Supervised dictionary learning with multiple classifier integration

Yuhui Quan, Yong Xu*, Yuping Sun, Yan Huang

School of Computer Science & Engineering,
South China University of Technology, Guangzhou 510006, China

Abstract

Supervised sparse coding has become a widely-used module in existing recognition systems, which unifies classifier training and dictionary learning to enforce discrimination in sparse codes. Many existing methods suffer from the insufficient discrimination when dealing with high-complexity data due to the use of simple supervised techniques. In this paper, we integrate multiple classifier training into dictionary learning to overcome such a weakness. A minimization model is developed, in which an ensemble of classifiers for prediction and a dictionary for representation are jointly learned. The ensemble of classifiers is constructed from a set of linear classifiers, each of which is associated with a group of atoms and applied to the corresponding sparse codes. Such a construction scheme allows the dictionary and all the classifiers to be simultaneously updated during training. In addition, we provide an interesting insight into label consistency from the view of multiple classifier learning by showing its relation with the proposed method. Compared with the existing supervised sparse coding approaches, our method is able to learn a compact dictionary with better discrimination and a set of classifiers with improved robustness. The experiments in several image recognition tasks show the improvement of the proposed method over several state-of-the-art approaches.

Keywords: Sparse coding, Supervised dictionary learning, Multiple classifier learning, Image classification

1. Introduction

In recent years, sparse models have been widely used in a variety of applications in computer vision and pattern recognition, e.g., image analysis [1, 2], image processing [3–6] and image recognition [7–15]. The philosophy of sparse modeling comes from the parsimony principle which refers to representing objects using as few variables as

*Corresponding author.
Email addresses: yuhui.quan@mail.scut.edu.cn (Yuhui Quan), yxu@scut.edu.cn (Yong Xu), sun.yp@mail.scut.edu.cn (Yuping Sun), h.y47@mail.scut.edu.cn (Yan Huang)

possible [16], and the success of sparse modeling is attributed to the fact that high-dimensional data of particular types often lie on some low-dimensional manifolds. Given a set of input data, sparse modeling aims at expressing each input data by a linear combination of a few elements taken from a set of representative patterns. The representative patterns are called atoms, and the total set of patterns is called dictionary. The coefficients of the linear combination are often referred to as sparse codes.

The dictionaries for sparse modeling are usually learned from data to maximize the efficiency of sparse approximation in terms of sparsity degree, which have shown improvement over the analytic dictionaries like wavelets in signal processing; see e.g. [3, 17, 18]. However, it is not optimal to use these dictionaries for classification problems where not only the sparsity but also the discriminability of sparse codes are pursued.[1] To enforce discrimination in sparse codes, the *supervised dictionary learning* methods [19–32] have been proposed to learn dictionaries in a supervised manner. The main idea of these methods is to couple the process of classifier training and the process of dictionary learning, which have exhibited impressive performance in a variety of recognition tasks. But there is still plenty of room for improvement. One possibility comes from the fact that many existing approaches (e.g. [19, 20, 25, 32]) only employ a single simple classifier in the learning process, whose discriminative power is insufficient to handle high-complexity data. This inspired us to integrate multiple classifier learning into supervised dictionary learning.

In this paper, we propose an effective supervised dictionary learning model which integrates multiple classifier training. Together with sparsity constraints, the objective function of the proposed model involves a simple ℓ_2 reconstructive term and a novel ensemble-based discriminative term. The discrimination term is defined by the prediction error summarized from a set of multi-class linear classifiers, each of which is associated with a group of atoms and applied to the sparse codes corresponding to the atom group. The proposed discrimination term has an interesting relation with the label consistency term used in the LC-KSVD method [32]. We then present an efficient numerical algorithm for solving the proposed model, in which the dictionary and classifiers are simultaneously updated. Used as the sparse coding module as well as the classification module, the proposed method is evaluated in several image recognition tasks, including the classification on faces, objects, scenes, actions, and dynamic textures. The experimental results have demonstrated the power of our method in discriminative sparse coding for classification.

[1] The dictionaries inducing discriminative sparse codes are often referred to as discriminative dictionaries.

1.1. Related work

As the goal of this paper is to develop of a dictionary learning method for sparse coding, we first give a detailed literature review on sparse dictionary learning. Then, we give a brief review on some multiple classifier learning methods which are related to our work.

1.1.1. Sparse dictionary learning

In the past, a large number of sparse models have been proposed and studied for visual recognition, whose applications cover building codebooks for local image descriptors [9, 33], learning image patch representations [19, 26], feature selection [34], and classification [7, 27]; see [16] for a comprehensive review. This paper focuses on sparse coding and dictionary learning for classification.

The power of sparse coding for classification stems from its capability for modeling particular types of signals. There are two main successful strategies for exploiting such a capability for classification. The first one learns a class-specific dictionary for each category of signals and classifies signals by comparing the reconstruction errors or sparsity obtained under the learned dictionary. Such a classification scheme is similar in spirit to the nearest neighbor classification and the nearest subspace classification. One seminal work is the SRC method [8] that constructs the dictionaries using training samples, which has shown success in face recognition.

However, the SRC method requires a large dictionary for guaranteed performance, which is infeasible in practice due to the heavy computational burden. Such a drawback can be overcome by learning small-size dictionaries instead of simply taking signals as atoms; see e.g. [19, 29, 35]. But learning class-specific dictionaries separately might cause ambiguities among the learned dictionaries, i.e., signals of some class may also be well represented by the dictionaries of other classes. Several approaches have been proposed to reduce such ambiguities. Mairal et al. [19] incorporated a discriminant defined on class-specific reconstruction errors into dictionary learning to enforce the discrimination of class-specific dictionaries. Ramirez et al. [26] encouraged the independence of class-specific dictionaries by prompting the mutual incoherence among the learned dictionaries, and discarded the shared atoms which have high coherences during classification. Yang et al. [27] proposed to jointly learn class-specific dictionaries by simultaneously regarding the global and intra-class reconstruction errors and the inter-class projection energy. In [36–38], an additional global dictionary is jointly learned with class-specific dictionaries, which improves the compactness and discrimination of class-specific dictionaries. It is worth mentioning that a generalization of learning class-specific dictionaries is the so-called structured dictionary learning, which groups atoms to define structured sparsity on sparse codes. This actually allows interactions between dictionary atoms; see [39] for an example of inducing tree sparsity during dictionary learning and [14] for the concurrent image classification and annotation by grouping dictionary atoms with both class labels and image tags.

The other strategy for using sparse coding for classification is viewing dictionary atoms as discriminative features and using the corresponding sparse codes as the higher-level representations of signals for classification. The proof of this concept was first demonstrated in [7] with an analytic dictionary and a cost function built upon Fisher discriminant. Bach et al. [34] used bootstrap to improve the stability of sparse codes which is crucial to classification. For further improvement on discrimination and performance, joint cost functions that involve both a discriminative term and a classical dictionary learning formulation have been proposed. Marial et al. incorporated the softmax discriminative cost into class-specific dictionary learning [19] as well as single reconstructive dictionary learning [21]. To integrate max-margin classification into dictionary learning, the hinge loss and logistic loss are exploited in [21, 23, 24, 40] for defining the discriminative cost. Pham et al. [20] combined the linear prediction cost with the K-SVD dictionary learning formulation for semi-supervised classification, and based on a similar model, Zhang et al. [25] developed a much more efficient algorithm. Besides the linear prediction cost, Jiang et al. [28, 32] additionally considered the label consistency of subdictionaries in defining the discriminative cost, which explicitly enforces sparsity with structures under some adaptive transform and leads to impressive results in a variety of recognition tasks. As will be seen in Section 3.2, this method is very closely related to our work, and its details are presented in Section 2.2.

As the discriminative terms are constructed in the setting of supervised learning, the methods based on the second strategy are often referred to as supervised dictionary learning in the literature.[2] It is noted that optimizing a joint cost function in supervised dictionary learning requires alternating between three submodules (i.e. sparse coding, dictionary learning, and classification parameters training), which often involves a series of computationally demanding solvers and suffers from the big potential of getting stuck at local minima of the subproblems. Thus, it is preferable to develop supervised learning methods which can simultaneously update the dictionary and classification parameters.

[2] In its most general definition, supervised dictionary learning also includes the dictionary learning methods using the first strategy as these methods assume class labels known.

The benefits of using simultaneous update have been demonstrated in [25, 28, 32], where the dictionary and linear classifier are simultaneously updated using the K-SVD algorithm [3] followed by a renormalization stage. Finally, we would like to mention that supervised dictionary learning is related to neural network, as the joint process of dictionary learning and classifier construction is similar to the back propagation in network training; see [16] for details.

1.1.2. Multiple classifier learning

In the supervised setting, multiple classifier learning refers to a machine learning paradigm where a set of base classifiers are trained and combined as a strong classifier to gain extra performance [41]. It is shown in the literature that the results from the ensemble of multiple classifiers are less dependent on peculiarities of training set and may learn a more expressive concept class than a single classifier; see e.g. [41–43].

Exiting multiple classifier learning methods differ in the strategy of forming the ensemble, which can be roughly divided into two categories when fixing the type of each base classifier to the same, i.e., random subsampling on data like bagging [41, 44] and boosting [1, 2, 22, 45], and random subspace projection on features [42, 43, 46]. We focus on the latter strategy which is adopted in this paper. In the past, the random subspace ensemble strategy has been applied to various types of base classifiers, e.g. linear classifier [43], decision tree [42], logistic regressor [47], etc. In our method, the linear predictor is used as the base classifier, as it is simple yet effective in practice. Note that our method built upon an explicit optimization model is much different from the method in [43] which is based on a looping two-stage scheme.

1.2. Motivation and contribution

Most of the aforementioned supervised dictionary learning methods (e.g. [19, 20, 25, 32]) only employ a single simple classifier in the learning process, which is insufficiently discriminative, lack of expressibility, and strongly dependent on the peculiarities of training data. The resulting dictionaries and sparse codes are not discriminative enough for high-complexity data. Thus, we were inspired to develop a new supervised method to overcome this problem. Motivated by the success of multiple classifier learning [41, 42, 46], we proposed to learn an ensemble of classifiers (i.e. multiple base classifiers) instead of a single one during dictionary learning. Considering the development of an efficient numerical solver, we construct multiple classifiers via random feature subset selection on sparse codes.

Another motivation is that subdictionaries of a discriminative dictionary should be also discriminative.[3] Thus, we group dictionary atoms and strengthen the discriminability of sparse codes associated with each atom group, which encourages the sparse codes to be not only globally discriminative but also partially distinct.

