary more discriminative. However, other dictionary learning algorithms rarely consider the protection of edge details of the reconstructed samples when they are used for face recognition, let alone consider the label information and edge details of the reconstructed samples at the same time. Thirdly, we introduce the analysis dictionary into our HPDL and use it to construct the time complexity drop term which makes the running time of HPDL less than many other dictionary learning algorithms.
3) To the best of our knowledge, we integrate the sparse representation and low-rank representation techniques into the transfer subspace learning framework and combine it with HPDL for face recognition for the first time. By learning the common subspace, the distribution differences of samples mapped into the common subspace are reduced. Thus our DLSL can achieve good recognition result when dealing with the face recognition problems involving noise, pose variation and occlusion, etc.
4) We achieve comparable or superior performance to state-of-the-art methods on FRGC, LFW, CVL, Yale B and AR face databases.

This rest of this paper is organized as follows. Firstly, a new subspace learning algorithm (SL) is introduced in Section 2. Next, a high-performance dictionary learning algorithm (HPDL) is presented in Section 3. Then, we introduce a face recognition algorithm based on dictionary learning and subspace learning (DLSL) in Section 4. After that, experiments are reported in Section 5. Finally, the conclusion is drawn in Section 6.

2. The proposed subspace learning algorithm

In this paper, we propose a subspace learning algorithm (SL) by using sparse constraint, low-rank technology and introducing a label relaxation model.

Fig. 2 shows the sketch map of SL. In Fig. 2, $\boldsymbol{S}^+ = [\,\boldsymbol{\Phi}\ \ \boldsymbol{B}\,] \in \Re^{d\times N}$ is the labeled source domain data (or called the training set in original space), where d is the dimension of each sample in $\boldsymbol{S}^+$, N is the number of samples in $\boldsymbol{S}^+$. $\boldsymbol{\Phi} \in \Re^{d\times n_1}$ is the gallery sample set. $\boldsymbol{B} \in \Re^{d\times n_2}$ is the intra-class variant sample set. Obviously, $N = n_1 + n_2$. $\boldsymbol{T}^+ \in \Re^{d\times q}$ is the unlabeled target domain data, which contains q samples. The dimension of each sample in $\boldsymbol{T}^+$ is d. $\boldsymbol{\Pi} \in \Re^{d\times p}$ is the transformation matrix. $\tilde{\boldsymbol{S}}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+ = \boldsymbol{\Pi}^T\left[\,\boldsymbol{\Phi}\ \ \boldsymbol{B}\,\right] = \left[\,\tilde{\boldsymbol{\Phi}}\ \ \tilde{\boldsymbol{B}}\,\right] \in \Re^{p\times N}$ is the training set in the common subspace, which can be obtained by mapping $\boldsymbol{S}^+$ to a common subspace using $\boldsymbol{\Pi}$. p is the dimension of each sample in $\tilde{\boldsymbol{S}}^+$. $\tilde{\boldsymbol{\Phi}} \in \Re^{p\times n_1}$ is the gallery sample set in the common subspace. $\tilde{\boldsymbol{B}} \in \Re^{p\times n_2}$ is the intra-class variant sample set in the common subspace. $\tilde{\boldsymbol{T}}^+ = \boldsymbol{\Pi}^T\boldsymbol{T}^+ \in \Re^{p\times q}$ is the test set in the common subspace, which can be obtained by mapping $\boldsymbol{T}^+$ to a common subspace using $\boldsymbol{\Pi}$. $\boldsymbol{Q} \in \Re^{N\times q}$ is the reconstruction matrix. $\tilde{\boldsymbol{E}} \in \Re^{p\times q}$ is the noise matrix.

The goal of subspace learning is to find an optimal transformation matrix that maps all samples into a common subspace where the target and source samples have the same distribution. Then we have

$$\boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} \tag{1}$$

where $\boldsymbol{Q}$ is the reconstruction coefficient matrix, which should have a block-wise structure.

To this end, the optimization problem

$$\min_{\boldsymbol{\Pi},\boldsymbol{Q}} \left\|\boldsymbol{\Pi}^T\boldsymbol{T}^+ - \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q}\right\|_F^2 \tag{2}$$

can be formulated for (1). In order to enforce $\boldsymbol{Q}$ to have a block-wise structure, we introduce a low-rank constraint to (2), and obtain

$$\min_{\boldsymbol{\Pi},\boldsymbol{Q}} \operatorname{rank}(\boldsymbol{Q}) \quad \text{s.t. } \boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q}. \tag{3}$$

Solving the rank minimization problem is an NP-hard problem. However, when the rank of $\boldsymbol{Q}$ is not too large, the low-rank constraint can be replaced by the nuclear norm $\|\cdot\|_*$ in the following equation

$$\min_{\boldsymbol{\Pi},\boldsymbol{Q}} \|\boldsymbol{Q}\|_* \quad \text{s.t. } \boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q}. \tag{4}$$

In order to make the reconstruction coefficient matrix sparse, we have applied a $\ell_{2,1}$-norm constraint on (4). The $\ell_{2,1}$-norm constraint is helpful to capture the local features which can improve the quality of image reconstruction. Besides, it is robust to outliers [43]. Thus (4) is formulated as

$$\min_{\boldsymbol{\Pi},\boldsymbol{Q}} \|\boldsymbol{Q}\|_* + \gamma\|\boldsymbol{Q}\|_{2,1} \quad \text{s.t. } \boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q}. \tag{5}$$

In practice, $\boldsymbol{\Pi}^T\boldsymbol{T}^+$ is not equal to $\boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q}$ because of the influence of noise. By introducing the noise matrix $\tilde{\boldsymbol{E}}$ in (5), we obtain

$$\min_{\boldsymbol{\Pi},\boldsymbol{Q},\tilde{\boldsymbol{E}}} \|\boldsymbol{Q}\|_* + \gamma\|\boldsymbol{Q}\|_{2,1} + \lambda\|\tilde{\boldsymbol{E}}\|_1 \quad \text{s.t. } \boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} + \tilde{\boldsymbol{E}}. \tag{6}$$

This makes this model more coincident with the real situation. Thus the optimization problem of our SL is

$$\begin{cases} \min\limits_{\boldsymbol{\Pi},\boldsymbol{Q},\tilde{\boldsymbol{E}}} \dfrac{1}{2}\Omega(\boldsymbol{\Pi},\boldsymbol{Y},\boldsymbol{S}^+) + \|\boldsymbol{Q}\|_* + \gamma\|\boldsymbol{Q}\|_{2,1} + \lambda\|\tilde{\boldsymbol{E}}\|_1 \\ \text{s.t. } \boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} + \tilde{\boldsymbol{E}} \end{cases} \tag{7}$$

where $\Omega(\boldsymbol{\Pi},\boldsymbol{Y},\boldsymbol{S}^+)$ is the subspace learning function. $\boldsymbol{Y} \in \Re^{p\times N}$ is the binary label matrix, each column of it represents the label of a sample. **Each column of $\boldsymbol{Y}$ has one and only one element is "1"**, the position of element "1" corresponds to the category of the sample. For example, if the unique "1" of a column appears in the third position, this indicates that the category of the sample is 3.

In order to distinguish the differences between different categories of samples, we need to design a proper $\Omega(\boldsymbol{\Pi},\boldsymbol{Y},\boldsymbol{S}^+)$. The regression method is often used to design $\Omega(\boldsymbol{\Pi},\boldsymbol{Y},\boldsymbol{S}^+)$. Conventional linear regression method assumes that training samples can be exactly transformed into a strict binary label matrix, that is

$$\Omega(\boldsymbol{\Pi},\boldsymbol{Y},\boldsymbol{S}^+) = \left\|\boldsymbol{\Pi}^T\boldsymbol{S}^+ - \boldsymbol{Y}\right\|_F^2 + \mu\|\boldsymbol{\Pi}\|_F. \tag{8}$$

However, the above assumption is too rigid, which will make $\boldsymbol{\Pi}$ has little freedom when $\boldsymbol{S}^+$ is transformed into the strict binary label. The strict binary label matrix cannot reflect the differences between different categories of samples. Inspired by [44], we introduce a relaxation model by learning a non-negative label relaxation matrix $\boldsymbol{M} \in \Re^{p\times N}$. The improved relaxation model not only can relax the strict binary label matrix into a slack variable matrix but also provides more freedom for $\boldsymbol{\Pi}$.

Now, we illustrate the process of relaxing the strict binary label matrix into a slack variable matrix. From $\boldsymbol{Y} \in \Re^{p\times N}$ we can see that $\boldsymbol{Y}$ contains the labels of N samples. Each column in $\boldsymbol{Y}$ represents the label of a sample. Each label in $\boldsymbol{Y}$ belongs to $\Re^{p\times 1}$. Our goal is to explore the label distance between different categories of samples. For the convenience of explanation, we assume that the first, second and third columns of $\boldsymbol{Y}$ respectively represent the labels of $ts1$, $ts2$ and $ts3$. $ts1$, $ts2$ **and** $ts3$ **are three training samples which respectively belong to the first, second and third categories**. Thus, the binary label matrix is defined as

$$\boldsymbol{Y} = \begin{bmatrix} 1 & 0 & 0 & \cdots \\ 0 & 1 & 0 & \cdots \\ 0 & 0 & 1 & \cdots \\ 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots \end{bmatrix}_{p\times N}. \tag{9}$$

It is easy to see that the Euclidean distance between any two samples from different categories is $\sqrt{2}$ when they are projected into their label space. For example, the label distance between $ts1$ and $ts2$ is $\sqrt{(1-0)^2+(0-1)^2+(0-0)^2+(0-0)^2+\cdots+(0-0)^2} = \sqrt{2}$.

Obviously, the definition of the binary label matrix in equation (9) can't reflect the characteristics of each sample. For example, although the label distance between $ts1$ and $ts2$ is equal to the distance between $ts2$ and $ts3$, the difference between $ts1$ and $ts2$ should be different from the difference between $ts2$ and $ts3$. Different samples have different characteristics and should be treated differently. For $\boldsymbol{M}$, its element m_{ij} is subject to $m_{ij} \geq 0$.

We introduce a binary label matrix relaxation model $\hat{\boldsymbol{Y}} = \tilde{\boldsymbol{Y}} + \tilde{\boldsymbol{Y}} \odot \boldsymbol{M}$, where '$\odot$' is the Hadamard product operator of matrices. $\tilde{\boldsymbol{Y}} = (\boldsymbol{Y} \odot \boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}$, $\boldsymbol{T}_{wo} \in \Re^{p\times N}$ is a matrix and all its elements are 2, $\boldsymbol{O}_{ne} \in \Re^{p\times N}$ is a matrix and all its elements are 1. $\hat{\boldsymbol{Y}}$ can be written as $\hat{\boldsymbol{Y}} = (((\boldsymbol{Y} \odot \boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}) + ((\boldsymbol{Y} \odot \boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}) \odot \boldsymbol{M})$. The expression of $\hat{\boldsymbol{Y}}$ is

$$\hat{\boldsymbol{Y}} = \begin{bmatrix} 1+m_{11} & -1-m_{12} & -1-m_{13} & \cdots \\ -1-m_{21} & 1+m_{22} & -1-m_{23} & \cdots \\ -1-m_{31} & -1-m_{32} & 1+m_{33} & \cdots \\ -1-m_{41} & -1-m_{42} & -1-m_{43} & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ -1-m_{p1} & -1-m_{p2} & -1-m_{p3} & \cdots \end{bmatrix}_{p\times N} \tag{10}$$

where the first, second and third columns of $\hat{\boldsymbol{Y}}$ respectively represent the labels of $ts1$, $ts2$ and $ts3$.

