Label Consistent K-SVD: Learning A Discriminative Dictionary for Recognition

Zhuolin Jiang, *Member, IEEE,* Zhe Lin, *Member, IEEE,* Larry S. Davis, *Fellow, IEEE*

Abstract—A label consistent K-SVD (LC-KSVD) algorithm to learn a discriminative dictionary for sparse coding is presented. In addition to using class labels of training data, we also associate label information with each dictionary item (columns of the dictionary matrix) to enforce discriminability in sparse codes during the dictionary learning process. More specifically, we introduce a new label consistency constraint called 'discriminative sparse-code error' and combine it with the reconstruction error and the classification error to form a unified objective function. The optimal solution is efficiently obtained using the K-SVD algorithm. Our algorithm learns a single over-complete dictionary and an optimal linear classifier jointly. The incremental dictionary learning algorithm is presented for the situation of limited memory resources. It yields dictionaries so that feature points with the same class labels have similar sparse codes. Experimental results demonstrate that our algorithm outperforms many recently proposed sparse coding techniques for face, action, scene and object category recognition under the same learning conditions.

Index Terms—Discriminative dictionary learning, incremental dictionary learning, supervised learning, label consistent K-SVD, discriminative sparse-code error.

1 Introduction

Sparse coding has been successfully applied to a variety of problems in computer vision and image analysis, including image denoising [1], image restoration [2]–[4], image classification [5]–[7] and visual saliency [8]. Sparse coding approximates an input signal, y, by a linear combination of a few items from an over-complete dictionary D. Learning the dictionary from the training samples instead of using off-the-shelf bases such as Fourier or wavelet bases, has been shown to produce state-of-art results [9]. [6] employs the entire set of training samples as the dictionary for discriminative sparse coding, and achieves impressive performances on face recognition. However, determining sparse codes from large dictionaries is computationally expensive, prohibiting real-time application, although some approaches [7], [10]–[13] have been proposed that provide efficient optimization algorithms for sparse coding.

To scale to large training sets, compact dictionary learning approaches have been developed in [6], [11], [12], [14]–[17]. In [6], training samples are manually selected to construct the dictionary. In [12], dictionary learning is accomplished by grouping features from training samples using k-means clustering. In [11], a dictionary learning algorithm, K-SVD, is introduced that generalizes k-means clustering and efficiently learns an over-complete dictionary from a set of training signals. This method has been applied to a variety of image processing problems, including infilling missing pixels and image compression. The method of optimal directions [17] follows closely the k-means outline and updates the dictionary efficiently during learning. [10] uses a Lagrange dual for learning dictionaries efficiently and the feature-sign search with L1 sparsity for learning sparse coefficients. To exploit possible semantic relationships between dictionary elements, [18] employs tree-structured sparse regularization to learn *structured* dictionaries embedded in a hierarchy.

All of these approaches are designed to produce dictionaries useful for images reconstruction. They do not utilize class information about images in the training set. We refer to such learning approaches as *unsupervised* dictionary learning; their goal is to minimize the residual error of reconstructing the original signals to construct a dictionary. Dictionaries learned in such an unsupervised fashion can be used for classification tasks; examples include [5], [6], [19], [20]. But recent research [9], [14]–[16], [21] indicates that dictionaries constructed via *supervised* learning yields better classification performances.

Existing *supervised* learning approaches can be roughly divided into three categories. The first class of approaches learn multiple dictionaries or category-specific dictionaries to promote discrimination between classes [14], [22]–[28]. [22] wraps the dictionary learning process inside a boosting procedure for learning multiple dictionaries. In [14], [24], [27], [28], one dictionary is learned for each class; Classification is based on the corresponding reconstruction error - it does not leverage the sparse codes. However, dictionary construction during training and class-wise sparse coding during testing are time-consuming when there are a large number of classes. [25] learns class-specific dictionaries with an incoherence promoting term, which encourages class-specific dictionaries to be independent. [26] learns multiple dictionaries for visually correlated object cat-

Zhuolin Jiang and Larry S. Davis are with the Institute for Advanced Computer Studies, University of Maryland, College Park, MD, 20742 USA E-Mail: {zhuolin—lsd}@umiacs.umd.edu.
Zhe Lin is with Advanced Technology Labs, Adobe, San Jose, CA, 95110 USA Email: zlin@adobe.com.

egories. The common visual properties of the group are characterized by a common shared dictionary and category-specific visual properties are captured by multiple category-specific dictionaries.

The second set of approaches learn a compact dictionary by merging or selecting dictionary items from an initially large dictionary. [29] merges the visual dictionary items by considering the trade-off between intra-class compactness and inter-class discrimination power. [30], [31] learn a dictionary through merging two items by maximizing the mutual information of class distributions. [32] presents an approach for dictionary learning of action attributes via information maximization. [33] constructs small dictionaries which maintain the performance of their larger counterparts by using agglomerative information bottleneck [34]. [35], [36] exploit submodularity to construct a dictionary from a set of dictionary item candidates. For these approaches, a large dictionary is required at the beginning to guarantee discriminative power of the reduced compact dictionary.

The last type of approach incorporates discriminative terms into the objective function during training such as [14]–[16], [21], [24], [37]–[42]. The discrimination criteria include softmax discriminative cost function [14], [24], [38], [39], Fisher discrimination criterion [21], [26], [40], linear predictive classification error [15], [16], hinge loss function [37], [42] and logistic loss function [38], [41]. [21] learns a structured dictionary with class labels via Fisher discriminative criterion. The approach in [15] iteratively updates the dictionary based on the outcome of a linear classifier; it may suffer from a local minimum problem because it alternates between dictionary construction and classifier design. [16] incorporates classification error into the objective function, but it does not guarantee the discriminability of the resulting sparse codes when using a small-size dictionary.

We present a supervised learning algorithm to learn a compact and discriminative dictionary for sparse coding. We explicitly incorporate a label consistency constraint called 'discriminative sparse-code error' and an 'optimal' classification performance criteria into the objective function and optimize it using the K-SVD algorithm. The learned dictionary is then both reconstructive and discriminative, in contrast to traditional purely constructive ones [6], [11], [17]. The learned dictionary encourages the signals from the same class to have similar sparse codes and those from different classes to have dissimilar sparse codes; hence we can achieve good accuracy on object classification even with a simple multiclass linear classifier, in contrast to other existing sparse coding approaches [5], [14], [37], [38] which learn one classifier for each pair of categories or one-against-all classifiers to obtain good performances. Our dictionary learning algorithm is solved exactly with the K-SVD algorithm. It learns a single compact discriminative dictionary and a universal multiclass linear classifier (for all categories) simultaneously. This is in contrast to dictionary learning approaches such as [37], [38] which iteratively solve subproblems in order to approximate a joint solution, or those approaches [14], [24], [40] which learn the dictionary and classifier separately. Our regression based classification scheme, which only involves matrix multiplication is very efficient, in contrast to the other approaches which first map the computed sparse coefficients to each class [21], [43] and then use the reconstruction error for classification. Our main contributions are:

- A new label consistency constraint called 'discriminative sparse-code error' is introduced and combined with reconstruction error and classification error to form a unified objective function.
- The optimal solution to the objective function is efficiently obtained using the K-SVD algorithm.
- A single compact discriminative dictionary and a universal multiclass linear classier (for all categories) are learned simultaneously.
- An incremental dictionary learning algorithm with the label consistency constraint is introduced.