The contribution of this paper is multi-fold. Firstly, by adapting subspace ensemble learning to supervised sparse coding, we propose an effective supervised dictionary learning model that unifies the processes of compact dictionary learning and multiple classifier training. From the perspective of joint dictionary and classifier construction, benefiting from using a set of classifiers instead of a single one, our method has several advantages over many existing methods: the resulting sparse codes has stronger discrimination for recognition and weaker dependence on the peculiarities of training data; and with the use of multiple classifiers our method is able to learn more expressive concepts for improving classification performance. From the perspective of associating class labels with dictionary atoms, our method generalizes the case of learning a classifier for each subdictionary (e.g. [30]), yielding a more compact dictionary. Secondly, an efficient numerical scheme is developed to solve the proposed model, in which the dictionary and classifiers are simultaneously updated instead of being sequentially learned. Such a scheme is nontrivial because our model is different from the K-SVD related models such as D-KSVD [25] and LC-KSVD [32] and cannot be directly solved by the K-SVD algorithm due to the existence of subspace projection matrices. In comparison with the methods (e.g. [20]) that alternate the dictionary learning process and the classifier training process, the simultaneous update of dictionary and classifiers in our algorithm can reduce the probability of getting stuck at local minima. Thirdly, it is shown that the discriminative term in the proposed model is closely related to the label consistency criterion used in [32]. This observation is nontrivial as it provides an interesting view for interpreting the LC-KSVD training model [28] and build up a bridge between label consistency and multiple classifier learning. Finally, it has been demonstrated in the experiments that, when deployed as the sparse coding and classification modules, our method outperforms recent related approaches in a variety of image recognition tasks.

It is worth mentioning that the combination of dictionary learning and multiple classifier learning can be directly done by some two-stage scheme or a simple looping of two processes. However, such ad-hoc methods cannot jointly learn dictionaries and classifiers or may not have explicit learning objectives, which is the weakness compared to our

[3] The subdictionary mentioned here is different from the previously discussed class-specific subdictionary in Section 1.1. It refers to the subset of dictionary atoms, which is not intended for a specific category.

method. It is also worth mentioning that existing ensemble methods can be directly applied to the whole discriminative dictionary learning process by viewing it as a classification process [1, 2]. But the computational cost of this scheme is very expensive and not acceptable in practice due to the need of dictionary learning in each component of ensemble.

The rest of this paper is organized as follows. Section 2 is devoted to the preliminaries on sparse coding and supervised dictionary learning. Our method is presented in Section 3. Experimental evaluation and result analysis are given in Section 4. Section 5 concludes the paper and discusses future work.

2. Dictionary learning for sparse coding: reconstructive, discriminative and supervised approaches

To begin with, we first give an introduction to the notations used in this paper. Bold upper letters are used for matrices, bold lower letters for column vectors, light lower letters for scalars, and calligraphic English alphabets for sets. More specifically, $\boldsymbol{y}_j$ denotes the jth column of the matrix $\boldsymbol{Y}$, y_i denotes the ith element of the vector $\boldsymbol{y}$, $Y_{i,j}$ denotes the entry of $\boldsymbol{Y}$ at ith row and jth column. A group $\mathcal{G}$ is a subset of indices in $\mathbb{Z}^+$, and $\boldsymbol{Y}_{[\mathcal{G}]}$ denotes the sub-matrix of $\boldsymbol{Y}$ which is composed of the rows of $\boldsymbol{Y}$ whose indices fall into the group $\mathcal{G}$. For $\boldsymbol{x} \in \mathbb{R}^N$, its ℓ_q norm $\|\boldsymbol{x}\|_q$ ($q \in [1, \infty)$) is defined as $\|\boldsymbol{x}\|_q = (\sum_{j=1}^N |x_j|^q)^{1/q}$, and its ℓ_0 norm $\|\boldsymbol{x}\|_0$ is defined as $\|\boldsymbol{x}\|_0 = \#\{j|x_j \neq 0\}$. For $\boldsymbol{X} \in \mathbb{R}^{N\times M}$, its Frobenius norm is defined as $\|\boldsymbol{X}\|_F = (\sum_{i=1}^N \sum_{j=1}^M |X_{i,j}|^2)^{1/2}$. Besides, $\boldsymbol{I}_M$ denotes the $M \times M$ identity matrix, $\mathbf{1}_M$ denotes $[\underbrace{1, \ldots, 1}_{M}]^\top$ and $\mathbf{0}_M$ denotes $[\underbrace{0, \ldots, 0}_{M}]^\top$.

2.1. Sparse coding and reconstructive dictionary learning

Let $\boldsymbol{y} \in \mathbb{R}^N$ denote an N-dimensional signal and $\boldsymbol{D} = \{\boldsymbol{d}_k\}_{m=1}^M \subset \mathbb{R}^N$ denote a dictionary composed of M atoms ($M > N$ for redundant dictionary). The goal of sparse coding is to find a linear expansion $\boldsymbol{D}\boldsymbol{c} = \sum_{m=1}^M c_m \boldsymbol{d}_m$ that approximates $\boldsymbol{y}$ as closely as possible by using at most T elements, which can be formulated as the following optimization problem:

$$\underset{\boldsymbol{c}\in\mathbb{R}^M}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{c}\|_2^2, \qquad \text{s.t. } \|\boldsymbol{c}\|_0 \leq T, \tag{1}$$

This minimization problem can be efficiently solved by the OMP algorithm [48].

The performance of sparse approximation can be further improved by replacing the fixed dictionary with an adaptive one. Dictionary learning for sparse coding aims at finding such an optimal dictionary and can be generally formulated as the following optimization problem:

$$\underset{\boldsymbol{D},\boldsymbol{C}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{C}\|_F^2, \qquad \text{s.t. } \forall i,\ \|\boldsymbol{c}_i\|_0 \leq T, \tag{2}$$

where $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_P] \in \mathbb{R}^{N\times P}$ denotes a training set of P signals and $\boldsymbol{C} = [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_P] \in \mathbb{R}^{M\times P}$ are the corresponding sparse codes. In case of an infinite number of solutions due to unbounded optimization, normalization constraint on the norm of each atom (i.e., $\|\boldsymbol{d}_m\|_2 = 1, m = 1, 2, \ldots, M$) is often imposed. As the dictionary $\boldsymbol{D}$ in the objective function of (2) is only involved in the reconstruction error term $\|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{C}\|_F^2$, the minimization model in (2) is often referred to as reconstructive dictionary learning. This learning problem can be efficiently solved by many numerical algorithms, e.g. K-SVD [3].

2.2. *Discriminative sparse coding and supervised dictionary learning*

Sparse codes generated by adaptive dictionaries can be directly used as features to train classifiers for recognition. Such a two-stage scheme (i.e. dictionary learning followed by classifier training) has been extensively used in many existing methods (e.g. [7, 9, 33, 49]), but often the generated sparse codes are insufficiently discriminative for complex recognition tasks. One alternative (e.g. [20, 21, 25]) is to unify dictionary learning and classifier training in an optimization model. For example, the D-KSVD method [25] jointly learns a dictionary and a multi-class linear classifier as follows:

$$\begin{gathered}\underset{\boldsymbol{D},\boldsymbol{W},\boldsymbol{C}}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{C}\|_F^2 + \frac{\beta}{2}\|\boldsymbol{L}-\boldsymbol{W}\boldsymbol{C}\|_F^2 + \frac{\gamma}{2}\|\boldsymbol{W}\|_F^2, \\ \text{s.t.} \quad \forall i, \|\boldsymbol{c}_i\|_0 \leq T,\ \forall j, \|\boldsymbol{d}_j\|_2 = 1,\end{gathered} \tag{3}$$

where $\boldsymbol{W} \in \mathbb{R}^{K\times M}$ is the multi-class linear classifier to be learned, $\boldsymbol{L} = [\boldsymbol{l}_1, \ldots, \boldsymbol{l}_P] \in \mathbb{R}^{K\times P}$ is the binary label matrix of P training samples from K categories, and $\boldsymbol{l}_p = [0, 0, \ldots, 1, \ldots, 0]^\top \in \mathbb{R}^K$ is the binary label vector of sample $\boldsymbol{y}_p$ where nonzero occurs at the kth entry if $\boldsymbol{y}_p$ belongs to the kth category. In (3), the discriminative term $\|\boldsymbol{L}-\boldsymbol{W}\boldsymbol{C}\|_F^2$ is the linear prediction error produced by $\boldsymbol{W}$ and $\boldsymbol{C}$, which is used to induce discriminability on $\boldsymbol{C}$; the regularization term $||\boldsymbol{W}||_F^2$ is used to control the energy of $\boldsymbol{W}$.

To further enhance the discriminability of sparse codes, the LC-KSVD method [32] induces discriminative structures on sparse codes using the following dictionary learning model:

$$\underset{\boldsymbol{D},\boldsymbol{W},\boldsymbol{C}}{\operatorname{argmin}} \frac{1}{2}||\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{C}||_F^2 + \frac{\alpha}{2}||\boldsymbol{Q}-\boldsymbol{A}\boldsymbol{C}||_F^2 + \frac{\beta}{2}||\boldsymbol{L}-\boldsymbol{W}\boldsymbol{C}||_F^2 + \frac{\gamma}{2}||\boldsymbol{W}||_F^2, \qquad \text{s.t.} \quad \forall i, ||\boldsymbol{c}_i||_0 \leq T, \ \forall j, ||\boldsymbol{d}_j||_2 = 1, \tag{4}$$

where the term $||\boldsymbol{Q}-\boldsymbol{A}\boldsymbol{C}||_F^2$ is the so-called label consistency criterion, $\boldsymbol{A}$ is a linear transformation matrix that transform the sparse codes $\boldsymbol{C}$ to be the predefined discriminative form $\boldsymbol{Q} = [\boldsymbol{q}_1, \ldots, \boldsymbol{q}_P] \in \mathbb{R}^{M\times P}$, and $\boldsymbol{Q}$ is defined as a binary matrix in which nonzero occurs at the entry of mth row and pth column if the atom $\boldsymbol{d}_m$ is expected to share class labels with the signal $\boldsymbol{y}_p$. In other words, $\boldsymbol{q}_p$ is a binary vector with the form $\boldsymbol{q}_p = [0, 0, \ldots, 1, \ldots, 1, \ldots, 0, 0]^\top \in \mathbb{R}^M$ in which the nonzero values occur at the indices where the input signal $\boldsymbol{y}_p$ and the atoms share the same category label. For example, assuming $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_5]$, $\boldsymbol{D} = [\boldsymbol{d}_1, \ldots, \boldsymbol{d}_5]$, where $\boldsymbol{y}_1$, $\boldsymbol{y}_2$, $\boldsymbol{y}_3$, $\boldsymbol{d}_1$, $\boldsymbol{d}_2$ are from the 1st class and $\boldsymbol{y}_4$, $\boldsymbol{y}_5$, $\boldsymbol{d}_3$, $\boldsymbol{d}_4$, $\boldsymbol{d}_5$ are from the 2nd class, then the corresponding matrix $\boldsymbol{Q}$ is expressed as[4]

$$\boldsymbol{Q} \equiv \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

The minimization problems in (3) and (4) can be efficiently solved by applying the K-SVD algorithm [3] followed by some renormalization strategies on $\boldsymbol{D}$ and $\boldsymbol{W}$, in which the regularization term $||\boldsymbol{W}||_F^2$ is dropped and the energy of $\boldsymbol{W}$ is implicitly controlled by renormalization.