From the expression of $\hat{\boldsymbol{Y}}$ we can easily prove that the differences between different categories of samples can be reflected in the label matrix. For example, the label distance between $ts1$ and $ts2$ is

$$\begin{aligned} d_{12} &= \Big(\big[(1+m_{11})-(-1-m_{12})\big]^2 + \big[(-1-m_{21})-(1+m_{22})\big]^2 \\ &\quad + \cdots + \big[(-1-m_{p1})-(-1-m_{p2})\big]^2\Big)^{\frac{1}{2}} \\ &= \big((2+m_{11}+m_{12})^2 + (-2-m_{21}-m_{22})^2 + \cdots \\ &\quad + (-m_{p1}+m_{p2})^2\big)^{\frac{1}{2}} \\ &= \big((2+m_{11}+m_{12})^2 + (2+m_{21}+m_{22})^2 + \cdots \\ &\quad + (-m_{p1}+m_{p2})^2\big)^{\frac{1}{2}}. \end{aligned}$$

The label distance between $ts1$ and $ts3$ is

$$\begin{aligned} d_{13} &= \big((2+m_{11}+m_{13})^2 + (-m_{21}+m_{23})^2 \\ &\quad + (-2-m_{31}-m_{33})^2 + \cdots + (-m_{p1}+m_{p3})^2\big)^{\frac{1}{2}} \\ &= \big((2+m_{11}+m_{13})^2 + (-m_{21}+m_{23})^2 \\ &\quad + (2+m_{31}+m_{33})^2 + \cdots + (-m_{p1}+m_{p3})^2\big)^{\frac{1}{2}}. \end{aligned}$$

In most cases, d_{12} is not equal to d_{13}. That is to say, our model can reflect the differences between different categories of samples.

By substituting $\boldsymbol{Y}$ with $\hat{\boldsymbol{Y}}$, the optimization problem of SL is

$$\begin{cases} \min\limits_{\boldsymbol{\Pi},\boldsymbol{Q},\tilde{\boldsymbol{E}},\boldsymbol{M}} \dfrac{1}{2}\Big(\big\|\boldsymbol{\Pi}^T\boldsymbol{S}^+ - \big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big) \\ \quad + \big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big)\odot\boldsymbol{M}\big)\big\|_F^2\Big) \\ \quad + \|\boldsymbol{Q}\|_* + \gamma\|\boldsymbol{Q}\|_{2,1} + \lambda\|\tilde{\boldsymbol{E}}\|_1 \\ \text{s.t. } \boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} + \tilde{\boldsymbol{E}}. \end{cases} \tag{11}$$

However, (11) is not convex. To solve this problem, we need to iteratively update each variable by fixing the other variables.

Thus (11) is formulated as

$$\begin{cases} \min\limits_{\boldsymbol{\Pi},\boldsymbol{Q},\boldsymbol{M},\tilde{\boldsymbol{E}},\boldsymbol{\Psi}_1,\boldsymbol{\Psi}_2} \dfrac{1}{2}\big\|\boldsymbol{\Pi}^T\boldsymbol{S}^+ - \big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big) \\ \quad + \big(\big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big)\big)\odot\boldsymbol{M}\big)\big\|_F^2 \\ \quad + \|\boldsymbol{\Psi}_1\|_* + \gamma\|\boldsymbol{\Psi}_2\|_{2,1} + \lambda\|\tilde{\boldsymbol{E}}\|_1 \\ \text{s.t. } \boldsymbol{\Pi}^T\boldsymbol{T}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} + \tilde{\boldsymbol{E}},\ \boldsymbol{\Psi}_1 = \boldsymbol{Q},\ \boldsymbol{\Psi}_2 = \boldsymbol{Q},\ \boldsymbol{M} \geq 0. \end{cases} \tag{12}$$

The above problem can be solved by minimizing the following augmented Lagrange multiplier (ALM) function

$$\begin{aligned} \Delta = {} & \frac{1}{2}\big\|\boldsymbol{\Pi}^T\boldsymbol{S}^+ - \big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big) \\ & + \big(\big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big)\big)\odot\boldsymbol{M}\big)\big\|_F^2 \\ & + \|\boldsymbol{\Psi}_1\|_* + \gamma\|\boldsymbol{\Psi}_2\|_{2,1} + \lambda\|\tilde{\boldsymbol{E}}\|_1 \\ & + \langle\boldsymbol{\Theta}_1, \boldsymbol{\Pi}^T\boldsymbol{T}^+ - \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} - \tilde{\boldsymbol{E}}\rangle \\ & + \langle\boldsymbol{\Theta}_2, \boldsymbol{Q} - \boldsymbol{\Psi}_1\rangle + \langle\boldsymbol{\Theta}_3, \boldsymbol{Q} - \boldsymbol{\Psi}_2\rangle \\ & + \frac{\zeta}{2}\big\|\boldsymbol{\Pi}^T\boldsymbol{T}^+ - \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} - \tilde{\boldsymbol{E}}\big\|_F^2 \\ & + \frac{\zeta}{2}\big(\|\boldsymbol{Q} - \boldsymbol{\Psi}_1\|_F^2 + \|\boldsymbol{Q} - \boldsymbol{\Psi}_2\|_F^2\big) \end{aligned} \tag{13}$$

where $\boldsymbol{\Theta}_1 \in \Re^{p\times q}$, $\boldsymbol{\Theta}_2 \in \Re^{N\times q}$ and $\boldsymbol{\Theta}_3 \in \Re^{N\times q}$ are Lagrange multipliers, $\zeta > 0$ is a penalty parameter. $\langle\cdot,\cdot\rangle$ is the inner production. We can use inexact ALM (IALM) [45–48] to solve the above problem.

The closed form solution of $\boldsymbol{\Pi}$ is

$$\begin{aligned} \boldsymbol{\Pi}^* = {} & \big(\boldsymbol{S}^+\big(\boldsymbol{S}^+\big)^T + \zeta\big(\boldsymbol{T}^+ - \boldsymbol{S}^+\boldsymbol{Q}\big)\big(\boldsymbol{T}^+ - \boldsymbol{S}^+\boldsymbol{Q}\big)^T + \sigma\boldsymbol{I}\big)^{-1} \\ & \times\Bigg(\boldsymbol{S}^+\Big(\big(\big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big) \\ & + \big(\big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big)\big)\odot\boldsymbol{M}\big)\Big)^T \\ & + \zeta\big(\boldsymbol{T}^+ - \boldsymbol{S}^+\boldsymbol{Q}\big)\Big(\tilde{\boldsymbol{E}} - \frac{\boldsymbol{\Theta}_1}{\zeta}\Big)^T\Bigg). \end{aligned} \tag{14}$$

The closed form solution of $\boldsymbol{Q}$ is

$$\begin{aligned} \boldsymbol{Q}^* = {} & \big(\zeta\big(\boldsymbol{S}^+\big)^T\boldsymbol{\Pi}\boldsymbol{\Pi}^T\boldsymbol{S}^+ + 2\zeta\boldsymbol{I}\big)^{-1}\Bigg(\Big(\boldsymbol{\Psi}_1 - \frac{\boldsymbol{\Theta}_2}{\zeta}\Big) + \Big(\boldsymbol{\Psi}_2 - \frac{\boldsymbol{\Theta}_3}{\zeta}\Big) \\ & - \big(\boldsymbol{S}^+\big)^T\boldsymbol{\Pi}\Big(\boldsymbol{\Pi}^T\boldsymbol{T}^+ - \tilde{\boldsymbol{E}} + \frac{\boldsymbol{\Theta}_1}{\zeta}\Big)\Bigg). \end{aligned} \tag{15}$$

The closed form solution of $\boldsymbol{\Psi}_1 \in \Re^{q\times N}$ is

$$\boldsymbol{\Psi}_1^* = \vartheta_{1/\zeta}\Big(\boldsymbol{Q} + \frac{\boldsymbol{\Theta}_2}{\zeta}\Big) \tag{16}$$

where $\vartheta_\tau(\Xi) = \tilde{\boldsymbol{U}}\Gamma_\tau(\boldsymbol{\Sigma})\tilde{\boldsymbol{V}}^T$ is a thresholding operator with respect to τ; $\Gamma_\tau(\Sigma_{ij}) = \text{sign}(\Sigma_{ij})\max(0, |\Sigma_{ij} - \tau|)$ is the soft-thresholding operator. τ is the singular value. $\boldsymbol{\Xi} = \tilde{\boldsymbol{U}}\boldsymbol{\Sigma}\tilde{\boldsymbol{V}}^T$ is the singular value decomposition of $\boldsymbol{\Xi}$.

Algorithm 1 Solving the problem in (11) by Inexact ALM.

Input: $\boldsymbol{S}^+$, $\boldsymbol{T}^+$, $\boldsymbol{Y}$, $\gamma = 1$ and $\lambda = 0.01$.
Initialize: $\boldsymbol{M} = \boldsymbol{1}$; $\boldsymbol{Q} = \boldsymbol{\Psi}_1 = \boldsymbol{\Psi}_2 = \boldsymbol{0}$; $\boldsymbol{\Theta}_1 = \boldsymbol{0}$, $\boldsymbol{\Theta}_2 = \boldsymbol{0}$, $\boldsymbol{\Theta}_3 = \boldsymbol{0}$; $\tilde{\boldsymbol{E}} = \boldsymbol{0}$; $\zeta_{\max} = 10^9$, $\rho = 1.2$, $\zeta = 0.2$, $\varepsilon = 10^{-9}$.
While not converged **do**

1: Fix the others and update $\boldsymbol{\Pi}$ by (14).
2: Fix the others and update $\boldsymbol{Q}$ by (15).
3: Fix the others and update $\boldsymbol{\Psi}_1$ by (16).
4: Fix the others and update $\boldsymbol{\Psi}_2$ by (17).
5: Fix the others and update $\tilde{\boldsymbol{E}}$ by (18).
item[6:] Fix the others and update $\boldsymbol{M}$ by (19).
7: Update $\boldsymbol{\Theta}_1$, $\boldsymbol{\Theta}_2$, $\boldsymbol{\Theta}_3$ and ζ by (20).
8: Check the convergence conditions:
$\|\boldsymbol{\Pi}^T\boldsymbol{T}^+ - \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} - \tilde{\boldsymbol{E}}\|_\infty < \varepsilon$, $\|\boldsymbol{Q} - \boldsymbol{\Psi}_1\|_\infty < \varepsilon$, $\|\boldsymbol{Q} - \boldsymbol{\Psi}_2\|_\infty < \varepsilon$

End while
Output: $\boldsymbol{\Pi}$, $\boldsymbol{Q}$, $\tilde{\boldsymbol{E}}$

$\boldsymbol{\Psi}_2 \in \Re^{q\times N}$ is updated by

$$\boldsymbol{\Psi}_2^* = \arg\min_{\boldsymbol{\Psi}_2}\Big\|\boldsymbol{Q} + \frac{\boldsymbol{\Theta}_3}{\zeta} - \boldsymbol{\Psi}_2\Big\|_F^2 + \frac{2\gamma}{\zeta}\|\boldsymbol{\Psi}_2\|_{2,1} \tag{17}$$

$\tilde{\boldsymbol{E}}$ is updated by

$$\tilde{\boldsymbol{E}}^* = \arg\min_{\tilde{\boldsymbol{E}}}\|\tilde{\boldsymbol{E}}\|_1 + \frac{\zeta}{2\lambda}\Big\|\boldsymbol{\Pi}^T\boldsymbol{T}^+ - \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} - \boldsymbol{E} + \frac{\boldsymbol{\Theta}_1}{\zeta}\Big\|_F^2. \tag{18}$$

The optimal solution of $\boldsymbol{M}$ is

$$\begin{aligned} \boldsymbol{M}^* = {} & \max\big(\big(\boldsymbol{\Pi}^T\boldsymbol{S}^+ - \big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big)\big) \\ & \odot\big((\boldsymbol{Y}\odot\boldsymbol{T}_{wo}) - \boldsymbol{O}_{ne}\big), \boldsymbol{0}\big). \end{aligned} \tag{19}$$

$\boldsymbol{\Theta}_1$, $\boldsymbol{\Theta}_2$, $\boldsymbol{\Theta}_3$ and ζ are updated by

$$\begin{cases} \boldsymbol{\Theta}_1 = \boldsymbol{\Theta}_1 + \zeta\big(\boldsymbol{\Pi}^T\boldsymbol{T}^+ - \boldsymbol{\Pi}^T\boldsymbol{S}^+\boldsymbol{Q} - \tilde{\boldsymbol{E}}\big) \\ \boldsymbol{\Theta}_2 = \boldsymbol{\Theta}_2 + \zeta(\boldsymbol{Q} - \boldsymbol{\Psi}_1) \\ \boldsymbol{\Theta}_3 = \boldsymbol{\Theta}_3 + \zeta(\boldsymbol{Q} - \boldsymbol{\Psi}_2) \\ \zeta = \min(\rho\zeta, \zeta_{\max}) \end{cases} \tag{20}$$

where ρ is the iteration step-size, and $\rho > 1$.

