Neural Networks 155 (2022) 498–511

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Support vector machine embedding discriminative dictionary pair learning for pattern classification

Jing Dong [a,*], Liu Yang [a], Chang Liu [b], Wei Cheng [c], Wenwu Wang [d]

[a] College of Electrical Engineering and Control Science, Nanjing Tech University, Nanjing, Jiangsu, China
[b] College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin, Heilongjiang, China
[c] Beijing Institute of Applied Meteorology, Beijing, China
[d] Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford GU2 7XH, UK

ARTICLE INFO

Article history:
Received 2 January 2022
Received in revised form 23 August 2022
Accepted 30 August 2022
Available online 7 September 2022

Keywords:
Discriminative dictionary learning
Pattern classification
Support vector machine
Sparse representation

ABSTRACT

Discriminative dictionary learning (DDL) aims to address pattern classification problems via learning dictionaries from training samples. Dictionary pair learning (DPL) based DDL has shown superiority as compared with most existing algorithms which only learn synthesis dictionaries or analysis dictionaries. However, in the original DPL algorithm, the discrimination capability is only promoted via the reconstruction error and the structures of the learned dictionaries, while the discrimination of coding coefficients is not considered in the process of dictionary learning. To address this issue, we propose a new DDL algorithm by introducing an additional discriminative term associated with coding coefficients. Specifically, a support vector machine (SVM) based term is employed to enhance the discrimination of coding coefficients. In this model, a structured dictionary pair and SVM classifiers are jointly learned, and an optimization method is developed to address the formulated optimization problem. A classification scheme based on both the reconstruction error and SVMs is also proposed. Simulation results on several widely used databases demonstrate that the proposed method can achieve competitive performance as compared with some state-of-the-art DDL algorithms.

1. Introduction

In recent years, sparse representation and dictionary learning have been widely used in a variety of problems, including signal processing (Dong et al., 2017; Rencker, Bach, Wang, & Plumbley, 2019; Xu, Xu, & Quan, 2021), multi-view clustering (Zhao, Li, Chen, Zheng, & Xie, 2022), key frame extraction (Li, Li, Tan, Ding, & Xie, 2022), and pattern recognition (Jiang, Lin, & Davis, 2013; Tang, Panahi, Krim, & Dai, 2019). Sparse representation refers to representing signals as sparse coefficients using dictionaries with over-complete bases. As dictionaries learned from signals of interest tend to represent the signals better than pre-defined dictionaries, dictionary learning has attracted much attention.

According to the different ways of coding signals, dictionaries used in sparse representation could be divided into analysis dictionary (Li et al., 2021; Rubinstein, Peleg, & Elad, 2013) and synthesis dictionary (Aharon, Elad, & Bruckstein, 2006; Dai, Xu, & Wang, 2012; Zhao et al., 2022). Therefore, dictionary learning can be further divided into analysis dictionary learning, synthesis dictionary learning and analysis–synthesis dictionary pair learning. In synthesis dictionary learning, a linear combination of dictionary atoms is used to reconstruct a signal. As obtaining representation coefficients based on synthesis dictionaries involves addressing NP-hard (Pati, Rezaiifar, & Krishnaprasad, 1993; Tropp & Gilbert, 2007) or large-scale convex optimization problems (Chen, Donoho, & Saunders, 1998), the computational complexity is high. To address this problem, dictionary learning based on the sparse analysis model (Nam, Davies, Elad, & Gribonval, 2013) has been studied. In this model, the representation coefficients can be obtained by multiplying the dictionary and the signal, which greatly reduces the computational complexity of representation (Dong et al., 2016; Ravishankar & Bresler, 2013). However, sparse synthesis model can better model the complex local structures of images in general and it leads to many state-of-the-art results in many image processing tasks. To leverage the advantages of both synthesis dictionaries and analysis dictionaries, an analysis–synthesis dictionary model is proposed (Rubinstein & Elad, 2014). In this model, analysis coding coefficients can be computed efficiently via linear projections and original signals can be reconstructed using the synthesis dictionary and analysis coding coefficients.

Dictionary learning has been widely applied to pattern classification and is referred to as discriminative dictionary learning (DDL) (Cai, Zuo, Lei, Feng, & Ping, 2014; Jiang et al., 2013). In

* Corresponding author.
E-mail addresses: jingdong@njtech.edu.cn (J. Dong), yangliu@njtech.edu.cn (L. Yang), liuchang407@hrbeu.edu.cn (C. Liu), chengw@mail.iap.ac.cn (W. Cheng), w.wang@surrey.ac.uk (W. Wang).

https://doi.org/10.1016/j.neunet.2022.08.031
0893-6080/

the past few years, deep learning has drawn significant attention and achieved promising performance in classification tasks. Among these deep-learning-based methods, convolutional neural network (CNN) based models (He, Zhang, Ren, & Sun, 2016; Krizhevsky, Sutskever, & Hinton, 2012; Szegedy et al., 2015) have become the most popular ones for classification. However, most deep learning methods including CNN-based methods usually need training sets of large size to achieve satisfactory results. Although transfer learning can be applied to improve the performance by pre-training the models on a generic large-scale dataset and then fine-tuning them to the small-sized target dataset, this would not help much when the content of the base and target datasets are very different. In addition, when enough training samples are not available, complex models such as CNNs tend to overfit the training set and may not provide good results for the test set. As compared with deep learning, DDL algorithms can achieve promising results on small datasets.

In DDL algorithms, label information of training data is introduced into dictionary learning models or classification schemes to enhance the capability of the model for class discrimination. One popular strategy used in DDL algorithms is to learn a common dictionary shared by all classes while enforcing the coding coefficients corresponding to different classes to be discriminative (Cai et al., 2014; Jiang et al., 2013; Mairal, Ponce, Sapiro, Zisserman, & Bach, 2009). For example, Mairal et al. (2009) proposed to jointly learn a single dictionary adapted to all classes and a linear classifier. The support vector machine guided dictionary learning algorithm (SVGDL) proposed in Cai et al. (2014) learns a single dictionary shared between all classes and promotes the discrimination capability of coding coefficients using the weighted sum of the squared distances of the coefficients. Label consistent K-SVD (LC-KSVD) algorithm (Jiang et al., 2013) learns a single overcomplete dictionary and a linear classifier jointly based on a label consistency constraint where samples from the same class have similar coding coefficients. Zhao et al. (2022) proposed to learn a partially shared dictionary using a differentiable scale-invariant function as the sparsity regularizer to address multi-view clustering tasks.

Another popular strategy in DDL methods attempts to learn class-specific sub-dictionaries for different classes (Ramirez, Sprechmann, & Sapiro, 2010; Yang, Zhang, Feng, & Zhang, 2011). The algorithm proposed by Ramirez et al. (2010) introduces an incoherence promoting term to encourage dictionaries associated with different classes to be independent. Yang et al. (2011) learned a structured dictionary consisting of sub-dictionaries corresponding to different classes, and imposed the Fisher criterion on coding coefficients to minimize within-class scatter and enlarge between-class scatter. Class-aware analysis dictionary learning (CADL) (Wang, Guo, Guo, Luo, & Kong, 2017) learns class-specific analysis dictionaries and imposes a max-margin regularization term on coding coefficients to further improve the discrimination ability. It achieves better performance than previous state-of-art methods by making full use of the discrimination ability of class-specific dictionaries and representation coefficients.

However, the DDL algorithms mentioned above are all based on the sparse synthesis model which results in high computational complexity in the coding stage. To address this issue and also explore the discrimination of the sparse analysis model, some DDL algorithms based on analysis dictionary learning have been proposed (Guo, Guo, Kong, Zhang, & He, 2016; Shekhar, Patel, & Chellappa, 2014; Tang et al., 2019). These algorithms learn analysis dictionaries from training data and adopt the discrimination of the corresponding analysis coefficients. Shekhar et al. (2014) learned a full rank analysis dictionary from the training data and trained an additional SVM classifier on the coefficients over the learned dictionary. Guo et al. (2016) introduced a code consistent term and a local topology preserving term and proposed a correntropy induced formulation to improve discrimination with analysis coefficients. Tang et al. (2019) enhanced the discrimination of coefficients by applying a structure constraint to the analysis dictionary.

To integrate the advantages of both synthesis and analysis dictionary learning, DDL algorithms based on synthesis–analysis dictionary pair learning have been also developed (Gu, Zhang, Zuo, & Feng, 2014). The projective dictionary pair learning (DPL) algorithm (Gu et al., 2014) learns a structured synthesis dictionary together and a structured analysis dictionary jointly to achieve the goal of signal representation and better discrimination capability. This framework can also increase the efficiency of the sparse coding step in conventional DDL models. Based on DPL, a semi-coupled DPL method (Chen et al., 2019) is specifically designed to address the classification problem of polarimetric synthetic aperture radar images. DPL has also been adapted to classify electroencephalogram signals in Ameri, Pouyan, and Abolghasemi (2016).

Although DPL is fast and effective for pattern classification tasks, it only considers the structures of the learned dictionaries and does not exploit the discrimination capability of coding coefficients in the process of dictionary learning. To address this issue, a discriminative analysis–synthesis dictionary learning method is proposed in this paper. It emphasizes the discrimination ability of both dictionary atoms and coding coefficients and learns a pair of structured dictionaries by promoting the discrimination of coding coefficients. In particular, support vector machine (SVM) classifiers associated with coding coefficients are embedded in the model of dictionary pair learning so that the discriminative capability of the coefficients can be further improved. The resulting optimization problem is addressed with multi-variable optimization, and an alternating optimization method is developed to solve the problem. In addition, a new classification scheme which merges the residual error based on the learned dictionary pair and the discrimination of the trained classifiers is proposed. The proposed algorithm is referred to as SVM embedding discriminative dictionary pair learning (SVM-DDPL). To verify the effectiveness of the proposed method, extensive experiments on several widely used databases for pattern classification were performed. Experimental results show that the proposed algorithm can improve the accuracy of classification as compared with previous algorithms. As compared with deep learning approaches, the proposed method does not rely on large-scale training sets and can obtain promising results for datasets of small sizes without pre-training. By learning dictionary pairs and SVM classifiers simultaneously, the proposed algorithm can achieve a balance between the data representation and discrimination ability. The main contributions of this work are summarized as follows.

- A discriminative analysis–synthesis dictionary pair learning model is proposed to exploit the discrimination ability of both dictionary pair and coding coefficients.
- An alternating optimization method is developed to address the optimization problem formulated from this model.
- A classification scheme that considers both reconstruction error over the learned dictionary pair and SVM responses is proposed.
- Experiments on several widely used databases are performed to compare the proposed algorithm with several state-of-the-art dictionary-learning-based algorithms and well-known deep learning methods.

The remainder of this paper is organized as follows. Section 2 introduces the projective dictionary pair learning algorithm. Section 3 presents the proposed algorithm including the proposed formulation, the corresponding optimization framework, and the classification scheme. Experimental results are given in Section 4, and Section 5 concludes the paper.

