Sparse Coding for Classification via Discrimination Ensemble*

Yuhui Quan[1,3], Yong Xu[1], Yuping Sun[2,3], Yan Huang[1,3], and Hui Ji[3]
[1]School of Computer Science & Engineering, South China Univ. of Tech., Guangzhou 510006, China
[2]School of Automation Science & Engineering, South China Univ. of Tech., Guangzhou 510006, China
[3]Department of Mathematics, National University of Singapore, Singapore 117542
{csyhquan@scut.edu.cn, yxu@scut.edu.cn, ausyp@scut.edu.cn, matjh@nus.edu.sg}

Abstract

Discriminative sparse coding has emerged as a promising technique in image analysis and recognition, which couples the process of classifier training and the process of dictionary learning for improving the discriminability of sparse codes. Many existing approaches consider only a simple single linear classifier whose discriminative power is rather weak. In this paper, we proposed a discriminative sparse coding method which jointly learns a dictionary for sparse coding and an ensemble classifier for discrimination. The ensemble classifier is composed of a set of linear predictors and constructed via both subsampling on data and subspace projection on sparse codes. The advantages of the proposed method over the existing ones are multi-fold: better discriminability of sparse codes, weaker dependence on peculiarities of training data, and more expressibility of classifier for classification. These advantages are also justified in the experiments, as our method outperformed several recent methods in several recognition tasks.

1. Introduction

In recent years, as a promising technique for efficiently representing high-dimensional data, sparse coding has seen its successful usages in a variety of recognition tasks, *e.g.*, face recognition [31, 36, 3], object classification [32, 17, 3], texture classification [26, 25], and action recognition [8, 39]. Given a set of input data, sparse coding aims at expressing each input data by a linear combination of only a few elements from a set of representative patterns. These representative patterns are called *atoms*, the set of all the atoms is called *dictionary*, and the coefficients of the linear combinations are called *sparse codes*. More specifically, consider a set of input signals $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_P\} \subset \mathbb{R}^N$, sparse coding is about determining a set of atoms $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \dots, \boldsymbol{d}_M\} \subset \mathbb{R}^N$, together with a set of coding vectors $\{\boldsymbol{c}_1, \dots, \boldsymbol{c}_P\} \subset \mathbb{R}^M$, so that each input vector $\boldsymbol{y}_j$ can be approximated by the linear combination $\boldsymbol{y}_j \approx \sum_{\ell=1}^{M} \boldsymbol{c}_j(\ell)\boldsymbol{d}_\ell$, where most entries of $\boldsymbol{c}_j$ are zeros or close to zeros. Let $\|\cdot\|_0$ denote the pseudo-norm that counts the number of non-zero elements. Then, the classic sparse coding problem can be formulated as the following optimization problem (e.g. [1]):

$$\min_{\boldsymbol{D},\boldsymbol{C}} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{C}\|_F^2, \quad \text{s.t. } \forall i,\ \|\boldsymbol{c}_i\|_0 \le T, \tag{1}$$

where $\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \dots, \boldsymbol{d}_M] \in \mathbb{R}^{N\times M}$ denotes the dictionary to be learned, $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_P] \in \mathbb{R}^{N\times P}$ denotes a matrix containing the input samples as column vectors, $\boldsymbol{C} = [\boldsymbol{c}_1, \dots, \boldsymbol{c}_P] \in \mathbb{R}^{M\times P}$ denotes the matrix containing the corresponding coding vectors, and the parameter T controls the sparsity degree on each coding vector. Furthermore, the normalization constraint on each atom is often imposed to avoid possible unbounded solutions, which states $\|\boldsymbol{d}_j\|_2 = 1$ for all j.

It can be seen that the dictionary learned using (1) only cares about the approximation error between the input data and the resultant succinct expression. In other words, the sparse codes obtained under the learned dictionary can be viewed as the cleaned up version of the input data. One may use such sparse codes as the features for classification. However, the additional discriminative information provided by these sparse codes over the original input signals is limited when being used in complex classification tasks, as they do not take account of the discriminability needed in classification. In recent years, there have been an abundant literature on discriminative sparse coding which is to learn a dictionary whose resultant sparse codes possess improved discriminative power; see e.g. [21, 22, 20, 33, 26, 15]. The basic idea of discriminative sparse coding for classification is to include some supervised learning processes into sparse coding. Most existing approaches for discriminative sparse

*The corresponding author Yong Xu thanks the supports by National Nature Science Foundations of China (61273255 and 61070091), Engineering & Technology Research Center of Guangdong Province for Big Data Analysis and Processing (2013-1589-1-11), Project of High Level Talents in Higher Institution of Guangdong Province (2013-2050205-47) and Guangdong Technological Innovation Project (2013KJCX0010).

coding consider the following variational model:

$$\min_{\boldsymbol{D},\boldsymbol{C}} \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{C}\|_F^2 + \gamma\mathcal{J}(\boldsymbol{C};\boldsymbol{L}) \tag{2}$$

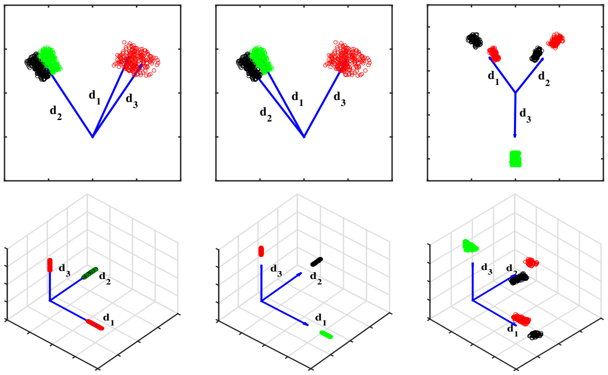
subject to $\|\boldsymbol{c}_i\|_0 \leq T$, $\|\boldsymbol{d}_j\|_2 \leq 1$, for all i and j, where γ is a weight, $\boldsymbol{L}$ is a matrix that encodes the label information of each training sample, and $\mathcal{J}(\cdot;\boldsymbol{L})$ denotes a penalty function that measures the discriminative error between the codes and labels.

1.1. Motivation

In recent years, several supervised learning techniques have been incorporated into discriminative sparse coding, such as linear prediction in [24, 37, 13], softmax regression in [21, 22], logistic regression in [22, 20], and maximum margin learning in [19, 33]. All these techniques focus on feeding back classification performance of a single classifier, which is rather rudimentary considering the great advances in supervised learning in recent years. For example, the powerful *ensemble learning* [6], a machine learning paradigm where a set of base classifiers are trained and combined as an ensemble classifier to gain extra performance, has not been fully exploited.