We can obtain $\boldsymbol{\Pi}$ by iteratively solving the above equations. Then, the discriminative subspace is obtained. After that, we obtain the test sample matrix $\tilde{\boldsymbol{T}}^+$ by mapping $\boldsymbol{T}^+$ into the discriminative subspace. Similarly, $\tilde{\boldsymbol{S}}^+$ is obtained. The expressions of $\tilde{\boldsymbol{T}}^+$ and $\tilde{\boldsymbol{S}}^+$ are

$$\begin{cases} \tilde{\boldsymbol{T}}^+ = \boldsymbol{\Pi}^T\boldsymbol{T}^+ \\ \tilde{\boldsymbol{S}}^+ = \boldsymbol{\Pi}^T\boldsymbol{S}^+ = \big[\,\tilde{\boldsymbol{\Phi}} \quad \tilde{\boldsymbol{B}}\,\big]. \end{cases} \tag{21}$$

3. The high-performance dictionary learning algorithm

For the obtained $\tilde{\boldsymbol{S}}^+ \in \Re^{p\times N}$, all samples in $\tilde{\boldsymbol{S}}^+$ are arranged into a matrix $\boldsymbol{X} = [\boldsymbol{X}_1, ..., \boldsymbol{X}_k, ..., \boldsymbol{X}_C]$, where $\boldsymbol{X}_k \in \Re^{p\times n}$ is the collection of training samples with the category k, n is the number of samples per category, C is the number of categories. Let $\boldsymbol{X} = [\boldsymbol{X}_1, ..., \boldsymbol{X}_k, ..., \boldsymbol{X}_C] = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N] \in \Re^{p\times N}$, where $N = n \times C$. The commonly used dictionary learning model can be written as $\boldsymbol{X} = \boldsymbol{D}\boldsymbol{A}$, where $\boldsymbol{D} = [\boldsymbol{d}_1, ..., \boldsymbol{d}_K] \in \Re^{p\times K}$ is the learned dictionary. $\boldsymbol{A} = [\boldsymbol{a}_1, ..., \boldsymbol{a}_N] \in \Re^{K\times N}$. $\boldsymbol{a}_i = [a_{1,i}, a_{2,i}, ..., a_{K,i}]^T \in \Re^{K\times 1}$ is the coding coefficient vector of $\boldsymbol{x}_i$, $i = 1, 2, ..., N$. K is the number of atoms. In our high-performance dictionary learning algorithm (HPDL), the atoms in $\boldsymbol{D}$ involve C categories. Let f denote the number of atoms from each category. Thus, $\boldsymbol{D}$ can also be written as $\boldsymbol{D} = [\boldsymbol{D}_1, ..., \boldsymbol{D}_C] \in \Re^{p\times K}$, where $\boldsymbol{D}_i$ is the sub-dictionary

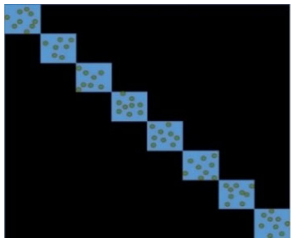


Fig. 3. Illustration of $\boldsymbol{A}$ learned using HPDL on CVL face database.

which contains all the learned dictionary atoms with the category i, $i = 1, 2, ..., C$.

As mentioned in [49], if we assign a label to an atom and construct the label constraint term, the discriminative ability of the learned dictionary can be improved.

We use the K-SVD algorithm to learn $\boldsymbol{D}_i$. If atom $d_j \in \boldsymbol{D}_i$, $j = 1, 2, ..., K$, $i = 1, 2, ..., C$, then its label denoted by $\tilde{l}_j = [0, ...1, ..., 0]^T \in \Re^{C\times 1}$. If the only nonzero element in $\tilde{l}_j$ is located in the ith entry, then d_j belongs to the ith class. Hence, the label matrix $\tilde{\boldsymbol{L}}$ of $\boldsymbol{D}$ can be written as $\tilde{\boldsymbol{L}} = [\tilde{l}_1, ..., \tilde{l}_K]^T \in \Re^{K\times C}$.

In [50], in order to group data $\{\boldsymbol{x}_i\}_{i=1}^N$ into C clusters $\{\Gamma_j\}_{j=1}^C$, the authors defined a cluster indicator matrix $\boldsymbol{Z} \in \Re^{N\times C}$: if $\boldsymbol{x}_i \in \Gamma_j$, $z_{ij} = 1$, else $z_{ij} = 0$, where z_{ij} is the i row j column element of $\boldsymbol{Z}$. Then, a weighted cluster indicator matrix $\boldsymbol{\Upsilon}$ is defined as

$$\boldsymbol{\Upsilon} = [\boldsymbol{\Upsilon}_1, \boldsymbol{\Upsilon}_2, ..., \boldsymbol{\Upsilon}_C] = \boldsymbol{Z}\left(\boldsymbol{Z}^T\boldsymbol{Z}\right)^{-\frac{1}{2}}. \tag{22}$$

Following [50], we use $\tilde{\boldsymbol{L}}$ to construct a weighted label matrix $\boldsymbol{G}$. The expression of $\boldsymbol{G}$ is

$$\boldsymbol{G} = \tilde{\boldsymbol{L}}\left(\tilde{\boldsymbol{L}}^T\tilde{\boldsymbol{L}}\right)^{-\frac{1}{2}} \in \Re^{K\times C}. \tag{23}$$

Ideally, the atoms should only reconstruct the same class of training samples [51], the learned dictionary should contain different class atoms and the atoms of each class should reconstruct one class of the training samples. To achieve this goal, we construct a label embedding term to learn the dictionary $\boldsymbol{D}$. The minimization of the label embedding term is

$$\min_{\boldsymbol{A}} \mathrm{Tr}\left(\boldsymbol{A}^T\boldsymbol{U}\boldsymbol{U}^T\boldsymbol{A}\right) \tag{24}$$

where $\boldsymbol{U} = \boldsymbol{G}\boldsymbol{G}^T \in \Re^{K\times K}$ is the scaled label matrix of the dictionary $\boldsymbol{D}$, it has a block-diagonal structure. Obviously, $\boldsymbol{U}$ is a constant. In order to achieve the above goal, it is necessary to show that the label embedding term can force $\boldsymbol{A}$ to be diagonal. Thus, we did an experiment to verify it. Fig. 3 shows the structure of $\boldsymbol{A}$, which is obtained by the verification experiment. From Fig. 3 we can see that $\boldsymbol{A}$ is close to block-diagonal. This illustrates that the label embedding term can facilitate the atoms of each class reconstruct one class of the training samples.

The purpose of dictionary learning is to learn a distinctive dictionary $\boldsymbol{D}$, which makes $\|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{A}\|_F^2$ the minimum.

However, the label information and the protection of edge details of reconstructed samples are not simultaneously considered in the dictionary learning process. Besides, many current dictionary learning algorithms require a lot of time to complete the classification task. In this paper, we propose a high-performance dictionary learning algorithm (HPDL). In HPDL, on the one hand, the label information is used to construct the label embedding term, which makes the atoms only reconstruct the same class of training samples. On the other hand, we construct a non-local self-similarity term and use it to protect the edge detail of the reconstructed sample [52]. The construction of the non-local self-similarity term is based on the following assumptions. If the given data point $\boldsymbol{x}_j$ is the jth most similar data point to $\boldsymbol{x}_i$ in a nonlocal neighborhood, then the corresponding coding coefficient $\boldsymbol{a}_j$ is also the jth most similar coding coefficients for $\boldsymbol{a}_i$ in a nonlocal neighborhood. Then the non-local self-similarity term can be defined as

$$\sum_{i=1}^{N}\left\|\boldsymbol{a}_i - \sum_j \boldsymbol{W}^{ji}\boldsymbol{a}_j\right\|_2^2 = \|\boldsymbol{A} - \boldsymbol{A}\boldsymbol{W}\|_F^2 \tag{25}$$

where $\boldsymbol{W}^{ji}$ is the weight between a data point $\boldsymbol{x}_j$ and $\boldsymbol{x}_i$.

As in [17], an analysis dictionary $\boldsymbol{P} \in \Re^{K\times p}$ is introduced into the objective function to learn the optimal coding coefficients, which greatly reduces the time complexity. $\boldsymbol{P}$ is subject to $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{X} \in \Re^{K\times N}$. Then the optimization problem with respect to the objective function of our HPDL is

$$\begin{cases} \min\limits_{\boldsymbol{P}_k,\boldsymbol{A}_k,\boldsymbol{D}_k} \sum\limits_{k=1}^{C}\|\boldsymbol{X}_k - \boldsymbol{D}_k\boldsymbol{A}_k\|_F^2 + \alpha\|\boldsymbol{A}_k - \boldsymbol{A}_k\boldsymbol{W}_k\|_F^2 \\ \quad + \beta\,\mathrm{Tr}\left(\boldsymbol{A}_k^T\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{A}_k\right) + \delta\|\boldsymbol{P}_k\bar{\boldsymbol{X}}_k\|_F^2 \\ \quad + \eta\|\boldsymbol{P}_k\boldsymbol{X}_k - \boldsymbol{A}_k\|_F^2 \\ \text{s.t. } \left\|d_k^i\right\|_2^2 \le 1 \end{cases} \tag{26}$$

where $\boldsymbol{D} = [\boldsymbol{D}_1, ..., \boldsymbol{D}_k, ..., \boldsymbol{D}_C]d_k^i$ is the ith atom in $\boldsymbol{D}_k$. $\boldsymbol{A}_k \in \Re^{m\times n}$ is the coding coefficient matrix which corresponds to the class k, the labels of whose corresponding coding atoms are embedded in the dictionary learning process. $\boldsymbol{P}_k \in \Re^{m\times p}$ is the analysis dictionary which corresponds to the class k, $\boldsymbol{P} = [\boldsymbol{P}_1; ...; \boldsymbol{P}_k; ...; \boldsymbol{P}_C]$. $\bar{\boldsymbol{X}}_k$ denotes the complementary data matrix of $\boldsymbol{X}_k$ in the whole training set $\boldsymbol{X}$. α, β, δ and η are the regularization parameters. $\boldsymbol{W}$ is the weight matrix, and

$$\boldsymbol{W}_k^{ij} = \begin{cases} \dfrac{2\exp(-\frac{\|X_k^i - X_k^j\|_2^2}{\hat{h}})}{\sum_i\sum_j \exp(-\frac{\|X_k^i - X_k^j\|_2^2}{\hat{h}})}, & X_k^i, X_k^j \in \boldsymbol{X}_k \text{ and } i \ne j \\ 0, & X_k^i, X_k^j \in \boldsymbol{X}_k \text{ and } i = j \end{cases} \tag{27}$$

where $\boldsymbol{W}_k^{ij}$ is the i row j column element of $\boldsymbol{W}_k \in \Re^{n\times n}$. $\boldsymbol{W}_k$ is the weight matrix $\boldsymbol{W}$ which corresponds to the class k. X_k^i is the ith element in $\boldsymbol{X}_k$, X_k^j is the jth element in $\boldsymbol{X}_k$. $\boldsymbol{U}_k \in \Re^{f\times f}$ is the scaled label matrix which corresponds to the class k. It is located in the diagonal of $\boldsymbol{U}$. $\hat{h}$ is a parameter, we let $\hat{h} = 0.02$.