2. Related work

2.1. Classical discriminative dictionary learning

By introducing label information of training data, DDL algorithms can promote the discriminative capability of representation coefficients to address classification tasks. Let $\mathbf{X} = [\mathbf{X_1}, \ldots, \mathbf{X}_k, \ldots, \mathbf{X}_K] \in \mathbb{R}^{p\times n}$ denote a set of n training samples from K classes, where $\mathbf{X}_k \in \mathbb{R}^{p\times n_k}$ is the subset of $\mathbf{X}$. $\mathbf{X}_k \in \mathbb{R}^{p\times n_k}$ denotes the training samples from the kth class where the total number of all training samples satisfies $n = \sum_{k=1}^{K} n_k$. The formulations of most classical DDL algorithms can be written as (Tang et al., 2019; Wang et al., 2017)

$$\arg\min_{\mathbf{D},\mathbf{A}} \mathcal{R}(\mathbf{X}, \mathbf{D}, \mathbf{A}) + \lambda_1 \|\mathbf{A}\|_p + \lambda_2 f(\mathbf{A}, \mathbf{Y}), \tag{2.1}$$

where $\mathbf{A}$ represents the coefficient matrix of $\mathbf{X}$ over a learned synthesis dictionary $\mathbf{D}$, and $\mathbf{Y}$ denotes the class label matrix of training samples. $\mathcal{R}(\mathbf{X}, \mathbf{D}, \mathbf{A})$ is the reconstruction residual obtained by approximating $\mathbf{X}$ using the dictionary $\mathbf{D}$ and coefficients $\mathbf{A}$. $\|\mathbf{A}\|_p$ is the ℓ_p norm regularizer to enforce $\mathbf{A}$ to be sparse, and typical choices are ℓ_1-norm or ℓ_0-norm. $f(\mathbf{A}, \mathbf{Y})$ denotes the term for promoting the discrimination ability of the model based on $\mathbf{A}$ and labels $\mathbf{Y}$. A typical DDL algorithm following this formulation is SVGDL (Cai et al., 2014), where the discrimination promoting term is based on the weighted sum of the squared distances between all pairs of coding vectors. However, learning a shared dictionary for all classes ignores the intrinsic variability of dictionaries between different classes.

Apart from the discrimination-promoting term, discrimination can also be enhanced by a using structured dictionary, and this technique is called the class-specific dictionary learning method. In class-specific dictionary learning, the learned dictionary $\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_K]$ consists of several sub-dictionaries, and the sub-dictionary $\mathbf{D}_k$ corresponds to class k. Then the reconstruction residual $\mathcal{R}(\mathbf{X}, \mathbf{D}, \mathbf{A})$ can be seen as the sum of the reconstruction error over the sub-dictionaries. This class-specific setting is effective for improving classification accuracy. For example, FDDL (Yang et al., 2011) learns a structured dictionary whose dictionary atoms correspond to different classes instead of learning a shared dictionary for all classes. To enhance the discrimination ability of coding coefficients achieved by class-specific dictionaries, FDDL imposes the Fisher discrimination criterion on the coding coefficients to obtain small within-class scatter and big between-class scatter.

2.2. Projective dictionary pair learning

The DPL algorithm (Gu et al., 2014) aims to learn a structured synthesis dictionary $\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_K]$ and a structured analysis dictionary $\mathbf{P} = [\mathbf{P}_1; \mathbf{P}_2; \ldots; \mathbf{P}_K]$ jointly, which can be written as

$$\{\mathbf{D}^*, \mathbf{P}^*\} = \arg\min_{\mathbf{D},\mathbf{P}} \sum_{k=1}^{K} \|\mathbf{X}_k - \mathbf{D}_k\mathbf{P}_k\mathbf{X}_k\|_F^2 + \lambda\|\mathbf{P}_k\overline{\mathbf{X}}_k\|_F^2, \text{ s.t.} \|\mathbf{d}_i\|_2^2 \le 1, \tag{2.2}$$

where $\|\cdot\|_F^2$ denotes the Frobenius norm, $\mathbf{X}_k$ denotes data samples from class k, and $\overline{\mathbf{X}}_k$ is the complementary set of $\mathbf{X}_k$ in the whole training set $\mathbf{X}$, i.e., data samples not from class k. Matrices $\mathbf{D}_k \in \mathbb{R}^{p\times m}$ and $\mathbf{P}_k \in \mathbb{R}^{m\times p}$ are the synthesis and analysis sub-dictionaries corresponding to class k, respectively. $\mathbf{d}_i$ represents the ith atom/column of the synthesis dictionary $\mathbf{D}$. The constraint $\|\mathbf{d}_i\|_2^2 \le 1$ is used to avoid trivial solutions of $\mathbf{D}_k$ and make the model more stable.

In this model, $\mathbf{P}_k$ can project data samples from class k to a new subspace, meanwhile project data samples from other classes to a nearly null space, which is formulated via the term $\|\mathbf{P}_k\overline{\mathbf{X}}_k\|_F^2$. In other words, the whole coefficient matrix $\mathbf{A} = \mathbf{PX}$ is approximately block-diagonal. The reconstruction error term $\sum_{k=1}^{K} \|\mathbf{X}_k - \mathbf{D}_k\mathbf{P}_k\mathbf{X}_k\|_F^2$ aims to find the synthesis sub-dictionary $\mathbf{D}_k$ that reconstructs the data matrix $\mathbf{X}_k$ from its projective coding matrix $\mathbf{P}_k\mathbf{X}_k$.

Based on the DPL model, the learned analysis sub-dictionary $\mathbf{P}_k^*$ tends to produce small coding coefficients for samples of classes other than k, and the synthesis dictionary $\mathbf{D}_k^*$ is trained to reconstruct samples of class k from their projective coefficients $\mathbf{P}_k^*\mathbf{X}_k$. Therefore, the residual $\|\mathbf{X}_k - \mathbf{D}_k^*\mathbf{P}_k^*\mathbf{X}_k\|_F^2$ will be smaller than $\|\mathbf{X}_k - \mathbf{D}_k^*\mathbf{P}_k^*\mathbf{X}_i\|_F^2$ $(i \ne k)$ as $\mathbf{D}_k^*$ is not trained to reconstruct $\mathbf{X}_i$ $(i \ne k)$. If the test sample $\mathbf{x}$ is from class k, the reconstruction residual $\|\mathbf{x} - \mathbf{D}_k^*\mathbf{P}_k^*\mathbf{x}\|_F^2$ tends to be much smaller than $\|\mathbf{x} - \mathbf{D}_i^*\mathbf{P}_i^*\mathbf{x}\|_F^2$ $(i \ne k)$. Based on this, the class label of $\mathbf{x}$ can be identified using the following classifier

$$k^* = \arg\min_{k\in\{1,2,\ldots,K\}} \|\mathbf{x} - \mathbf{D}_k\mathbf{P}_k\mathbf{x}\|_F^2. \tag{2.3}$$

3. Proposed algorithm

In this section, the formulation of the proposed algorithm is presented first, and then the optimization strategy to address the formulated optimization problem is given in detail. The classification scheme is also presented.

3.1. Problem formulation

As mentioned in Section 2.2, the original DPL algorithm learns a structured pair of analysis dictionary and synthesis dictionary so that training signals of each class can be approximately reconstructed by the sub-dictionary pair corresponding to this class. Based on this formulation, test signals are classified into the class whose dictionary pair gives the best approximation. However, the discrimination of coding coefficients is not considered explicitly in both training and testing processes, which limits the performance of classification. To address this issue, additional classifiers for coding coefficients can be learned in the training process and also taken into account in the test process. By introducing a regularizer related to the coding coefficients, we propose the following formulation

$$\begin{aligned}
&\{\mathbf{P}^*, \mathbf{D}^*, \mathbf{A}^*, \mathbf{U}^*, \mathbf{b}^*\} \\
&= \arg\min_{\mathbf{P},\mathbf{D},\mathbf{A},\mathbf{U},\mathbf{b}} \sum_{k=1}^{K} \|\mathbf{X}_k - \mathbf{D}_k\mathbf{A}_k\|_F^2 + \lambda_1\|\mathbf{P}_k\mathbf{X}_k - \mathbf{A}_k\|_F^2 \\
&\quad + \lambda_2\|\mathbf{P}_k\overline{\mathbf{X}}_k\|_F^2 + \lambda_3 f(\mathbf{A}, \mathbf{y}^k, \mathbf{u}_k, b_k) + \alpha\|\mathbf{P}_k\|_F^2 \\
&\qquad \text{s.t. } \|\mathbf{d}_i\|_2^2 \le 1,
\end{aligned} \tag{3.1}$$

where $\mathbf{X}_k \in \mathbb{R}^{p\times n_k}$, $\mathbf{P}_k \in \mathbb{R}^{m\times p}$, $\mathbf{D}_k \in \mathbf{R}^{p\times m}$, and $\mathbf{A}_k \in \mathbf{R}^{m\times n_k}$ denote the training samples, the analysis sub-dictionary, the synthesis sub-dictionary and the representation coefficient matrix of the kth class, respectively. $\mathbf{P} = [\mathbf{P}_1; \mathbf{P}_2; \ldots; \mathbf{P}_K]$ and $\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_K]$ denote the structured analysis dictionary and synthesis dictionary corresponding to all classes. The coding coefficients of all samples are represented as $\mathbf{A} = [\mathbf{A}_1, \ldots, \mathbf{A}_k, \ldots, \mathbf{A}_K]$. Samples not from the kth class are denoted as $\overline{\mathbf{X}}_k = [\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}, \mathbf{X}_{k+1}, \ldots, \mathbf{X}_K]$. Parameters λ_1, λ_2, λ_3, and $\alpha > 0$ are constant scalars to balance the importance of different terms in the objective function.

The regularizer $f(\mathbf{A}, \mathbf{y}^k, \mathbf{u}_k, b_k)$ is based on support vector machine, and attempts to differentiate coding coefficients corresponding to different classes by learning K hyperplanes represented by $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K]$ and $\mathbf{b} = [b_1, b_2, \ldots, b_K]^T$. In particular, the regularizer is defined as

$$f(\mathbf{A}, \mathbf{y}^k, \mathbf{u}_k, b_k) = \frac{1}{2}\|\mathbf{u}_k\|_2^2 + \mathcal{C}\sum_{i=1}^{n} l(\mathbf{a}_i, y_i^k, \mathbf{u}_k, b_k), \tag{3.2}$$

where $l(\mathbf{a}_i, y_i^k, \mathbf{u}_k, b_k)$ is the quadratic hinge loss function (Yang, Yu, Gong, & Huang, 2009), $\mathcal{C}$ is a fixed constant, and $\mathbf{a}_i$ is the ith column of $\mathbf{A}$, i.e., the coding coefficients corresponding to the ith sample. The vector $\mathbf{y}^k = [y_1^k, y_2^k, \ldots, y_n^k]^T \in \mathbb{R}^n$ represents the label vector of the kth class. If the training sample $\mathbf{x}_i$ is from class k, $y_i^k = 1$, otherwise $y_i^k = -1$. As coding coefficients are treated as training samples of SVM explicitly, the discrimination of coding coefficients can be promoted effectively.