This paper is organized as follows: Section 2 presents the objective function for learning a reconstructive dictionary and a discriminative dictionary. Section 3 describes a label consistent K-SVD algorithm for simultaneously learning a dictionary with a combined discriminative and reconstructive criteria, and an optimal multiclass linear classifier. An incremental dictionary learning approach with the label consistency constraint is also presented. Section 4 describes the classification approach. Section 5 describes implementation details. Section 6 presents experimental results and analysis. Section 7 concludes the paper and discusses future work.

2 Dictionary Learning

2.1 Dictionary Learning for Reconstruction

2.1.1 *Sparsity Constraint using L_0-norm*

Let Y be a set of n-dimensional N input signals, *i.e.* $Y = [y_1 ... y_N] \in R^{n \times N}$. Learning a reconstructive dictionary with K items for sparse representation of Y can be accomplished by solving the following problem:

$$< D, X >= \arg\min_{D,X} \|Y - DX\|_2^2 \quad s.t. \forall i, \|x_i\|_0 \le T \quad (1)$$

where $\|Y - DX\|_2^2$ denotes the reconstruction error, and $D = [d_1 ... d_K] \in R^{n \times K}$ ($K > n$, making the dictionary over-complete [1]) is the learned dictionary, $X = [x_1, ..., x_N] \in R^{K \times N}$ are the sparse codes of input signals Y, and T is a sparsity constraint factor (each signal has fewer than T non-zero items in its decomposition).

The construction of D is achieved by minimizing the reconstruction error and satisfying the sparsity constraints. The K-SVD algorithm [11] is an iterative approach to minimize the energy in (1) and learns a reconstructive dictionary for sparse representations of signals.

1. It means that the number of dictionary items is larger than the dimensionality of signals, which is often recommended in image processing tasks because it captures a large number of patterns in the input data [10]. However, an over-complete dictionary is not always required for discrimination tasks.

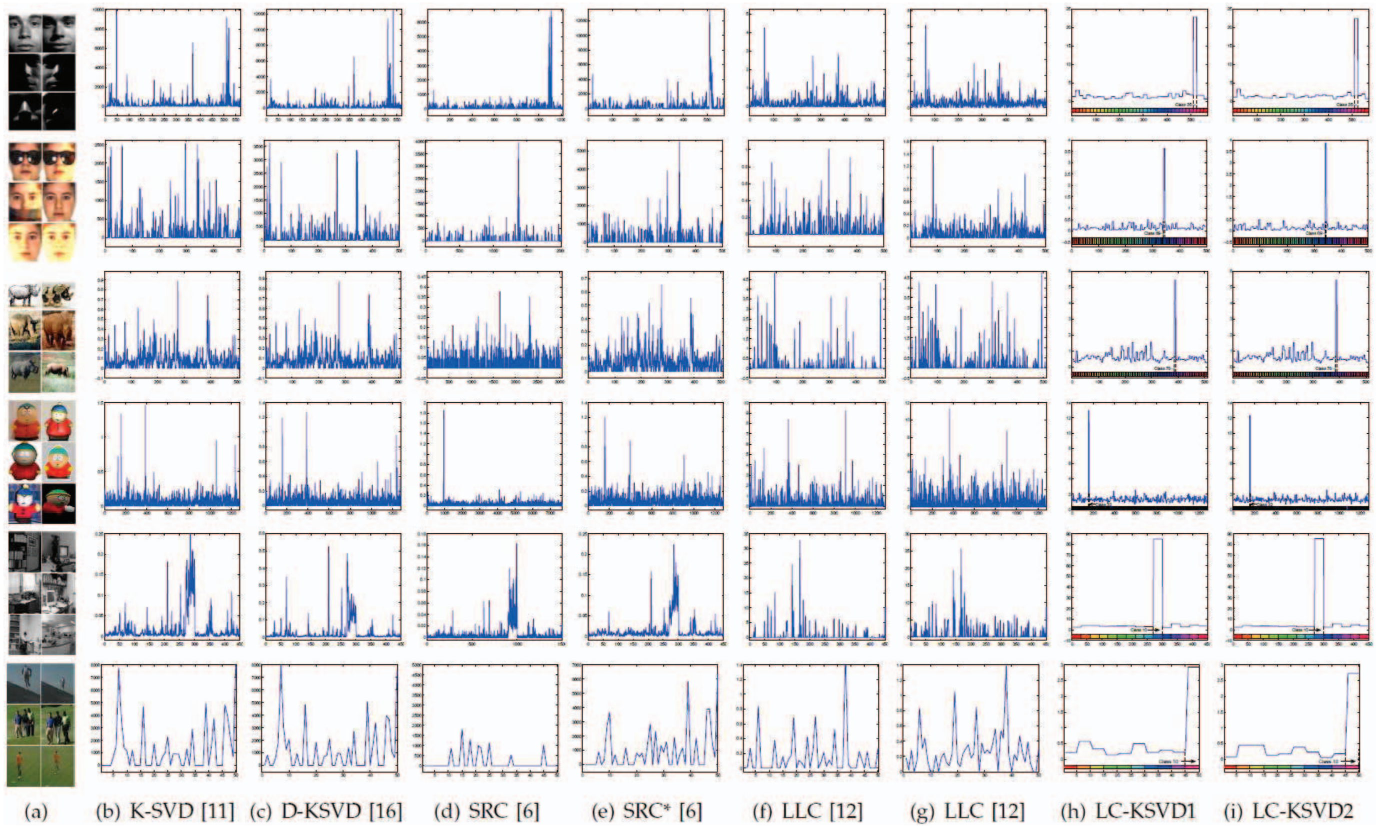


Fig. 1. Examples of sparse codes using different dictionary learning approaches on the six evaluated datasets. Each waveform indicates a sum of absolute sparse codes for different testing samples from the same class. The curves in 1_{st}, 2_{nd}, 3_{rd}, 4_{th}, 5_{th} and 6_{th} row correspond to class 35 (32 testing frames) in Extended YaleB, class 69 (6 testing frames) in AR Face, class 78 (29 testing frames) in Caltech101, class 32 (71 testing frames) in Caltech256, class 10 (115 testing frames) in fifteen scene categories and class 10 (4 testing sequences) in UCF dataset respectively. (a) are sample images from these classes. (f) and (g) are the sparse codes using LLC with 30 and 70 local bases, respectively, in five datasets except the UCF dataset, where we used 10 and 30 correspondingly. Each color from the color bars in (h) and (i) represents one class for a subset of dictionary items. The black dashed lines demonstrate that the curves are highly peaked in one class.

It is highly efficient and works well in applications such as image restoration and compression.

Given D, sparse coding computes the sparse representation x_i of y_i by solving:

$$x_i = x^*(y_i, D) \equiv \arg\min_{x} \|y_i - Dx\|_2^2 \quad s.t. \quad \|x\|_0 \leq T \tag{2}$$

The Orthogonal Matching Pursuit algorithm (OMP) [44] can be used to solve (2).