3. Our method

In this section, we introduce multiple classifier learning to supervised dictionary learning by defining a novel discriminative term based on multiple classifier system. An interesting relation between the proposed discriminative term and the label consistency regularization [32] is shown. Then built upon the proposed discriminative term, a

[4] In the implementation of LC-KSVD, the number of shared atoms in each category is the same and there is no overlap between the shared atoms from different categories. In other words, the atom are partitioned into disjoint groups of the same size.

supervised dictionary learning model is proposed. Finally an efficient numerical scheme is developed for solving the proposed model.

3.1. Ensemble discrimination for supervised dictionary learning

The single linear classifier used in D-KSVD is rather weak in terms of discrimination. Motivated by the success of the multiple classifier learning methods [41, 42, 46], we propose to learn multiple classifiers instead of a single one. Meanwhile, we argue that the subdictionaries of a discriminative dictionary should be also discriminative, implying that subsets of discriminative sparse codes should be also discriminative. Thus, we construct classifiers using different subsets of sparse codes. Such a construction scheme also facilitates the development of an efficient numerical solver, as shown in Section 3.3. The basic procedure of the construction is as follows. First, each dictionary atom is assigned to several predefined groups. Accordingly, the sparse codes associated with the same atom group are grouped together. Then on each group of sparse codes a multi-class linear classifier is applied, and the prediction errors of these classifiers are used to define the discriminative term.

Let $\{\mathcal{G}_z\}_{z=1}^{Z}$ be a predefined set of groups for grouping dictionary atoms and sparse codes. Each element of $\mathcal{G}_z$ is an integer within $[1, M]$. Define $\boldsymbol{P}_{\mathcal{G}_z}$ as the operator that draws the rows from $\boldsymbol{C}$ whose indices fall into $\mathcal{G}_z$, i.e., $\boldsymbol{P}_{\mathcal{G}_z}\boldsymbol{C} = \boldsymbol{C}_{[\mathcal{G}_z]}$. Then we define the discriminative ensemble error as

$$f(\boldsymbol{C}) = \sum_{z=1}^{Z} \frac{\beta_z}{2} \|\boldsymbol{L} - \boldsymbol{W}_z \boldsymbol{P}_{\mathcal{G}_z} \boldsymbol{C}\|_F^2, \tag{5}$$

where $\boldsymbol{W}_z$ is the multi-class linear classifier (often called a base classifier in the context of ensemble learning) learned from the zth group of sparse codes $\boldsymbol{P}_{\mathcal{G}_z}\boldsymbol{C}$. In other words, the discriminative ensemble error can be viewed as the weighted average linear prediction error over groups of sparse codes. The operators $\{\boldsymbol{P}_{\mathcal{G}_z}\}_z$ act as the subspace projectors for constructing ensembles. This is similar in spirit to the subspace ensemble methods [42, 46]. In our proposal, we project sparse codes into different subspaces with random subset selection, and on each subspace we apply a linear classifier during the dictionary learning process.

The discriminative ensemble error defined in (5) can be directly used as a discrimination term and incorporated into many existing supervised dictionary learning models. In this paper, based on the discriminative ensemble error, we construct a new dictionary learning model as follows:

$$\begin{aligned} \underset{\boldsymbol{D},\{\boldsymbol{W}_z\}_{z=1}^{Z},\boldsymbol{C}}{\operatorname{argmin}} \ & \frac{1}{2}\|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{C}\|_F^2 + \sum_{z=1}^{Z}\Big(\frac{\beta_z}{2}\|\boldsymbol{L}-\boldsymbol{W}_z\boldsymbol{P}_{\mathcal{G}_z}\boldsymbol{C}\|_F^2 + \frac{\gamma_z}{2}\|\boldsymbol{W}_z\|_F^2\Big), \\ & \text{s.t.} \quad \forall i, \|\boldsymbol{c}_i\|_0 \le T, \ \forall j, \|\boldsymbol{d}_j\|_2 = 1, \end{aligned} \tag{6}$$

where β_z and γ_z are the scalars controlling the relative contribution of each term, and the regularization term $\|\boldsymbol{W}_z\|_F^2$ is for controlling the energy of the classifier $\boldsymbol{W}_z$. An efficient numerical scheme for solving (6) is detailed in Section 3.3. For brevity, we refer to our method as *Multiple Classifiers based Dictionary Learning* (MCDL).

The effectiveness of our method can be interpreted from two perspectives. One is the power of using multiple classifiers. Compared with the methods [20, 25] using a single classifier, our method learns a set of classifiers, which is helpful to reduce the dependence of sparse codes on the peculiarities of training set and learn more expressive concepts. The other perspective is that subdictionaries of a discriminative dictionary should be also discriminative. By grouping dictionary atoms and strengthening discriminability of sparse codes within each group, our method encourages the sparse codes to be not only globally discriminative but also partially distinct. As a result, the overall discriminability of sparse codes is strengthened.

3.2. Relation with label consistency

It is obvious that when $Z = 1$ and $\boldsymbol{P}_{\mathcal{G}_1} = \boldsymbol{I}_M$, the discriminative ensemble error degenerates into global linear prediction error term used in [25, 28, 32]. In the next, we show an interesting connection between the proposed discriminative term defined in (5) and the label consistency term defined in (4). For this purpose, we rewrite the discriminative ensemble error as

$$f(\boldsymbol{C}) = \|\boldsymbol{B}(\mathbf{1}_Z \otimes \boldsymbol{L}) - \boldsymbol{B}\boldsymbol{W}\boldsymbol{P}\boldsymbol{C}\|_F^2\,, \tag{7}$$

where $\boldsymbol{P} = (\boldsymbol{P}_{\mathcal{G}_1}^\top, .., \boldsymbol{P}_{\mathcal{G}_Z}^\top)^\top$, $\boldsymbol{W} = diag(\boldsymbol{W}_1, \dots, \boldsymbol{W}_Z)$ is the block-diagonal matrix constructed by stacking $\boldsymbol{W}_1, \dots, \boldsymbol{W}_Z$ along diagonal, $\boldsymbol{B} = diag(\sqrt{\beta_1}\mathbf{1}_{|\mathcal{G}_1|}, \dots, \sqrt{\beta_Z}\mathbf{1}_{|\mathcal{G}_Z|})$ is a diagonal matrix with $[\sqrt{\beta_1}\mathbf{1}_{|\mathcal{G}_1|}^\top, \dots, \sqrt{\beta_Z}\mathbf{1}_{|\mathcal{G}_Z|}^\top]^\top$ as diagonal, and the operation $\otimes$ denotes Kronecker product. Now consider a more complex configuration of $\{\boldsymbol{P}_{\mathcal{G}_z}\}$ as follows. We set $Z = M/K$ and set $\beta_i = 1$ for all possible i (i.e. $\boldsymbol{B} = \boldsymbol{I}_M$).[5] We also set $\{\mathcal{G}_z\}_{z=1}^{Z}$ to satisfy: *(1)* $\bigcup_{z=1}^{Z}\mathcal{G}_z = \{1, \dots, M\}$;

[5] Without loss of generality, we suppose M is divisible by K and hence Z is an integer.

(2) $\forall i \neq j,\ \mathcal{G}_i \cap \mathcal{G}_j = \emptyset$. By definition, it is easy to verify that there exist two permutation matrices $\boldsymbol{R}$ and $\boldsymbol{S}$, such that $\boldsymbol{R}(\mathbf{1}_Z \otimes \boldsymbol{L}) = \boldsymbol{Q}$ and $\boldsymbol{SP} = \boldsymbol{I}_M$. After some simple substitutions and using $\boldsymbol{RR}^\top = \boldsymbol{I}_P$ and $\boldsymbol{SS}^\top = \boldsymbol{I}_M$, we have

$$f(\boldsymbol{C}) = \left\|\boldsymbol{R}^\top\boldsymbol{Q} - \boldsymbol{WS}^\top\boldsymbol{C}\right\|_F^2 = \left\|\boldsymbol{Q} - \boldsymbol{RWS}^\top\boldsymbol{C}\right\|_F^2, \tag{8}$$

which is actually equivalent to the label consistency term $\|\boldsymbol{Q} - \boldsymbol{AC}\|_F^2$ in the LC-KSVD model defined in (4) by setting $\boldsymbol{AS} = \boldsymbol{RW}$. In other words, the LC-KSVD training model in (4) can be reinterpreted as a subspace ensemble learning based model - sparse codes are projected into different subspaces and then an ensemble of classifiers is constructed from a set of linear classifiers, each of which is trained on some subspace of sparse codes.[6] In comparison with the LC-KSVD model, the proposed model has more flexibility in designing the subspace projections and determining the weights of base classifiers.

3.3. Optimization

Solving the minimization problem of (6) is nontrivial. Unlike the cases of the D-KSVD and LC-KSVD methods, in our method the K-SVD algorithm cannot be directly applied to solving (6) due to the presence of matrices $\{\boldsymbol{P}_{\mathcal{G}_z}\}_{z=1}^Z$. Intuitively, we need to alternately estimate the unknown variables $\boldsymbol{D}$, $\boldsymbol{C}$ and $\{\boldsymbol{W}_z\}_{z=1}^Z$ one at a time while fixing others. But in the next it is shown that by exploiting the structures of $\boldsymbol{W}_z\boldsymbol{P}_{\mathcal{G}_z}$, we can simultaneously update $\boldsymbol{D}$ and $\{\boldsymbol{W}_z\}_{z=1}^Z$. This is more efficient than the alternating optimization between three submodules (e.g. [20]) and largely reduces the potential of getting stuck at local minima of the subproblems.

To derive an efficient solver, we drop the regularization term $\sum_{z=1}^Z \frac{\gamma_z}{2}\|\boldsymbol{W}_z\|_F^2$ in (6) and control the energy of $\boldsymbol{W}_z$ by post-renormalization, which is the same strategy used in the D-KSVD and LC-KSVD methods. Then the MCDL optimization model can be written as

$$\underset{\boldsymbol{D},\boldsymbol{C},\{\boldsymbol{W}_z\}_{z=1}^Z}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{UC}\|_F^2, \quad \text{s.t. } \forall i, \|\boldsymbol{c}_i\|_0 \leq T,\ \forall j, \|\boldsymbol{u}_j\|_2 = 1, \tag{9}$$

where

[6] Our discussion is focused on the training model of LC-KSVD, regardless of how to use the learned classifiers in the latter stage.

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{Y} \\ \sqrt{\beta_1}\boldsymbol{L} \\ \cdots \\ \sqrt{\beta_Z}\boldsymbol{L} \end{bmatrix} \quad \text{and} \quad \boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M] = \begin{bmatrix} \boldsymbol{D} \\ \sqrt{\beta_1}\boldsymbol{W}_1\boldsymbol{P}_{\mathcal{G}_1} \\ \cdots \\ \sqrt{\beta_Z}\boldsymbol{W}_Z\boldsymbol{P}_{\mathcal{G}_Z} \end{bmatrix}.$$

We solve the minimization problem of (9) with an alternative iteration scheme. At the beginning of the $(\ell + 1)$th iteration, $\boldsymbol{D}$ and $\{\boldsymbol{W}_z\}_{z=1}^{Z}$ are fixed, and the calculation of $\boldsymbol{C}$ becomes the classical sparse coding problem

$$\boldsymbol{C}^{(\ell+1)} = \underset{\boldsymbol{C}}{\operatorname{argmin}} \frac{1}{2}||\boldsymbol{X} - \boldsymbol{U}^{(\ell)}\boldsymbol{C}||_F^2, \qquad \text{s.t. } \forall i, ||\boldsymbol{c}_i||_0 \leq T, \tag{10}$$

which is solved by the OMP algorithm [48].