In the proposed formulation (3.1), a structured dictionary pair and a set of SVM classifiers corresponding to different classes are jointly learned, which leads to improved discrimination ability of both dictionaries and coding coefficients.

3.2. Optimization strategy

In this section, we will describe the optimization framework to find the optimal solution of (3.1) in detail. The dictionary pair $\mathbf{P}$ and $\mathbf{D}$ are initialized as random matrices, and then the optimization problem can be addressed by updating each variable while keeping the others fixed, i.e., solving the sub-problem with respect to each variable alternatively as follows.

(1) Fix $\mathbf{P}$, $\mathbf{D}$, $\mathbf{U}$ and $\mathbf{b}$, update $\mathbf{A}$:
Ignoring terms unrelated to $\mathbf{A}$, the update of $\mathbf{A}$ can be realized by updating each column $\mathbf{a}_i$ in parallel, which corresponds to solving the optimization problems with respect to $\mathbf{a}_i$, for $i = 1, \ldots, n$,

$$\begin{aligned}\mathbf{a}_i^* = \arg\min_{\mathbf{a}_i} &\|\mathbf{x}_i - \mathbf{D}\mathbf{a}_i\|_2^2 + \lambda_1\|\mathbf{P}\mathbf{x}_i - \mathbf{a}_i\|_2^2 \\ &+ \lambda_3\mathcal{C}\sum_{k=1}^{K} l(\mathbf{a}_i, y_i^k, \mathbf{u}_k, b_k),\end{aligned} \tag{3.3}$$

where $\mathbf{x}_i \in \mathbb{R}^n$ is the ith column of the data matrix $\mathbf{X}$, and the superscript $*$ denotes the estimation of an optimal variable. The definition of the quadratic hinge loss function is (Yang et al., 2009)

$$\begin{aligned}&l(\mathbf{a}_i, y_i^k, \mathbf{u}_k, b_k) \\ &= \begin{cases} \|1 - y_i^k(\mathbf{u}_k^T\mathbf{a}_i + b_k)\|_2^2, & 1 - y_i^k(\mathbf{u}_k^T\mathbf{a}_i + b_k) > 0, \\ 0, & \text{otherwise}.\end{cases}\end{aligned} \tag{3.4}$$

By taking the derivative of (3.3) with respect to $\mathbf{a}_i$ and setting it to zero, the closed-form solution of $\mathbf{a}_i$ can be obtained. Specifically, when $1 - y_i^k(\mathbf{u}_k^T\mathbf{a}_i + b_k) > 0$,

$$\begin{aligned}\mathbf{a}_i = &\left[\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{I} + \lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k\mathbf{u}_k^T\right]^{-1} \\ &* \left[\mathbf{D}^T\mathbf{x}_i + \lambda_1\mathbf{P}\mathbf{x}_i + \lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k(\mathbf{y}_i^k)^T - \lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k b_k\right].\end{aligned} \tag{3.5}$$

Otherwise,

$$\mathbf{a}_i = (\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{I})^{-1}(\lambda_1\mathbf{P}\mathbf{x}_i + \mathbf{D}^T\mathbf{x}_i). \tag{3.6}$$

(2) Fix $\mathbf{A}$, $\mathbf{U}$, $\mathbf{b}$, and $\mathbf{D}$, update $\mathbf{P}$:
The analysis dictionary $\mathbf{P}$ is updated by optimizing sub-dictionaries $\mathbf{P}_k$ with $k = 1, \ldots, K$ in parallel, and the update of $\mathbf{P}_k$ involves solving the sub-problem as follows

$$\mathbf{P}_k^* = \arg\min_{\mathbf{P}_k} \lambda_1\|\mathbf{P}_k\mathbf{X}_k - \mathbf{A}_k\|_F^2 + \lambda_2\|\mathbf{P}_k\overline{\mathbf{X}}_k\|_F^2 + \alpha\|\mathbf{P}_k\|_F^2. \tag{3.7}$$

By setting the derivative with respect to $\mathbf{P}_k$ to zero, the optimal solution for $\mathbf{P}_k$ can be directly obtained, that is

$$\mathbf{P}_k = \lambda_1\mathbf{A}_k\mathbf{X}_k^T(\lambda_1\mathbf{X}_k\mathbf{X}_k^T + \lambda_2\overline{\mathbf{X}}_k\overline{\mathbf{X}}_k^T + \alpha\mathbf{I})^{-1}. \tag{3.8}$$

(3) Fix $\mathbf{A}$, $\mathbf{U}$, $\mathbf{b}$, and $\mathbf{P}$, update $\mathbf{D}$:
The optimization of $\mathbf{D}$ is equivalent to updating synthesis sub-dictionaries $\mathbf{D}_k$ for $k = 1, \ldots, K$. When $\mathbf{A}$, $\mathbf{U}$, $\mathbf{b}$ and $\mathbf{P}$ are fixed, the optimization sub-problem with respect to $\mathbf{D}_k$ can be written as

$$\mathbf{D}_k^* = \arg\min_{\mathbf{D}_k} \|\mathbf{X}_k - \mathbf{D}_k\mathbf{A}_k\|_F^2 \qquad \text{s.t. } \|\mathbf{d}_i\|_2^2 \leq 1. \tag{3.9}$$

This problem can be addressed effectively via the alternating direction method of multipliers (ADMM) (Boyd, Parikh, Chu, Peleato, & Eckstein, 2011; Gu et al., 2014). Specifically, by introducing an auxiliary variable $\mathbf{S}_k$, the problem can be reformulated as

$$\{\mathbf{D}_k^*, \mathbf{S}_k^*\} = \arg\min_{\mathbf{D}_k, \mathbf{S}_k} \|\mathbf{X}_k - \mathbf{D}_k\mathbf{A}_k\|_F^2 \qquad \text{s.t. } \mathbf{D}_k = \mathbf{S}_k,\ \|\mathbf{s}_i\| \leq 1. \tag{3.10}$$

The optimal solution of (3.10) can be obtained by alternatively updating the variables as follows (Gu et al., 2014)

$$\begin{cases}\mathbf{D}_k^{(r+1)} = \arg\min_{\mathbf{D}_k} \|\mathbf{X}_k - \mathbf{D}_k\mathbf{A}_k\|_F^2 + \rho\|\mathbf{D}_k - \mathbf{S}_k^{(r)} + \mathbf{T}_k^{(r)}\|_F^2 \\ \mathbf{S}_k^{(r+1)} = \arg\min_{\mathbf{S}_k} \|\mathbf{D}_k^{(r+1)} - \mathbf{S}_k + \mathbf{T}_k^{(r)}\|_F^2, \text{ s.t. } \|\mathbf{s}_i\|_2^2 \leq 1 \\ \mathbf{T}_k^{(r+1)} = \mathbf{T}_k^{(r)} + \mathbf{D}_k^{(r+1)} - \mathbf{S}_k^{(r+1)},\end{cases} \tag{3.11}$$

where r is the iteration number.

(4) Fix $\mathbf{P}$, $\mathbf{D}$ and $\mathbf{A}$, update $\mathbf{U}$, and $\mathbf{b}$:
When $\mathbf{P}$, $\mathbf{D}$ and $\mathbf{A}$ are fixed, the optimization of $\mathbf{U}$ and $\mathbf{b}$ is a multi-class SVM problem, which can be divided into K linear one-against-all SVM subproblems. The gradient-based optimization method to train linear SVM solvers proposed in Yang et al. (2009) can be used to estimate $\mathbf{u}_k$'s and b_k's.

3.3. Classification scheme

Once the structured analysis dictionary $\mathbf{P}$, structured synthesis dictionary $\mathbf{D}$ and the SVM classifiers defined by $< \mathbf{U}, \mathbf{b} >$ are learned, we can perform the classification task as follows. Assuming $\mathbf{x}$ denotes a test sample, the reconstruction error $\|\mathbf{x} - \mathbf{D}_k\mathbf{P}_k\mathbf{x}\|_2^2$ should be small if $\mathbf{x}$ is from the kth class based on the first two terms of the proposed formulation. In addition, the response to the SVM classifier corresponding to the kth class, i.e., $\mathbf{u}_k\mathbf{z} + b_k$ will be large, where $\mathbf{z}$ represents coefficients of $\mathbf{x}$ over the whole analysis dictionary $\mathbf{P}$, i.e., $\mathbf{z} = \mathbf{P}\mathbf{x}$. By fusing the discrimination of the reconstruction error and the SVM classifiers, the label of the test sample $\mathbf{x}$ can be determined by

$$k^* = \arg\min_{k \in \{1,2,\ldots,K\}} \beta\|\mathbf{x} - \mathbf{D}_k\mathbf{P}_k\mathbf{x}\|_2^2 - (\mathbf{u}_k^T\mathbf{z} + b_k), \tag{3.12}$$

where $\beta > 0$ is a scale parameter to balance the two terms. The fusion of the reconstruction error and SVM response in the classification scheme aims to make effective use of the discrimination of both dictionary pairs and coefficients, which is also consistent with the formulation of the dictionary pair learning stage. Note that, in the proposed formulation (3.1), the signals from each class are supposed to be reconstructed by the dictionary pairs corresponding to that class, and the representation coefficients tend to be discriminative over the hyperplanes defined by the SVM classifiers. By solving the problem above, the test sample $\mathbf{x}$ from the ith ($i \neq k$) class can be separated from the samples of the kth class due to larger SVM classifier response and smaller reconstruction residual.

The whole procedure of the proposed SVM-DDPL algorithm is summarized in Algorithm 1. In particular, Lines 3–8 describe the operations in each iteration of the training stage of SVM-DDPL, where line 3 updates the coding coefficients $\mathbf{A}$ via Eqs. (3.5) or (3.6), and lines 4–8 updates the analysis sub-dictionary $\mathbf{P}_k$, synthesis sub-dictionary $\mathbf{D}_k$ and SVM classifier parameters $\mathbf{u}_k$ and b_k corresponding to each class k with $k = 1, \ldots, K$. Line 11 shows the classification of a given test sample in the test stage.

Algorithm 1 The proposed SVM-DDPL algorithm

Initialization: Training samples $\mathbf{X}$ and regularization parameters λ_1, λ_2, λ_3, α, $\mathcal{C}$, and β. Initialize the iteration counter $t = 1$, analysis dictionary $\mathbf{P}$, and synthesis dictionary $\mathbf{D}$ as random matrices. $\mathbf{U}$ and $\mathbf{b}$ are initialized as the all-zero matrix.