2.1.2 *Sparsity Constraint using L_1-norm*

An alternative formulation for (1) is to replace L_0-norm regularization with L_1-norm regularization to enforce sparsity:

$$< D, X >= \arg\min_{D,X} \|Y - DX\|_2^2 + \gamma\|X\|_1 \tag{3}$$

where γ is a parameter to balance the reconstruction error and sparsity.

Similarly, given D, the sparse representation x_i of an input signal y_i can be computed as:

$$x_i = x^*(y_i, D) \equiv \arg\min_{x} \|y_i - Dx\|_2^2 + \gamma\|x\|_1, \tag{4}$$

which can be optimized by many efficient L_1 optimization approaches, such as [10], [13].

2.2 Dictionary Learning for Classification

The sparse code x can be directly used as a feature for classification. A good classifier $f(x)$ can be obtained by determining its model parameters $W \in R^{m\times K}$ satisfying:

$$W = \arg\min_{W} \sum_i \mathcal{L}\{h_i, f(x_i, W)\} + \lambda_1\|W\|_F^2 \tag{5}$$

where m is the number of categories, $\mathcal{L}$ is the classification loss function, h_i is the label of y_i and λ_1 is a regularization parameter (which prevents overfitting). Typical loss functions are the logistic loss function [38], square hinge loss [37] and a simple quadratic loss function [15], [16].

Separating the dictionary learning from the classifier learning might make D suboptimal for classification. It is possible to jointly learn the dictionary and classification model, as in [15], [16], [37], [38], which attempt to optimize the learned dictionary for classification tasks.

In this case, an objective function for learning D and W jointly can be defined as:

$$< D, W, X >= \arg \min_{D,W,X} \|Y - DX\|_2^2$$
$$+ \sum_i \mathcal{L}\{h_i, f(x_i, W)\} + \lambda_1 \|W\|_F^2 \ s.t. \forall i, \|x_i\|_0 \leq T \quad (6)$$

Practically, these approaches appear to require learning relatively large dictionaries to achieve good classification performance, leading to high computation cost. This problem is aggravated when good classification results can only be obtained using a classification architecture based on multiple pairwise classifiers or one-against-all classifiers as in [37], [38].

Alternatively, the sparse code x_i can be directly used as a feature descriptor of signal y_i in a classical risk minimization formulation:

$$< D, W >= \arg \min_{D,W} \sum_i \mathcal{L}\{h_i, f(x^*(y_i, D), W)\} + \frac{\nu_1}{2} \|W\|_F^2 \quad (7)$$

where the dictionary D is not explicitly defined in the energy function (7) but implicitly in the sparse coding step, *i.e.* Equations (2) or (4). The dictionary can be obtained by using stochastic gradient descent, as in [9], [37], where implicit differentiation is adopted to compute the gradient of (7) with respect to the dictionary.

Using a single dictionary and a multiclass classifier is efficient for classification, especially with many classes, and it allows feature sharing amongst the classes [9]. We will show that good classification results can be obtained using only a small, single unified dictionary (and a single multiclass linear classifier) by a simple extension to the objective function for joint dictionary and classifier construction. This extension enforces a label consistency constraint on the sparse codes w.r.t. the learned dictionary - intuitively that the class distribution that a dictionary element 'contributes' to during classification are highly peaked in one class. We refer to this method as label consistent K-SVD (LC-KSVD) since it employs the original K-SVD algorithm to obtain its solution.

3 Label Consistent K-SVD

We aim to leverage the supervised information (i.e. labels) of input signals to learn a reconstructive and discriminative dictionary. Each dictionary item will be chosen so that it represents a subset of the training signals ideally from a single class, so each dictionary item d_k can be associated with a particular label. Hence there is an explicit correspondence between dictionary items and the labels in our approach.

We subsequently focus on the effects of adding a label consistency regularization term, and a joint classification error and label consistency regularization term into the objective function in (1) for learning a dictionary with more balanced reconstructive and discriminative power. We refer to them as LC-KSVD1 and LC-KSVD2, respectively, in the following.

3.1 LC-KSVD1

The performance of the linear classifier depends on the discriminability of the input sparse codes x. For obtaining discriminative sparse codes x with the learned D, an objective function for dictionary construction is defined as:

$$< D, A, X >= \arg \min_{D,A,X} \|Y - DX\|_2^2$$
$$+ \alpha \|Q - AX\|_2^2 \ s.t. \forall i, \|x_i\|_0 \leq T \quad (8)$$

where α controls the relative contribution between reconstruction and label consistency regularization, and $Q = [q_1 ... q_N] \in R^{K \times N}$ are the 'discriminative' sparse codes of input signals Y for classification. We say that $q_i = [q_i^1 ... q_i^K]^t = [0 ... 1, 1, ... 0]^t \in R^K$ is a 'discriminative' sparse code corresponding to an input signal y_i, if the non-zero values of q_i occur at those indices where the input signal y_i and the dictionary item d_k share the same label. For example, assuming $D = [d_1 ... d_6]$ and $Y = [y_1 ... y_6]$, where y_1, y_2, d_1 and d_2 are from class 1, y_3, y_4, d_3 and d_4 are from class 2, and y_5, y_6, d_5 and d_6 are from class 3, Q can be defined as:

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

where each column corresponds to a discriminative sparse code for an input signal.

A is a linear transformation matrix. Here we identify a linear transformation, $g(x; A) = Ax$, which transforms the original sparse codes x to be most discriminative in sparse feature space R^K.

The term $\|Q - AX\|_2^2$ represents the discriminative sparse-code error, which enforces that the transformed sparse codes AX approximate the discriminative sparse codes Q. It forces the signals from the same class to have very similar sparse representations (*i.e.* encouraging label consistency in the resulting sparse codes), which results in good classification performance even using a simple linear classifier.

3.2 LC-KSVD2

As in [15], [16], [37], [38], we aim to include the classification error as a term in the objective function for dictionary learning, in order to make the dictionary optimal for classification. Here we use a linear predictive classifier $f(x; W) = Wx$. An objective function for learning a dictionary D having both reconstructive and discriminative power can be defined as follows:

$$< D, W, A, X >= \arg \min_{D,W,A,X} \|Y - DX\|_2^2$$
$$+ \alpha \|Q - AX\|_2^2 + \beta \|H - WX\|_2^2 \ s.t. \forall i, \|x_i\|_0 \leq T \quad (9)$$

where the term $\|H - WX\|_2^2$ represents the classification error. W denotes the classifier parameters. $H =$

$[h_1...h_N] \in R^{m\times N}$ are the class labels of input signals Y. $h_i = [0,0...1...0,0]^t \in R^m$ is a label vector corresponding to an input signal y_i, where the non-zero position indicates the class of y_i. α and β are the scalars controlling the relative contribution of the corresponding terms.

In order to further understand the objective function in (9), we can rewrite the objective function into another simplified form. Assuming $A \in R^{K\times K}$ is invertible and discriminative sparse codes $X' = AX$, then $D' = DA^{-1}$,$W' = WA^{-1}$. The objective function in (9) can be rewritten as:

$$< D', W', X' >= \arg\min_{D',W',X'} \|Y - D'X'\|_2^2$$
$$+\alpha\|Q - X'\|_2^2 + \beta\|H - W'X'\|_2^2 \; s.t. \forall i, \|x_i\|_0 \leq T \quad (10)$$

The first term represents the reconstruction error, the second term the discriminative sparse-code error, and the third term the classification error. The second term $\|Q - X'\|_2^2$ makes the sparse codes discriminative between classes while the third term $\|H-W'X'\|$ supports learning an optimal classifier.