The benefits of introducing ensemble learning to sparse coding are multi-fold. Firstly, the size of training data is often limited in real applications, which could be due to the cost of data collection (*e.g.* face images [31]) or the computational cost of using a training set of large size (*e.g.* classifying objects of over thousands of categories [7]). As a result, the dictionary learning, as well as the single classifier training, is often sensitive to the shape of training data. Ensemble learning allows the combination of multiple classifiers which can effectively reduce such sensitivity. Secondly, when using a single classifier, the applicability of discriminative sparse coding is often limited owing to the imperfectness of the used learning algorithms, *e.g.*, the linear classifier used in [24, 37, 14] is inappropriate for linearly inseparable data, and the fisher discriminant used in [35, 34] is optimal only when the data from each category are realized from the normal distribution. In contrast, using ensemble learning can avoid such imperfectness by integrating multiple classifiers. Lastly, the hypothesis space being searched might not contain the true target function, while ensembles can give some good approximation [6].

Motivated by the likely benefits of ensemble learning over single classifier training, there have been several attempts to incorporate ensemble learning into discriminative sparse coding; see *e.g.* [38, 40, 41]. However, there are plenty of room for further improvement in all these methods in both theoretical and applied perspectives. For example, the supervised information is not fully utilized in [41], a two-stage scheme used in [40] does not feed back classification performance for dictionary learning, and an iterative re-sampling scheme is directly used in [38] for learning multiple dictionaries and classifiers, which lacks an unified variational model. All these inspired us to study new variational approaches for ensemble learning based discriminative sparse coding. See Figure 1 for an illustration of our motivation to introduce ensemble learning to sparse coding.



(a) (b) (c)

Figure 1. Motivation of sparse coding with discriminative ensemble. (a) Reconstructive sparse coding only considers minimizing the reconstruction error. Thus, inter-class signals which have high correlations are likely to share atoms during dictionary leaning, which yields similar inter-class sparsity patterns and decreases the discrimination of sparse codes, *e.g.*, green points and black points are mixed together in the K-SVD [1] coding space. (b) Jointly learning a linear classifier could address the issue, as labels of signals are utilized to enforce the separability of inter-class sparsity patterns, *e.g.*, green points and black points are separated in the D-KSVD [37] coding space. (c) However, discrimination terms based on a single linear classifier are insufficient in many scenarios, as inter-class sparsity patterns are unnecessarily linearly separable due to the multi-modal distribution (*e.g.* black points or red points are distributed in two clusters in the D-KSVD [37] coding space), peculiarity and outliers of training data. On the other hand, using highly nonlinear classifier would result in complex optimization models that are challenging to solve. In contrast, integrating multiple linear classifiers can overcome the weakness of single linear classifier while keeping the simplicity of the model.

1.2. Main Contributions

This paper aims to develop a new discriminative sparse coding method which is built upon ensemble learning. We first construct a new variational model of the form (2) that embeds ensemble learning into sparse coding by considering an ensemble classifier in defining the term $\mathcal{J}$. Then, we propose an alternating iterative scheme to solve the resultant optimization problem. The proposed discriminative sparse coding method can be applied to classification by voting the predictions from all base classifiers in the ensemble. Compared to the classic sparse coding methods, *e.g.* the K-SVD method [1] and the proximal method [2, 3], the sparse codes from the proposed method are much more discriminative when being used in classification. Compared to the single classifier based discriminative sparse coding methods [24, 37, 14], the proposed method is built upon multiple classifiers in an ensemble setting, which is capable to tackle the insufficiency of training data and improve the robustness of classification. Compared to the methods that assign labels to atoms for adding discrimination [40, 13], the proposed method can be regarded as a generalization which projects the sparse codes to a set of subspaces and learns a classifier on each subspace, yielding a compact dictionary without explicit label assignment. In comparison to other existing ensemble-based dictionary learning methods [38, 40], the proposed method provides a variational model with better theoretical justification, and avoids learning multiple dictionaries for ensemble as done in [40, 41].

2. Discriminative Sparse Coding

Nowadays, sparse coding has emerged as one promising technique in a wide range of applications, including image recovery, analysis, and classification. In classic sparse modeling problem, the sparse coding aims at finding sparse representation of input data under an adaptive dictionary. There has been an abundant literature on its analysis and algorithms. For example, the SRC method [31] considers the sparse approximation problem with the dictionary constructed by concatenating all training samples. The well-known K-SVD method [1] considers the model (1) and provides a fast numerical solver, and the proximal method for solving the same problem is proposed in [2] with rigorous convergence analysis. All these methods only concern the sparse approximation of input data. The label information of the training samples in supervised setting are ignored. As a result, the obtained sparse codes often do not provide additional discriminative information over the input feature vectors when used for classification. Thus, many methods have been proposed to utilize the labels of data for discriminative sparse coding, *e.g.* [21, 37, 26, 35, 13]. In the next, we give a brief review on the existing discriminative sparse coding techniques, which can be mainly classified into two categories based on the usage of label information.

2.1. Joint dictionary learning and classifier training

There are two approaches for combining classification and sparse coding. One is a two-stage approach which first runs sparse coding and then uses the obtained sparse codes as the features to train classifiers; see *e.g.* [11, 32, 30]. Such a two-stage scheme is not optimal for discrimination as it does not relate classifiers to the process of sparse coding. Thus, a better approach is to simultaneously run classifier training and sparse coding, which often can be formulated as a variational model:

$$\begin{aligned}&\min_{\boldsymbol{D},\boldsymbol{W},\boldsymbol{C}} \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{C}\|_F^2+\gamma\mathcal{J}(\boldsymbol{C},\boldsymbol{W};\boldsymbol{L})\\ &\text{s.t.}\quad \|\boldsymbol{c}_i\|_0\le T,\ \|\boldsymbol{d}_j\|_2\le 1,\ \text{for all } i,j,\end{aligned}\qquad(3)$$

where $\mathcal{J}(\cdot,\cdot;\boldsymbol{L})$ denotes a classification loss function, $\boldsymbol{W}$ denotes the classifier parameters related to $\mathcal{J}$, and $\boldsymbol{L}=[\boldsymbol{l}_1,\boldsymbol{l}_2,\dots,\boldsymbol{l}_P]\in\mathbb{R}^{K\times P}$ is the binary label matrix of P training samples from K categories, where $\boldsymbol{l}_p=[0,0,\dots,1,\dots,0,0]^\top\in\mathbb{R}^K$ denotes the binary label vector of the pth sample $\boldsymbol{y}_p$ in which nonzero occurs at the kth entry if $\boldsymbol{y}_p$ belongs to the kth category.