1: // Support vector machine embedding discriminative dictionary pair learning:
2: **while** the maximal iteration number is not reached **do**
3: Update each column of $\mathbf{A}$ via (3.5) or (3.6).
4: **for** $k = 1$ to K **do**
5: Update $\mathbf{P}_k$ via (3.8).
6: Update $\mathbf{D}_k$ via (3.11).
7: Update $\mathbf{u}_k$ and b_k via the SVM solver in Yang et al. (2009).
8: **end for**
9: **end while**
10: // Classification based on the learned dictionary pair and SVM classifiers:
11: Obtain the class label of a given test sample $\mathbf{x}$ via (3.12).

3.4. Computational complexity

In the training stage of the proposed algorithm, $\mathbf{A}$, $\mathbf{P}$, $\mathbf{D}$, and $\{\mathbf{U}, \mathbf{b}\}$ are updated alternatively. The update of $\mathbf{A}$, $\mathbf{P}$ and $\mathbf{D}$ mainly involve matrix multiplications and inversions. In each iteration, the time complexities of updating $\mathbf{A}$, and $\mathbf{P}$ are $O(m^2K^2p + m^3K^3)$ and $O(mn_kp + p^2n_k + p^3 + mp^2)$, respectively. In the update of $\mathbf{P}$, the matrix inverse $(\lambda_1\mathbf{X}_k\mathbf{X}_k^T + \lambda_2\overline{\mathbf{X}}_k\overline{\mathbf{X}}_k^T + \alpha\mathbf{I})^{-1}$ is not changed over iterations and can be pre-computed to accelerate the training process. The time complexity of updating $\mathbf{D}$ is $O(T(m^2n_k + m^3 + mn_kp + pm^2))$, where T denotes the iteration number in ADMM for updating $\mathbf{D}$. We experimentally found that in most cases T is less than 20. The update of $\{\mathbf{U}, \mathbf{b}\}$ has a time complexity of $O(n_kK)$.

In the testing stage, the classification is very efficient. The time complexities of computing the reconstruction error term and SVM-based term are $O(mp)$ and $O(m)$, respectively. As a result, the total complexity of the testing stage to classify samples of K categories is $O(Kmp)$.

4. Simulation results

Experiments are performed to demonstrate the convergence of the proposed algorithm, illustrate its the sensitivity with different parameter settings, and compare its performance with existing DDL algorithms.

Fig. 1. Samples of the AR database.

Fig. 2. Samples of the Extended YaleB database.

4.1. Databases introduction and settings

In the experiments, four widely used pattern classification databases are utilized, i.e., AR (Martinez & Benavente, 1998), Extended YaleB (Georghiades, Belhumeur, & Kriegman, 2001), Caltech101 (Li, Fergus, & Perona, 2004), Scene15 (Lazebnik, Schmid, & Ponce, 2006a), and Isolet (Fanty & Cole, 1990). AR and Extended YaleB are used for face recognition, Caltech101 is used for object categorization and Scene15 is used for scene categorization. Isolet is used for spoken letter recognition. The details of each database, settings of training and test data, feature extraction of databases, and parameter settings of the proposed algorithm are presented as follows.

(1) AR: The AR database (Martinez & Benavente, 1998) involves many variations such as illumination, expressions, sunglass, and scarf occlusion. It contains face images of 100 people consisting of 50 men and 50 women of size 165 $\times$ 120. Each individual and each category contains no fewer than 26 images. Sample images are shown in Fig. 1. In our experiments with AR, 20 images per individual are randomly selected as training data and the remanding 6 images are used as test data.

(2) Extended YaleB: The Extended YaleB database (Georghiades et al., 2001) contains 2,414 front face images involving large variations in illumination and expressions. The images are from 38 individuals and each individual has about 64 images cropped to 168 $\times$ 192 pixels. Sample images from Extended YaleB are shown in Fig. 2. For this database, we randomly select 32 images per person as training data, and the remaining are used as testing data.

(3) Caltech101: The Caltech101 database (Li et al., 2004) contains 9,144 images from 102 object categories including 101 object classes and a background class. The object categories include animals, vehicles, flowers, plants, and so on. The number of images in each category varies from 31 to 800. Moreover, the images from each category have large shape variability in

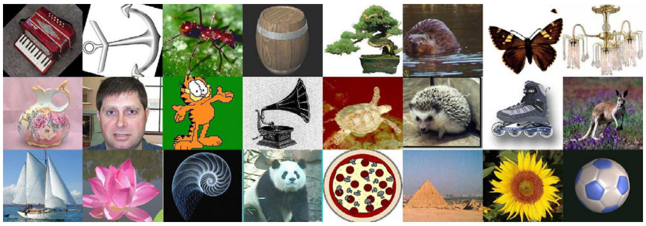


Fig. 3. Samples of objects from the Caltech101 dataset.

Fig. 4. Samples of scenes from the Scene15 database.

object size, pose, and location, which increases the difficulty of classification. Some image samples from Caltech101 are shown in Fig. 3. Following the experimental settings in Gu et al. (2014), we randomly select 30 images per class for training and the remaining for testing.

(4) Scene15: The Scene15 database contains 4,485 images from 15 natural scene categories. Each category contains at least 200 images and the average image size is about 250×300 pixels. This database includes kitchen, suburb, living, forest, coast, industrial, office, highway, tall building, mountain, inside city, bedroom, street, room, open country, and store scene categories. Some examples from this database are illustrated in Fig. 4. Following the experimental settings in Jiang et al. (2013), we randomly select 100 images per category for training and use the remaining for testing.

(5) Isolet: Isolet is widely used for spoken letter recognition (Fanty & Cole, 1990). In this database, 150 speakers spoke the name of each letter of the alphabet twice. Hence, one speaker has 52 training examples. The speakers are grouped into sets of 30 speakers each and are referred to as isolet1, isolet2, isolet3, isolet4, and isolet5. The spoken letter database we use is provided by Fanty and Cole (1990)[1] and the dimension of data is 617.

(6) Feature extraction of databases: Following conventional experimental settings of DDL (Jiang et al., 2013), training and testing of the algorithms are both based on feature vectors extracted from samples. For experiments with the AR and Extended YaleB databases, random face features are used as in Jiang et al. (2013). Specifically, a sample image is projected into a feature vector of fixed dimension by a random matrix, and the resulting feature vector is then normalized. The dimensions of the extracted features for AR and Extended YaleB are 540 and 504, respectively. For Caltech101 and Scene15, features provided by Jiang et al. (2013)[2] are used. In particular, spatial pyramid features (Lazebnik, Schmid, & Ponce, 2006b) are extracted from dense scale-invariant feature transform (SIFT) descriptors of 16×16 patches from each image, and the feature dimension is reduced to 3,000 by principle component analysis. More details can be found in Jiang et al. (2013). For experiments with the Isolet database, the described features contain spectral coefficients, contour features, sonorant features, pre-sonorant features, and post-sonorant features.

(7) Parameter settings of SVM-DPPL: In the model of SVM-DDPL, main parameters include regularization parameters λ_1, λ_2, λ_3, α, $\mathcal{C}$, the number of atoms m in each sub-dictionary in the process of discriminative dictionary pair learning, and the parameter β in the classification scheme. As the parameter selection issue remains an open problem, the parameters of the proposed algorithm are selected in a heuristic way. In particular, α is fixed as $1e-4$, and $\mathcal{C}$ is fixed as 0.2. The number of atoms m in the sub-dictionaries is set based on the number of training samples. For experiments with more training samples, m is set as larger values. Other parameters are tuned using a grid search strategy, and the detailed settings for each database are summarized in Table 1.

4.2. Convergence of SVM-DDPL

The convergence of SVM-DDPL is illustrated in Fig. 5. It can be seen that with the increase of the iteration number, the objective function value decreases quickly on Caltech101 and Scene15 databases. In our experiments, the proposed SVM-DDPL algorithm converges in less than 20 iterations.

4.3. Experiments with different parameters

Taking the Extended YaleB database as an example, different values of the parameters are tested to demonstrate the sensitivity of SVM-DDPL. The results with different parameter settings are presented in Fig. 6.

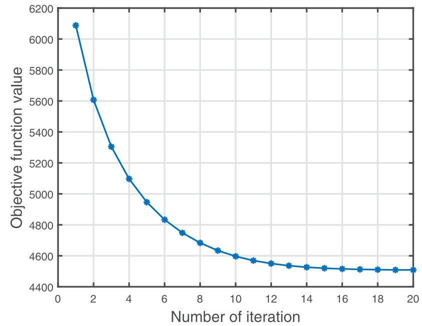
In particular, we test the different settings of λ_1, λ_2 and λ_3 with other parameters fixed as $\alpha = 1e-4, \mathcal{C} = 0.2$ and $\beta = 5$. Fig. 6(a)–(c) show the results with only one parameter of λ_1, λ_2 and λ_3 being fixed. When λ_3 is fixed, as shown in Fig. 6(a), the recognition rate is not sensitive to λ_1 compared to λ_2. The recognition rate decreases seriously when λ_2 is smaller than $1e-4$. Fig. 6(b) shows the results with $\lambda_1 = 2e-2$. It can be seen that λ_2 has a stronger influence on the classification accuracy of SVM-DDPL than λ_3. Fig. 6(c) shows the results with different settings of λ_1 and λ_3 with $\lambda_2 = 2e-3$. In this case, recognition rates are all greater than 90% when λ_1 is smaller than $1e-1$, which shows that the algorithm is not sensitive with settings of λ_1 and λ_3.

The influence of α is presented in Fig. 6(d) with $\lambda_1 = 2e-2$, $\lambda_2 = 2e-3$, $\lambda_3 = 2e-6$, $\beta = 5$ and $\mathcal{C} = 0.2$ fixed. The recognition accuracy decreases substantially when $\alpha > 1e-3$. Fig. 6(e) shows the influence of β with other parameters fixed as $\lambda_1 = 2e-2$, $\lambda_2 = 2e-3$, $\lambda_3 = 2e-6$, $\alpha = 1e-4$, and $\mathcal{C} = 0.2$, and the recognition rate reaches the maximum with $\beta = 5$. Results with different settings of $\mathcal{C}$ are presented in Fig. 6(f), where other parameters are set as $\lambda_1 = 2e-2$, $\lambda_2 = 2e-3$, $\lambda_3 = 2e-6$, $\alpha = 1e-4$ and $\beta = 5$. It can be seen that the results with various $\mathcal{C}$ are all above 97.3%, which shows that the algorithm is not sensitive to the change of parameter $\mathcal{C}$.