$\sum_{k=1}^{C}\|\boldsymbol{X}_k - \boldsymbol{D}_k\boldsymbol{A}_k\|_F^2$ is the reconstruction error term. $\sum_{k=1}^{C}\|\boldsymbol{A}_k - \boldsymbol{A}_k\boldsymbol{W}_k\|_F^2$ is the non-local self-similarity term, which preserves the edge detail of the reconstructed sample. $\sum_{k=1}^{C}\mathrm{Tr}(\boldsymbol{A}_k^T\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{A}_k)$ is the label embedding term, which forces the atoms in the dictionary $\boldsymbol{D}$ only reconstruct the training samples with the same class. $\sum_{k=1}^{C}\delta\|\boldsymbol{P}_k\bar{\boldsymbol{X}}_k\|_F^2 + \eta\|\boldsymbol{P}_k\boldsymbol{X}_k - \boldsymbol{A}_k\|_F^2$ is the time complexity drop term, which reduces the time complexity of dictionary learning algorithm.

The main steps of solving (26) are as follows.

Step 1 (Update $\boldsymbol{A}$): $\boldsymbol{A}$ can be updated by

$$\begin{aligned} \min_{\boldsymbol{A}_k} &\sum_{k=1}^{C}\|\boldsymbol{X}_k - \boldsymbol{D}_k\boldsymbol{A}_k\|_F^2 + \alpha\|\boldsymbol{A}_k - \boldsymbol{A}_k\boldsymbol{W}_k\|_F^2 \\ &+ \beta\,\mathrm{Tr}\left(\boldsymbol{A}_k^T\boldsymbol{U}_k\boldsymbol{U}_k^T\boldsymbol{A}_k\right) + \eta\|\boldsymbol{P}_k\boldsymbol{X}_k - \boldsymbol{A}_k\|_F^2. \end{aligned} \tag{28}$$

By setting the derivative with respect to $\boldsymbol{A}_k$ to zero, we have

$$\begin{aligned} &\left[\boldsymbol{D}_k^T\boldsymbol{D}_k + \beta\boldsymbol{U}_k\boldsymbol{U}_k^T\right]\boldsymbol{A}_k \\ &\quad + \boldsymbol{A}_k\left[(\alpha+\eta)\mathbf{I} - \alpha\left(\boldsymbol{W}_k + \boldsymbol{W}_k^T\right) + \alpha\boldsymbol{W}_k\boldsymbol{W}_k^T\right] \\ &= \left(\boldsymbol{D}_k^T\boldsymbol{X}_k + \eta\boldsymbol{P}_k\boldsymbol{X}_k\right). \end{aligned} \tag{29}$$

Algorithm 2 DLSL.

Input: Training sample set $\boldsymbol{S}^+$, test sample set $\boldsymbol{T}^+$. $\alpha = 0.3$, $\beta = 0.3$, $\delta = 0.003$, $\eta = 0.8$, $\xi = 10^{-6}$, $\hat{h} = 0.02$, $\zeta = 1$.

1: Obtain $\boldsymbol{\Pi}$ by subspace learning.
2: Obtain $\tilde{\boldsymbol{S}}^+$ and $\tilde{\boldsymbol{T}}^+$.
3: Compute $\boldsymbol{U}$.
4: Obtain the dictionary $\boldsymbol{P}$ and $\boldsymbol{D}$.
5: Compute the residual.
$identity(\boldsymbol{y}) = \arg\min_i \|\boldsymbol{y} - \boldsymbol{D}_i\boldsymbol{P}_i\boldsymbol{y}\|_2$

Output: The labels of test samples

(29) is a Sylvester equation, which can be solved by the method in [53].

Step 2 (Update $\boldsymbol{P}$): $\boldsymbol{P}$ can be updated by

$$\min_{\boldsymbol{P}_k} \sum_{k=1}^{C} \delta\|\boldsymbol{P}_k\bar{\boldsymbol{X}}_k\|_F^2 + \eta\|\boldsymbol{P}_k\boldsymbol{X}_k - \boldsymbol{A}_k\|_F^2. \tag{30}$$

The closed form solution of (32) is

$$\boldsymbol{P}_k = \eta\boldsymbol{A}_k\boldsymbol{X}_k^T\left(\eta\boldsymbol{X}_k\boldsymbol{X}_k^T + \delta\bar{\boldsymbol{X}}_k\bar{\boldsymbol{X}}_k^T + \xi\mathbf{I}\right)^{-1} \tag{31}$$

where ξ is a small number, which ranges from 10^{-7} to 10^{-5}.

Step 3 (Update $\boldsymbol{D}$): $\boldsymbol{D}$ can be updated by

$$\begin{cases} \min\limits_{\boldsymbol{D}_k} \sum\limits_{k=1}^{C} \|\boldsymbol{X}_k - \boldsymbol{D}_k\boldsymbol{A}_k\|_F^2 \\ \text{s.t. } \left\|d_k^i\right\|_2^2 \le 1. \end{cases} \tag{32}$$

The optimal solution of (32) can be obtained by introducing a variable $\boldsymbol{V}$ and using the ADMM algorithm. Thus, we obtain

$$\begin{cases} \boldsymbol{D}^{(h+1)} = \arg\min\limits_{\boldsymbol{D}} \sum\limits_{k=1}^{C} \|\boldsymbol{X}_k - \boldsymbol{D}_k\boldsymbol{A}_k\|_F^2 + \zeta\left\|\boldsymbol{D}_k - \boldsymbol{V}_k^h + \boldsymbol{J}_k^h\right\|_F^2 \\ \boldsymbol{V}^{(h+1)} = \arg\min\limits_{\boldsymbol{V}} \sum\limits_{k=1}^{C} \zeta\left\|\boldsymbol{D}_k^{(h+1)} - \boldsymbol{V}_k + \boldsymbol{J}_k^h\right\|_F^2, \quad \text{s.t. } \left\|v_k^i\right\|_2^2 \le 1 \\ \boldsymbol{J}^{(h+1)} = \boldsymbol{J}_k^h + \boldsymbol{D}_k^{(h+1)} - \boldsymbol{V}_k^{(h+1)}. \end{cases} \tag{33}$$

In our HPDL model, because $\boldsymbol{D}_k^*$ is trained to reconstruct the samples of class k, thus $\|\boldsymbol{X}_k - \boldsymbol{D}_k^*\boldsymbol{P}_k^*\boldsymbol{X}_k\|_F^2 \ll \|\boldsymbol{X}_i - \boldsymbol{D}_k^*\boldsymbol{P}_k^*\boldsymbol{X}_i\|_F^2$.

Hence, we naturally have the following classifier

$$identity(\boldsymbol{y}) = \arg\min_i \|\boldsymbol{y} - \boldsymbol{D}_i\boldsymbol{P}_i\boldsymbol{y}\|_2 \tag{34}$$

where $\boldsymbol{y} \in \Re^{p\times 1}$ is an arbitrary test sample.

4. Face recognition algorithm based on dictionary learning and subspace learning

4.1. DLSL algorithm

In this paper, we propose a face recognition algorithm based on dictionary learning and subspace learning (DLSL). In DLSL, we first use the subspace learning algorithm (SL) to learn a common subspace where the distribution differences between the source and target domains reduce. Then the training set and test set are mapped into the common subspace which is robust to noise.

Finally, the test samples are recognized by the proposed HPDL in the common subspace. Fig. 1 shows the flow chart of DLSL.

4.2. Computational complexity

The computational complexity of DLSL is evaluated in the present section. In SL stage, its major computational burden lies in updating $\boldsymbol{\Pi}$, $\boldsymbol{Q}$ and $\boldsymbol{\Psi}_1$. The complexity of updating $\boldsymbol{\Pi}$ is $\boldsymbol{O}(d^2(N+q) + d^3 + d(q+N)C)$. The complexity of updating $\boldsymbol{Q}$ is $\boldsymbol{O}((d+N)qC + NdC + N^3 + d^2N^2)$. The complexity of updating $\boldsymbol{\Psi}_1$ is $\boldsymbol{O}(q^3)$. Thus, the main computational complexity of SL is $\boldsymbol{O}(t(d^2(N+q) + d^3 + q^3 + N^3))$, where t is the number of iterations. In the dictionary learning stage, when we train the HPDL, the major computation burden comes from updating $\boldsymbol{P}$ with a $\boldsymbol{O}(mnp + p^3 + mp^2)$ complexity. In many applications, n and m are much smaller than p. When $\boldsymbol{P}$ is updated in each iteration, the major complexity is to solve the inverse of $\mathrm{H} = \eta\boldsymbol{X}_k\boldsymbol{X}_k^T + \delta\bar{\boldsymbol{X}}_k\bar{\boldsymbol{X}}_k^T + \xi\mathbf{I}$, which has a $\boldsymbol{O}(p^3)$ complexity. However, H will not change in the iterations, thus its inverse can be precomputed. This greatly reduces complexity. That's why we call $\sum_{k=1}^{C}\delta\|\boldsymbol{P}_k\bar{\boldsymbol{X}}_k\|_F^2 + \eta\|\boldsymbol{P}_k\boldsymbol{X}_k - \boldsymbol{A}_k\|_F^2$ the time complexity drop term. In the test phase, the computation of class-specific reconstruction error only has a complexity of $\boldsymbol{O}(mp)$. Thus, the total complexity of HPDL to classify a query sample is $\boldsymbol{O}(mpC)$. In our method, d is greater than p. Hence, the complexity of our DLSL is approximately $\boldsymbol{O}(t(d^2(N+q) + d^3 + q^3 + N^3))$.

4.3. Advantages and disadvantages

The advantages and disadvantages of DLSL are summarized as follows. DLSL can reduce the distribution differences between the source and target domains by SL so that it can effectively deal with the face recognition problems involving pose variation, occlusion, etc. Besides, DLSL learns a discriminative dictionary by HPDL, which further enhances the performance of DLSL. Despite the promising performance of DLSL, there is still room for further improvement. For example, in DLSL the optimization of subspace learning and the optimization of dictionary learning are separate, thus DLSL can only obtain the suboptimal solution. This limits the performance of DLSL. In our future research work, we will integrate subspace learning and dictionary learning into an objective function for optimization so that our DLSL can get the optimal solution. In the aspect of real-time, although DLSL is better than many other current dictionary learning algorithms, it is still difficult to meet real-time requirements when dealing with large databases. Hence, our other research direction in the future is to make DLSL better meet the real-time requirements.

5. Experiments

In this section, we used several databases to test and verify the effectiveness of the proposed DLSL. The databases used include the FRGC face database, LFW face database, CVL face database, Yale B face database and AR face database. We compare our approach with other methods on the robustness to different cases, such as noise and large pose variation. In the comparison method, SRC [5], RRC [8], ESRC [9], OPR [54] and FSSP [55] are the sparse representation methods. K-SVD [12], LC-KSVD [13], D-KSVD [14], FDDL [16], DPL [17], DLRD [56], LRDL [57] are the dictionary learning methods. ConvNet-RBM [35], MDFR [36] and RRNN [37] are the deep learning methods or neural network methods. Razzaghi's method [27], TTRLSR [22], DKTL [28], DSTL [29], Xu's method [30], CD-ALPHN [58], TSSA [59], Appice's method [60] and SPDA [61] are the transductive transfer learning methods.

5.1. Face recognition with noise

We use FRGC and LFW databases to test DLSL's robustness to noise.

Table 1
Recognition rates (%) of different algorithms on the FRGC database with noise.