After $\boldsymbol{C}^{(\ell+1)}$ is calculated, we simultaneously update $\boldsymbol{D}$ and $\{\boldsymbol{W}_z\}_{z=1}^{Z}$. To this end, we rewrite $\boldsymbol{u}_j$ as

$$\boldsymbol{u}_j = \begin{bmatrix} \boldsymbol{u}_{j,0} \\ \boldsymbol{u}_{j,1} \\ \cdots \\ \boldsymbol{u}_{j,Z} \end{bmatrix},$$

where $\boldsymbol{u}_{j,0} = \boldsymbol{d}_j$ and $\boldsymbol{u}_{j,z}$ is the jth column of $\sqrt{\beta_z}\boldsymbol{W}_z\boldsymbol{P}_{\mathcal{G}_z}$ for $z = 1, \ldots, Z$. Note that $\boldsymbol{W}_z\boldsymbol{P}_{\mathcal{G}_z}$ is actually the matrix constructed by inserting $\boldsymbol{0}_K$s into $\boldsymbol{W}_z$ as columns at the indices that fall into $\bar{\mathcal{G}}_z = \{1, 2, \ldots, M\} - \mathcal{G}_z$. In other words, we have $\boldsymbol{u}_{j,z} = \boldsymbol{0}_K$ if $j \in \bar{\mathcal{G}}_z$ and otherwise $\boldsymbol{u}_{j,z}$ is the $(r^{-1}(j, \mathcal{G}_z))$-th column of $\sqrt{\beta_z}\boldsymbol{W}_z$, where function $r(j, \mathcal{G})$ returns the jth smallest element in $\mathcal{G}$ and $r^{-1}(j, \mathcal{G}) = \#\{x \in \mathcal{G} : x \leq j\}$. Define $\Omega_j \subseteq \{N + 1, N + 2, \ldots, N + ZK\}$ as the set of indices of zeros from such $\boldsymbol{0}_K$s in $\boldsymbol{u}_j$, and define $\boldsymbol{V}_\Omega$ as the projection matrix such that $\boldsymbol{V}_\Omega\boldsymbol{u}$ projects $\boldsymbol{u}$ onto Ω.[7] Then the update of $\boldsymbol{D}$ and $\{\boldsymbol{W}_z\}_{z=1}^{Z}$ can be reformulated as the minimization w.r.t. $\boldsymbol{U}$ as follows:

$$\boldsymbol{U}^{(\ell+1)} = \underset{\boldsymbol{U}}{\operatorname{argmin}} \frac{1}{2}||\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{C}^{(\ell+1)}||_F^2, \qquad \text{s.t. } \boldsymbol{U} \in \mathcal{U}, \tag{11}$$

where $\mathcal{U} = \{\boldsymbol{U} \in \mathbb{R}^{(N+KZ)\times M} : ||\boldsymbol{u}_j||_2 = 1, \boldsymbol{V}_{\Omega_j}\boldsymbol{u}_j = \boldsymbol{0}, j = 1, \ldots, M\}$ denotes the feasible set for $\boldsymbol{U}$.

The minimization in (11) is solved by the proximal gradient method. Let $I_{\mathcal{U}}(\boldsymbol{U})$ be the indicator function of $\boldsymbol{U}$ such that $I_{\mathcal{U}}(\boldsymbol{U}) = 0$ if $\boldsymbol{U} \in \mathcal{X}$ and $+\infty$ otherwise. Then $\boldsymbol{U}^{(\ell+1)} = [\boldsymbol{u}_1^{(\ell+1)}, \cdots, \boldsymbol{u}_M^{(\ell+1)}]$ is updated column by column as

[7] Note that Ω_j is determined by $\{\mathcal{G}_z\}_{z=1}^{Z}$. Thus Ω_j and $\boldsymbol{V}_{\Omega_j}$ can be precomputed for acceleration.

follows:

$$\boldsymbol{u}_j^{(\ell+1)} \in \mathrm{Prox}_{\mu_j^{(\ell)}}^{I(\hat{\boldsymbol{U}}_j^{(\ell)})}(\boldsymbol{u}_j^{(\ell)} - \frac{1}{\mu_j^{(\ell)}} \nabla_{\boldsymbol{u}_j} Q(\boldsymbol{C}^{(\ell+1)}, \widetilde{\boldsymbol{U}}^{(\ell)})), \tag{12}$$

where $\mu_j^{(\ell)}$ determines the step size, $Q(\boldsymbol{C}, \boldsymbol{U}) = \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{C}\|_F^2$, $\mathrm{Prox}_t^F(\boldsymbol{x})$ is the proximal operator defined as

$$\mathrm{Prox}_t^F(\boldsymbol{x}) := \underset{\boldsymbol{u}}{\mathrm{argmin}}\, F(\boldsymbol{u}) + \frac{t}{2}\|\boldsymbol{u} - \boldsymbol{x}\|_F^2, \tag{13}$$

and

$$\begin{cases} \hat{\boldsymbol{U}}_j^{(\ell)} = [\boldsymbol{u}_1^{(\ell+1)}, \cdots, \boldsymbol{u}_{j-1}^{(\ell+1)}, \boldsymbol{u}_j, \boldsymbol{u}_{j+1}^{(\ell)}, \cdots, \boldsymbol{u}_M^{(\ell)}]; \\ \widetilde{\boldsymbol{U}}_j^{(\ell)} = [\boldsymbol{u}_1^{(\ell+1)}, \cdots, \boldsymbol{u}_{j-1}^{(\ell+1)}, \boldsymbol{u}_j^{(\ell)}, \boldsymbol{u}_{j+1}^{(\ell)}, \cdots, \boldsymbol{u}_M^{(\ell)}]. \end{cases} \tag{14}$$

By direct calculation, the minimization problem of (12) is equivalent to

$$\boldsymbol{u}_j^{(\ell+1)} = \underset{\substack{\|\boldsymbol{u}_j\|_2 = 1 \\ \boldsymbol{V}_{\Omega_j}\boldsymbol{u}_j = \boldsymbol{0}}}{\mathrm{argmin}}\, \frac{1}{2}\|\boldsymbol{u}_j - \boldsymbol{s}_j^{(\ell)}\|_2^2, \tag{15}$$

where $\boldsymbol{s}_j^{(\ell)} = \boldsymbol{u}_j^{(\ell)} - \frac{1}{\mu_j^{(\ell)}}\nabla_{\boldsymbol{u}_j} Q(\boldsymbol{C}^{(\ell+1)}, \widetilde{\boldsymbol{U}}_j^{(\ell)})$. This minimization problem has a closed-form solution

$$\boldsymbol{u}_j^{(\ell+1)} = (\boldsymbol{I} - \boldsymbol{V}_{\Omega_j})\boldsymbol{s}_j^{(\ell)} / \|(\boldsymbol{I} - \boldsymbol{V}_{\Omega_j})\boldsymbol{s}_j^{(\ell)}\|_2. \tag{16}$$

After several iterations, we finally obtain the optimal solution $\bar{\boldsymbol{U}}$ for (9), which is column-wise normalized and can be directly decomposed into $\bar{\boldsymbol{D}}$, $\{\bar{\boldsymbol{W}}_z\}_{z=1}^Z$ and several $\boldsymbol{0}_K s$. Then the final learned dictionary $\boldsymbol{D}$ and base classifiers $\{\boldsymbol{W}_z\}_{z=1}^Z$ are computed by renormalization.

$$\begin{aligned} &\boldsymbol{D} = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \cdots, \boldsymbol{d}_M\} = \left\{\frac{\bar{\boldsymbol{d}}_1}{\|\bar{\boldsymbol{d}}_1\|_2}, \frac{\bar{\boldsymbol{d}}_2}{\|\bar{\boldsymbol{d}}_2\|_2}, \cdots, \frac{\bar{\boldsymbol{d}}_M}{\|\bar{\boldsymbol{d}}_M\|_2}\right\}; \\ &\boldsymbol{W}_z = \{\boldsymbol{w}_{z,1}, \boldsymbol{w}_{z,2}, \cdots, \boldsymbol{w}_{z,|\mathcal{G}_z|}\} = \left\{\frac{\bar{\boldsymbol{w}}_{z,1}}{\|\bar{\boldsymbol{d}}_{r(1,\mathcal{G}_z)}\|_2}, \frac{\bar{\boldsymbol{w}}_{z,2}}{\|\bar{\boldsymbol{d}}_{r(2,\mathcal{G}_z)}\|_2}, \cdots, \frac{\bar{\boldsymbol{w}}_{z,|\mathcal{G}_z|}}{\|\bar{\boldsymbol{d}}_{r(|\mathcal{G}_z|,\mathcal{G}_z)}\|_2}\right\}, \quad \forall z \in \{1, \cdots, Z\}. \end{aligned} \tag{17}$$

3.4. Classification Scheme

Once the dictionary $\boldsymbol{D}$ and the base classifiers $\{\boldsymbol{W}_z\}_{z=1}^{Z}$ have been learned, given a test sample $\boldsymbol{y}_{test}$, we compute the corresponding sparse code $\boldsymbol{c}_{test}$ by solving

$$\underset{\boldsymbol{c}}{\operatorname{argmin}} \|\boldsymbol{y}_{test} - \boldsymbol{D}\boldsymbol{c}\|_2^2, \quad \text{s.t. } \|\boldsymbol{c}\|_0 \leq T, \tag{18}$$

with the OMP algorithm [48]. Then the label of $\boldsymbol{c}_{test}$ is predicted by taking the class index which corresponds to the maximal prediction score computed by applying $\{\boldsymbol{W}_z\}_{z=1}^{Z}$ to $\boldsymbol{c}_{test}$, i.e.

$$\text{label}(\boldsymbol{y}_{test}) = \underset{i}{\operatorname{argmax}}\, l_i, \tag{19}$$

where $\boldsymbol{l} = [l_1, \ldots, l_K]$ is the prediction score vector (unnecessarily binary) generated by

$$\boldsymbol{l} = \sum_{z=1}^{Z} \beta_z \boldsymbol{W}_z \boldsymbol{P}_{\mathcal{G}_z} \boldsymbol{c}_{test}. \tag{20}$$