The dictionary learned in this way is adaptive to the underlying structure of the training data (leading to a good representation for each member in the set with strict sparsity constraints), and generates discriminative sparse codes X and addresses the desirable property of the discriminability of classifier construction regardless of the size of the dictionary. These sparse codes can be utilized directly by a classifier, such as in [6]. The discriminative property of sparse code x is very important for the performance of a linear classifier.

In the following section, we describe the optimization procedure for LC-KSVD2. For LC-KSVD1, it utilizes a similar procedure except that $\beta = 0$. However, the classifier W for LC-KSVD1 is trained separately using (17), after D, A and X are computed using (8).

3.3 Optimization

We use the efficient K-SVD algorithm to find the optimal solution for all parameters simultaneously. Equation (9) can be rewritten as:

$$< D, W, A, X >= \arg\min_{D,W,A,X} \left\| \begin{pmatrix} Y \\ \sqrt{\alpha}Q \\ \sqrt{\beta}H \end{pmatrix} - \begin{pmatrix} D \\ \sqrt{\alpha}A \\ \sqrt{\beta}W \end{pmatrix} X \right\|_2^2 \quad s.t. \forall i, \|x_i\|_0 \leq T \; (11)$$

Let $Y_{new} = (Y^t, \sqrt{\alpha}Q^t, \sqrt{\beta}H^t)^t$, $D_{new} = (D^t, \sqrt{\alpha}A^t, \sqrt{\beta}W^t)^t$. The matrix D_{new} is L_2 normalized column-wise. The optimization of (11) is equivalent to solving the following problems:

$$< D_{new}, X >= \arg\min_{D_{new},X} \{\|Y_{new} - D_{new}X\|_2^2\}$$
$$s.t. \forall i, \|x_i\|_0 \leq T \quad (12)$$

This is exactly the problem that K-SVD [11] solves. Following K-SVD, d_k and its corresponding coefficients, the k-th row in X, denoted as x_R^k, are updated at a time. Let $E_k = (Y - \sum_{j\neq k} d_j x_R^j)$, and $\widetilde{x}_R^k$, $\widetilde{E}_k$ denote the result of discarding the zero entries in x_R^k and E_k, respectively. d_k and $\widetilde{x}_R^k$ can be computed by:

$$< d_k, \widetilde{x}_R^k >= \arg\min_{d_k,\widetilde{x}_R^k} \{\|\widetilde{E}_k - d_k\widetilde{x}_R^k\|_F^2\} \quad (13)$$

A SVD operation is performed for $\widetilde{E}_k$, *i.e.* $U\Sigma V^t = \mathtt{SVD}(\widetilde{E}_k)$. Then d_k and $\widetilde{x}_R^k$ are computed as:

$$d_k = U(:,1), \quad \widetilde{x}_R^k = \Sigma(1,1)V(:,1) \quad (14)$$

Finally the non-zero values in x_R^k are replaced by $\widetilde{x}_R^k$. The proposed LC-KSVD2 algorithm is summarized in Algorithm 1.

LC-KSVD learns D, A and W simultaneously, which reduces the possibility of converging to local minima and is scalable to a large number of classes. In addition, it allows us to easily combine another discriminative term, *i.e.* discriminative sparse-code error, into the objective function. It produces a discriminative sparse representation regardless of the size of the dictionary. Fig. 1 shows examples of sparse codes of one testing class from six evaluated datasets using different approaches. From this figure, we can see that LC-KSVD ensures that signals from the same class have similar sparse codes, which is very important for linear classification.

Algorithm 1 Label Consistent K-SVD

```
Input: $Y$, $Q$, $H$, $\alpha$, $\beta$, $T$, $K$
Output: $D$, $A$, $W$
Compute $D^{(0)}$, $A^{(0)}$, $W^{(0)}$:
    Compute $D^{(0)}$ by combining class-specific dictionary items
    for each class using original K-SVD [11];
    Compute sparse codes $X^{(0)}$ for $Y$ using (2);
    Compute $A^{(0)}$ and $W^{(0)}$ using (16)(17);
Initialize $Y_{new} = \begin{pmatrix} Y \\ \sqrt{\alpha}Q \\ \sqrt{\beta}H \end{pmatrix}$ $D_{new} = \begin{pmatrix} D^{(0)} \\ \sqrt{\alpha}A^{(0)} \\ \sqrt{\beta}W^{(0)} \end{pmatrix}$
Update $D_{new}$ by solving (12) using original K-SVD [11];
Obtain $D$, $A$, $W$ from $D_{new}$ by using (23).
```

3.3.1 *Initialization of LC-KSVD*

We need to initialize the parameters $D^{(0)}$, $A^{(0)}$ and $W^{(0)}$ for LC-KSVD. For $D^{(0)}$, we employ several iterations of K-SVD within each class and then combine all the outputs (*i.e* dictionary items learning from each class) of each K-SVD. The label of each dictionary item d_k is then initialized based on the class it corresponds to and will remain fixed during the entire dictionary learning process [2], although d_k is updated during the learning process. Dictionary elements are uniformly allocated to each class with the number of the elements proportional to the dictionary size.

In order to initialize $A^{(0)}$, we employ the multivariate ridge regression model [45], with the quadratic loss and L_2 norm regularization, as follows:

$$A = \arg\min_A \|Q - AX\|^2 + \lambda_2\|A\|_2^2 \quad (15)$$

2. We do associate a unique and fixed class label to each dictionary item, but an input signal of a class certainly can (and does) use dictionary items from other classes, as the sparse codes in Fig. 1 illustrate.

which yields the following solution:

$$A = QX^t(XX^t + \lambda_2 I)^{-1} \tag{16}$$

Similarly, for $W^{(0)}$, we again use the ridge regression model and obtain the following solution:

$$W = HX^t(XX^t + \lambda_1 I)^{-1} \tag{17}$$

Given the initialized $D^{(0)}$, we apply the original K-SVD algorithm to compute the sparse codes X of training signals Y. Then X can be used to compute the initial $A^{(0)}$ in (16) and $W^{(0)}$ in (17).