As a discriminative term, the classification loss function $\mathcal{J}(\boldsymbol{C},\boldsymbol{W};\boldsymbol{L})$ in (3) varies in different methods - softmax discriminative cost [21, 22], linear prediction error [24, 37, 14, 13], hinge loss [19, 33], and logistic loss [22, 20], to name a few. Take the linear prediction error for example, the classification loss function $\mathcal{L}$ is defined as

$$\mathcal{J}(\boldsymbol{C},\boldsymbol{W};\boldsymbol{L})=\|\boldsymbol{L}-\boldsymbol{W}\boldsymbol{C}\|_F^2,\qquad(4)$$

where $\boldsymbol{W}\in\mathbb{R}^{K\times M}$ is a classic multi-class linear predictor. Such a simple discrimination term has demonstrated moderate performance improvement in face recognition [37]. But a global linear classifier is often still not powerful enough in many challenging classification tasks.

Most existing approaches solve the problem (3) via an alternating iteration scheme which alternatively updates the estimations in three submodules, *i.e.* sparse coding, dictionary learning and classifier training.

2.2. Associating dictionary atoms with class labels

The discriminability in sparse codes can be further improved by learning a dictionary with labeled atoms. More specifically, each dictionary atom is associated with one or more class labels. During the process of dictionary learning, each input signal is encouraged to have significant responses on the atoms whose class labels are shared with the signal. In this scheme, a dictionary is partitioned into several discriminative sub-dictionaries, and distinct sparsity patterns (*e.g.* positions or magnitude spectrum of non-zero elements) are induced in the sparse coefficients of inter-class signals, which is likely to increase the distance of sparse codes among different classes.

When they are disjoint and learned independently from inner-class samples, sub-dictionaries become naive class-specific dictionaries, which have been employed in many previous studies; see *e.g.* [21, 28]. The main drawback of these methods is that the learned class-specific dictionaries do not encode correlation between classes. On the one hand, the learned class-specific dictionary in each class might also represent data from other classes equally well, which results in decreased discriminative power of sparse codes. On the other hand, the samples from different classes do not share any dictionary atom, which makes the resultant representation less efficient in terms of characterizing the underlying structures. Several schemes have been proposed to tackle these issues - adding an additional globally shared pool of atoms [40, 16], reducing mutual coherence between atoms and detecting shared atoms among class-specific dictionaries [26], etc.

A promising alternative to using naive class-specific dictionaries is to jointly learn sub-dictionaries, *e.g.* [35]. To induce discriminability in sparse codes according to subdictionaries, one way is to group sparse codes according to the label consistency between dictionary atoms and data samples. Then group sparsity is imposed on the grouped sparse codes, *e.g.* [10, 12]. The resultant structured sparse codes are more discriminative for classification than the purely sparse ones. Another way is to induce separability in sparse codes with certain class separation criterion, see *e.g.* [35, 5]. Take the label consistency criterion [14, 13] for example, the discrimination term is defined as follows:

$$\mathcal{J}(\boldsymbol{C}, \boldsymbol{A}; \boldsymbol{B}) = \|\boldsymbol{B} - \boldsymbol{A}\boldsymbol{C}\|_F^2, \tag{5}$$

where $\boldsymbol{A}$ is a linear transformation matrix to be learned, $\boldsymbol{B} \in \mathbb{R}^{M\times P}$ is a predefined binary matrix for label consistency where $\boldsymbol{B}_{m,p}$ is nonzero if the atom $\boldsymbol{d}_m$ is expected to share class label with the signal $\boldsymbol{y}_p$.

3. Main Body

In this section, we develop an ensemble based discriminative sparse coding method for classification. Instead of learning a single linear classifier defined in (4), we train multiple linear classifiers based on different subspaces of sparse codes from different subsets of input signals during dictionary learning. By jointly learning a dictionary for sparse coding and training an ensemble classifier for classification, the benefits of the proposed method are two-fold: better discriminability of sparse coding and better robustness in classification.

3.1. Ensemble based discriminative sparse coding

Let $\{\boldsymbol{W}_z \in \mathbb{R}^{K\times M_z}\}_{z=1}^Z$ be a set of multi-class linear classifiers to be learned. We propose the following variational model for discriminative sparse coding:

$$\begin{aligned}\min_{\boldsymbol{D},\{\boldsymbol{W}_z\}_{z=1}^Z,\boldsymbol{C}} & \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{C}\|_F^2 + \sum_{z=1}^Z \gamma_z \|\boldsymbol{W}_z\|_F^2 + \\ & \sum_{z=1}^Z \beta_z \|\boldsymbol{L}\boldsymbol{Q}_z - \boldsymbol{W}_z\boldsymbol{P}_z\boldsymbol{C}\,\boldsymbol{Q}_z\|_F^2 \qquad (6) \\ \text{s.t.} \quad & \|\boldsymbol{c}_i\|_0 \le T, \ \|\boldsymbol{d}_j\|_2 \le 1, \ \text{for all } i, j,\end{aligned}$$

where β_zs and γ_zs are the scalars controlling relative contribution of each term, $\boldsymbol{P}_z \in \mathbb{R}^{M_z\times M}$ is a subspace ensemble constructor which projects coding vector of each sample (*i.e.* each column of $\boldsymbol{C}$) onto certain subspace, and $\boldsymbol{Q}_z \in \mathbb{R}^{P\times P}$ is a subsample ensemble constructor which selects coding vectors of certain samples (*i.e.* some columns of $\boldsymbol{C}$) for classification. There are three main terms in (6):

- The first term is a fidelity term for the consistency between signals and codes;
- The second term is a discrimination term built upon an ensemble of classifiers, where $\{\boldsymbol{P}_z\}_{z=1}^Z$ is used for constructing subspace ensemble while $\{\boldsymbol{Q}_z\}_{z=1}^Z$ for constructing subsample ensemble;
- The last term is to control the energy of the classifiers to avoid over-fitting.

Compared to the single linear classifier based approaches (e.g. [37]), by using the ensemble of linear classifiers, the proposed method is able to reduce the dependence of sparse codes on peculiarities of training set and learn more expressive concepts for further performance gain in classification.