Noise	Gaussian noise				Multiplicative noise				Poisson noise			
Corrupted ratio (%)	7	10	13	Average	7	10	13	Average	7	10	13	Average
SNR (dB)	15.3	11.5	6.7		14.2	10.3	6.1		17.9	13.1	8.2	
SRC [5]	76.63	74.29	73.03	74.65	74.11	71.23	70.03	71.78	79.56	77.24	76.02	77.60
RRC [8]	80.33	78.28	75.16	77.92	77.82	75.39	73.01	75.40	82.19	80.05	76.86	79.70
K-SVD [12]	82.12	80.20	77.32	79.88	80.35	78.06	75.68	78.03	84.66	83.01	80.22	82.63
LC-KSVD [13]	82.44	80.42	77.92	80.26	79.95	77.26	74.78	77.33	85.03	83.02	79.67	82.57
D-KSVD [14]	82.78	81.21	79.23	81.07	80.61	79.06	77.38	79.01	85.42	84.16	82.03	83.87
FDDL [16]	84.44	81.24	78.35	81.31	83.08	80.17	77.68	80.31	87.21	84.09	80.46	83.92
ConvNet-RBM [35]	87.32	84.61	82.75	84.89	86.67	84.62	82.07	84.45	89.62	86.23	84.11	86.65
MDFR [36]	85.79	82.11	80.11	82.67	84.39	81.08	79.17	81.54	87.03	83.69	81.29	84.00
PLR [62]	86.68	83.05	81.46	83.73	84.95	81.34	78.65	81.64	88.91	85.36	83.41	85.89
RRNN [37]	87.25	85.28	82.41	84.98	85.62	83.19	79.56	82.79	89.68	87.06	84.55	87.09
OPR [54]	86.27	82.37	80.59	83.07	85.04	79.68	77.43	80.71	89.01	84.58	82.21	85.26
Razzaghi [27]	87.56	86.01	84.22	85.93	87.19	84.76	83.02	84.99	91.29	89.35	86.37	89.00
TTRLSR [22]	85.92	83.17	82.44	83.84	85.11	82.43	80.10	82.54	88.63	86.52	84.81	86.65
DKTL [28]	86.47	85.88	84.13	85.49	86.91	84.50	82.55	84.65	90.01	87.22	84.05	87.09
DSTL [29]	85.56	83.01	81.37	83.31	84.66	82.08	79.55	82.09	88.18	86.36	84.20	86.24
Xu [30]	87.22	85.69	84.01	85.64	86.69	84.45	82.11	84.41	90.07	87.24	84.16	87.15
CD-ALPHN [58]	86.48	84.20	83.01	84.56	86.53	84.91	83.01	84.81	90.92	88.53	86.22	88.55
TSSA [59]	84.68	83.25	81.58	83.17	85.49	83.94	82.59	84.00	89.15	87.48	85.04	87.22
Appice [60]	85.05	84.61	83.26	84.30	85.76	84.01	82.10	83.95	89.40	87.92	85.82	87.71
SPDA [61]	87.55	85.91	84.29	85.91	87.27	85.39	83.28	85.31	90.58	88.10	85.63	88.10
DLSL	**90.21**	**88.53**	**86.08**	**88.27**	**90.66**	**87.86**	**85.34**	**87.95**	**93.11**	**91.47**	**88.73**	**91.10**

Table 2
Recognition rates (%) of different algorithms on the LFW database with noise.

Noise	Gaussian noise				Multiplicative noise				Poisson noise			
Corrupted ratio (%)	7	10	13	Average	7	10	13	Average	7	10	13	Average
SNR (dB)	15.3	11.5	6.7		14.2	10.3	6.1		17.9	13.1	8.2	
SRC [5]	76.30	74.18	72.59	74.35	72.24	71.06	69.57	70.95	80.23	79.01	77.36	78.86
RRC [8]	80.27	78.36	77.01	78.54	75.39	73.81	71.25	73.48	82.37	80.24	78.91	80.50
K-SVD [12]	77.26	75.29	72.13	74.89	74.22	72.64	70.08	72.31	80.95	78.34	76.96	78.75
LC-KSVD [13]	78.61	75.57	73.19	75.79	73.61	72.03	70.68	72.10	81.68	79.85	77.62	79.71
D-KSVD [14]	79.36	77.81	75.35	77.50	74.02	72.35	71.27	72.54	82.89	81.03	78.99	80.97
FDDL [16]	80.17	78.27	77.12	78.52	75.93	72.88	72.09	73.63	83.26	81.72	80.20	81.72
ConvNet-RBM [35]	79.37	77.04	75.24	77.21	75.47	71.49	70.47	72.47	81.10	80.16	77.37	79.54
MDFR [36]	80.92	77.92	75.20	78.01	77.91	76.11	73.56	75.86	81.83	80.35	78.03	80.07
PLR [62]	81.33	79.03	77.28	79.21	78.82	77.16	74.27	76.75	83.05	81.28	79.78	81.37
RRNN [37]	80.49	77.42	75.07	77.66	78.03	77.02	73.79	76.28	82.52	79.47	77.95	79.98
OPR [54]	82.11	80.46	77.82	80.13	79.38	77.36	75.61	77.45	85.37	83.15	82.26	83.59
Razzaghi [27]	83.05	81.33	77.96	80.78	79.55	77.52	77.03	78.03	86.92	85.39	82.11	84.80
TTRLSR [22]	79.57	77.28	75.22	77.35	78.06	76.11	75.01	76.39	84.58	82.22	78.55	81.78
DKTL [28]	82.19	80.26	77.13	79.86	78.88	77.01	75.87	77.25	86.05	84.92	81.48	84.15
DSTL [29]	79.33	77.01	74.86	77.06	77.93	75.90	75.06	76.29	84.07	82.10	77.42	81.19
Xu [30]	81.33	79.46	76.61	79.13	78.83	76.30	75.22	76.78	85.21	82.71	80.07	82.66
CD-ALPHN [58]	83.70	80.57	77.81	80.69	78.56	77.31	75.26	77.04	85.39	83.05	82.11	83.51
TSSA [59]	83.68	81.44	78.22	81.11	78.39	77.14	75.09	76.87	85.20	82.86	81.78	83.28
Appice [60]	81.67	79.59	76.29	79.18	77.40	75.48	73.89	75.59	84.11	82.11	80.75	82.32
SPDA [61]	84.30	82.24	78.34	81.62	80.91	77.53	76.92	78.45	87.48	85.03	82.57	85.02
DLSL	**85.27**	**83.42**	**80.93**	**83.20**	**82.47**	**80.28**	**78.46**	**80.40**	**89.48**	**87.46**	**84.49**	**87.14**

FRGC version 2.0 is a large-scale face database. Here, we use a subset of FRGC to do experiments. The subset contains 7318 images which belong to 316 different people. These images involve large lighting, accessory (e.g., glasses), expression variation and image blur, etc. Some samples of FRGC are shown in Fig. 4. For each person, we randomly select six images as the training set, and the rest images are used for the test set.

Labeled Faces in the Wild (LFW) is a face database, which contains more than 13000 images. Fig. 5 shows some samples of LFW. Here, the whole LFW database is used for experiments. We randomly select 60% of the samples from each subject as the training set. Here, the number of training sample per subject may not be an integer, hence we round off it. The rest samples are used as the test set.

For each test image in FRGC and LFW, we respectively add Gaussian noise, multiplicative noise and Poisson noise to it. That is, for any test image m, we add a noise n to it. Then we obtain the noisy image $\tilde{m}$. $\tilde{m} = m + \hat{\eta}n$, where $\hat{\eta}$ is the corruption ratio. In our experiments, each image is cropped to 32×42. The corruption ratios are 7%, 10%, and 13% respectively. We also mark the corresponding signal to noise ratio (SNR) in Table 1 and 2.

Fig. 4. Some samples in the FRGC face database.

Table 1 lists the recognition rates of different algorithms on the FRGC database. In the experiments of adding Gaussian noise,

Fig. 5. Some samples in the LFW face database.

Fig. 6. Some samples in the CVL database.

the average recognition rate of DLSL is 88.27%, which is the highest recognition rates, with about 11%, 7%, 6%, 5%, 5%, 5%, 5%, 5%, 4%, 4%, 4%, 3% and 3% improvements over RRC, FDDL, MDFR, OPR, PLR, TTRLSR, DSTL, TSSA, ConvNet-RBM, RRNN, CD-ALPHN, SPDA and Xu's method respectively. In the experiments of adding multiplicative noise, the average recognition rate of DLSL is 87.95%, which is higher than those of other algorithms. In the experiments of adding Poisson noise, DLSL also outperforms other algorithms.

Table 2 shows the recognition rates of different algorithms on the LFW database. In the experiments of adding Gaussian noise, we observe that the average recognition rate of DLSL is 83.20%, which is the highest recognition rates, with about 6%, 6%, 5%, 5%, 4%, 4%, 3% and 3% improvements over TTRLSR, RRNN, MDFR, FDDL, DKTL, Appice's method, Razzaghi's method and OPR respectively. In the experiments of adding multiplicative noise, the average recognition rate of DLSL is 80.40%, which is also higher than those of other algorithms. In the experiments of adding Poisson noise, the average recognition rate of DLSL is 87.14%, which is the highest recognition rates, with about 8%, 6%, 6%, 6%, 6%, 4%, 4% and 3% improvements over RRNN, TTRLSR, DSTL, FDDL, PLR, OPR, TSSA and DKTL respectively.

5.2. Face recognition with pose variation

Here, we use the CVL database to evaluate the performance of DLSL. CVL database involves 114 different people. For each person, there are seven different images. **The images in the CVL database mainly involve large pose variation.** As in [55], we use a subset of CVL database, which contains 770 images of 110 people. For each person, four to six images are respectively selected as the training set, and the rest images are used for the test set. Each image in the subset is cropped to 30×40. Fig. 6 shows some samples of CVL database.

Table 3 depicts the recognition rates of different algorithms on the CVL database. In Table 3, when the numbers of training samples per class are four, five and six, the recognition rates of DLSL are 66.33%, 90.56% and 95.45% respectively, which are higher than those of other algorithms. The average recognition rate of DLSL is 84.11%, which is the highest recognition rates, with about 9%, 7%, 7%, 6%, 6%, 5%, 5%, 4%, 4% and 4% improvements over RRNN, TSSA, OPR, PLR, DSTL, MDFR, ConvNet-RBM, DKTL, SPDA and Xu's method respectively.

Table 3
Recognition rates (%) of different algorithms on the CVL database.

The number of training samples per class	4	5	6	Average
SRC [5]	39.33	46.00	62.00	49.11
RRC [8]	36.67	40.00	58.00	44.89
FSSP (1) [55]	42.00	53.00	70.00	55.00
FSSP (2) [55]	41.33	53.00	70.00	54.78
ESRC [9]	38.07	44.65	58.25	46.99
DPL [17]	31.97	42.22	51.52	41.90
K-SVD [12]	42.44	53.56	62.33	52.78
LC-KSVD [13]	48.32	55.02	76.01	59.78
D-KSVD [14]	44.51	49.85	73.01	55.79
FDDL [16]	52.48	59.02	79.23	63.58
ConvNet-RBM [35]	60.56	87.15	91.78	79.83
MDFR [36]	60.22	86.93	91.23	79.46
PLR [62]	59.45	86.52	90.36	78.77
RRNN [37]	56.21	83.11	86.89	75.40
OPR [54]	58.26	84.19	89.33	77.26
Razzaghi [27]	63.44	88.35	92.57	81.45
TTRLSR [22]	61.56	85.47	89.78	78.93
DKTL [28]	62.68	87.22	91.58	80.49
DSTL [29]	60.77	85.20	88.93	78.30
Xu [30]	62.49	87.38	91.44	80.43
CD-ALPHN [58]	61.70	85.02	88.92	78.54
TSSA [59]	60.77	84.80	87.23	77.60
Appice [60]	61.45	85.62	88.93	78.66
SPDA [61]	63.58	86.92	90.89	80.46
DLSL	**66.33**	**90.56**	**95.45**	**84.11**

Fig. 7. Some samples in the Yale B database.

5.3. Face recognition with illumination variation

Yale B database contains 5760 single source images of 10 subjects. For every subject in a particular pose, an image with ambient illumination was also captured. Fig. 7 shows some samples in the Yale B database. For each subject, we randomly select 100 images as the training set, and the rest images are used as the test set. The size of each image is cropped to 32×42.

Table 4 presents the recognition rates of different algorithms on the Yale B database. In Table 4 we can see that our DLSL outperforms other algorithms. For example, the recognition rate of DLSL is about 7%, 6%, 6%, 5%, 4%, 3% and 3% higher than those of OPR, TSSA, RRNN, TTRLSR, DSTL, FDDL and Xu's method respectively.

5.4. Face recognition with expression variation

In this section, we use the LFW face database to evaluate the performance of our DLSL. Here, a subset LFWc is used in the experiments. In LFWc, each subject contains no less than eight images and no more than 13 images. The total number of images in LFWc is 1087. For each subject in LFWc, we randomly select five images

Table 4
Recognition rates (%) of different algorithms on the Yale B database.