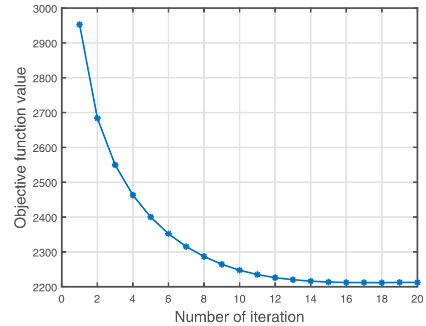
The parameter β is to balance the effect of the terms in the classification scheme and the value of β is tuned to achieve the best performance. For an individual database, the employment of larger values of β indicates that the reconstruction error has a greater impact than the cases with smaller β's. However, it does not mean the reconstruction error term dominates the classification results, as the reconstruction error and SVM responses are obtained based on different principles and the values of these two terms are not comparable. For different databases, the values of β can be different. In the experiments, β is tuned for each database to achieve the best results.

[1] https://archive.ics.uci.edu/ml/datasets/ISOLET

[2] http://www.zhuolin.umiacs.io/projectlcksvd.html

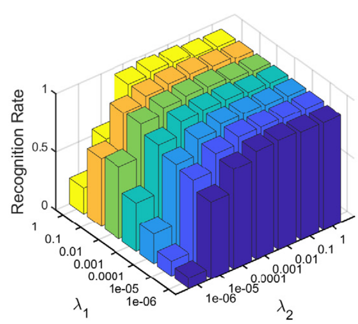


(a) Convergence on Caltech101

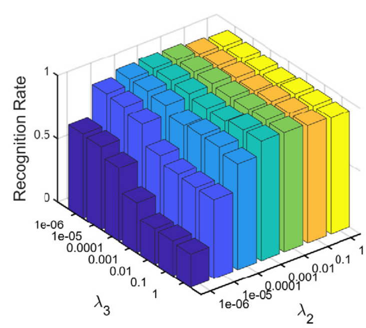


(b) Convergence on Scene15

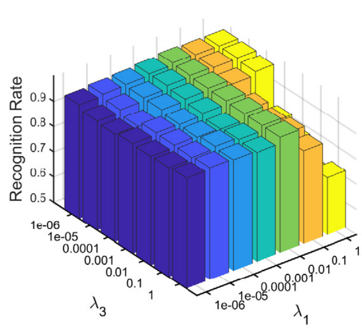
Fig. 5. Convergence curve of SVM-DDPL on Caltech101 and Scene15.



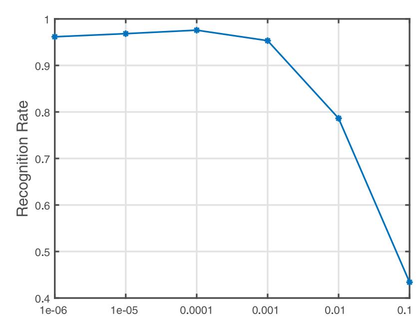
(a) Results with different values of λ_1 and λ_2 with $\lambda_3 = 2e-6$ and $\beta = 5$.



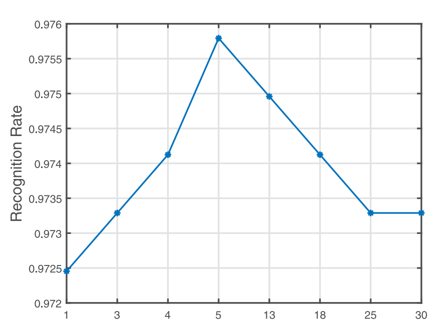
(b) Results with different values of λ_2 and λ_3 with $\lambda_1 = 2e-2$ and $\beta = 5$.



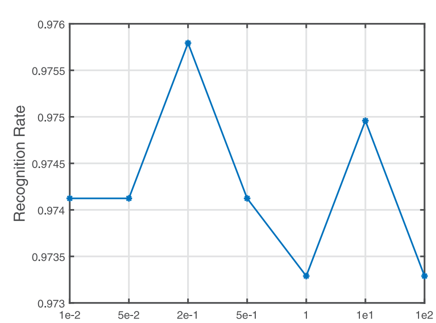
(c) Results with different values of λ_1 and λ_3 with $\lambda_2 = 2e-3$ and $\beta = 5$.



(d) Results with different values of α.



(e) Results with different values of β.



(f) Results with different values of $\mathcal{C}$.

Fig. 6. Recognition rates obtained via SVM-DDPL with different settings of λ_1, λ_2, λ_3, α, β and $\mathcal{C}$ on the Extended YaleB database.

Table 1
Parameter settings of SVM-DDPL in each database.

Parameters	AR	Extended YaleB	Caltech101	Scene15	Isolet
λ_1	8e−1	2e-2	3e−2	8e−2	1e-1
λ_2	1e−3	2e-3	1e−5	1e−5	1e-3
λ_3	1e−5	2e-6	1e−5	1e−5	1e-5
m	6	10	10	20	10
β	0.1	5	1	7	1

4.4. Comparison with other algorithms

To verify the performance of SVM-DDPL, it is compared with other DDL methods in terms of classification accuracy as well as training time and testing time. All simulations were performed in Matlab 2020a with an Intel Core i5 CPU at 2.90 GHz and 16 GB memory. The testing time is defined as the average processing time to classify a single image. We measure the performance of each algorithm by repeating the experiment over 5 realizations and report the averaged results.

4.4.1. Competing algorithms

The competing algorithms are Fisher Discrimination Dictionary Learning (FDDL)[3] (Yang et al., 2011), LC-KSVD[4] (Jiang et al., 2013), SVGDL[5] (Cai et al., 2014), SADL[6] (Tang et al., 2019), CADL[7] (Wang et al., 2017), and DPL[8] (Gu et al., 2014) algorithms. Among these algorithms, FDDL, LC-KSVD, and SVGDL are synthesis dictionary learning based algorithms, SADL and CADL learn a structured analysis dictionary, while DPL is based on dictionary pair learning. We have also used two latest dictionary-learning-based algorithms as baselines, i.e., relaxed block-diagonal dictionary pair learning with a locality constraint (RBD-DPL)[9] (Chen, Wu, & Kittler, 2021) and twin-incoherent self-expressive latent DPL (SLatDPL)[10] (Zhang et al., 2021). RBD-DPL introduces relaxed block-diagonal structures to representations to enhance the discriminability of dictionaries. SLatDPL integrates the coefficient learning and salient feature extraction into a unified model and uses a self-expressive adaptive locality-preserving framework with a twin-incoherence constraint.

Note the DPL algorithm can be seen as an ablation version of the proposed algorithm without the coding coefficient discriminative term, and the comparison with DPL aims to validate the impact of the coding coefficient discriminative term introduced in the proposed algorithm. The proposed SVM-DDPL algorithm jointly learns the dictionary pairs and SVM classifiers so that the training process can comprehensively exploit dictionaries and SVM classifiers. In contrast, training SVM after the dictionary pair learning separates dictionary learning and SVM training into two independent stages. With this framework, the discriminative capability of the representation coefficients in dictionary learning is not fully exploited and the SVM parameters are optimized based only on the fixed coefficients. As a result, this framework is not as flexible as the joint learning paradigm used in the proposed algorithm. To compare the performance of these two training schemes, we perform experiments by training SVM after learning the dictionary pair, which is referred to as DPL+SVM. The numbers of dictionary atoms are the same as those used in the proposed algorithm, and other parameters follow the original settings of DPL. The classification is also based on the proposed fusing classification scheme.

The results of the competing algorithms are from the original papers of the algorithms or obtained based on the released implementation. Specifically, for the experiments that have been performed in the original papers under the same experimental settings as ours, the results reported in the original papers are presented. For other experiments, the parameters of the competing algorithms are tuned carefully as suggested in their original papers and the best results are presented.

4.4.2. Results on AR

The averaged recognition rates on AR are reported in Table 2. The accuracy of the proposed algorithm is slightly higher than that of the previous algorithms and achieves the best result on the AR database. SVM-DDPL achieves 98.7% accuracy on the AR database, which has 0.4% improvements over DPL. The training time of SVM-DDPL is longer than DPL as SVM classifiers are trained together with dictionary learning.

4.4.3. Results on Extended YaleB

The recognition rates on Extended YaleB are presented in Table 3. For Extended YaleB, as shown in Table 3, the proposed algorithm performs better than FDDL, LC-KSVD, SADL, SVGDL, and DPL, and achieves 97.6% accuracy. The training time of this algorithm is shorter than that of SVGDL when it obtains the maximum accuracy, which may be because it has a shorter encoding time.

4.4.4. Results on Caltech101

The experimental results on the Caltech101 database of different algorithms are listed in Table 4. It can be seen that the recognition rate of the SVM-DDPL algorithm on the Caltech101 database is better than other algorithms. The recognition rate of this algorithm is 3.6% higher than that of the DPL algorithm. This further shows that the simultaneous introduction of SVM classifiers and structural dictionary pair is helpful to improve the classification accuracy.

In addition, we randomly select 5, 10, 15, 20, 25, and 30 samples per category as training samples and test on the remaining. The comparison results with other DDL methods are shown in Table 5.

As we can seen from Table 5, the classification rate on the Caletch101 database will be improved with the increase in the number of training samples per class. In general, with different numbers of training samples, the proposed algorithm outperforms DPL. The best result is obtained by the proposed algorithm when the number of training samples per class is 30.

4.4.5. Results on Scene15

The classification accuracies obtained by different algorithms are shown in Table 6. Our proposed algorithm outperforms all competing algorithms except for SADL. It achieves 0.8% improvement as compared with DPL. The confusion matrix of SVM-DDPL is shown in Fig. 7. It can be seen that the recognition accuracy of our algorithm for the scenes of the bedroom and living room can reach 100%.

[3] https://github.com/JumperWang/FDDL
[4] http://www.zhuolin.umiacs.io/projectlcksvd.html
[5] http://www4.comp.polyu.edu.hk/~cslzhang/code/SVGDL.zip
[6] https://github.com/wtang0512/
[7] https://github.com/eeGuoJun/SPL2017
[8] http://www4.comp.polyu.edu.hk/~cslzhang/code/DPL_NIPS14.zip
[9] https://github.com/chenzhe207/RBD-DPL
[10] https://github.com/Daitu/SLatDPL

Table 2
Recognition rates (%) on the AR database.

Methods	Classification accuracy (%)	Training time (s)	Testing time (s)
FDDL (Yang et al., 2011)	96.9	91.0	3.5e−3
LC-KSVD (Jiang et al., 2013)	97.8	39.7	2e−4
SVGDL (Cai et al., 2014)	96.7	760.0	2.0e−5
SADL (Tang et al., 2019)	97.2	32.6	1.3e−5
CADL (Wang et al., 2017)	98.6	107.9	7.1e−5
DPL (Gu et al., 2014)	98.3	11.3	3.9e−4
RBD-DPL (Chen et al., 2021)	97.6	2.84	1.6e−5
SLatDPL (Zhang et al., 2021)	**98.8**	54.8	2.0e−3
DPL+SVM	98.4	12.0	5.8e−5
SVM-DDPL	98.7	120.2	1.8e−4

Table 3
Recognition rates (%) on the Extended YaleB database.