Remark - *An interesting observation on the connection between label consistency and ensemble learning*. The label consistency term defined in (5) can be also understood from the viewpoint of ensemble learning. First, assuming each class shares label with H atoms and each atom only shares label with one class, it is easy to verify that there exists a permutation matrix $\boldsymbol{R}$ such that $\boldsymbol{R}(\boldsymbol{1}_Z \otimes \boldsymbol{L}) = \boldsymbol{B}$. Then we can rewrite (5) as

$$\mathcal{J}(\boldsymbol{C}, \bar{\boldsymbol{A}}; \boldsymbol{L}) = \sum_{h=1}^H \|\boldsymbol{L} - \bar{\boldsymbol{A}}_h\boldsymbol{C}\|_F^2, \tag{7}$$

where $\bar{\boldsymbol{A}}_h \in \mathbb{R}^{K\times P}$ is the hth block of $\bar{\boldsymbol{A}}$ which is defined as $\bar{\boldsymbol{A}} = [\bar{\boldsymbol{A}}_1, \bar{\boldsymbol{A}}_2, ..., \bar{\boldsymbol{A}}_H] = \boldsymbol{R}\boldsymbol{A}$. Thus, the label consistency term can be viewed as a discrimination term defined as the summation of prediction errors from a set of linear classifiers $\{\boldsymbol{A}_h\}_{h=1}^H$, which is a special case of the ensemble discrimination term in (6). Note that the base learners $\{\boldsymbol{A}_h\}_{h=1}^H$ learned in LC-KSVD are utilized in learning but not classification. In comparison, we utilize the base learners in both learning and classification for improvement.

3.2. Construction of ensemble classifier

We now give a detailed description on the implementation of the ensemble construction operators $\{\boldsymbol{P}_z\}_{z=1}^Z$ and $\{\boldsymbol{Q}_z\}_{z=1}^Z$ in (6). As suggested in [6], the correlation of each pair of base classifiers in the ensemble should be as low as possible for promising diversity and improvement. One often-used technique to form ensemble with independent bases is done by random injection. In this paper, we configure $\{\boldsymbol{P}_z\}_{z=1}^Z$ and $\{\boldsymbol{Q}_z\}_{z=1}^Z$ as follows[1]:

- *Identical projection*: Set $\boldsymbol{P}_1 = \boldsymbol{I}_M$ and $\boldsymbol{Q}_1 = \boldsymbol{I}_P$. This is an ordinary base which results in (4).
- *Feature selection*: For $z = 2, ..., H_1 + 1$, set $\boldsymbol{Q}_z = \boldsymbol{I}_P$ and set $\boldsymbol{P}_z \in \mathbb{R}^{K\times M}$ to be a feature selection matrix such that $\boldsymbol{P}_z\boldsymbol{C}$ selects K rows from $\boldsymbol{C}$. More specifically, $\boldsymbol{P}_z$ is a binary matrix with K nonzeros generated

[1] Recall that $M/P/K$ are the number of atoms/signals/categories.

by randomly deleting $M-K$ rows from $\boldsymbol{I}_M$. The row positions of 1s in $\boldsymbol{P}_z$ indicate the selected dimensions of sparse codes for training $\boldsymbol{W}_z$.

- *Random projection*: For $z = H_1+2, ..., H_2+H_1+1$, set $\boldsymbol{Q}_z = \boldsymbol{I}_P$ and set $\boldsymbol{P}_z \in \mathbb{R}^{\frac{M}{2}\times M}$ to be a random Gaussian matrix with zero mean. Compared to the feature selection, the random projection can guarantee a global preservation of inter-point distances.
- *Data subsampling*: For $z = H_2+H_1+2, ..., H_3+H_2+H_1+1$, set $\boldsymbol{P}_z = \boldsymbol{I}_M$ and set $\boldsymbol{Q}_z \in \mathbb{R}^{P\times P}$ to be a diagonal projection matrix that selects sparse codes from a subset of training samples from each class. More concretely, $\boldsymbol{Q}_z$ is a binary diagonal matrix where the pth diagonal element being 1 indicates that the pth signal is used for training $\boldsymbol{W}_z$.[2] The $\boldsymbol{Q}_z$s are generated by thresholding randomly permuted indices.

Empirically, the performance of our method is insensitive to the randomness from the generation schemes above.

3.3. Algorithm

We use an alternating iterative scheme to solve the problem (6), which alternatively updates the unknowns $\boldsymbol{D}$, $\boldsymbol{C}$ and $\{\boldsymbol{W}_z\}_{z=1}^{Z}$ as follows[3]: for $\ell = 1, 2, \ldots,$

$$\begin{cases} \boldsymbol{C}^{(\ell+1)} = \underset{\boldsymbol{C}}{\operatorname{argmin}} \sum_{z=1}^{Z} \beta_z \|\boldsymbol{L} - \boldsymbol{W}_z^{(\ell)}\boldsymbol{P}_z\boldsymbol{C}^{(\ell)}\boldsymbol{Q}_z\|_F^2 \\ \qquad\qquad + \|\boldsymbol{Y} - \boldsymbol{D}^{(\ell)}\boldsymbol{C}\|_F^2, \;\; \text{s.t. } \|\boldsymbol{c}_i\|_0 \le T \text{ for all } i; \\ \boldsymbol{D}^{(\ell+1)} = \underset{\boldsymbol{D}}{\operatorname{argmin}} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{C}^{(\ell)}\|_F^2, \;\; \text{s.t. } \|\boldsymbol{d}_j\|_2 = 1 \text{ for all } j; \\ \boldsymbol{W}_z^{(\ell+1)} = \underset{\boldsymbol{W}}{\operatorname{argmin}} \|\boldsymbol{L} - \boldsymbol{W}\boldsymbol{P}_z\boldsymbol{C}^{(\ell)}\boldsymbol{Q}_z\|_F^2 + \frac{\gamma_z}{\beta_z}\|\boldsymbol{W}\|_F^2. \end{cases}$$

3.3.1 Sparse approximation

At the beginning of the $(l+1)$th iteration, we update the sparse codes with the learned dictionary and classifiers from the previous step by solving the following problem:

$$\boldsymbol{C}^{(\ell+1)} = \underset{\boldsymbol{C}}{\operatorname{argmin}} \sum_{z=1}^{Z} \beta_z \|\boldsymbol{L} - \boldsymbol{W}_z^{(\ell)}\boldsymbol{P}_z\boldsymbol{C}\boldsymbol{Q}_z\|_F^2 \\ + \|\boldsymbol{Y} - \boldsymbol{D}^{(\ell)}\boldsymbol{C}\|_F^2 \quad \text{s.t. } \forall i, \; \|\boldsymbol{c}_i\|_0 \le T.$$