Algorithm	Recognition rate
NPE [63]	81.33
LSDA [64]	82.46
LatLRR [65]	85.31
ESRC [9]	81.20
RRC [8]	84.28
DPL [17]	86.49
K-SVD [12]	83.44
LC-KSVD [13]	88.21
D-KSVD [14]	85.03
FDDL [16]	89.72
ConvNet-RBM [35]	87.24
MDFR [36]	88.37
PLR [62]	88.02
RRNN [37]	86.30
OPR [54]	85.81
Razzaghi [27]	90.22
TTRLSR [22]	87.33
DKTL [28]	89.20
DSTL [29]	88.39
Xu [30]	89.51
CD-ALPHN [58]	90.14
TSSA [59]	86.91
Appice [60]	88.52
SPDA [61]	89.57
DLSL	**92.33**

Table 5
Recognition rates (%) of different algorithms on the LFW database.

Algorithm	Recognition rate
NPE [63]	74.29
LSDA [64]	78.24
LatLRR [65]	81.27
ESRC [9]	74.60
RRC [8]	70.41
DPL [17]	87.15
K-SVD [12]	85.26
LC-KSVD [13]	86.01
D-KSVD [14]	85.38
FDDL [16]	86.33
ConvNet-RBM [35]	91.21
MDFR [36]	91.30
PLR [62]	88.47
RRNN [37]	91.25
OPR [54]	90.03
Razzaghi [27]	94.01
TTRLSR [22]	90.30
DKTL [28]	92.51
DSTL [29]	91.47
Xu [30]	92.35
CD-ALPHN [58]	91.33
TSSA [59]	90.86
Appice [60]	91.35
SPDA [61]	93.01
DLSL	**95.82**

as the training set, and the rest images are used as the test set. The size of each image is 64×64.

Table 5 shows the recognition rates of different algorithms on the LFW database. It can be seen from Table 5, the recognition rate of DLSL is 95.82%, which is the highest recognition rates, with about 9%, 8%, 7%, 5%, 5%, 5%, 4%, 4%, 4% and 4% improvements over FDDL, DPL, PLR, OPR, TTRLSR, TSSA, DSTL, RRNN, ConvNet-RBM and Appice's method respectively.

5.5. Face recognition with occlusion

In this section, we use the AR database to test the robustness of our DLSL against occlusion. AR database includes more than 4000 images, which involves 126 different persons. For each person, there are 26 different images, which involve illumination variation, expression variation and occlusion. Fig. 8 shows some samples in the AR database. Here, the size of each image is cropped to 64×64. The experimental conditions are set as follows.


(a) Sunglasses (b) Scarves

Fig. 8. Some samples in the AR database.

Table 6
Recognition rates (%) of different algorithms on the AR database.

Algorithm	Sunglasses	Scarf	Average
NPE [63]	83.75	82.58	83.17
LSDA [64]	83.66	81.83	82.75
LatLRR [65]	76.52	75.24	75.88
DLRD [56]	85.26	83.01	84.14
LRDL [57]	87.21	83.96	85.59
DPL [17]	85.33	84.12	84.73
K-SVD [12]	82.19	81.02	81.61
LC-KSVD [13]	83.56	82.23	82.90
D-KSVD [14]	83.79	82.88	83.34
FDDL [16]	84.12	82.15	83.14
ConvNet-RBM [35]	85.92	83.57	84.74
MDFR [36]	86.78	84.01	85.39
PLR [62]	86.25	82.31	84.28
RRNN [37]	88.15	84.55	86.35
OPR [54]	87.33	84.21	85.77
Razzaghi [27]	88.82	85.93	87.37
TTRLSR [22]	85.46	82.11	83.78
DKTL [28]	87.45	84.23	85.84
DSTL [29]	86.29	83.01	84.65
Xu [30]	87.24	85.62	86.43
CD-ALPHN [58]	86.55	84.28	85.41
TSSA [59]	86.04	84.09	85.06
Appice [60]	87.15	85.42	86.28
SPDA [61]	88.12	85.90	87.01
DLSL	**90.33**	**87.88**	**89.11**

1) *Sunglasses:* For each person, we choose seven neutral images and one image with sunglasses as the training set. The rest neutral images and the rest images with sunglasses as the test set. That is to say, for each person, eight images are used as the training set and 12 images are used as the test set.
2) *Scarf:* The setting of experimental conditions in this part is the same as the above Sunglasses experiment.

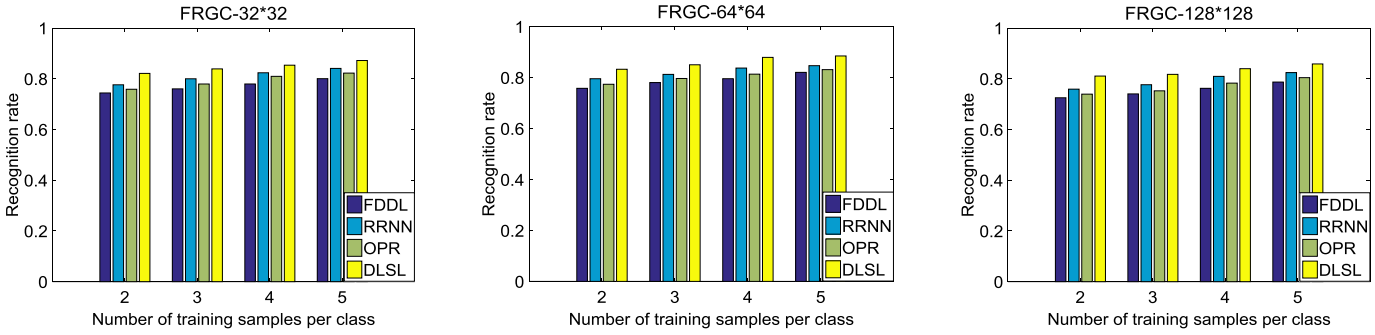
Table 6 depicts the recognition rates of different algorithms in two different scenarios. We can observe that DLSL achieves the best results in the sunglasses case and scarf case. The average recognition rate of DLSL is 89.11%, with about 8%, 6%, 5%, 5%, 5%, 5%, 4%, 4%, 4%, 4%, 4%, 3% and 3% improvements over K-SVD, FDDL, DPL, ConvNet-RBM, PLR, DSTL, DKTL, MDFR, OPR, TSSA, CD-ALPHN, RRNN and Xu's method respectively.

5.6. Comparison of running time of various dictionary learning algorithms

To illustrate that DLSL is a fast algorithm compared with other dictionary learning algorithms. We do the contrast experiment. The setting of the contrast experiment is exactly the same as the experiments in Section 5.1 to Section 5.5.

Table 7
The average running time of five dictionary learning algorithms on different databases.

Database/Algorithm	FDDL	D-KSVD	LC-KSVD	K-SVD	DLSL
FRGC (noise)	170023.21 s	97136.26 s	69433.28 s	4042.32 s	963.05 s
LFW (noise)	140885.87 s	82389.29 s	56430.86 s	3261.93 s	802.42 s
CVL	9418.36 s	4820.92 s	3520.82 s	213.92 s	49.75 s
Yale B	127023.77 s	75600.81 s	55588.83 s	3158.37 s	773.28 s
LFW	11846.93 s	6158.55 s	4787.25 s	304.26 s	72.91 s
AR	31416.29 s	15646.21 s	13003.47 s	763.27 s	173.47 s
Average	81769.07 s	46958.67 s	33794.08 s	1957.34 s	472.48 s



(a) size 32×32 (b) size 64×64 (c) size 128×128

Fig. 9. The recognition rates of different algorithms on FRGC database.

Here, K-SVD [12],[1] LC-KSVD [13],[2] D-KSVD [14][1] and FDDL [16][2] are used as the comparison methods. We test all these methods on each database and the average running time is shown in Table 7.

We can see that the running time of our DLSL is about 1/4 of that of K-SVD, 1/72 of that of LC-KSVD, 1/99 of that of D-KSVD, 1/173 of that of FDDL. Hence, DLSL is a relatively fast dictionary learning algorithm.

5.7. Influence of the number of training samples on recognition performance

In this section, we use different sizes of samples to illustrate the influence of the number of training samples on the recognition rate. The size of each sample is set to 32×32, 64×64 and 128×128 respectively. Here, FDDL [16], OPR [54] and RRNN [37] are used as the comparison methods. For FRGC database, two to five images are respectively selected from each subject as the training set, the rest images are used as the test set. For CVL and LFW (LFWc) databases, three to six images are respectively selected from each subject as the training set, and the rest images are used as the test set. For AR database, a subset is used here. In the subset, each subject contains 14 neutral images and six images with sunglasses. Here, for each subject we respectively select one image with sunglasses and four to seven neutral images as the training set, the rest images are used as the test set.

Fig. 9 shows the recognition rates of different algorithms on the FRGC database. Fig. 10 shows the average recognition rates of the four algorithms on different databases with different sizes of samples.

As can be seen from Fig. 9 that the recognition rate of DLSL will be higher with more training samples. When the numbers of training samples per class are two, three, four and five, the recognition rates of DLSL are higher than those of FDDL, RRNN and OPR. In Fig. 10(a), it can be seen that the average recognition rates of our DLSL are the highest in different sizes of image. For each algorithm, the highest average recognition rate is obtained when the size of the image is 64×64, followed by the size 32×32 and 128×128.

Fig. 11 shows the recognition rates of different algorithms on the CVL database. Fig. 10(b) shows the average recognition rates of the four methods on CVL database with different sizes of samples.

It can be seen from Fig. 11 that the recognition rate of DLSL increases when the number of training samples per class increases. When the numbers of training samples per class are three, four, five and six respectively, the recognition rates of DLSL are higher than those of FDDL, RRNN and OPR. In Fig. 10(b), we can see that the average recognition rates of DLSL are the highest in different sizes of image. For each algorithm, the highest average recognition rate is obtained when the size of the image is 64×64, followed by the size 32×32 and 128×128.

Fig. 12 shows the recognition rates of different algorithms on the LFW (LFWc) database. Fig. 10(c) shows the average recognition rates of the four algorithms on LFW (LFWc) database with different sizes of the sample.

As can be seen from Fig. 12 that the recognition rate of DLSL will be higher with more training samples. When the numbers of training samples per class are three, four, five and six respectively, the recognition rates of DLSL are higher than those of FDDL, RRNN and OPR. Fig. 10(c) illustrates that the average recognition rates of DLSL are the highest in different sizes of image. For each algorithm, the highest average recognition rate is obtained when the size of the image is 64×64, followed by the size 32×32 and 128×128.

Fig. 13 shows the recognition rates of different algorithms on the AR database. Fig. 10(d) shows the average recognition rates of the four algorithms on the AR database with different sizes of the samples.

It can be seen from Fig. 13 that the recognition rate of DLSL will be higher with more training samples. When the numbers of training samples per class are five, six, seven and eight respectively, the recognition rates of DLSL are higher than those of FDDL, RRNN and OPR. In Fig. 10(d), we can see that the average recognition rates of DLSL are the highest in different sizes of image. For each algorithm,

[1] The MATLAB code for K-SVD and D-KSVD is directly taken from: http://www.yongxu.org/lunwen.html.
[2] The MATLAB code for LC-KSVD and FDDL is directly taken from: http://www.personal.psu.edu/thv102/disdictlearn.html.