Methods	Classification accuracy (%)	Training time (s)	Testing time (s)
FDDL (Yang et al., 2011)	95.8	23.0	7.3e−3
LC-KSVD (Jiang et al., 2013)	96.7	6.8	3.2e−5
SVGDL (Cai et al., 2014)	96.1	220.0	7.9e−6
SADL (Tang et al., 2019)	96.3	39.2	7.6e−6
CADL (Wang et al., 2017)	97.3	120.3	1.9e−5
DPL (Gu et al., 2014)	97.5	4.4	1.7e−4
RBD-DPL (Chen et al., 2021)	97.3	2.98	1.8e−4
SLatDPL (Zhang et al., 2021)	**98.8**	36.8	1.1e−3
DPL+SVM	97.1	6.1	2.5e−5
SVM-DDPL	97.6	69.5	1.5e−4

Table 4
Recognition rates (%) on the Caltech101 database.

Methods	Classification accuracy (%)	Training time (s)	Testing time (s)
FDDL (Yang et al., 2011)	73.1	529.8	5.7e−3
LC-KSVD (Jiang et al., 2013)	73.6	373.9	1.8e−4
SVGDL (Cai et al., 2014)	76.7	1500.0	1.2e−5
SADL (Gu et al., 2014)	74.5	847.5	4.8e−5
CADL (Wang et al., 2017)	75.0	2501.1	4.2e−4
DPL (Tang et al., 2019)	73.9	134.6	1.3e−3
RBD-DPL (Chen et al., 2021)	72.8	58.6	1.2e−4
SLatDPL (Zhang et al., 2021)	74.6	740.3	6.6e−2
DPL+SVM	76.1	139.2	2.2e−4
SVM-DDPL	**77.5**	1913.2	1.3e−3

Table 5
Recognition rates (%) on Caltech101 with different numbers of training samples per class.

# of training samples per class	5	10	15	20	25	30
FDDL	53.6	63.6	66.8	69.8	71.7	73.1
LC-KSVD	54.0	63.1	67.7	70.5	72.3	73.6
SVGDL	55.3	64.3	69.6	72.3	75.1	76.7
SADL	48.6	59.8	63.8	66.1	71.2	74.5
CADL	49.7	60.2	66.9	70.1	73.7	75.0
DPL	47.2	57.7	63.2	66.3	68.8	73.9
RBD-DPL	46.6	56.2	62.4	67.0	69.1	72.8
SLatDPL	47.9	57.2	63.8	68.3	71.2	74.6
DPL+SVM	47.9	60.1	65.3	68.9	70.6	76.1
SVM-DDPL	48.8	62.3	67.5	71.5	73.5	77.5

Table 6
Recognition rates (%) on the Scene15 database.

Methods	Classification accuracy (%)	Training time (s)	Testing time (s)
FDDL (Yang et al., 2011)	92.3	5.5	3.3e−3
LC-KSVD (Jiang et al., 2013)	92.9	3.4	2.3e−5
SVGDL (Cai et al., 2014)	96.8	66.3	2.7e−6
SADL (Tang et al., 2019)	**98.5**	174.2	2.4e−5
CADL (Wang et al., 2017)	97.3	523.1	2.6e−4
DPL (Gu et al., 2014)	96.9	12.2	2.0e−4
RBD-DPL (Chen et al., 2021)	98.0	24.6	2.2e−4
SLatDPL (Zhang et al., 2021)	94.5	807.6	1.1e−2
DPL+SVM	97.2	14.7	3.2e−4
SVM-DDPL	97.7	153.1	2.1e−4

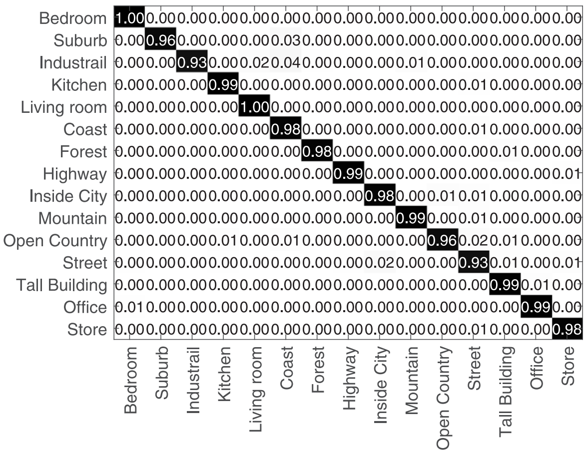


Fig. 7. Confusion matrix of SVM-DDPL on the Scene15 database.

4.4.6. Results on Isolet

Following the settings in Zhang, Li, Chow, Zhang, and Yan (2016) and Wang, Yang, and Li (2019), the experiments on Isolet are performed by varying the number of labeled spoken letters from 3 to 30 with interval 3. We first tune the optimal parameters of each method on Isolet5 and perform experiments on Isolet2 to Isolet4 with the same parameters. The results of different algorithms with varying numbers of labeled samples in each class are shown in Fig. 8.

As we can see, the recognition rates obtained by all algorithms increase with the increase of the number of labeled training samples. The proposed SVM-DDPL algorithm outperforms other algorithms in most cases. As compared with original DPL methods, SVM-DDPL improves recognition rates by at least 0.5% in all cases. The experimental results on the Isolet database show our SVM-DDPL of imposing SVM response constraint on sparse coding coefficients can produce considerable performance improvement for spoken letter recognition.

4.5. Evaluation of classification scheme

In the proposed algorithm, we employ a fusion classification scheme by considering both reconstruction error and SVM responses of the coding coefficients. If only the reconstruction error term or the SVM-related term is considered in classification, the advantages offered by the dictionary pair learning stage cannot be fully utilized, and the final classification performance may not be as good as that of the proposed classification scheme based on both terms.

To compare different classification schemes, the proposed SVM-DDPL and DPL+SVM are tested using different classification schemes while keeping the same dictionary learning stage. Specifically, classification schemes using reconstruction error, SVM, and the fusion of these two are referred to Err, SVM and Err+SVM, respectively. The results of SVM-DDPL and DPL+SVM with different classification schemes are shown in Fig. 9. Note that DPL+SVM using the Err classification scheme is actually reduced to the DPL algorithm (Gu et al., 2014).

From Fig. 9, it can be seen that for DPL+SVM the SVM classification scheme performs better than the Err classification scheme for AR and Caltech101 while the Err classification scheme obtains better results for the remaining two databases. This is probably because that the reconstruction error and SVM classifier are not balanced jointly in the training process as the SVM classifier in DPL+SVM is trained independently after the dictionary is learned in DPL. For the SVM-DDPL algorithm, the SVM classification scheme achieves better results than the Err classification scheme on all databases except for Extended YaleB, which illustrates the effectiveness of the SVM classifier in SVM-DDPL. We can also see that the fusion classification scheme (i.e. Err+SVM) helps improve the accuracies of classification as compared with the other two classification schemes, i.e. Err and SVM. For example, for the algorithm DPL+SVM, the fusion classification scheme obtains better results on AR, Caltech101, and Scene15 than the individual classification schemes Err and SVM. Similarly, for SVM-DDPL, the fusion classification scheme performs better on all the databases than the individual classification schemes Err and SVM. On all the datasets, the SVM-DDPL achieves the best performance when using the fusion classification scheme.

4.6. Discrimination of the proposed model

The goal of classification is to classify samples into their correct categories, and discrimination is the key point to achieving this goal. In particular, discrimination refers to the ability to improve intraclass similarities and interclass differences by mapping the original samples or features to other spaces. In the proposed algorithm, discrimination of both dictionary pairs and representation coefficients is exploited. In the training process, a structured dictionary pair is learned so that a sample can be represented more accurately by the sub-dictionary pair corresponding to its class, which improves the discrimination of dictionary pairs. The discrimination of representation coefficients is promoted via the SVM-based regularizer where hyperplanes are trained in an embedding way to separate the coefficients corresponding to different classes. In the training process, the learned dictionary pair and representation coefficients are both utilized to classify samples by fusing reconstruction error based on sub-dictionary pairs and SVM responses in the classification scheme.

As the final purpose of promoting discrimination is to perform the classification task, discrimination can be evaluated in terms of classification accuracies. The comparison of the proposed algorithm and other baselines in Section 4.4 can demonstrate the superiority of the proposed algorithm in terms of discrimination. In addition, the discrimination can be evaluated by observing the intraclass similarities and interclass differences directly. For the proposed algorithm, representation error and SVM responses can reflect the discrimination of the learned dictionary pair and representation coefficients, respectively. Taking Scene15 and Calteh101 as examples, Figs. 10–11 illustrate the representation error, SVM responses, and the fusion of them for testing samples. It can be seen that all matrices are block-diagonal. The blocks of the representation error matrices show the representation error of the samples using the sub-dictionary pair of the associated class tends to be smaller than the representation error obtained by sub-dictionary pairs of other classes, which illustrates the discrimination of the learned dictionary pair. The blocks in SVM response matrices can illustrate the discrimination of the representation coefficients. By comparing subfigures (c) with the associated subfigures (a) and (b), it can be observed that the blocks tend to be clearer in subfigures (c) for both databases, which demonstrates that the fusion of representation error and SVM response can improve the overall discrimination capability.

4.7. Comparison with deep learning methods

In particular, we further compare the proposed method with some well-known convolutional neural networks, namely, AlexNet (Krizhevsky et al., 2012), GoogLeNet (Szegedy et al., 2015), VGGNet (Simonyan & Zisserman, 2015), ResNet-18 (He

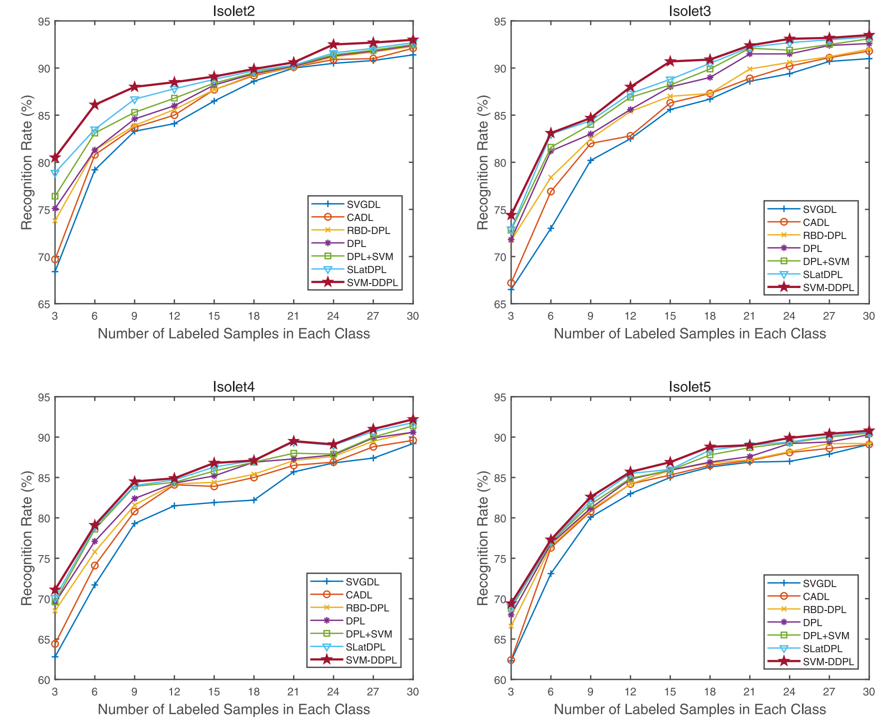


Fig. 8. Recognition results of different algorithms with varying numbers of labeled samples in each class.