This problem is column separable with respect to $\boldsymbol{C}$. Thus, we update $\boldsymbol{C} = [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_P]$ column by column as follows: for $i = 1, \ldots, P$,

$$\boldsymbol{c}_i^{(\ell+1)} = \underset{\boldsymbol{c}}{\operatorname{argmin}} \sum_{\substack{z=1 \\ \boldsymbol{Q}_z(i)=1}}^{Z} \beta_z \|\boldsymbol{l}_i - \boldsymbol{W}_z^{(\ell)}\boldsymbol{P}_z\boldsymbol{c}\|_2^2 \\ + \|\boldsymbol{y}_i - \boldsymbol{D}^{(\ell)}\boldsymbol{c}\|_2^2, \quad \text{s.t. } \|\boldsymbol{c}\|_0 \le T \tag{8}$$

where $\boldsymbol{Q}_z(i)$ denotes the ith diagonal element of $\boldsymbol{Q}$. This problem can be rewritten as

$$\boldsymbol{c}_i^{(\ell+1)} = \underset{\boldsymbol{c}}{\operatorname{argmin}} \|\boldsymbol{x} - \boldsymbol{U}_i^{(\ell)}\boldsymbol{c}\|_F^2, \quad \text{s.t. } \|\boldsymbol{c}\|_0 \le T, \tag{9}$$

where $\boldsymbol{U}_i^{(\ell)} = (\boldsymbol{D}^{\top(\ell)}, \ldots, \sqrt{\beta_z}(\boldsymbol{W}_z^{(\ell)}\boldsymbol{P}_z)^\top, \ldots)^\top$ and $\boldsymbol{x} = (\boldsymbol{y}_i^\top, \ldots, \sqrt{\beta_z}\boldsymbol{l}_i^\top, \ldots)^\top$ for all possible zs subject to $Q_z(i) = 1$. This is a classic sparse coding problem which is solved by OMP [29].

3.3.2 Dictionary refinement

After the sparse codes have been updated, the refinement of dictionary becomes the following problem:

$$\boldsymbol{D}^{(\ell+1)} = \underset{\boldsymbol{D}}{\operatorname{argmin}} \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{C}^{(\ell)}\|_F^2, \text{ s.t. } \forall j, \; \|\boldsymbol{d}_j\|_2 = 1.$$

By applying projected gradient descent, we update the dictionary atom by atom as follows: for $j = 1, \ldots, M$,

$$\begin{cases} \boldsymbol{s}_j^{(\ell)} = \boldsymbol{d}_j^{(\ell)} - \frac{1}{\mu_j^\ell}\nabla_{\boldsymbol{d}_j}\mathcal{F}(\boldsymbol{C}^{(\ell+1)}, \widetilde{\boldsymbol{D}}_j^{(\ell)}; \boldsymbol{Y}), \\ \boldsymbol{d}_j^{(\ell+1)} = \underset{\|\boldsymbol{d}_j\|_2=1}{\operatorname{argmin}} \|\boldsymbol{d}_j - \boldsymbol{s}_j^{(\ell)}\|_2, \end{cases} \tag{10}$$

where μ_j^ℓ is the step size, $\mathcal{F}(\boldsymbol{C}, \boldsymbol{D}; \boldsymbol{Y}) = \|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{C}\|_F^2$,

$$\widetilde{\boldsymbol{D}}_j^{(\ell)} = [\boldsymbol{d}_1^{(\ell+1)}, \cdots, \boldsymbol{d}_{j-1}^{(\ell+1)}, \boldsymbol{d}_j^{(\ell)}, \boldsymbol{d}_{j+1}^{(\ell)}, \cdots, \boldsymbol{d}_m^{(\ell)}],$$

The problem (10) has a closed-form solution

$$\boldsymbol{d}_j^{(\ell+1)} = \boldsymbol{s}_j^{(\ell)} / \|\boldsymbol{s}_j^{(\ell)}\|_2. \tag{11}$$

3.3.3 Classifier training

With the sparse codes fixed, the training of classifiers is about solving the following problem:

$$\boldsymbol{W}_z^{(\ell+1)} = \underset{\boldsymbol{W}}{\operatorname{argmin}} \|\boldsymbol{L} - \boldsymbol{W}\boldsymbol{M}_z^{(\ell)}\|_F^2 + \frac{\gamma_z}{\beta_z}\|\boldsymbol{W}\|_F^2,$$

where $\boldsymbol{M}_z^{(\ell)} = \boldsymbol{P}_z\boldsymbol{C}^{(\ell)}\boldsymbol{Q}_z$. This is a ridge regression problem with the explicit solution given by

$$\boldsymbol{W}_z^{(\ell+1)} = \boldsymbol{L}\boldsymbol{M}_z^{(\ell)\top}(\boldsymbol{M}_z^{(\ell)}\boldsymbol{M}_z^{(\ell)\top} + \frac{\gamma_z}{\beta_z}\boldsymbol{I})^{-1}, \tag{12}$$

which can be efficiently computed by the conjugate gradient method as $(\boldsymbol{M}_z^{(\ell)}\boldsymbol{M}_z^{(\ell)\top} + \frac{\gamma_z}{\beta_z}\boldsymbol{I})$ is positive definite.

3.4. Classification strategy

Once the dictionary $\boldsymbol{D}$ and the classifiers $\{\boldsymbol{W}_z\}_{z=1}^{Z}$ have been learned, the classification is done as follows. Given a test sample $\boldsymbol{y}^{\text{test}}$, we compute the corresponding sparse code $\boldsymbol{c}^{\text{test}}$ by solving the sparse approximation problem

$$\boldsymbol{c}^{\text{test}} = \underset{\boldsymbol{c}}{\operatorname{argmin}} \|\boldsymbol{y}^{\text{test}} - \boldsymbol{D}\boldsymbol{c}\|_2^2, \quad \text{s.t. } \|\boldsymbol{c}\|_0 \le T, \tag{13}$$

[2] Setting $\boldsymbol{Q}_z$ rectangular instead of square is more succinct. However, we adopt the square case for the convenience of presenting our algorithm.

[3] In the following parts, we omit $\boldsymbol{Q}_z$ in $\boldsymbol{L}\boldsymbol{Q}_z$ for the convenience of presenting our algorithm. This does not affect the optimization procedure due to the nature of $\boldsymbol{Q}_z$.

using OMP [29]. Then the prediction score of $\boldsymbol{c}^{\text{test}}$ on the zth classifier $\boldsymbol{W}_z$ is computed by

$$\boldsymbol{s}_z^{\text{test}} = \boldsymbol{W}_z \boldsymbol{P}_z \boldsymbol{c}^{\text{test}}, \tag{14}$$

and all the scores are voted as follows:

$$\boldsymbol{s}^{\text{test}} = \sum_{z=1}^{Z} \beta_z (\mathbf{V} \circ \boldsymbol{s}_z^{\text{test}}), \tag{15}$$

where $\mathbf{V}$ is an operator that sets the maximum element of the input vector to 1 and sets the remaining elements to 0s. The label of y^{test} is finally determined by taking the class index which corresponds to the maximal value in $\boldsymbol{s}^{\text{test}}$.