3.4 Incremental Dictionary Learning

LC-KSVD needs to access the entire training set at each iteration to optimize the objective function. In order to learn a discriminative dictionary with limited memory resources, we employ the learning framework in (7). We learn D, A and W by minimizing the following objective function with the sparsity constraint using L_1-norm regularization:

$$\begin{aligned}\min_{D,W,A} \quad & \sum_i \mathcal{L}^i(D, y_i, W, h_i, A, q_i) + \frac{\nu_1}{2}\|W\|_F^2 + \frac{\nu_2}{2}\|A\|_F^2 \\ s.t. \quad & x_i = \arg\min_x \|y_i - Dx\|_2^2 + \gamma\|x\|_1, i \in \{1...N\} \\ & \|d_j\|_2^2 \leq 1, j \in \{1...K\}. \end{aligned} \tag{18}$$

where $\mathcal{L}^i(D, y_i, W, h_i, A, q_i) \triangleq \mu\|q_i - Ax_i\|_2^2 + (1 - \mu)\|h_i - Wx_i\|_2^2$. Let $\mathcal{L}_1^i(D, y_i, A, q_i) = \|q_i - Ax_i\|_2^2$ and $\mathcal{L}_2^i(D, y_i, W, h_i) = \|h_i - Wx_i\|_2^2$ in the following. [3]Similar to (9), (18) also incorporates the reconstruction error, the discriminative sparse code error and the classification error into the objective function of dictionary learning. Hence minimizing (18) also leads to a dictionary with a reconstructive and discriminative power.

The objective function in (18) is highly nonlinear and highly nonconvex. We use a stochastic gradient descent algorithm to optimize (18). The main difficulty is to compute the gradients of the sparse code x_i with respect to D, because D is not explicitly defined in (18) but implicitly in the sparse coding step (see (4)). So we have to compute the gradient of $\mathcal{L}^i$ for y_i with respect to D using the chain rule: $\frac{\partial \mathcal{L}^i}{\partial D} = \mu \frac{\partial \mathcal{L}_1^i}{\partial x_i}\frac{\partial x_i}{\partial D} + (1-\mu)\frac{\partial \mathcal{L}_2^i}{\partial x_i}\frac{\partial x_i}{\partial D}$.

There is no analytical link between x_i and D. Following [7], [9], [37], we overcome this by using implicit differentiation on the fixed point equations. We establish the fixed point equation for (4): $D^t(Dx - y) = -\gamma sign(x)$. Then we compute the derivative of D on both sides:

$$\frac{\partial x_\Lambda}{\partial D_\Lambda} = (D_\Lambda^t D_\Lambda)^{-1}\left(\frac{\partial D_\Lambda^t y}{\partial D_\Lambda} - \frac{\partial D_\Lambda^t D_\Lambda}{\partial D_\Lambda} x\right) \tag{19}$$

where Λ is the index set of non-zero sparse coefficients of x_i and $\bar{\Lambda}$ denote the index set of zeros coefficients.

3. Since we used very small values for ν_1 and ν_2, the main difference between (18) and (9) is the sparsity constraint: (18) uses L_1-norm while (9) uses L_0-norm. The theoretical results on the equivalence between L_0 and L_1 regularization pointed out by [43], [46], imply that (18) and (9) are generally achieving the same result. The L_0-norm is not differentiable, so we use L_1-norm in (18) in order to perform stochastic gradient descent.

We set the gradients to be zeros for $x_{i,\bar{\Lambda}}$. Here we define an auxiliary variable $\phi_u \in R^K, u \in \{1, 2\}$ with $\phi_{u,\bar{\Lambda}} = 0$, and $\phi_{u,\Lambda} = (D_\Lambda^t D_\Lambda)^{-1}\frac{\partial \mathcal{L}_u^i}{\partial x_i}$, where $\frac{\partial \mathcal{L}_1^i}{\partial x_i} = A^t(Ax_i - q_i)$ and $\frac{\partial \mathcal{L}_2^i}{\partial x_i} = W^t(Wx_i - h_i)$. Hence we obtain: $\frac{\partial \mathcal{L}_1^i}{\partial D} = -D\phi_1 x_i^t + (y_i - Dx_i)\phi_1^t$, $\frac{\partial \mathcal{L}_2^i}{\partial D} = -D\phi_2 x_i^t + (y_i - Dx_i)\phi_2^t$. So the gradient of $\mathcal{L}^i$ with respect to D for input signal y_i can be rewritten as:

$$\frac{\partial \mathcal{L}^i}{\partial D} = \mu\frac{\partial \mathcal{L}_1^i}{\partial D} + (1-\mu)\frac{\partial \mathcal{L}_2^i}{\partial D} \tag{20}$$

The gradients of (18) with respect to A and W can be easily obtained by

$$\frac{\partial \mathcal{L}^i}{\partial A} = \mu(Ax_i - q_i)x_i^t + \nu_2 A, \tag{21}$$

$$\frac{\partial \mathcal{L}^i}{\partial W} = (1-\mu)(Wx_i - h_i)x_i^t + \nu_1 W. \tag{22}$$

We apply an on-line method that reads a small batch of training descriptors y at a time and incrementally updates D, A and W. We first use the previous initialization of LC-KSVD to initialize D, A and W. Then we loop through all the training descriptors to update them incrementally. The parameters D, A and W are updated in a gradient descent fashion. This is different from (9) which can be optimized by K-SVD. The learning rate [4] is set to $\min(\rho, \rho i_0/i)$, where ρ is a constant, $i_0 = \hat{T}/10$ and $\hat{T}$ is the number of iterations. Finally, we project the dictionary items onto the unit circle. The proposed incremental dictionary learning is summarized in Algorithm 2.

Algorithm 2 Incremental Dictionary Learning

```
Input: $Y$, $Q$, $H$, $D^{(0)}$, $A^{(0)}$, $W^{(0)}$, $\mu$, $\gamma$, $\rho$, $\nu_1$, $\nu_2$, $\hat{T}$
Output: $D$, $A$, $W$
for $t = 1...\hat{T}$ do
    Permute training samples $Y$
    for $i = 1...N$ do
        Compute the sparse code $x_i$ for $y_i$ by (4);
        Find the active set $\Lambda_i$ and compute the auxiliary variables $\phi_1$ and $\phi_2$;
        Choose the learning rate $\rho_t = \min(\rho, \rho i_0/i)$
        Update $D$, $A$ and $W$ by (20)(21)(22);
        $D^{(t)} = D^{(t)} - \rho_t \frac{\partial \mathcal{L}^i}{\partial D^{(t)}}$,
        $A^{(t)} = A^{(t)} - \rho_t \frac{\partial \mathcal{L}^i}{\partial A^{(t)}}$,
        $W^{(t)} = W^{(t)} - \rho_t \frac{\partial \mathcal{L}^i}{\partial W^{(t)}}$,
        Project the columns of $D^{(t)}$ onto the unit circle
    end for
    Update $D^{(t+1)} = D^{(t)}$, $A^{(t+1)} = A^{(t)}$, $W^{(t+1)} = W^{(t)}$;
end for
```

4 Classification Approach

We obtain $D = \{d_1...d_K\}$, $A = \{a_1...a_K\}$ and $W = \{w_1...w_K\}$ from D_{new} by employing the K-SVD algorithm [11]. We cannot simply use D, A and W for testing since D, A and W are L_2-normalized in D_{new} jointly in the LC-KSVD algorithm, *i.e.* $\forall k, \|(d_k^t, \sqrt{\alpha}a_k^t, \sqrt{\beta}w_k^t)^t\|_2 = 1$. The desired dictionary $\hat{D}$, transform parameters $\hat{A}$ and classifier parameters $\hat{W}$ are computed as follows:

$$\hat{D} = \left\{\frac{d_1}{\|d_1\|_2}...\frac{d_K}{\|d_K\|_2}\right\}, \hat{A} = \left\{\frac{a_1}{\|d_1\|_2}...\frac{a_K}{\|d_K\|_2}\right\}, \hat{W} = \left\{\frac{w_1}{\|d_1\|_2}...\frac{w_K}{\|d_K\|_2}\right\} \tag{23}$$

4. This strategy is adopted from [9], [47]. The algorithm converges very fast, typically in 10 iterations as shown in Figure 12(a).