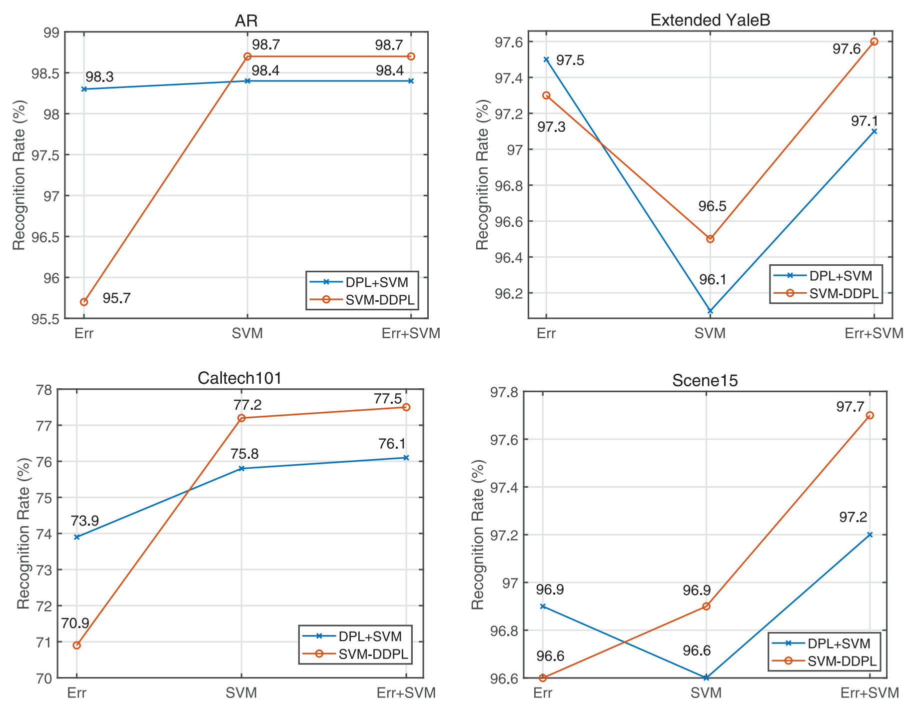
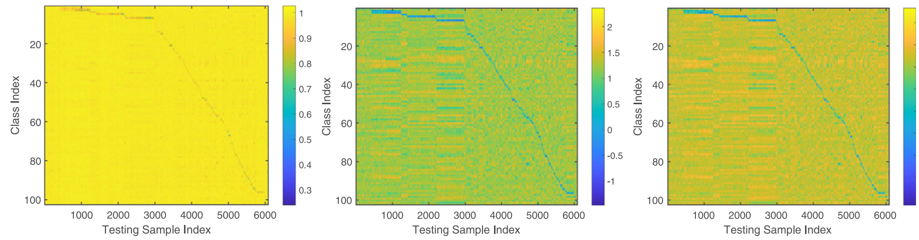


Fig. 9. Recognition rates (%) obtained by DPL+SVM and SVM-DDPL using different classification schemes.

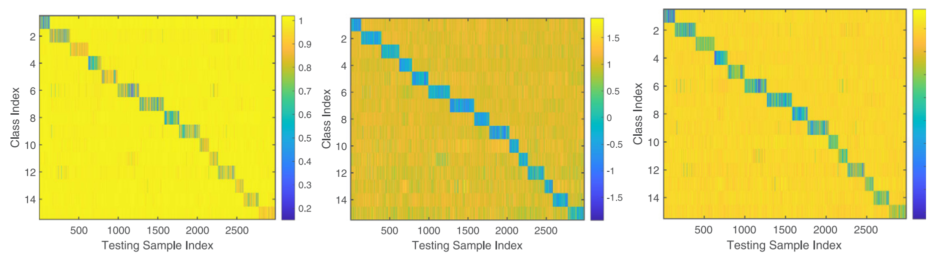
et al., 2016) and PCANet (Chan et al., 2015). For VGGNet, the architecture with 16 layers is used, and for PCANet, the two-stage version is used. For the deep neural networks including AlexNet, GooLeNet, VGGNet, and ResNet-18, both the network models with random initialization and the models pre-trained on ImageNet database (Deng et al., 2009) are tested. The numbers of training and testing samples are the same as those used in the proposed SVM-DDPL algorithm. Experimental results are summarized in Table 7, where the results in parentheses are obtained with randomly initialized network models.

From the results, it can be seen that deep neural networks, such as AlexNet, GoogLeNet, VGGNet, and ResNet-18 do not perform well if they are not pre-trained on the large-scale database. Using pre-trained models can improve the results of deep learning methods significantly. For the Caltech101 database, deep learning methods with pre-trained models achieve better results.



(a) $\|\mathbf{x}-\mathbf{D}_k\mathbf{P}_k\mathbf{x}\|_2^2$ (b) $-(\mathbf{u}_k^T\mathbf{z}+b_k)$ (c) $\beta\|\mathbf{x}-\mathbf{D}_k\mathbf{P}_k\mathbf{x}\|_2^2-(\mathbf{u}_k^T\mathbf{z}+b_k)$

Fig. 10. Discrimination for Caltech101.



(a) $\|\mathbf{x}-\mathbf{D}_k\mathbf{P}_k\mathbf{x}\|_2^2$ (b) $-(\mathbf{u}_k^T\mathbf{z}+b_k)$ (c) $\beta\|\mathbf{x}-\mathbf{D}_k\mathbf{P}_k\mathbf{x}\|_2^2-(\mathbf{u}_k^T\mathbf{z}+b_k)$

Fig. 11. Discrimination for Scene15.

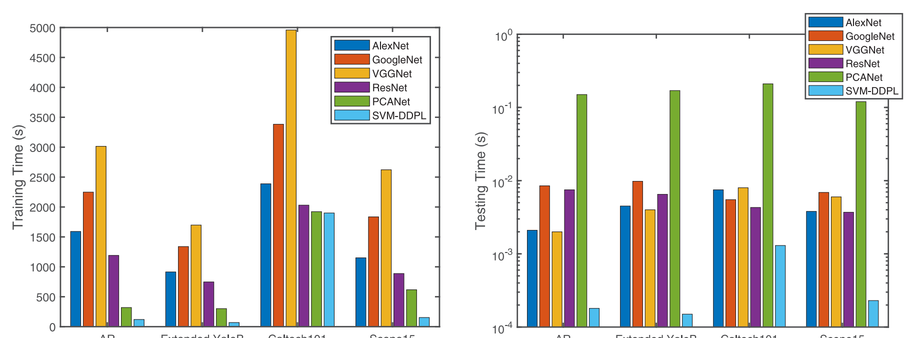


Fig. 12. Training time and testing time of SVM-DDPL and deep learning methods.

Table 7
The recognition rates (%) as compared with deep learning methods.

Algorithms	AR	Extended YaleB	Caltech101	Scene15
AlexNet (Krizhevsky et al., 2012)	85.3 (81.1)	61.7 (57.1)	79.6 (49.1)	81.2 (60.3)
GoogLeNet (Szegedy et al., 2015)	89.3 (85.9)	78.2 (69.2)	85.5 (53.7)	87.7 (67.5)
VGGNet (Simonyan & Zisserman, 2015)	85.1 (79.1)	86.9 (56.2)	87.8 (39.5)	88.8 (52.1)
ResNet-18 (He et al., 2016)	98.6 (86.2)	85.4 (72.3)	**89.4** (58.6)	92.3 (74.9)
PCANet (Chan et al., 2015)	95.6	**98.2**	68.5	87.2
SVM-DDPL	**98.7**	97.6	77.5	**97.7**

This is because the contents of this database are similar to ImageNet which is used to pre-train the models. For face databases AR and Extended YaleB, and scene database Scene15, as their contents are quite different from the contents of ImageNet, the improvement of the pre-trained models is limited. For these three databases, the proposed algorithm obtains better results than deep networks including AlexNet, GoogLeNet, VGGNet, and ResNet-18. The shallow network PCANet achieves the highest accuracy on Extended YaleB, but its performance on the other databases is not as good as the proposed algorithm. In summary, as compared with deep learning approaches, the proposed method does not rely on a large-scale training set and can obtain promising results for datasets of small sizes without pre-training.

The training time and testing time of SVM-DDPL and deep learning methods are shown in Fig. 12. It can be seen that deep learning methods need more time to reach the current accuracies than the proposed algorithm, in terms of both training time and testing time.

5. Conclusions

We have proposed a novel discriminative dictionary pair learning model by introducing a differentiable SVM term to the original DPL model. The additional discriminative term is associated with coding coefficients and can further improve the discrimination of coding coefficients in the process of dictionary pair learning. Based on this model, a pair of structured synthesis–analysis dictionary together with a set of SVM classifiers can be learned jointly. The corresponding optimization problem is addressed by updating optimal variables alternatively. In the classification stage, the reconstruction residual and the support vector machine classifier are used collaboratively to determine the categories of samples. Simulations on the AR, Extended YaleB, Caltech101, Scene15, and Isolet databases were performed and the results have demonstrated the superiority of the proposed DPL-SVM as compared with other state-of-the-art DDL methods.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61906087). The authors wish to thank the Associate Editor and the anonymous reviewers for their contributions to improving the quality of the paper.

Appendix

In this appendix, the derivation of Eq. (3.5) is detailed.