Remark - The convergence of the above algorithm cannot be guaranteed. In fact, the proximal method [2] with theoretically guaranteed convergence can be adapted to our settings with very little modifications. However, the theoretical convergence does not provide any practical benefit and it indeed performs slightly worse than our algorithm.

4. Experiments

In existing literature, there are various protocols for evaluating discriminative sparse coding methods. We adopted the experimental setting from [13], which uses five datasets and covers a variety of recognition tasks ranging from face recognition and object classification to scene classification and action recognition. The datasets and protocols are detailed in the next subsection.

Throughout the experiments, we set $H_1 = H_2 = H_3 = H$ for simplicity. The resultant number of base classifiers is $Z = 3H + 1$. The weights of all the classifiers are set the same, *i.e.* $\beta_z = \beta$ and $\gamma_z = \gamma$ for all possible z. Then, the parameters of our method are reduced to five scalars: the number of base classifiers Z, the discrimination weights β and γ, the sparsity degree T, and the dictionary size M. The parameters β and γ are determined by cross-validation, M is set to be a multiple of the number of categories on the dataset, T is set according to [13], and H is set 10 when the dimensions of input signals are over 1000 and set 8 otherwise. For initialization, we calculate $\boldsymbol{D}^{(0)}$ and $\boldsymbol{C}^{(0)}$ using K-SVD and initialize $\boldsymbol{W}^{(0)}$ using (12) with $\boldsymbol{C}^{(0)}$.

4.1. Datasets and protocols

- **Ex. YaleB** [9]: The extended YaleB dataset contains 2414 images of 38 human frontal faces. There are about 64 images taken under different illumination conditions and expressions for each person. Each original face image is cropped to 192 × 168 pixels and then projected onto a 504-dimensional feature vector by random projection. The dataset is randomly split into two halves. One half which contains 32 images per person is used for training, and the other half for test. We set $T = 40$ and $M = 532$.
- **AR Face** [13]: The AR Face dataset consists of over 4000 frontal images from 126 individuals, in which 26 pictures were taken in two separate sessions for each individual. A subset with 2600 images from 50 male subjects and 50 female subjects is used. Each image is cropped to 165 × 120 and then projected onto a 540-dimensional feature vector by random projection. For each person, 20 images are collected for training and the rest are for test. We set $T = 40$ and $M = 500$.
- **Caltech-101** [7]: The Caltech-101 dataset is composed of 8677 images from 101 object categories and 467 images from an additional background category. The number of samples per category is greatly unbalanced, varying from 31 to 800. The 3000-dimensional SIFT-based spatial pyramid feature [18] is used to represent each image. We trained on 15 samples per category and tested on the rest. The dictionary size is set equal to the size of training set (*i.e.* 1530). The parameter T is set to 45.
- **Scene-15** [18]: The Scene-15 dataset contains 4485 images of 15 categories of scenes. The number of samples per category varies from 210 to 410. Similar to the case in Caltech-101, a 3000-dimensional SIFT-based spatial pyramid feature [18] is extracted from each image. From each category, 100 images are collected for training and the rest for test. The parameters are set as follows: $T = 50$ and $M = 600$.
- **UCF Action** [23]: The UCF Sports Action dataset consists of 150 action videos of 10 categories. The number of samples per category varies from 14 to 35. The action bank feature [27] is extracted from each sample and then projected onto a 100-dimensional vector by PCA. The performance is measured by the five-fold cross validation (*i.e.* one fold for test and the remaining four folds for training). We set $M = 50$ and $T = 10$.

4.2. Methods for comparison

Our purpose here is not to compete with the top recognition systems like deep networks, but to demonstrate the improvement of the proposed method over the related ones. Thus, our method is compared against some recent sparse coding methods that are closely-related to ours, including[4]

- *SRC* [31], sparse representation based classification via stacking training samples as a dictionary, which was implemented with two different dictionary configurations: SRC for the case where all training samples are used for dictionary construction, and SRC* for the case where the dictionary size is the same as ours;
- *K-SVD* [1], reconstructive sparse coding via solving (1), which is applied to classification via a two-stage strategy: sparse coding followed by single linear classifier training;

[4] We observed noticeable improvement from state-of-the-art deep networks over our method. But such a comparison is not fair, as deep networks learn features from the data while our method uses handcrafted features.

- *Joint* [24], unifying classifier learning and sparse representation into one optimization framework;
- *D-KSVD* [37], simultaneously learning a dictionary and a linear classifier by solving (3) and (4) via K-SVD;
- *L0DL* [2], a convergent sparse coding method that jointly learns a single classifier and a dictionary;
- *LC-KSVD* [13], sparse coding with label consistency regularization (5) and single linear classifier training (4);
- *DLSI* [26], class-specific dictionary learning with incoherence control on dictionary atoms;
- *FDDL* [35], Fisher discriminant dictionary learning;
- *LLC* [30], coding with locality but not sparsity of codes.

In the next, we denote our method by EasyDL (Ensemble Classifier based Dictionary Learning; 'EC' and 'easy' are homophones). For fair comparison, the dictionary sizes of all the compared methods except SRC are set the same.

4.3. Results and analysis

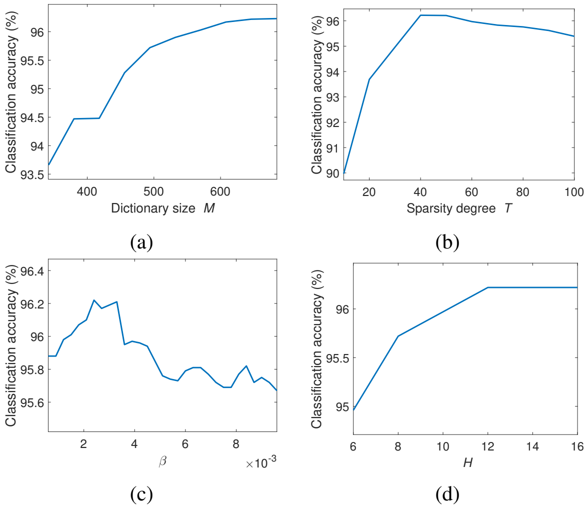
Overall performance. The classification accuracies of all the compared methods are summarized in Table 1. It can be seen that our method is very competitive among all the compared methods. In the evaluation on face recognition, EasyDL outperformed all other compared methods except SRC. The impressive performance of SRC is attributed to its large dictionary size. It can be found from the results of SRC* that, the performance of SRC decreases dramatically when the dictionary size gets small.