Given an input signal y_i with its corresponding discriminative code q_i and the label vector h_i, the relationship between the desired ($\hat{D}$, $\hat{A}$, $\hat{W}$) and the learned (D, A, W) is established according to:

$$\begin{aligned} y_i &\approx Dx_i = \sum_k x_{k,i} d_k = \sum_k x_{k,i} \|d_k\|_2 \frac{d_k}{\|d_k\|_2} = \sum_k \hat{x}_{k,i} \hat{d}_k = \hat{D}\hat{x}_i \\ q_i &\approx Ax_i = \sum_k x_{k,i} a_k = \sum_k x_{k,i} \|d_k\|_2 \frac{a_k}{\|d_k\|_2} = \sum_k \hat{x}_{k,i} \hat{a}_k = \hat{A}\hat{x}_i \\ h_i &\approx Wx_i = \sum_k x_{k,i} w_k = \sum_k x_{k,i} \|d_k\|_2 \frac{w_k}{\|d_k\|_2} = \sum_k \hat{x}_{k,i} \hat{w}_k = \hat{W}\hat{x}_i \end{aligned} \tag{24}$$

where $\hat{d}_k = \frac{d_k}{\|d_k\|_2}$, $\hat{a}_k = \frac{a_k}{\|d_k\|_2}$ and $\hat{w}_k = \frac{w_k}{\|d_k\|_2}$ are the k-th column of $\hat{D}$, $\hat{A}$ and $\hat{W}$, respectively.

For a test image y_i, we first compute its sparse representation $\hat{x}_i$ with dictionary $\hat{D}$ by solving Equations (2) or (4). Then we simply use the linear predictive classifier to estimate a label vector $l = \hat{W}\hat{x}_i$. The label of y_i is the index corresponding to the largest element of l.

5 Implementation Details

In this section, we provide implementation details of our approach. The parameters α and β are fixed for each dataset and determined by n-fold cross validation on the training data. Their learned values [5] for different datasets are presented in the following experiment section. The sparsity factor $T = 30$ is used in our experiments except the UCF dataset, where $T = 10$ is used.

The optimization of (8) for LC-KSVD1 and LC-KSVD2 employ a similar procedure except for the classification error term in (11). The linear classifier W is learned separatively from the training data, using the multivariate ridge regression model as in (17).

The feature descriptors used in the Extended YaleB database and AR Face database are random faces [6], [16]. Each face image is projected onto a n-dimensional feature vector with a randomly generated matrix from a zero-mean normal distribution. Each row of the matrix is L_2 normalized. Following [16], the dimension of a random-face feature in Extended YaleB is $n = 504$ while the dimension in AR face is $n = 540$.

For the Caltech101 dataset, we first extract SIFT descriptors from 16×16 patches which are densely sampled using a grid with a step size of 6 pixels; then we extract the spatial pyramid feature [48] based on the extracted SIFT features with three grids of size 1×1, 2×2 and 4×4. To train the codebook for the spatial pyramid, we use the standard k-means clustering with $k = 1024$. Finally, the spatial pyramid feature is reduced to 3000 dimensions by PCA. Please note that the sparse coding in our approach is used to encode the extracted spatial pyramid features. This is very different from approaches, such as [5], [12], which use sparse coding to encode SIFT descriptors from local patches.

5. From our experiments, $\alpha = 4.0$ and $\beta = 2.0$ achieved good performances for datasets using random face features (such as AR and YaleB); while α and β are set to 0.001 for spatial pyramid features. The surprising small value for alpha and beta for spatial pyramid features is due to the scale of those features. These are large and sparse vectors which when normalized result in feature vectors with very small component values; so, the reconstruction errors have very small magnitudes, and the relative weights of α and β compared to the scale of reconstruction errors is similar for all of the datasets.

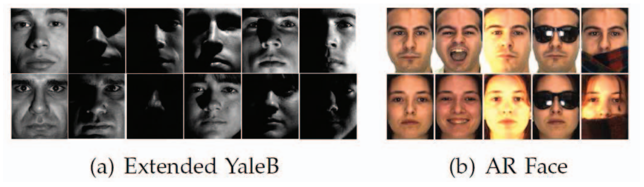


Fig. 2. Evaluated Face databases.

For the Caltech101 dataset, in each spatial sub-region of the spatial pyramid, the vector quantization codes are pooled together to form a pooled feature. The pooled features from each sub-region are concatenated and normalized as the final spatial pyramid feature of an image. There are two pooling methods: (1) sum pooling [48]: $x_{out} = x_1+, ..., +x_n$; (2) max pooling [5]: $x_{out} = max(x_1, ..., x_n)$, where x_i is the vector quantization code. Then these pooled features are normalized by: (1) L_1 normalization: $x_{out} = x_{out}/\sum_i x_i$; (2) L_2 normalization: $x_{out} = x_{out}/\|x_{out}\|_2$. Different combinations are evaluated in the experiment section.

For the Caltech256 dataset, we compute HOG descriptors from each patch at three scales, 16×16, 25×25 and 31×31. The dimension of each HOG descriptor is 128. Then we compute the spatial pyramid features using 1×1, 2×2, and 4×4 sub-regions. Finally we reduce the dimensionality of the features to 305 using PCA.

For the Fifteen Scene Category dataset, we compute the spatial pyramid feature using a four level spatial pyramid and a SIFT-descriptor codebook with a size of 200. The final spatial pyramid features are reduced to 3000 by PCA. The feature descriptors used in the UCF action dataset are the action bank feature representations provided by [49]. We again use PCA to reduce the feature dimension to 100.

For the incremental dictionary learning, we set the parameters ν_1 and ν_2 to be 10^{-5}. The constant ρ is selected from $\{10^{-6}, 10^{-5}, ..., 10\}$. The parameters μ and γ are set to 0.6 and 0.5 respectively in the experiment.

6 Experiments

We evaluate our approach on two face databases: Extended YaleB database [50] and AR face database [51], two object category datasets: Caltech101 [52] and Caltech256 [53], a scene category dataset: Fifteen scene categories [48], and a realistic action dataset: UCF sports action [54]. We compare our approaches with K-SVD [6] [11], D-KSVD [16], sparse representation-based classification (SRC) [6] and the recently proposed LLC algorithm [12] [7]. Our source code can be downloaded from: http://www.umiacs.umd.edu/~zhuolin/projectlcksvd.html.

6. An algorithm that directly uses the dictionary learned by original K-SVD algorithm and employs a linear classifier learned by (17).

7. D-KSVD and SRC results are based on our own implementations, K-SVD and LLC implementations are provided by the authors, which allowed us to standardize the learning parameters across methods.