Based on Eqs. (3.3) and (3.4), when $1 - y_i^k(\mathbf{u}_k^T\mathbf{a}_i + b_k) > 0$, the update of $\mathbf{a}_i$ corresponds to solving the following optimization problem

$$\begin{aligned}&\arg\min_{\mathbf{a}_i} \|\mathbf{x}_i - \mathbf{D}\mathbf{a}_i\|_2^2 + \lambda_1\|\mathbf{P}\mathbf{x}_i - \mathbf{a}_i\|_2^2 \\ &+ \lambda_3\mathcal{C}\sum_{k=1}^{K}\|1 - y_i^k(\mathbf{u}_k^T\mathbf{a}_i + b_k)\|_2^2.\end{aligned} \tag{A.1}$$

The closed-form solution to this problem can be obtained by taking the derivative of (A.1) with respect to $\mathbf{a}_i$ and setting it to zero. In particular, the gradient of (A.1) with respect $\mathbf{a}_i$ can be written as

$$\begin{aligned}\nabla_{\mathbf{a}_i} =& -2\mathbf{D}^T(\mathbf{x}_i - \mathbf{D}\mathbf{a}_i) - 2\lambda_1(\mathbf{P}\mathbf{x}_i - \mathbf{a}_i) \\ &+ \lambda_3\mathcal{C}\sum_{k=1}^{K}[-2y_i^k\mathbf{u}_k(1 - y_i^k\mathbf{u}_k^T\mathbf{a}_i - y_i^k\mathbf{b}_k)] \\ =& -2\mathbf{D}^T\mathbf{x}_i + 2\mathbf{D}^T\mathbf{D}\mathbf{a}_i - 2\lambda_1\mathbf{P}\mathbf{x}_i + 2\lambda_1\mathbf{a}_i - 2\lambda_3\mathcal{C}\sum_{k=1}^{K}y_i^k\mathbf{u}_k \\ &+ 2\lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k\mathbf{u}_k^T\mathbf{a}_i + 2\lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k\mathbf{b}_k \\ =& 2\left[\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{I} + \lambda_3\mathcal{C}\sum_{k=1}^{K}(y_i^k)^2\mathbf{u}_k\mathbf{u}_k^T\right]\mathbf{a}_i \\ &- 2\left[\mathbf{D}\mathbf{x}_i + \lambda_1\mathbf{P}\mathbf{x}_i + \lambda_3\mathcal{C}\sum_{k=1}^{K}y_i^k\mathbf{u}_k - \lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k b_k\right].\end{aligned} \tag{A.2}$$

Let $\nabla\mathbf{a}_i = 0$, and the closed-form solution can be obtained, that is

$$\begin{aligned}\mathbf{a}_i =& \left[\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{I} + \lambda_3\mathcal{C}\sum_{k=1}^{K}(y_i^k)^2\mathbf{u}_k\mathbf{u}_k^T\right]^{-1} \\ &* \left[\mathbf{D}\mathbf{x}_i + \lambda_1\mathbf{P}\mathbf{x}_i + \lambda_3\mathcal{C}\sum_{k=1}^{K}y_i^k\mathbf{u}_k - \lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k b_k\right].\end{aligned} \tag{A.3}$$

As $y_i^k = 1$ or -1, the solution can be simplified as

$$\begin{aligned}\mathbf{a}_i =& \left(\mathbf{D}^T\mathbf{D} + \lambda_1\mathbf{I} + \lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k\mathbf{u}_k^T\right)^{-1} \\ &* \left(\mathbf{D}\mathbf{x}_i + \lambda_1\mathbf{P}\mathbf{x}_i + \lambda_3\mathcal{C}\sum_{k=1}^{K}y_i^k\mathbf{u}_k - \lambda_3\mathcal{C}\sum_{k=1}^{K}\mathbf{u}_k b_k\right).\end{aligned} \tag{A.4}$$

References

Aharon, M., Elad, M., & Bruckstein, A. (2006). K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, *54*(11), 4311–4322.

Ameri, R., Pouyan, A., & Abolghasemi, V. (2016). Projective dictionary pair learning for EEG signal classification in brain computer interface applications. *Neurocomputing*, *218*, 382–389.

Boyd, S., Parikh, N., Chu, E., Peleato, B., & Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, *3*(1), 1–122.

Cai, S., Zuo, W., Lei, Z., Feng, X., & Ping, W. (2014). Support vector guided dictionary learning. In *Proceedings of the 13th European conference on computer vision* (pp. 624–639). Switzerland.

Chan, T.-H., Jia, K., Gao, S., Lu, J., Zeng, Z., & Ma, Y. (2015). PCANet: A simple deep learning baseline for image classification. *IEEE Transactions on Image Processing*, *24*(12), 5017–5032.

Chen, S. S., Donoho, D. L., & Saunders, M. A. (1998). Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, *20*(1), 33–61.

Chen, Y., Jiao, L., Li, Y., Li, L., Zhang, D., Ren, B., et al. (2019). A novel semicoupled projective dictionary pair learning method for PolSAR image classification. *IEEE Transactions on Geoscience and Remote Sensing*, *57*(4), 2407–2418.

Chen, Z., Wu, X.-J., & Kittler, J. (2021). Relaxed block-diagonal dictionary pair learning with locality constraint for image recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 1–15.

Dai, W., Xu, T., & Wang, W. (2012). Simultaneous codeword optimization (SimCO) for dictionary update and learning. *IEEE Transactions on Signal Processing*, *60*(12), 6340–6353.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255).

Dong, J., Han, Z., Zhao, Y., Wang, W., Prochazka, A., & Chambers, J. (2017). Sparse analysis model based multiplicative noise removal with enhanced regularization. *Signal Processing*, *137*, 160–176.

Dong, J., Wang, W., Dai, W., Plumbley, M. D., Han, Z.-F., & Chambers, J. (2016). Analysis SimCO algorithms for sparse analysis model based dictionary learning. *IEEE Transactions on Signal Processing*, *64*(2), 417–431.

Fanty, M., & Cole, R. (1990). Spoken letter recognition. In *Neural information processing systems* (pp. 220–226).

Georghiades, A., Belhumeur, P., & Kriegman, D. (2001). From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(6), 643–660.

Gu, S., Zhang, L., Zuo, W., & Feng, X. (2014). Projective dictionary pair learning for pattern classification. In *Advances in Neural Information Processing Systems*: *vol. 27*.

Guo, J., Guo, Y., Kong, X., Zhang, M., & He, R. (2016). Discriminative analysis dictionary learning. In *AAAI conference on artificial intelligence* (pp. 1617–1623).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Jiang, Z., Lin, Z., & Davis, L. S. (2013). Label consistent K-SVD: Learning a discriminative dictionary for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(11), 2651–2664.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. In *Neural information processing systems*: *vol. 25*, (no. 2), (pp. 1097–1105).

Lazebnik, S., Schmid, C., & Ponce, J. (2006a). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE conference on computer vision and pattern recognition*: *vol. 2*, (pp. 2169–2178).

Lazebnik, S., Schmid, C., & Ponce, J. (2006b). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE conference on computer vision and pattern recognition* (pp. 2169–2178).

Li, F.-F., Fergus, R., & Perona, P. (2004). Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In *IEEE conference on computer vision and pattern recognition workshop* (p. 178).

Li, Z., Li, Y., Tan, B., Ding, S., & Xie, S. (2022). Structured sparse coding with the group log-regularizer for key frame extraction. *IEEE/CAA Journal of Automatica Sinica*, http://dx.doi.org/10.1109/JAS.2022.105602.

Li, B., Rencker, L., Dong, J., Luo, Y., Plumbley, M. D., & Wang, W. (2021). Sparse analysis model based dictionary learning for signal declipping. *IEEE Journal of Selected Topics in Signal Processing*, *15*(1), 25–36.

Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., & Bach, F. (2009). Supervised dictionary learning. In *Advances in neural information processing systems*: *vol. 21*.

Martinez, A., & Benavente, R. (1998). *The AR face database*: *CVC technical report*.

Nam, S., Davies, M. E., Elad, M., & Gribonval, R. (2013). The cosparse analysis model and algorithms. *Applied and Computational Harmonic Analysis*, *34*(1), 30–56.

Pati, Y. C., Rezaiifar, R., & Krishnaprasad, P. (1993). Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *27th Asilomar conference on signals, systems, and computers* (pp. 40–44).

Ramirez, I., Sprechmann, P., & Sapiro, G. (2010). Classification and clustering via dictionary learning with structured incoherence and shared features. In *IEEE conference on computer vision and pattern recognition* (pp. 3501–3508).

Ravishankar, S., & Bresler, Y. (2013). Learning sparsifying transforms. *IEEE Transactions on Signal Processing*, *61*(5), 1072–1086.

Rencker, L., Bach, F., Wang, W., & Plumbley, M. D. (2019). Sparse recovery and dictionary learning from nonlinear compressive measurements. *IEEE Transactions on Signal Processing*, *67*(21), 5659–5670.

Rubinstein, R., & Elad, M. (2014). Dictionary learning for analysis-synthesis thresholding. *IEEE Transactions on Signal Processing*, *62*(22), 5962–5972.

Rubinstein, R., Peleg, T., & Elad, M. (2013). Analysis K-SVD: A dictionary-learning algorithm for the analysis sparse model. *IEEE Transactions on Signal Processing*, *61*(3), 661–677.

Shekhar, S., Patel, V. M., & Chellappa, R. (2014). Analysis sparse coding models for image-based classification. In *IEEE international conference on image processing*.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *3rd International conference on learning representations*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2015). Going deeper with convolutions. In *2015 IEEE conference on computer vision and pattern recognition* (pp. 1–9).

Tang, W., Panahi, A., Krim, H., & Dai, L. (2019). Analysis dictionary learning based classification: Structure for robustness. *IEEE Transactions on Image Processing*, *28*(12), 6035–6046.

Tropp, J., & Gilbert, A. (2007). Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, *53*(12), 4655–4666.

Wang, J., Guo, Y., Guo, J., Luo, X., & Kong, X. (2017). Class-aware analysis dictionary learning for pattern classification. *IEEE Signal Processing Letters*, *24*(12), 1822–1826.

Wang, W., Yang, C., & Li, Q. (2019). Discriminative analysis dictionary and classifier learning for pattern classification. In *2019 IEEE international conference on image processing* (pp. 385–389).

Xu, R., Xu, Y., & Quan, Y. (2021). Factorized tensor dictionary learning for visual tensor data completion. *IEEE Transactions on Multimedia*, *23*, 1225–1238.

Yang, J., Yu, K., Gong, Y., & Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *IEEE conference on computer vision and pattern recognition* (pp. 1794–1801).

Yang, M., Zhang, L., Feng, X., & Zhang, D. (2011). Fisher discrimination dictionary learning for sparse representation. In *International conference on computer vision* (pp. 543–550).

Zhang, Z., Li, F., Chow, T. W. S., Zhang, L., & Yan, S. (2016). Sparse codes auto-extractor for classification: A joint embedding and dictionary learning framework for representation. *IEEE Transactions on Signal Processing*, *64*(14), 3790–3805.

Zhang, Z., Sun, Y., Wang, Y., Zhang, Z., Zhang, H., Liu, G., et al. (2021). Twin-incoherent self-expressive locality-adaptive latent dictionary pair learning for classification. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(3), 947–961.

Zhao, H., Li, Z., Chen, W., Zheng, Z., & Xie, S. (2022). Accelerated partially shared dictionary learning with differentiable scale-invariant sparsity for multi-view clustering. *IEEE Transactions on Neural Networks and Learning Systems*, http://dx.doi.org/10.1109/TNNLS.2022.3153310.