3.5. Initialization and configuration

The MCDL method requires an elaborative initialization of $\boldsymbol{D}$ and $\{\boldsymbol{W}_z\}_{z=1}^{Z}$. The initialization is crucial because the MCDL model is non-convex due to the existence of ℓ_0 norm and the alternating optimization. For the initialization of $\boldsymbol{D}$, several iterations of K-SVD is run within each category and all the output dictionary atoms are collected as $\boldsymbol{D}^{(0)}$. The base classifiers $\{\boldsymbol{W}_z\}_{z=1}^{Z}$ are initialized by solving

$$\underset{\boldsymbol{W}_z}{\operatorname{argmin}} \frac{\beta_z}{2}\|\boldsymbol{L} - \boldsymbol{W}_z \boldsymbol{P}_{\mathcal{G}_z} \boldsymbol{C}\|_F^2 + \frac{\gamma_z}{2}\|\boldsymbol{W}_z\|_F^2, \tag{21}$$

which is the ridge regression with explicit solution

$$\boldsymbol{W}_z^{(0)} = \boldsymbol{L}\boldsymbol{C}^\top \boldsymbol{P}_{\mathcal{G}_z}^\top \left(\boldsymbol{P}_{\mathcal{G}_z} \boldsymbol{C}\boldsymbol{C}^\top \boldsymbol{P}_{\mathcal{G}_z}^\top + \frac{\gamma_z}{\beta_z} \boldsymbol{I}_{|\mathcal{G}_z|}\right)^{-1}. \tag{22}$$

The performance of MCDL largely depends on the construction of multiple classifiers. There are plenty of ways to set up the groups $\{\mathcal{G}_z\}_{z=1}^{Z}$. Followings are some examples:

- *Single group:* $Z = 1;\ \mathcal{G} = \{1, \ldots, M\};$
- *Disjoint groups:* $\bigcup_{z=1}^{Z} \mathcal{G}_z = \{1, \ldots, M\};\ \forall i \neq j,\ |\mathcal{G}_i| = |\mathcal{G}_j|,\ \mathcal{G}_i \cap \mathcal{G}_j = \varnothing;$
- *Sharing groups:* $\bigcup_{z=1}^{Z} \mathcal{G}_z = \{1, \ldots, M\};\ \exists \mathcal{S} \neq \varnothing, \mathcal{S} \subseteq \{1, \ldots, M\}, \forall i \neq j,\ |\mathcal{G}_i| = |\mathcal{G}_j|,\ \mathcal{G}_i \cap \mathcal{G}_j = \mathcal{S}.$

The first setting is a trivial case which corresponds to the global linear prediction cost in (3). The second setting corresponds to disjoint groups, and as described in Section 3.2, it can be viewed as a label consistency regularization. The last setting allows group overlap, where a few dictionary atoms are reused by classifiers. From the perspective of label consistency, it allows some dictionary atoms to be associated with multiple categories, which is similar in spirit to the methods [30, 36, 37] that learn both class-specific dictionaries and a shared dictionary.

4. Experiment

4.1. Protocol

There are various methodologies employed in existing literature for evaluating discriminative dictionary learning methods. We adopted the experimental protocol of [32], as it covers a variety of recognition tasks, including face recognition, object classification, scene classification, and action recognition. The data used for these experiments are available online.[8] In addition, we applied our method to classifying static and dynamic textures. The characteristics of all the benchmark datasets for our evaluation are summarized in Table 1, and the experimental protocols on the datasets are presented in following subsections. The methods for comparison mainly include

- one of the most popular ℓ_0 norm based dictionary learning methods - KSVD [3],[9]
- the sparse representation based classification method - SRC [8],
- four sparsity-based supervised dictionary learning methods, including D-KSVD [25], LC-KSVD [32], Joint [20], and DLSI [26],[10]
- a dictionary learning method based on locality instead of sparsity of codes - LLC [33].

[8] http://www.umiacs.umd.edu/~zhuolin/projectlcksvd.html

[9] A two-step strategy described in Section 1.1 is employed for classification.

[10] In [32], two versions of LC-KSVD are presented. We select the improved version for comparison. The parameter setting and all the results are consistent with [32].

Note that the KSVD, SRC and LLC methods are three representative reconstructive dictionary learning methods with successful results in image recognition and restoration. Other compared methods are all discriminative dictionary learning methods that are closely-related to ours and have yielded state-of-the-art results. Both these methods and ours view dictionary atoms as discriminative features and use sparse code as higher-level representation of signals for classification. Also note that the dictionary used in the original SRC method is much larger than other compared methods, as SRC stacks all the training samples as a dictionary. In order to have a fair evaluation, we also include a reduced version of SRC (denoted by SRC*) for comparison, whose dictionary size is set the same as ours. In addition to the aforementioned methods, we also include some classical task-specific methods for comparison.

Table 1: Characteristics of the datasets for experimental evaluation. The columns from left to right are the name of datasets, number of data samples, number of categories, number of training samples, number of test samples, and number of experiment repetitions.

Dataset	#Sample	#Class	#Training	#Test	#Repetition
Ext. YaleB	2414	38	1216	1198	30
AR Face	2600	100	2000	600	30
Scene-15	4485	15	1500	2985	30
Caltech-101	9144	102	(5:5:30)×102	–	10
UCF Action	150	10	140 / 120	10 / 30	30
ALOT	25000	250	(1/4 1/2) × 25000	(3/4 1/2) × 25000	30
Dyntex++	3600	100	1800	1800	30

For simplicity, we set the weight of each classifier to be the same, i.e., $\beta_z = \beta$ for all possible z, where β is a predefined scalar for weighting the discriminative terms. Then, the parameters of our method are reduced to three scalars: the discrimination weight β, the sparsity degree T, and the dictionary size M. In our experiments, β is determined by cross-validation, M is set to be a multiple of the number of categories in each dataset (i.e. M is divisible by K), and T is set according to the complexity of dataset. The values of these parameters for each dataset are presented in the following subsections respectively.

According to the group settings presented in Section 3.5, we adopted the following four configurations for grouping in our experiment, i.e.

- *G1*. A single group $\mathcal{G} = \{1, \ldots, M\}$ is used with $Z = 1$, i.e. $\boldsymbol{P}_{\mathcal{G}} = \boldsymbol{I}$.
- *G2*. Totally M/K disjoint groups of the same size are used. The sizes of all the groups are set to be K. The groups are generated by uniform partitioning on random permutation of integers $\{1, \ldots, M\}$.

- *G3*. Totally M/K sharing groups of the same size are used. The number of the atom indices shared by all the groups is set to be M/K (i.e., $\mathcal{G}_i \cap \mathcal{G}_j = \mathcal{S}$ for all $i \neq j$, and $|\mathcal{S}| = M/K$). To generate the groups, M/K atoms indices are randomly picked up as $\mathcal{S}$ for sharing. Then the remaining indices are randomly partitioned to M/K subsets with equal sizes. Each subset is united with the shared indices $\mathcal{S}$ to form the group.

- *G4*. Totally M/K sharing groups of the same size are used. The number of the atom indices shared by all the groups is set to be $2M/K$. The generation of the groups is similar to that of G3.

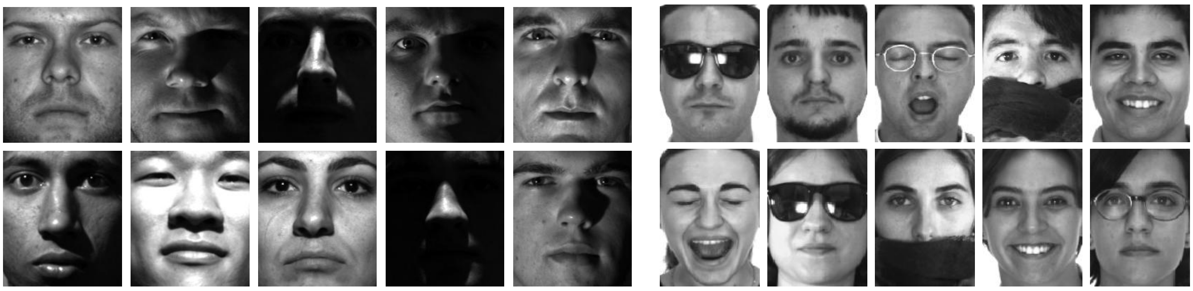
We generated the groups according to these four configurations G1, G2, G3, and G4 respectively. The final $\{\mathcal{G}_z\}_z$ is the union of all the generated groups above. In practice, it is observed that the performance of MCDL is insensitive to the randomness of the generation of $\{\mathcal{G}_z\}_z$.

4.2. Face recognition

Face recognition is one widely-studied recognition problem with applications ranging from checking identities at international borders and searching mugshots in national criminal databases to tagging faces in photos on social media websites. We demonstrate the effectiveness of our method in face recognition with two popular benchmark datasets:

- The extended YaleB dataset [50] contains 2414 images of 38 human frontal faces with different illumination conditions and expressions, as shown in Figure 1 (a). There are about 64 images for each person. The original images were cropped to 192×168 pixels. As done in [25], each face image is projected into a 504-dimensional feature vector using a random matrix of zero-mean normal distribution. The dataset is randomly split into two halves. One half with 32 images per person is used for training and the other half is used for test. The parameters of our method on this dataset are set as follows: $\beta = 2.7 \times 10^{-3}$, $T = 50$, and $M = 570$.

- The AR face dataset [51] consists of over 4000 frontal images from 126 individuals, in which 26 pictures were taken in two separate sessions for each individual. The main characteristic of the AR dataset is that it includes frontal views of faces with different facial expressions, lighting conditions and occlusions, as shown in Figure 1 (b). Following the standard evaluation procedure [25, 28, 32], we draw a subset from the original dataset. The resulting dataset consists of 2600 images from 50 male subjects and 50 female subjects. We randomly pick up 20 images from each person for training and the remaining images for test. Each face image is cropped to 165 × 120 and then reduced to be a 540-dimensional feature vector by random projection as above. The parameters of our method on this dataset are set as follows: $\beta = 2.5 \times 10^{-1}$, $T = 40$, and $M = 500$.



(a) Extended YaleB

(b) AR face

Figure 1: Some sample images from two face datasets. (a) The extended YaleB dataset; (b) The AR face dataset.

Table 2: The recognition accuracies (%) of the compared methods on two face datasets.

Dataset	KSVD [3]	SRC [8]	SRC* [8]	D-KSVD [25]	LC-KSVD [32]	LLC [33]	Joint [20]	DLSI [26]	MCDL
Ex.YaleB	93.10	97.02	80.50	94.10	95.00	90.70	93.88	89.00	95.79 ± 0.72
AR Face	86.50	97.50	68.50	88.80	93.70	88.70	88.24	89.80	95.21 ± 1.20

The experimental results on these two datasets are summarized in Table 2. It can be seen that our method outperforms all the compared methods except SRC. It is worth mentioning that the SRC method stacks all training samples as a big dictionary, which is computationally infeasible for real applications. Moreover, the performance of the SRC method decreases dramatically with the dictionary size reduced. When the dictionaries are set in the same size, the SRC* method performs worse than our method. This is not surprising as the SRC method could be roughly viewed as a generalized nearest subspace classifier, whose discriminative power largely depends on the number of training samples.