6.1 Extended YaleB

The Extended YaleB database contains $2,414$ frontal-face images of 38 persons [50]. There are about 64 images for each person. The original images were cropped to 192×168 pixels. This database is challenging due to varying illumination conditions and expressions as shown in Fig. 2(a). We randomly select half of the images per category as training and the other half for testing. The dictionary consists of 570 items, which corresponds to an average of 15 items per person. Unlike K-SVD [11] and D-KSVD [16], there is an explicit correspondence between the dictionary items and the labels of people in our approach, which is similar to the SRC algorithm [6], but our approach uses fewer training samples.

We measure the performance of the SRC algorithm using dictionaries with two different sizes (all training samples and 15 samples per person). For fair comparison, the number of local bases, which determines the sparsity of the LLC codes, is chosen as identical to the sparsity factor (*i.e.* $T = 30$) used in our implementation. For evaluating the importance of the number of local bases, we also evaluate LLC with 70 local bases.

The experimental results are summarized in Table 1. Most of the failure cases are from images taken under extremely bad illumination conditions. Hence we perform another experiment with these bad images excluded (about 10 for each person). The results of this experiment are listed in the third column of Table 1. Our approaches always achieve better results than KSVD, D-KSVD, LLC. Additionally, our approach outperforms SRC when using the same size dictionary (denoted SRC*).

In addition, we compare with SRC in terms of the computation time for classifying one test image. The time is computed as an average over all the test images. As shown in Table 2, our approach is approximately 22 times faster than SRC*. If a database is provided with more categories, a small dictionary for sparse coding can save even more time (see the results for AR database).

α and β are determined by 5-fold cross validation on the training dataset. The effects of parameter selection are show in Fig. 3. We can observe that good performance is achieved at $\alpha = 4.0$ and $\beta = 2.0$.

6.2 AR Face

The AR face database consists of over $4,000$ color images of 126 persons. Each person has 26 face images taken during two sessions. Compared to the Extended YaleB, these images in Fig. 2(b) include more facial variations including different illumination conditions, different expressions and different facial 'disguises' (sunglasses and scarves). Following the standard evaluation procedure, we use a subset of the database consisting of 2600 images from 50 male subjects and 50 female subjects. For each person, we randomly select 20 images for training and the other 6 for testing. Each face image, of size 165×120 pixels, is projected onto a 540-dimensional vector with a randomly generated matrix. The learned dictionary has 500 dictionary items, *i.e.* 5 items per person. As discussed earlier, there is an explicit correspondence between the dictionary items and the labels of people. $\alpha = 3.0$ and $\beta = 2.0$ are used in the experiment.

We evaluate our approaches using random face features and compared with state-of-art approaches. For SRC, we learn two dictionaries with two different dictionary sizes. All the approaches use the same learning parameters. The recognition results are summarized in Table 3. Our approaches outperform K-SVD, D-KSVD, LLC and SRC*. Note that SRC degrades dramatically when using only 5 samples per person.

In addition, we compare with SRC in terms of the computation time for classification. As shown in Table 4, our approach is approximately 35 times faster than SRC*. As expected, a small dictionary can save more time for a database consisting of many training images, and the performance does not degrade much compared to using the entire set of training images as the dictionary.

TABLE 1
Recognition results using random-face features on the Extended YaleB database. The 2nd column is the result when we used all 64 images for each person. The 3rd column is the result when we removed 10 poor-quality images for each person.

Method	Accuracy (%)	Accuracy (%)
K-SVD(15 per person) [11]	93.1	98.0
D-KSVD(15 per person) [16]	94.1	98.0
SRC(all train. samp.) [6]	**97.2**	**99.0**
SRC*(15 per person) [6]	80.5	86.7
LLC(30 local bases) [12]	82.2	92.1
LLC(70 local bases) [12]	90.7	96.7
LC-KSVD1(15 per person)	94.5	98.3
LC-KSVD2(15 per person)	95.0	98.8
LC-KSVD2(all train. samp.)	96.7	**99.0**

TABLE 2
Computation time for classifying a test image on the Extended YaleB database.

Method	Average Time (ms)
SRC(all training samples) [6]	20.78
SRC*(15 per person) [6]	11.22
LC-KSVD1(15 per person)	0.52
LC-KSVD2(15 per person)	0.49

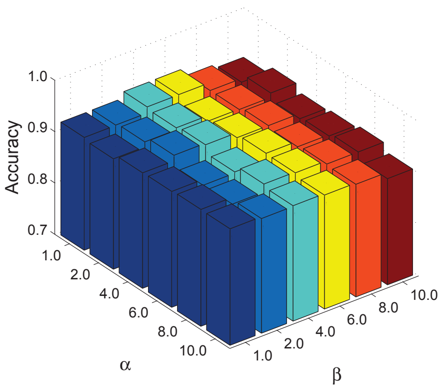


Fig. 3. Effects of parameter selection of α and β on the classification accuracy on the Extended YaleB database.

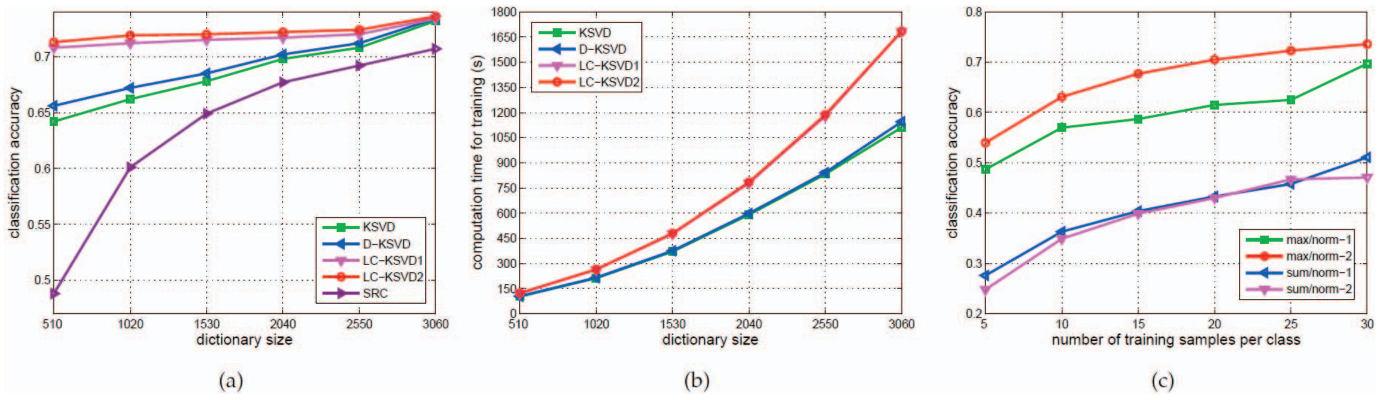


Fig. 4. Performance comparisons on the Caltech101. (a) Performance on the Caltech101 with varying dictionary size; (b) Training time on the Caltech101 with varying dictionary size; (c) Performance on the Caltech101 with different spatial-pyramid-matching settings.

TABLE 3
Recognition results using random face features on the AR face database.

Method	Accuracy (%)
K-SVD(5 per person) [11]	86.5
D-KSVD(5 per person) [16]	88.8
SRC(all train. samp.) [6]	97.5
SRC*(5 per person) [6]	66.5
LLC(30 local bases) [12]	69.5
LLC(70 local bases) [12]	88.7
LC-KSVD1(5 per person)	92.5
LC-KSVD2(5 per person)	93.7
LC-KSVD2(all train. samp.)	**97.8**

TABLE 4
Computation time for classifying a test image on the AR face database.