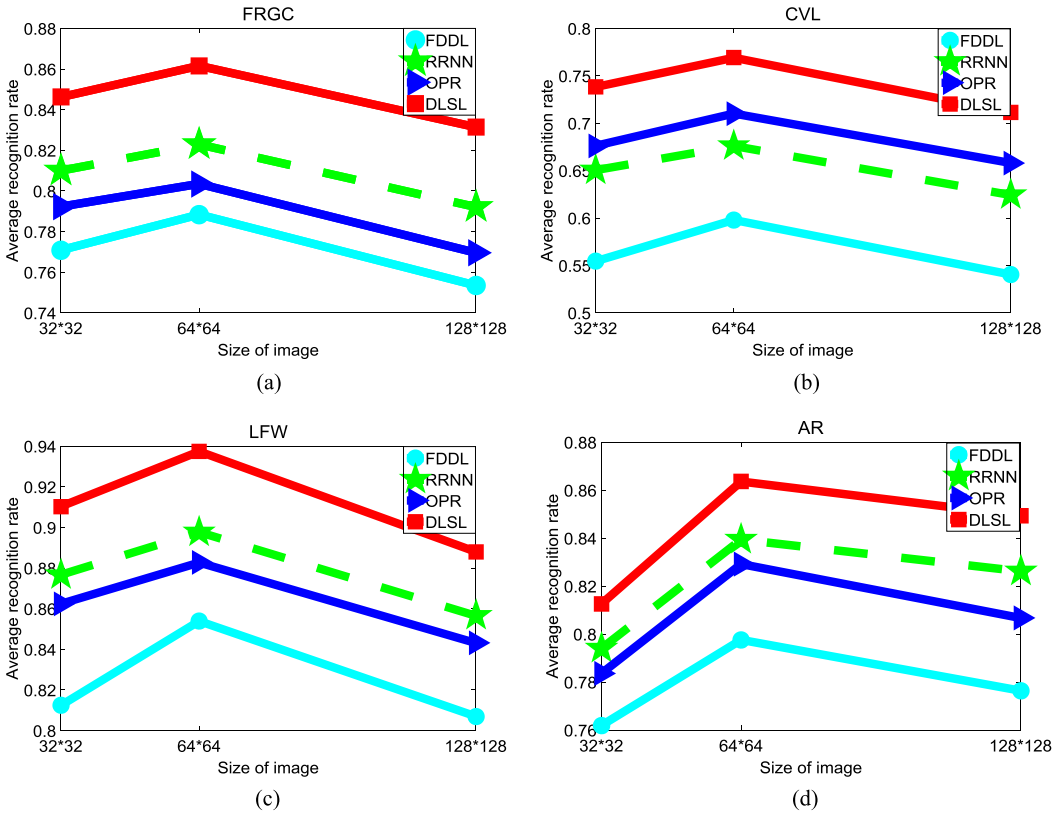


Fig. 10. The average recognition rates of different algorithms on different databases.

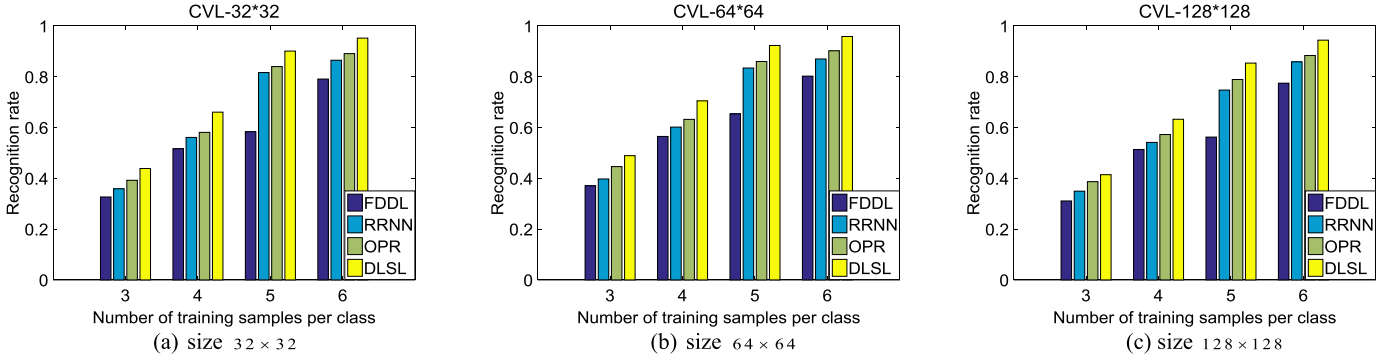


Fig. 11. The recognition rates of different algorithms on the CVL database.

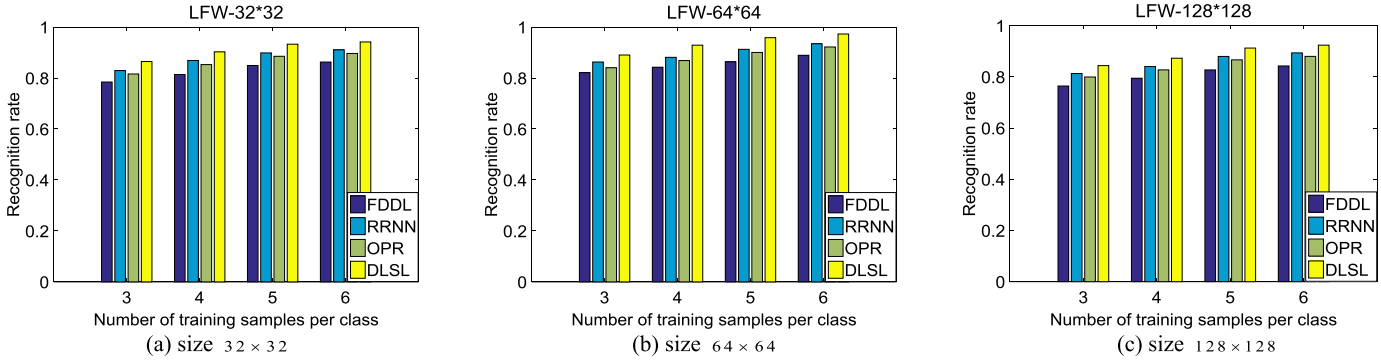


Fig. 12. The recognition rates of different algorithms on the LFW database.

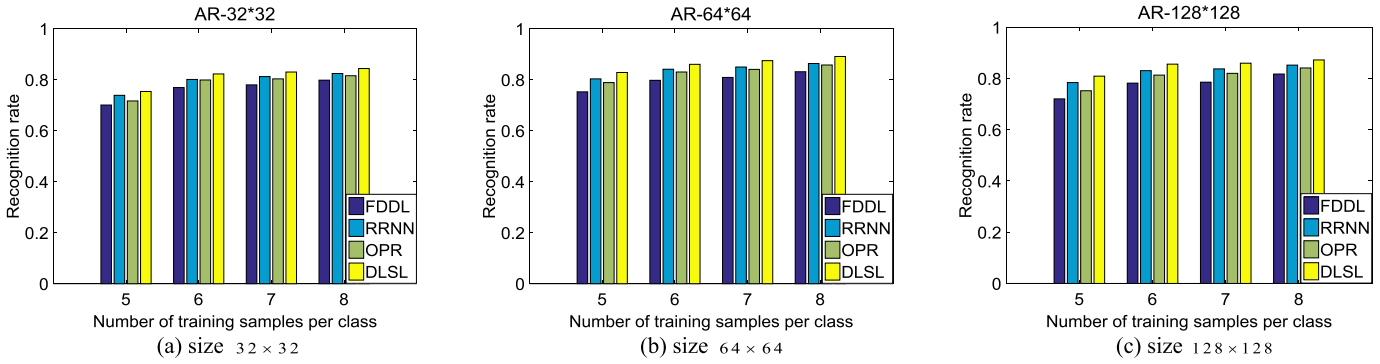


Fig. 13. The recognition rates of different algorithms on AR database.

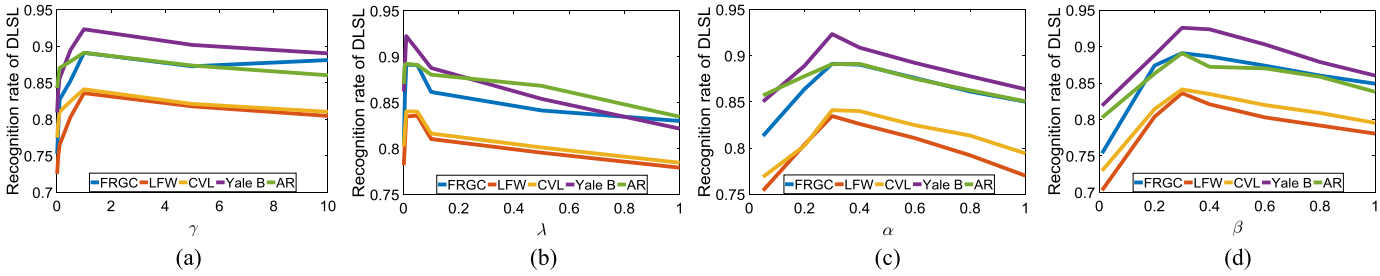


Fig. 14. The influence of important hyper-parameters on DLSL.

the highest average recognition rate is obtained when the size of the image is 64×64, followed by the size 128×128 and 32×32.

From the above experiments, we can see that when the number of training samples increases, the recognition rates will increase generally, and the recognition rates of our DLSL are higher than those of other algorithms.

5.8. Parameter sensitivity

There are several important hyper-parameters in our DLSL, including γ, λ, α, β. We use FRGC, LFW, CVL, Yale B and AR databases to explore the influence of these parameters on the recognition rate of DLSL. The setting of experimental conditions in this part is exactly the same as that in Section 5.1, 5.2, 5.3 and 5.5. For the databases with multiple experimental conditions, we take the average of the experimental results as the final experimental result. Fig. 14 shows the influence of γ, λ, α, β on DLSL. As it can be observed in Fig. 14(a), when γ is in the interval 0 to 1, the recognition rate of DLSL increases with the increase of γ. When γ is equal to 1, the recognition rate of DLSL is almost the highest. From Fig. 14(b) we can see that when the value of λ is very small, the recognition rate of DLSL achieves the best result. In Fig. 14(c) and (d), it can be seen that the recognition rate of DLSL increases at first and then decreases with the increase of α and β. When α and β are equal to 0.3, DLSL achieves the best recognition performance.

6. Conclusion

In this paper, we propose a face recognition algorithm based on dictionary learning and subspace learning (DLSL). The main idea of DLSL is as follows. By learning the common subspace, the distribution differences between different domains are reduced. Thus, DLSL can effectively deal with face recognition problems involving noise, pose variation and occlusion, etc. Experimental results on five databases illustrate that DLSL outperforms the traditional dictionary learning methods and is competitive with some current state-of-the-art methods.

Conflict of interest statement

We declare that there are no known conflicts of interests associated with this publication.

Acknowledgment

This work is supported in part by National Natural Science Foundation of China under grants 61771145, 61371148.

References

[1] J. Yang, L. Luo, J. Qian, Y. Tai, F. Zhang, Y. Xu, Nuclear norm based matrix regression with applications to face recognition with occlusion and illumination changes, IEEE Trans. Pattern Anal. Mach. Intell. 39 (1) (2017) 156–171.
[2] J. Yang, A. Frangi, J. Yang, D. Zhang, J. Zhong, KPCA plus LDA: a complete kernel fisher discriminant framework for feature extraction and recognition, IEEE Trans. Pattern Anal. Mach. Intell. 27 (2) (2005) 230–244.
[3] S. Zafeiriou, G. Tzimiropoulos, M. Petrou, T. Stathaki, Regularized kernel discriminant analysis with a robust kernel for face recognition and verification, IEEE Trans. Neural Netw. Learn. Syst. 23 (3) (2012) 526–534.
[4] M. Yang, L. Zhang, S. Shiu, D. Zhang, Robust kernel representation with statistical local features for face recognition, IEEE Trans. Neural Netw. Learn. Syst. 24 (6) (2013) 900–912.
[5] J. Wright, A. Yang, A. Ganesh, S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2) (2009) 210–227.
[6] R. He, W. Zheng, B. Hu, Maximum correntropy criterion for robust face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1561–1576.
[7] X. Li, D. Dai, X. Zhang, C. Ren, Structured sparse error coding for face recognition with occlusion, IEEE Trans. Image Process. 22 (5) (2013) 1889–1900.
[8] M. Yang, L. Zhang, J. Yang, D. Zhang, Regularized robust coding for face recognition, IEEE Trans. Image Process. 22 (5) (2013) 1753–1766.
[9] W. Deng, J. Hu, J. Guo, Extended SRC: undersampled face recognition via intraclass variant dictionary, IEEE Trans. Pattern Anal. Mach. Intell. 34 (9) (2012) 1864–1870.
[10] W. Deng, J. Hu, J. Guo, In defense of sparsity based face recognition, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2013.
[11] K. Huang, D. Dai, C. Ren, Z. Lai, Learning kernel extended dictionary for face recognition, IEEE Trans. Neural Netw. Learn. Syst. 28 (5) (2017) 1082–1094.
[12] M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation, IEEE Trans. Signal Process. 54 (11) (2006) 4311–4322.