4.3. Object classification

Identifying objects in images is an interesting but very challenging task in computer vision. We tested the effectiveness of our method in object classification on the Caltech-101 [52] dataset, which consists of 8677 images from 101 object categories and 467 images from an additional background category. The number of images in each category is greatly unbalanced, varying from 31 to 800, and significant shape and appearance variabilities are presented in each category. See Figure 2 for some sample images from the Caltech-101 dataset.

Figure 2: Sample images from the Caltech-101 dataset.

The 3000-dimensional SIFT-based spatial pyramid feature used in [28, 32, 53] is extracted from each image and used as the input of all the compared methods. Following the common experimental protocol, we randomly pick up 5, 10, 15, 20, 25, and 30 samples per category for training and test on the remaining samples. This process is repeated 10 times with different splits of training and test set and finally the average classification accuracy is reported. The dictionary size M is set proportional to the size of training set, i.e. 510, 1020, 1530, 2040, 2550, and 3060 respectively. The parameter T is set to 45 and β is 5×10^{-4}.

The methods for comparison are the same as in Section 4.2. The results are shown in Figure 3. It can be observed that our approach performs the best among all the compared methods, no matter how large the training set is.

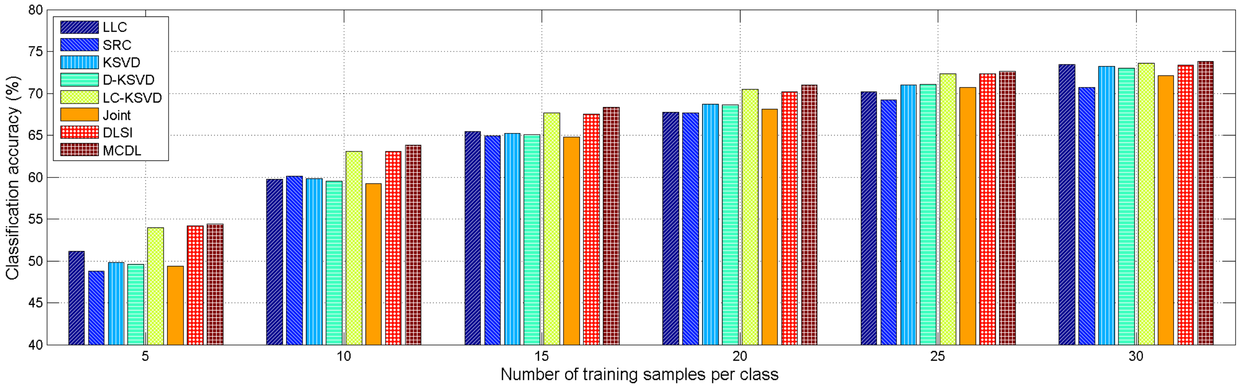
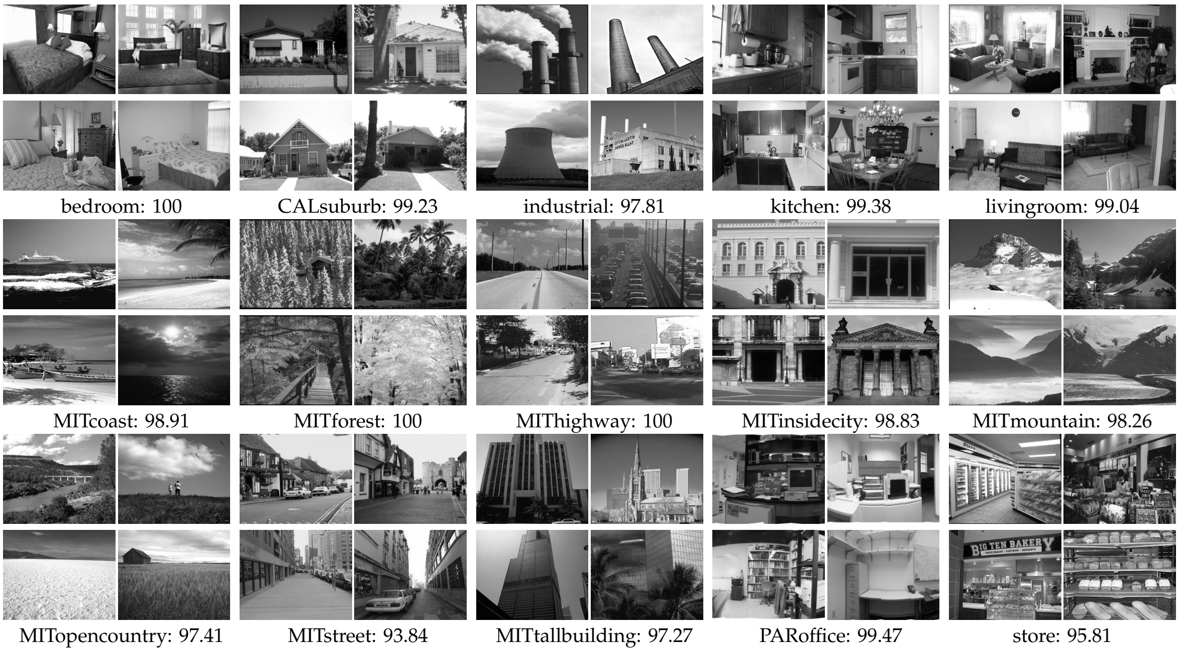


Figure 3: Classification accuracies (%) of the compared methods using different sizes of training set on the Caltech-101 dataset.

4.4. Scene classification

The ability of computer to distinguish scenes is very useful, as it can serve to provide priors for the presence of actions, surfaces and objects, as well as their locations and scales. We applied our method to scene classification and evaluated the performance on the Scene-15 dataset [53]. The Scene-15 dataset contains 15 scene categories, including bedroom, suburb, industrial, kitchen, living room, coast, forest, highway, inside city, mountain, open country, street, tall building, office, and store. The number of images per category varies from 210 to 410. The resolution of each image is about 250 × 300. See Figure 4 for the sample images from each category in the dataset.



bedroom: 100
CALsuburb: 99.23
industrial: 97.81
kitchen: 99.38
livingroom: 99.04
MITcoast: 98.91
MITforest: 100
MIThighway: 100
MITinsidecity: 98.83
MITmountain: 98.26
MITopencountry: 97.41
MITstreet: 93.84
MITtallbuilding: 97.27
PARoffice: 99.47
store: 95.81

Figure 4: Per-class classification accuracy (%) by the proposed method on the Scene-15 dataset.

Similar to the case in Caltech-101, the 3000-dimensional SIFT-based spatial pyramid features [28, 32, 53] extracted from the images are used as the input of all the compared methods. Same as the standard experimental protocol used in [28, 32, 53], from each category 100 images are randomly picked up for training and the rest for test. For the configuration of parameters, we set $\beta = 6 \times 10^{-4}$, $T = 55$ and $M = 450$.

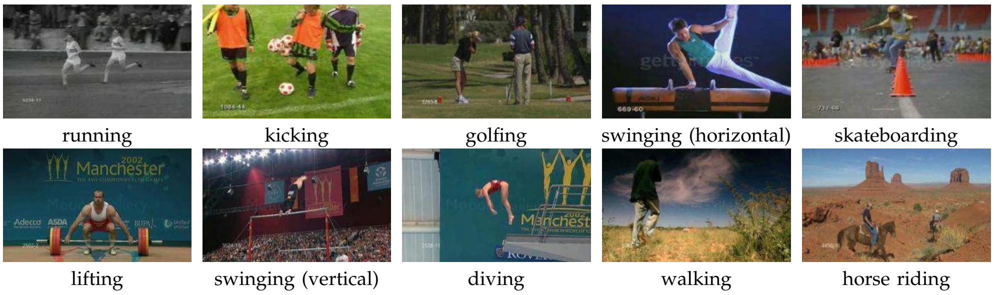
Besides the compared methods used in the previous subsections, several state-of-the-art scene classification methods [9, 10, 40, 49, 53, 54] are also included for comparison. Table 3 summarizes the experimental results. It can be seen that MCDL outperforms other compared methods. The classification accuracy achieved by our method on each category is shown in Figure 4.

Table 3: Classification accuracies (%) of the compared methods on the Scene-15 dataset.

Method	Accuracy	Method	Accuracy	Method	Accuracy
Lazebnik et al. [53]	81.40 ± 0.50	Boureau et al. [49]	84.30 ± 0.50	Joint [20]	88.20
Gemert et al. [54]	76.67 ± 0.39	SRC [8]	91.80	DLSI [26]	92.46
Yang et al. [9]	80.28 ± 0.93	SRC* [8]	77.62	LLC [33]	89.20
Gao et al. [10]	89.75 ± 0.50	KSVD [3]	86.70	LC-KSVD [32]	92.90
Lian et al. [40]	86.43 ± 0.41	D-KSVD [25]	89.10	MCDL	97.35 ± 0.31

4.5. Action recognition

Action recognition refers to the process of labeling videos or images that contains human motion with action categories. Interactive applications like human-computer interaction and games benefit from the advances in automatic action recognition. To evaluate the performance of our method in action recognition, the UCF sport dataset [55] was selected, which is composed of 150 video sequences from 10 action categories collected from various broadcast sport channels such as BBC and ESPN. The action categories include running, kicking, golfing, swinging (horizontal), swinging (vertical), skateboarding, lifting, diving, walking, and horse riding, as shown in Figure 5. The number of samples in each category varies from 14 to 35.



running kicking golfing swinging (horizontal) skateboarding

lifting swinging (vertical) diving walking horse riding

Figure 5: Key frames of selected samples from the UCF sport dataset.

The 100-dimensional reduced action bank feature [56] is adopted to represent each video. The recognition performance is measured by two kind of schemes. One is the leave-one-video-out evaluation scheme, in which one video from each class is collected for test and the remaining videos for training. This scheme has been used in many evaluations of sparse dictionary learning methods, e.g. [28, 32]. The other scheme is the five-fold cross validation, in which one fold is used for test and the remaining four folds for training. This scheme has been widely used in a lot of literature on action recognition, e.g. [55–60]. The dictionary size of MCDL is set 70 for the leave-one-video-out scheme and 50 for the five-fold cross validation scheme, and for both cases the parameters β and T are set to 0.5 and 10 respectively.

The performance comparison is shown in Table 4. It can be seen that for both evaluation schemes, our approach performs the best among all the compared methods. Note that the D-KSVD method performed worse than SRC. The reason is that the single linear classifier used in D-KSVD is rather weak in terms of discrimination and has large dependence on the peculiarities of data especially when the number of training samples is small. This justifies the motivation of this paper to inject multiple classifier learning into supervised dictionary learning.

Table 4: Classification accuracies (%) on the UCF sports action dataset.