Regarding object classification, our method achieved the best result. We tested the performance on the smaller-size training sets. The results show that EasyDL performs consistently well, even in the case where training samples are insufficient, *e.g.*, accuracy of 54.4% is achieved using 5 samples for training. The most competitive method to ours is LC-KSVD, which can be viewed as an ensemble-based method during learning, as shown in Section 3.1. In comparison, EasyDL achieved better results by integrating all base learners for classification and learning compact dictionaries without explicit assignment of labels. We also tested the performance of our method on Caltech with 30 training samples per class and compared it with all the methods reviewed in [4]. The results show that our method performs worse than [4] with a gap of 1.86%, but outperforms other compared methods. It is noted that [4] tackles the feature pooling stage in image classification, which is different from ours, and our method can be used as a classification module and combined with [4] for improvement.

On Scene-15 and UCF, EasyDL performs slightly better than FDDL and shows noticeable improvement over other compared methods. The Fisher discriminant used in FDDL is optimal only when signals from each category are sampled from the normal distribution, implying that FDDL is vulnerable to outliers presented in training data. In contrast, EasyDL tackles imperfectness of data by using ensemble classifiers. Therefore, noticeable performance improvement of EasyDL over FDDL is observed on other datasets.

In summary, all the experimental results demonstrate the effectiveness of our method.

Contribution of ensemble components. The performance of EasyDL was tested with the identical projection plus different combinations of the other three ensemble components (*i.e.* feature selection, random projection and data subsampling). The results on Extended YaleB are listed in Table 3. It is seen that a single component yields moderately good results, and further performance improvement can be gained by combining different ensemble components. This verifies the necessity of using different types of ensemble in EasyDL. Notice that the improvement by the combination of feature selection and random projection is very marginal, as these two components are similar in that they are both for subspace ensemble. Also notice that the subspace ensemble and subsample ensemble are complementary in EasyDL, as noticeable improvement can be observed from the combination of data subsampling and feature selection (or random projection). We also varied the value of H and tested the performance changes of EasyDL. The results are shown in Fig. 2(d). We can see that the performance of EasyDL increases with more classifiers involved, and it becomes saturate when H is sufficiently large.



(a)

(b)

(c)

(d)

Figure 2. Influence of parameter selection in EasyDL.

Influence of parameters. We analyze the influence of the parameters β, T and M by alternatively adjusting one while fixing the other two. The results on Extended YaleB are shown in Fig. 2. We can see from Figure 2(a) that the performance of EasyDL is not sensitive to β within a small range, but exhibits some disturbances due to the non-convexity of the learning model. As β becomes larger, the discrimination

Table 1. Classification accuracies (%) of the compared methods on the test datasets

Dataset	SRC	SRC*	K-SVD	Joint	D-KSVD	L0DL	LC-KSVD	DLSI	FDDL	LLC	EasyDL
Ex. YaleB	**97.20**	80.50	93.10	93.88	94.10	95.66	95.00	89.00	91.90	90.70	96.22
AR Face	**97.50**	68.50	86.50	88.24	88.80	94.40	93.70	89.80	92.00	88.70	94.40
Caltech-101	64.90	64.90	65.20	52.10	65.10	67.58	67.70	61.39	66.80	65.43	**68.40**
Scene-15	91.80	77.62	86.70	88.20	89.10	88.84	92.90	92.46	98.35	89.20	**98.46**
UCF Action	90.40	80.62	86.80	86.00	88.10	86.85	91.20	88.74	91.32	87.50	**91.40**

Table 2. Training time (seconds per iteration) and test time (milliseconds per sample) of the tested methods on five datasets.

Dataset				Training time (s) per iteration				Test time (ms) per sample			
Name	Dim×#Sample	#Training	#Class	D-KSVD	LC-KSVD	FDDL	EasyDL	D-KSVD	LC-KSVD	SRC	EasyDL
Ext. YaleB	504 × 2414	1216	38	2.39	0.83	80.22	21.79	0.10	0.25	30.34	0.43
AR Face	540 × 2600	2000	100	2.64	1.20	153.1	64.80	0.06	0.24	91.12	0.50
Caltech-101	3000 × 9144	1515	102	14.82	8.52	9891	601.83	0.84	0.85	247.54	1.37
Scene-15	3000 × 4485	1500	15	28.47	3.24	60.75	44.64	0.34	0.34	202.83	0.43
UCF Action	100 × 150	140	10	0.14	0.01	0.31	0.16	0.04	0.03	0.53	0.30

of sparse codes increases while the representative power of the dictionary decreases. Thus, an acceptable β should balance the discrimination and representation. In Fig. 2(b), the performance of EasyDL drops a lot when T is small. The reason is obvious: the subspaces of data cannot be fully characterized by a limited number of atoms, making the sparse codes lose discriminability. When T is larger than 50, the performance of EasyDL decreases slightly. This is not surprising as representing samples by many atoms might cause over-fitting. From Fig. 2(c) we can see that the classification accuracy increases as the dictionary becomes larger. But the increment becomes ignorable when the dictionary is sufficiently large.

Table 3. Classification results on the extended YaleB dataset obtained by using different combinations of ensembles.

Ensemble type	Switch [Y=Yes, N=No]						
Feature selection	Y	N	N	Y	Y	N	Y
Random projection	N	N	Y	N	Y	Y	Y
Data subsampling	N	Y	N	Y	N	Y	Y
Accuracy (%)	94.6	94.8	94.2	96.0	94.7	95.7	96.2

4.4. Efficiency

The computational efficiency of EasyDL is compared to D-KSVD, LC-KSVD, SRC, and FDDL. All the compared methods are tested under the same environment: MATLAB on an Intel Quad-Core CPU. Both the time costs of dictionary learning and classification are reported in Table 2. In dictionary learning, EasyDL is slower than LC-KSVD and D-KSVD. The time cost of EasyDL on Extended YaleB is around seven times of D-KSVD on average, yet acceptable. In classification, the time cost of EasyDL is slightly worse than D-KSVD and LC-KSVD but significantly less than SRC. The scalability of EasyDL is better than FDDL and SRC, but still with noticeable increase of the computational time as the scale of problem gets large.