[13] Z. Jiang, Z. Lin, L. Davis, Label consistent K-SVD: learning a discriminative dictionary for recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (11) (2013) 2651–2664.
[14] Q. Zhang, B. Li, Discriminative K-SVD for dictionary learning in face recognition, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2010.
[15] M. Yang, L. Zhang, J. Yang, D. Zhang, Metaface learning for sparse representation based face recognition, in: Proc. IEEE International Conference on Image Processing, 2010.
[16] M. Yang, L. Zhang, X. Feng, D. Zhang, Sparse representation based fisher discrimination dictionary learning for image classification, Int. J. Comput. Vis. 109 (3) (2014) 209–232.
[17] S. Gu, L. Zhang, W. Zuo, X. Feng, Projective dictionary pair learning for pattern classification, in: Proc. International Conference on Neural Information Processing System, 2014.
[18] M. Harandi, C. Sanderson, S. Shirazi, B. Lovell, Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching, in: Proc. of the International Conference on Computer Vision and Pattern Recognition, 2011.
[19] Z.W. Huang, R.P. Wang, S.G. Shan, X.L. Chen, Projection metric learning on Grassmann manifold with application to video based face recognition, in: Proc. of the International Conference on Computer Vision and Pattern Recognition, 2015.
[20] M. Shao, D. Kit, Y. Fu, Generalized transfer subspace learning through low-rank constraint, Int. J. Comput. Vis. 109 (1) (2014) 74–93.
[21] M. Kan, J. Wu, S. Shan, X. Chen, Domain adaptation for face recognition: targetize source domain bridged by common subspace, Int. J. Comput. Vis. 109 (1) (2014) 94–109.
[22] W.M. Zheng, Y. Zong, X. Zhou, M. Xin, Cross-domain color facial expression recognition using transductive transfer subspace learning, IEEE Trans. Affect. Comput. 9 (1) (2018) 21–37.
[23] Q. Wu, X. Zhou, Y. Yan, H. Wu, H. Min, Online transfer learning by leveraging multiple source domains, Knowl. Inf. Syst. 52 (3) (2017) 687–707.
[24] F. Zhu, L. Shao, Weakly-supervised cross-domain dictionary learning for visual recognition, Int. J. Comput. Vis. 109 (1–2) (2014) 42–59.
[25] Y.W. He, Y.J. Tian, D.L. Liu, Multi-view transfer learning with privileged learning framework, Neurocomputing 335 (2019) 131–142.
[26] A. Arnold, R. Nallapati, W.W. Cohen, A comparative study of methods for transductive transfer learning, in: Proc. 7th IEEE Int. Conf. on Data Mining-Workshops, 2007.
[27] P. Razzaghi, P. Razzaghi, K. Abbasi, Transfer subspace learning via low-rank and discriminative reconstruction matrix, Knowl.-Based Syst. 163 (2019) 174–185.
[28] L. Zhang, J. Yang, D. Zhang, Domain class consistency based transfer learning for image classification across domains, Inf. Sci. 418 (2017) 242–257.
[29] J.Z. Lin, C. He, Z.J. Wang, S.Y. Li, Structure preserving transfer learning for unsupervised hyperspectral image classification, IEEE Geosci. Remote Sens. Lett. 14 (10) (2017) 1656–1660.
[30] Y. Xu, X.Z. Fang, J. Wu, X.L. Li, D. Zhang, Discriminative transfer subspace learning via low-rank and sparse representation, IEEE Trans. Image Process. 25 (2) (2016) 850–863.
[31] M. Yang, L. Zhang, J. Yang, D. Zhang, Robust sparse coding for face recognition, in: Proc. of IEEE Conf. Comput. Vis. Pattern Recog., 2011.
[32] E.J. Candès, X. Li, Y. Ma, J. Wright, Robust principal component analysis?, J. ACM 58 (3) (2011) 11.
[33] M. Yin, S. Xie, Z. Wu, Y. Zhang, J. Gao, Subspace clustering via learning an adaptive low-rank graph, IEEE Trans. Image Process. 27 (8) (2018) 3716–3728.
[34] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Ma, Y. Xu, Robust recovery of subspace structures by low-rank representation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 1–14.
[35] Y. Sun, X. Wang, X. Tang, Hybrid deep learning for face verification, IEEE Trans. Pattern Anal. Mach. Intell. 38 (10) (2016) 1997–2009.
[36] A. Al-Waisy, R. Qahwaji, S. Ipson, S. Al-Fahdawi, A multimodal deep learning framework using local feature representations for face recognition, Mach. Vis. Appl. 29 (1) (2018) 35–54.
[37] Y. Li, W. Zheng, Z. Cui, T. Zhang, Face recognition based on recurrent regression neural network, Neurocomputing 297 (2018) 50–58.
[38] S. Pan, I. Tsang, J. Kwok, Q. Yang, Domain adaptation via transfer component analysis, IEEE Trans. Neural Netw. 22 (2) (2011) 199–210.
[39] Y. Shi, F. Sha, Information-theoretical learning of discriminative clusters for unsupervised domain adaptation, in: Proc. ICML, 2012.
[40] I.H. Jhuo, D. Liu, D.T. Lee, S.F. Chang, Robust visual domain adaptation with low-rank reconstruction, in: Proc. IEEE CVPR, 2012.
[41] M. Shao, C. Castillo, Z. Gu, Y. Fu, Low-rank transfer subspace learning, in: Proc. IEEE ICDM, 2012.
[42] Z. Ma, Y. Yang, N. Sebe, A. Hauptmann, Knowledge adaptation with partially shared features for event detection using few exemplars, IEEE Trans. Pattern Anal. Mach. Intell. 36 (9) (2014) 1789–1802.
[43] C. Ding, D. Zhou, X. He, H. Zha, R_1-PCA: rotational invariant L_1-norm principal component analysis for robust subspace factorization, in: Proc. 23rd Int. Conf. Mach. Learn., 2006.
[44] S. Xiang, F. Nie, G. Meng, C. Pan, C. Zhang, Discriminative least squares regression for multiclass classification and feature selection, IEEE Trans. Neural Netw. Learn. Syst. 23 (11) (2012) 1738–1754.
[45] J. Wright, A. Ganesh, S. Rao, Y. Peng, Y. Ma, Robust principal component analysis: exact recovery of corrupted low-rank matrices via convex optimization, in: Proc. Adv. NIPS, 2009.
[46] L. Zhuang, S. Gao, J. Tang, J. Wang, Z. Lin, Y. Ma, N. Yu, Constructing a nonnegative low-rank and sparse graph with data-adaptive features, IEEE Trans. Image Process. 24 (11) (2015) 3717–3728.
[47] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low representation, in: Proc. ICML, 2010.
[48] Z. Lin, M. Chen, Y. Ma, The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices, Eprint Arxiv, 2010.
[49] Z. Jiang, Z. Lin, L.S. Davis, Learning a discriminative dictionary for sparse coding via label consistent K-SVD, in: Proc. IEEE Conf. CVPR, 2011.
[50] J. Ye, Z. Zhao, M. Wu, Discriminative K-means for clustering, in: Proc. Adv. NIPS, 2007.
[51] Y. Zhang, Z. Jiang, L. Davis, Learning structured low-rank representations for image classification, in: Proc. IEEE Conf. CVPR, 2013.
[52] J. Li, J. Wu, H. Deng, J. Liu, A self-learning image super-resolution method via sparse representation and non-local similarity, Neurocomputing 184 (2016) 196–206.
[53] R. Bartels, G. Stewart, Solution of the matrix formula $AX + XB = C$ [F4], Commun. ACM 15 (9) (1972) 820–826.
[54] Y. Tai, J. Yang, Y. Zhang, L. Luo, J. Qian, Y. Chen, Face recognition with pose variations and misalignment via orthogonal procrustes regression, IEEE Trans. Image Process. 25 (6) (2016) 2673–2683.
[55] Q. Feng, C. Yuan, J. Pan, J. Yang, Y. Chou, Y. Zhou, W. Li, Superimposed sparse parameter classifiers for face recognition, IEEE Trans. Cybern. 47 (2) (2017) 378–390.
[56] L. Ma, C. Wang, B. Xiao, W. Zhou, Sparse representation for face recognition based on discriminative low-rank dictionary learning, in: Proc. 25th IEEE Conf. Comput. Vis. Pattern Recognit., 2012.
[57] P. Zhou, C. Fang, Z. Lin, C. Zhang, E. Chang, Dictionary learning with structured noise, Neurocomputing 273 (2018) 414–423.
[58] R.M. Marcacini, R.G. Rossi, I.P. Matsuno, Cross-domain aspect extraction for sentiment analysis: a transductive learning approach, Decis. Support Syst. 114 (2018) 70–80.
[59] H. Sun, S. Liu, S.L. Zhou, H.X. Zou, Transfer sparse subspace analysis for unsupervised cross-view scene model adaptation, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 9 (7) (2016) 2901–2909.
[60] A. Appice, P. Guccione, D. Malerba, Transductive hyperspectral image classification: toward integrating spectral and relational features via an iterative ensemble system, Mach. Learn. 103 (2016) 343–375.
[61] T. Xiao, P. Liu, W. Zhao, H.W. Liu, X.L. Tang, Structure preservation and distribution alignment in discriminative transfer subspace learning, Neurocomputing 337 (2019) 218–234.
[62] J. Li, Y. Kong, H. Zhao, J. Yang, Y. Fu, Learning fast low-rank projection for image classification, IEEE Trans. Image Process. 25 (10) (2016) 4803–4814.
[63] X. He, D. Cai, S. Yan, H. Zhang, Neighborhood preserving embedding, in: Proc. 10th IEEE Int. Conf. Comput. Vis., 2005.
[64] D. Cai, X. He, K. Zhou, J. Han, H. Bao, Locality sensitive discriminant analysis, in: Proc. 20th Int. Joint Conf. Artif. Intell., 2007.
[65] G. Liu, S. Yan, Latent low-rank representation for subspace segmentation and feature extraction, in: Proc. 13th IEEE Int. Conf. Comput. Vis., 2011.

Mengmeng Liao received the M.S. degree in circuits and systems from Xidian University, Xi'an, China, in 2016. He is a doctoral candidate with Fudan University, Shanghai, China. His research interests include face recognition, sparse representation computer vision, pattern recognition. He has published several papers of face recognition. He was awarded the third prize of the National College Student Mathematics Competition of China in 2011.

Xiaodong Gu received the M.S. degree in communication and information system from Soochow (Suzhou) University, China, in 2000 and PhD degree in signal and information processing from Peking (Beijing) University, Beijing, China, in 2003. From 2003 to 2005, he had been a Postdoctoral Fellow with Electronic Science and Technology Postdoctoral Research Station, and with the Department of Electronic Engineering, Fudan University, Shanghai, China. Currently he is a Full Professor with the Department of Electronic Engineering, Fudan University. He had been a Visiting Fellow with Princeton University,

USA from 2010 to 2011. He had been a Guest Professor with Kyushu Institute of Technology, Japan in 2009. He is or has been the principal investigators of ten projects supported by National Natural Science Foundation of China, Shanghai National Natural Science Foundation, Postdoctoral Science Foundation of China, etc. He also has been the vice principal investigators of more than ten research projects. He has published more than one hundred academic papers in journals and conference proceedings, one monograph with Science Press, Beijing, and four book chapters invited by Nova Science Publishers, New York. His current research interests include pulse coupled neural networks, convolutional neural networks, artificial neural networks, image processing, deep learning, brain-inspired models, and pattern recognition. He has been serving as an Editorial Board Member of *Neural Networks* since 2014. He was the recipient of 2005 Excellent Dissertation Award of Peking University and the recipient of 2003 Excellent Graduate of Peking University. He was the recipient of 2007 Excellent Postdoctoral Fellow of Fudan University (10/553). He was awarded 2013 Shanghai Excellent Invention Trials Silver Award (Third place). He was awarded the third prize of 2016 Shanghai Natural Science Award (First place).