Method	Evaluation	Accuracy	Method	Evaluation	Accuracy
SRC [8]	fivefold cross	90.40	Qiu et al. [61]	fivefold cross	83.60
SRC* [8]	fivefold cross	80.62	Yao et al. [62]	fivefold cross	86.60
LLC [33]	fivefold cross	87.50	Yeffet et al. [57]	leave-one-video-out	79.20
KSVD [3]	fivefold cross	86.80	Wu et al. [60]	leave-one-video-out	91.30
D-KSVD [25]	fivefold cross	88.10	Le et al. [59]	leave-one-video-out	86.50
Joint [20]	fivefold cross	86.00	Sadanand et al. [56]	leave-one-video-out	95.00
DLSI [26]	fivefold cross	88.74	Kovashka et al. [58]	leave-one-video-out	87.30
LC-KSVD [32]	fivefold cross	91.20	LC-KSVD [32]	leave-one-video-out	95.70
MCDL	fivefold cross	91.65 ± 0.24	MCDL	leave-one-video-out	95.90 ± 0.11

4.6. *Texture classification*

Understanding visual textures, either static or dynamic, is fundamental to many computer vision and image processing tasks such as scene classification, video understanding, visual retrieval and image-guided diagnosis. We tested the performance of our approach in both the static and dynamic texture classification tasks with two datasets:

- The ALOT dataset [63] is a large static texture dataset, encompassing 25000 texture images from 250 classes. Its texture categories range from hand-made textures to natural textures, and from regular textures to random textures. See Figure 6 for some examples of the dataset. The 700-dimensional multi-scale LBP histogram is extracted from each texture image for classification. We trained our model with a quarter of and a half of the samples respectively, and the rest are used for test. The dictionary size of MCDL is set 1080 and the parameters β and T are set to 1×10^{-3} and 5 respectively.

- The DynTex++ dataset [65] contains 3600 dynamic texture video sequences from 36 categories. The categories of dynamic textures on this dataset range from waves on beach to vehicle traffic on road. See Figure 7 for the snapshots of the dataset. The 177-dimensional LBP-TOP histogram [64] is extracted from each dynamic texture sequence for classification. One half of the samples are used for training and the other half are for test. The dictionary size of MCDL is set 1080 and the parameters β and T are set to 1×10^{-5} and 25 respectively.

Figure 6: Sample images from the ALOT dataset.

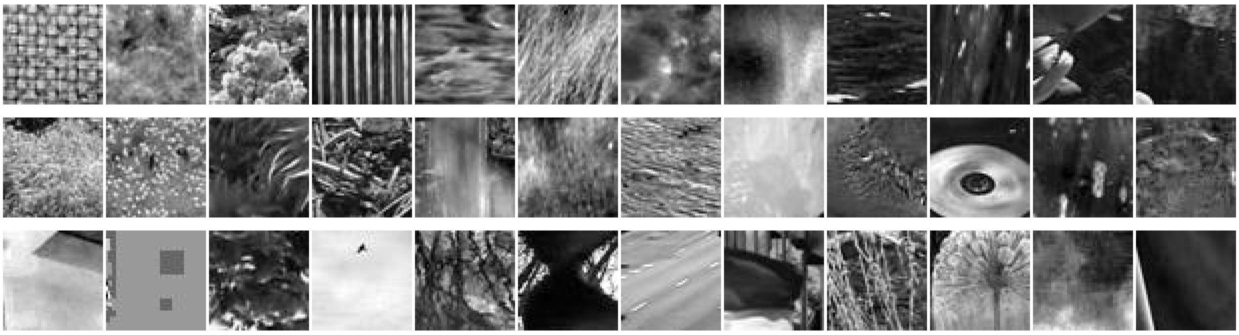


Figure 7: Key frames of selected samples from the DynTex++ dataset.

The results are listed in Table 5 and Table 6. In addition to the comparison with several sparse coding and dictionary learning methods including SRC*, KSVD, Joint, D-KSVD and LC-KSVD, we compare our method with the representative method [66] in the static texture classification and the state-of-the-art methods [64, 65, 67] in the dynamic texture classification. The dictionary sizes and sparsity degrees of all the compared dictionary learning methods

Table 5: Classification accuracies (%) on the ALOT dataset.

# Training sample	SRC* [8]	KSVD [3]	Joint [8]	D-KSVD [25]	LC-KSVD [32]	Xu et al. [66]	MCDL
25	83.38 ± 0.34	84.14 ± 0.35	84.22 ± 0.42	84.71 ± 0.46	85.05 ± 0.34	82.12 ± 0.38	85.85 ± 0.32
50	89.45 ± 0.33	89.88 ± 0.46	90.02 ± 0.37	90.22 ± 0.35	90.52 ± 0.42	86.64 ± 0.36	91.78 ± 0.36

Table 6: Classification accuracies (%) on the DynTex++ dataset.

SRC* [8]	KSVD [3]	Joint [8]	D-KSVD [25]	LC-KSVD [32]	Ghanem et al. [65]	Zhao et al. [64]	Xu et al. [68]	MCDL
88.53 ± 0.83	89.31 ± 0.58	89.40 ± 0.57	89.27 ± 0.56	89.67 ± 0.50	63.70	89.80	89.90	90.35 ± 0.66

are set the same as ours. It can be seen from Table 5 that in the static texture classification our method outperforms other compared methods and shows noticeable performance improvement over [66] which employs a more complicated feature extraction process instead of the simple LBP histograms. On this dataset, we also tested the ensemble linear classifier (ELC) constructed from random subspace ensemble using the same number of base classifiers as ours. The experimental results show ELC is inferior to MCDL with a performance gap of 12.2% classification accuracy when using half of samples for training. This demonstrates that the higher-level representations from sparse coding are helpful to the performance improvement of ensemble classifiers. From Table 6 we can see that our method performs the best among all the compared approaches in the dynamic texture classification. To further understand the performance of our method, we conduct the t-test analysis on the classification results, and the results show that our method significantly outperforms other compared methods.

4.7. Influence of components and parameters

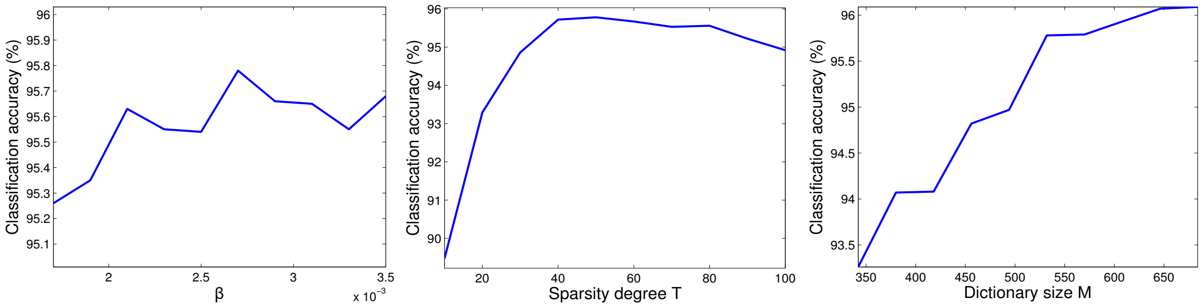
To analyze the contribution of each component in our method, we tested the performance of MCDL using different combinations of the four group configurations (i.e. G1, G2, G3 and G4). The results on the extended YaleB dataset are listed in Table 7. It can be seen that using single group configuration yields reasonable but insufficient discrimination. With more group configurations added, the performance of MCDL increases. This not only verified the necessity of using the four group configurations to generate classifiers, but also demonstrated the power of multiple classifier learning in discriminative sparse coding.

To analyze the influence of the discrimination parameter β, the sparsity parameter T and the dictionary size M in

Table 7: The recognition accuracies (%) of MCDL on the extended YaleB dataset using different group configurations.

G1	G2	G3	G4	G1+G2	G1+G2+G3	G1+G2+G4	G1+G2+G3+G4
94.20	94.56	94.35	94.25	95.02	95.64	95.60	95.79

MCDL, we conducted a test on the extended YaleB dataset by alternatively fixing two of the parameters and adjusting the rest one. The classification results corresponding to different values of the parameters are plotted in Figure 8. It can be seen from Figure 8 (a) that the performance of MCDL is not sensitive to β within a small range. As β increases, the discrimination of sparse codes is improved while the representative power of the learned dictionary is degraded. Thus, the best choice of β for classification is to strike the balance of discrimination and representation. In Figure 8 (b), the performance of MCDL drops much when T is small. This is because using too few dictionary atoms cannot well capture the characteristics and variation of data, making the sparse codes insufficiently discriminative. When T is larger than 40, the performance of MCDL decreases slightly. The reason is that using too many dictionary atoms for representation might cause the over-fitting problem. From Figure 8 (c) we can see that the classification accuracy increases as the dictionary becomes larger. But the increment gets small when the dictionary is sufficiently large.



(a) (b) (c)

Figure 8: Classification accuracies (%) versus the parameters β, T and M on the extended YaleB dataset.

4.8. Efficiency

In order to evaluate the computational efficiency of our approach, we compared our method with several baseline methods in terms of the average running time during the training phase and the test stage in different applications.

More specifically, for each tested method, both the average training time per iteration during dictionary learning and the average test time for an input sample during classification are reported. To have a fair comparison, all the tested methods are implemented under the same computational environment. The software environment is the MATLAB 2013a platform run on the Windows 7 operating system, and the hardware environment is a PC with Intel Dual-Core i7-3770 3.4GHz CPU and 32GB memory. We compared our method with three closely-related methods: D-KSVD, LC-KSVD, and SRC. The results are listed in Table 8.[11] It can be seen that in training MCDL is slower than LC-KSVD but faster than D-KSVD. Although on average the training time of MCDL is about twice as much as that of LC-KSVD, the time cost of MCDL is still acceptable. The test time of MCDL is on a par with D-KSVD and LC-KSVD, and is much less than SRC. It can be also seen that the scalability of MCDL is much better than SRC, as SRC cannot scale well to large dictionary (i.e. a large number of training samples) in terms of computational time.

Table 8: Training time (seconds per iteration) and test time (milliseconds per sample) of the tested methods on six datasets.

Dataset	Training time (s) per iteration			Test time (ms) per sample			
Name	D-KSVD	LC-KSVD	MCDL	D-KSVD	LC-KSVD	SRC	MCDL
Ext. YaleB	2.34	0.80	1.76	0.10	0.26	30.94	0.29
AR Face	2.57	1.16	2.12	0.06	0.24	79.25	0.27
Scene-15	20.84	3.18	5.04	0.33	0.33	183.51	0.36
Caltech-101	70.91	36.48	71.52	1.70	1.70	769.66	1.90
UCF Action	0.12	0.01	0.02	0.04	0.03	0.47	0.07
Dyntex++	4.64	1.83	3.09	0.31	0.31	26.03	0.35

5. Conclusions

Aiming at enhancing discriminability in sparse codes, in this paper, we proposed an effective and efficient supervised dictionary learning method for sparse coding by integrating multiple classifier learning into dictionary learning. The advantages of the proposed method over existing approaches are multi-fold: better discriminability of sparse codes, weaker dependence on peculiarities of training data, and more expressibility of classifier for prediction. We also provided an interesting insight into label consistency from the perspective of multiple classifier learning by showing its relation with the proposed discriminative term. We applied the proposed method to several image classification

[11] Note that LC-KSVD is faster than D-KSVD because the implementation of LC-KSVD is based on the approximated K-SVD algorithm which is more efficiently than the exact K-SVD algorithm used in the original implementation of D-KSVD. Also note that the SRC method does not require training and thus the training time of SRC is omitted.

tasks to demonstrate its great potential in applications of pattern recognition. The experimental results showed that our method is very competitive in terms of both classification accuracy and computational efficiency. In the future, we would like to investigate the exploitation of more multiple classifier learning techniques in supervised dictionary learning, such as boosting, bagging, and random projection.