5. Conclusion

As the proverb goes, the wisdom of the masses exceeds that of the wisest individual. We introduced ensemble classifier to discriminative sparse coding, where an ensemble classifier composed of multiple linear predictors is learned during dictionary learning. The integration of sparse coding and ensemble classifier learning not only reduces the bias of classifier but also improves the discriminability of dictionary. The proposed method was tested on several image classification tasks, and it consistently outperformed many existing sparse coding approaches. In future, we would like to further investigate the integration of ensemble learning and sparse coding, such as ensemble of nonlinear classifiers, iterative ensemble construction during learning, and unsupervised ensemble learning with dictionary learning.

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process*, 54(11):4311–4322, 2006. 1, 2, 3, 6

[2] C. Bao, H. Ji, Y. Quan, and Z. Shen. l0 norm based dictionary learning by proximal methods with global convergence. In *CVPR*, pages 3858–3865, 2013. 2, 3, 6, 7

[3] C. Bao, Y. Quan, and H. Ji. A convergent incoherent dictionary learning algorithm for sparse coding. In *ECCV*, pages 302–316. Springer, 2014. 1, 2

[4] Y.-L. Boureau, N. L. Roux, F. Bach, J. Ponce, and Y. LeCun. Ask the locals: multi-way local pooling for image recognition. In *ICCV*, pages 2651–2658. IEEE, 2011. 7

[5] S. Cai, W. Zuo, L. Zhang, X. Feng, and P. Wang. Support vector guided dictionary learning. In *ECCV*, pages 624–639. Springer, 2014. 4

[6] T. G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15. 2000. 2, 4

[7] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *COMPUT VIS IMAGE UND*, 106(1):59–70, 2007. 2, 6

[8] S. Gao, I. W. Tsang, L.-T. Chia, and P. Zhao. Local features are not lonely - Laplacian sparse coding for image classification. In *CVPR*, pages 3555–3561, 2010. 1

[9] A. S. Georghiades, P. N. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(6):643–660, 2001. 6

[10] J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. *J MACH LEARN RES*, 12:3371–3412, 2011. 4

[11] K. Huang and S. Aviyente. Sparse representation for signal classification. In *NIPS*, pages 609–616, 2006. 3

[12] R. Jenatton, J. Mairal, F. R. Bach, and G. R. Obozinski. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, pages 487–494, 2010. 4

[13] Z. Jiang, Z. Lin, and L. Davis. Label consistent K-SVD: Learning a discriminative dictionary for recognition. *IEEE Trans. Pattern Anal. Mach. Intell*, 35(11):2651–2664, 2013. 2, 3, 4, 6, 7

[14] Z. Jiang, Z. Lin, and L. S. Davis. Learning a discriminative dictionary for sparse coding via label consistent K-SVD. In *CVPR*, pages 1697–1704, 2011. 2, 3, 4

[15] Z. Jiang, G. Zhang, and L. S. Davis. Submodular dictionary learning for sparse coding. In *CVPR*, pages 3418–3425, 2012. 1

[16] S. Kong and D. Wang. A dictionary learning approach for classification: separating the particularity and the commonality. In *ECCV*, pages 186–199. Springer, 2012. 3

[17] N. Kulkarni and B. Li. Discriminative affine sparse codes for image classification. In *CVPR*, pages 1609–1616. IEEE, 2011. 1

[18] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, volume 2, pages 2169–2178, 2006. 6

[19] X.-C. Lian, Z. Li, B.-L. Lu, and L. Zhang. Max-margin dictionary learning for multiclass image categorization. In *ECCV*, pages 157–170. Springer, 2010. 2, 3

[20] X.-C. Lian, Z. Li, C. Wang, B.-L. Lu, and L. Zhang. Probabilistic models for supervised dictionary learning. In *CVPR*, pages 2305–2312, 2010. 1, 2, 3

[21] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *CVPR*, pages 1–8, 2008. 1, 2, 3

[22] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. R. Bach. Supervised dictionary learning. In *NIPS*, pages 1033–1040, 2009. 1, 2, 3

[23] J. A. Mikel Rodriguez and M. Shah. Action mach: A spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008. 6

[24] D.-S. Pham and S. Venkatesh. Joint learning and dictionary construction for pattern recognition. In *CVPR*, pages 1–8, 2008. 2, 3, 7

[25] Y. Quan, Y. Huang, and H. Ji. Dynamic texture recognition via orthogonal tensor dictionary learning. In *ICCV*, pages 73–81, 2015. 1

[26] I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *CVPR*, pages 3501–3508, 2010. 1, 3, 4, 7

[27] S. Sadanand and J. J. Corso. Action bank: A high-level representation of activity in video. In *CVPR*, pages 1234–1241, 2012. 6

[28] R. Sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos. Positive definite dictionary learning for region covariances. In *ICCV*, pages 1013–1019, 2011. 3

[29] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inf. Theory*, 53(12):4655–4666, 2007. 5, 6

[30] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, pages 3360–3367, 2010. 3, 7

[31] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell*, 31(2):210–227, 2009. 1, 2, 3, 6

[32] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, pages 1794–1801, 2009. 1, 3

[33] J. Yang, K. Yu, and T. Huang. Supervised translation-invariant sparse coding. In *CVPR*, pages 3517–3524, 2010. 1, 2, 3

[34] M. Yang, D. Dai, L. Shen, and L. Van Gool. Latent dictionary learning for sparse representation based classification. In *CVPR*, June 2014. 2

[35] M. Yang, D. Zhang, and X. Feng. Fisher discrimination dictionary learning for sparse representation. In *ICCV*, pages 543–550, 2011. 2, 3, 4, 7

[36] M. Yang, D. Zhang, and J. Yang. Robust sparse coding for face recognition. In *CVPR*, pages 625–632, 2011. 1

[37] Q. Zhang and B. Li. Discriminative K-SVD for dictionary learning in face recognition. In *CVPR*, pages 2691–2698, 2010. 2, 3, 4, 7

[38] W. Zhang, A. Surve, X. Fern, and T. Dietterich. Learning non-redundant codebooks for classifying complex objects. In *ICML*, pages 1241–1248. ACM, 2009. 2, 3

[39] Y. Zhang, Z. Jiang, and L. S. Davis. Learning structured low-rank representations for image classification. In *CVPR*, pages 676–683, 2013. 1

[40] N. Zhou, Y. Shen, J. Peng, and J. Fan. Learning inter-related visual dictionary for object recognition. In *CVPR*, pages 3490–3497, 2012. 2, 3

[41] Z. Zhu, Q. Chen, and Y. Zhao. Ensemble dictionary learning for saliency detection. *IMAGE VISION COMPUT*, 32(3):180–188, 2014. 2, 3