Acknowledgement. Yuping Sun would like to thank the support by China Scholarship Council Program. Yong Xu would like to thank the support by National Nature Science Foundations of China (61273255, 61211130308 and 61070091), Fundamental Research Funds for the Central Universities (SCUT 2013ZG0011) and GuangDong Technological innovation project (2013KJCX0010).

[1] Karthikeyan Natesan Ramamurthy, Jayaraman J Thiagarajan, Prasanna Sattigeri, and Andreas Spanias. Ensemble sparse models for image analysis. *arXiv preprint arXiv:1302.6957*, 2013.

[2] Zhenfeng Zhu, Qian Chen, and Yao Zhao. Ensemble dictionary learning for saliency detection. *Image and Vision Computing*, 32(3):180–188, 2014.

[3] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[4] Jianchao Yang, John Wright, Thomas Huang, and Yi Ma. Image super-resolution as sparse representation of raw image patches. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[5] Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen. Blind motion deblurring from a single image using sparse approximation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 104–111. IEEE, 2009.

[6] Michael Elad, Mario AT Figueiredo, and Yi Ma. On the role of sparse and redundant representations in image processing. *Proceedings of the IEEE*, 98(6):972–982, 2010.

[7] Ke Huang and Selin Aviyente. Sparse representation for signal classification. In *Advances in Neural Information Processing Systems*, pages 609–616, 2006.

[8] John Wright, Allen Y Yang, Arvind Ganesh, Shankar S Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[9] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801. IEEE, 2009.

[10] Shenghua Gao, Ivor Waihung Tsang, Liang-Tien Chia, and Peilin Zhao. Local features are not lonely–laplacian sparse coding for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3555–3561. IEEE, 2010.

[11] Meng Yang, D Zhang, and Jian Yang. Robust sparse coding for face recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 625–632. IEEE, 2011.

[12] Yangmuzi Zhang, Zhuolin Jiang, and Larry S Davis. Learning structured low-rank representations for image classification. In *Proceedings*

of IEEE Conference on Computer Vision and Pattern Recognition, pages 676–683. IEEE, 2013.

[13] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Multipath sparse coding using hierarchical matching pursuit. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 660–667. IEEE, 2013.

[14] Shenghua Gao, Liang-Tien Chia, and Ivor W Tsang. Multi-layer group sparse coding for concurrent image classification and annotation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2809–2816. IEEE, 2011.

[15] Julien Mairal, Francis Bach, and Jean Ponce. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):791–804, 2012.

[16] Julien Mairal, Francis Bach, and Jean Ponce. Sparse modeling for image and vision processing. *Foundations and Trends in Computer Graphics and Vision*, 8(2-3):85–283, December 2014.

[17] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.

[18] Chenglong Bao, Hui Ji, Yuhui Quan, and Zuowei Shen. l0 norm based dictionary learning by proximal methods with global convergence. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3858–3865, 2014.

[19] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Discriminative learned dictionaries for local image analysis. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[20] Duc-Son Pham and Svetha Venkatesh. Joint learning and dictionary construction for pattern recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[21] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R Bach. Supervised dictionary learning. In *Advances in Advances in Neural Information Processing Systems*, pages 1033–1040, 2009.

[22] Wei Zhang, Akshat Surve, Xiaoli Fern, and Thomas Dietterich. Learning non-redundant codebooks for classifying complex objects. In *Proceedings of International Conference on Machine Learning*, pages 1241–1248. ACM, 2009.

[23] Xiao-Chen Lian, Zhiwei Li, Changhu Wang, Bao-Liang Lu, and Lei Zhang. Probabilistic models for supervised dictionary learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2305–2312. IEEE, 2010.

[24] Jianchao Yang, Kai Yu, and Thomas Huang. Supervised translation-invariant sparse coding. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3517–3524. IEEE, 2010.

[25] Qiang Zhang and Baoxin Li. Discriminative K-SVD for dictionary learning in face recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2698. IEEE, 2010.

[26] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3501–3508. IEEE, 2010.

[27] Meng Yang, David Zhang, and Xiangchu Feng. Fisher discrimination dictionary learning for sparse representation. In *Proceedings of IEEE International Conference on Computer Vision*, pages 543–550. IEEE, 2011.

[28] Zhuolin Jiang, Zhe Lin, and Larry S Davis. Learning a discriminative dictionary for sparse coding via label consistent K-SVD. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1697–1704. IEEE, 2011.

[29] Ravishankar Sivalingam, Daniel Boley, Vassilios Morellas, and Nikolaos Papanikolopoulos. Positive definite dictionary learning for region covariances. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1013–1019. IEEE, 2011.

[30] Ning Zhou, Yi Shen, Jinye Peng, and Jianping Fan. Learning inter-related visual dictionary for object recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3490–3497. IEEE, 2012.

[31] Zhuolin Jiang, Guangxiao Zhang, and Larry S Davis. Submodular dictionary learning for sparse coding. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3418–3425. IEEE, 2012.

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

[33] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3360–3367. IEEE, 2010.

[34] Francis R Bach. Bolasso: Model consistent lasso estimation through the bootstrap. In *Proceedings of the international conference on Machine learning*, pages 33–40. ACM, 2008.

[35] Haoran Wang, Chunfeng Yuan, Weiming Hu, and Changyin Sun. Supervised class-specific dictionary learning for sparse modeling in action recognition. *Pattern Recognition*, 45(11):3902–3911, 2012.

[36] Shu Kong and Donghui Wang. A dictionary learning approach for classification: Separating the particularity and the commonality. In *Proceedings of European Conference on Computer Vision*, pages 186–199. Springer, 2012.

[37] Donghui Wang and Shu Kong. A classification-oriented dictionary learning model: Explicitly learning the particularity and commonality across categories. *Pattern Recognition*, 47(2):885–898, 2014.

[38] Shenghua Gao, IW Tsang, and Yi Ma. Learning category-specific dictionary and shared dictionary for fine-grained image categorization. *IEEE Transactions on Image Processing*, 23(2):623–634, 2014.

[39] Rodolphe Jenatton, Julien Mairal, Francis R Bach, and Guillaume R Obozinski. Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of International Conference on Machine Learning*, pages 487–494, 2010.

[40] Xiao-Chen Lian, Zhiwei Li, Bao-Liang Lu, and Lei Zhang. Max-margin dictionary learning for multiclass image categorization. In *Proceedings of European Conference on Computer Vision*, pages 157–170. Springer, 2010.

[41] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[42] Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[43] Marina Skurichina and Robert PW Duin. Bagging, boosting and the random subspace method for linear classifiers. *Pattern Analysis & Applications*, 5(2):121–135, 2002.

[44] Marina Skurichina and Robert PW Duin. Bagging for linear classifiers. *Pattern Recognition*, 31(7):909–930, 1998.

[45] P Jonathon Phillips, Patrick J Flynn, Todd Scruggs, Kevin W Bowyer, Jin Chang, Kevin Hoffman, Joe Marques, Jaesik Min, and William Worek. Overview of the face recognition grand challenge. In *IEEE Conference on Computer vision and pattern recognition*, volume 1, pages 947–954. IEEE, 2005.

[46] Robert Bryll, Ricardo Gutierrez-Osuna, and Francis Quek. Attribute bagging: Improving accuracy of classifier ensembles by using random feature subsets. *Pattern recognition*, 36(6):1291–1302, 2003.

[47] Dacheng Tao, Xiaoou Tang, Xuelong Li, and Xindong Wu. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(7):1088–1099, 2006.

[48] Yagyensh Chandra Pati, Ramin Rezaiifar, and PS Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of Asilomar Conference on Signals, Systems and Computers*, pages 40–44. IEEE, 1993.

[49] Y-L Boureau, Francis Bach, Yann LeCun, and Jean Ponce. Learning mid-level features for recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2559–2566. IEEE, 2010.

[50] Athinodoros S. Georghiades, Peter N. Belhumeur, and David Kriegman. 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, 2001.

[51] Aleix M Martinez. The AR face database. *CVC Technical Report*, 24, 1998.

[52] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[53] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178. IEEE, 2006.

[54] Jan C van Gemert, Jan-Mark Geusebroek, Cor J Veenman, and Arnold WM Smeulders. Kernel codebooks for scene categorization. In *Proceedings of European Conference on Computer Vision*, pages 696–709. Springer, 2008.

[55] Rodriguez Mikel, Ahmed Javed, and Shah Mubarak. Action mach: A spatio-temporal maximum average correlation height filter for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008.

[56] Sreemananth Sadanand and Jason J Corso. Action bank: A high-level representation of activity in video. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1234–1241. IEEE, 2012.

[57] Lahav Yeffet and Lior Wolf. Local trinary patterns for human action recognition. In *Proceedings of IEEE International Conference on Computer Vision*, pages 492–497. IEEE, 2009.

[58] Adriana Kovashka and Kristen Grauman. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2046–2053. IEEE, 2010.

[59] Quoc V Le, Will Y Zou, Serena Y Yeung, and Andrew Y Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3361–3368. IEEE, 2011.

[60] Xinxiao Wu, Dong Xu, Lixin Duan, and Jiebo Luo. Action recognition using context and appearance distribution features. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 489–496. IEEE, 2011.

[61] Qiang Qiu, Zhuolin Jiang, and Rama Chellappa. Sparse dictionary-based representation and recognition of action attributes. In *Proceedings of IEEE International Conference on Computer Vision*, pages 707–714. IEEE, 2011.

[62] Angela Yao, Juergen Gall, and Luc Van Gool. A hough transform-based voting framework for action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2061–2068. IEEE, 2010.

[63] Gertjan J Burghouts and Jan-Mark Geusebroek. Material-specific adaptation of color invariant features. *Pattern Recognition Letters*, 30(3):306–313, 2009.

[64] Guoying Zhao and Matti Pietikainen. Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):915–928, 2007.

[65] Bernard Ghanem and Narendra Ahuja. Maximum margin distance learning for dynamic texture recognition. In *European Conference on Computer Vision*, pages 223–236. Springer, 2010.

[66] Yong Xu, Hui Ji, and Cornelia Fermüller. Viewpoint invariant texture description using fractal analysis. *International Journal of Computer Vision*, 83(1):85–100, 2009.

[67] Hui Ji, Xiong Yang, Haibin Ling, and Yong Xu. Wavelet domain multifractal analysis for static and dynamic texture classification. *Image Processing, IEEE Transactions on*, 22(1):286–299, 2013.

[68] Yong Xu, Yuhui Quan, Haibin Ling, and Hui Ji. Dynamic texture classification using dynamic fractal analysis. In *International Conference on Computer Vision*, pages 1219–1226. IEEE, 2011.