Method	Average Time (ms)
SRC(all training samples) [6]	83.79
SRC*(5 per person) [6]	17.76
LC-KSVD1(5 per person)	0.541
LC-KSVD2(5 per person)	0.479

TABLE 5
Recognition results using spatial pyramid features on the Caltech101 dataset.

number of train. samp.	5	10	15	20	25	30
Malik [55]	46.6	55.8	59.1	62.0	-	66.20
Lazebnik [48]	-	-	56.4	-	-	64.6
Griffin [53]	44.2	54.5	59.0	63.3	65.8	67.60
Irani [56]	-	-	65.0	-	-	70.40
Grauman [57]	-	-	61.0	-	-	69.10
Pham [15]	-	-	42.0	-	-	-
Gemert [58]	-	-	-	-	-	64.16
Yang [5]	-	-	67.0	-	-	73.20
Wang [12]	51.15	59.77	65.43	67.74	70.16	73.44
SRC [6]	48.8	60.1	64.9	67.7	69.2	70.7
K-SVD [11]	49.8	59.8	65.2	68.7	71.0	73.2
D-KSVD [16]	49.6	59.5	65.1	68.6	71.1	73.0
LC-KSVD1	53.5	61.9	66.8	70.3	72.1	73.4
LC-KSVD2	**54.0**	**63.1**	**67.7**	**70.5**	**72.3**	**73.6**

TABLE 6
Computation time (ms) for classifying a test image on the Caltech101 dataset.

Dictionary size	510	1020	1530	2040	2550	3060
SRC [6]	173.44	343.12	520.88	662.40	835.34	987.55
LC-KSVD1	0.59	1.09	1.62	2.21	2.83	3.50
LC-KSVD2	0.54	0.98	1.44	1.94	2.50	3.17

6.3 Caltech101

The Caltech101 dataset [52] contains 9144 images from 102 classes (*i.e.* 101 object classes and a 'background' class) including animals, vehicles, flowers, etc. The samples from each category have significant shape variability. The number of images in each category varies from 31 to 800. Following the common experimental settings, we train on 5, 10, 15, 20, 25 and 30 samples per category and test on the rest. We repeat the experiments 10 times with different random spits of the training and testing images to obtain reliable results. The final recognition rates are reported as the average of each run. α and β are set to 0.001 in the experiment.

We evaluate our approach using spatial pyramid features and compare with K-SVD [11], D-KSVD [16], SRC [6] and other state-of-art approaches [5], [12], [15], [48], [53], [55]–[58]. The comparative results are shown in Table 5. We used 510, 1020, 1530, 2040, 2550 and 3060 dictionary items, respectively, for 5, 10, 15, 20, 25 and 30 training samples per category. Our approaches consistently outperform all the competing approaches. The basic reason for the good recognition performance, even with only a few training examples, is that the new label consistency constraint encourages the input signals from the same class to have similar sparse codes and those from different classes to have dissimilar sparse codes.

We randomly select 30 images per category as training data, and evaluate our approach using different dictionary sizes $K = 510$, 1020, 1530, 2040, 2550 and 3060; We compare the classification accuracy of K-SVD [11], D-KSVD [16] and SRC [6]. Fig. 4(a) shows that our approaches maintain a high classification accuracy and outperform the other three competing approaches significantly even using a smaller size dictionary.

We also compare the dictionary training time with K-SVD and D-KSVD. As shown in Fig. 4(b), we can train approximately 13 times faster when we use a dictionary with a size of 510 rather than 3060. More importantly, as shown in Fig. 4(a), the classification accuracy of LC-KSVD degrades only slightly when using the dictionary

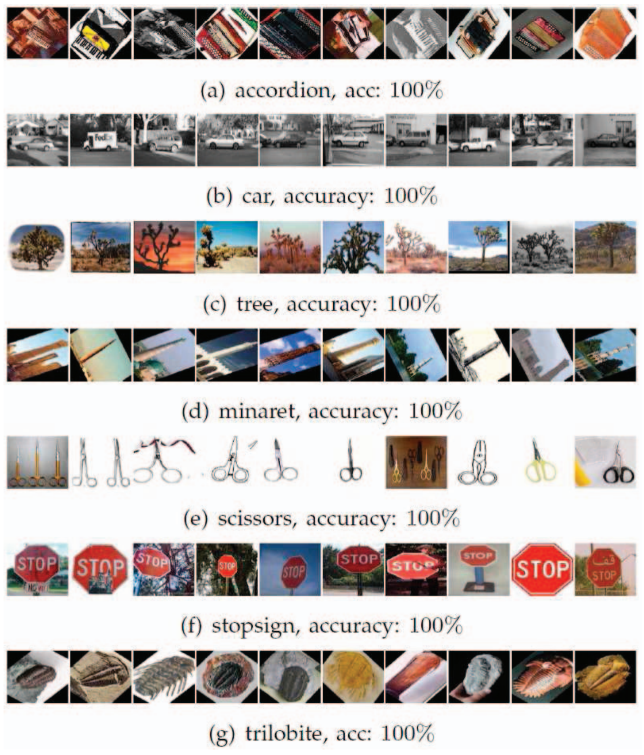


Fig. 5. Example images from classes with high classification accuracy from the caltech101 dataset.

of size 510. We compare our approach with SRC in terms of computation time of classifying one test image, using different dictionary sizes. As shown in Table 6, the computation time of our approach is significantly faster (a factor of more than 310) than SRC.

In addition, we evaluate different combinations of pooling methods and normalization methods for computing spatial pyramid features. As can be seen from Fig. 4(c), 'max pooling' followed by 'L_2 normalization' generates the best classification accuracy. Note that 'sum pooling' followed by 'L_1 normalization' generates the histograms which were used in the original spatial pyramid features [48]. There were a total of 9 classes achieving 100% classification accuracy when using 30 training images per category. Fig. 5 shows some samples from seven of these classes.

6.4 Caltech256

The Caltech256 dataset [53] contains $30,607$ images of 256 categories. There are at least 80 images per category. Compared to Caltech101 dataset, it is much more difficult due to the large variations in object location, pose and size. α and β are set to 0.001 in the experiment.

We evaluated our approaches on both 15 and 30 training images per class and compare with K-SVD [11], D-KSVD [16], SRC [6], LLC [12] and the state-of-the-art approaches [5], [53], [58]. We evaluate LLC with 70 local bases. The comparisons are shown in Table 7. We used 3840 and 7680 dictionary item respectively for 15 and 30 training samples per category. Our results are better than those in [58] in the case of 30 training samples per category, and marginally better than [5], [53] for both cases. Note that [5], [53] train sophisticate SVM classifiers to obtain good performance while our results are obtained by using only a simple linear classifier. In addition, our approaches outperform the other competing dictionary learning approaches including K-SVD, D-KSVD, LLC and SRC.

We randomly select 30 images per category as training data, and evaluate our approach using dictionary sizes from 1280 to 7680, and compare the results with those from K-SVD, D-KSVD, SRC and LLC. As shown in Fig. 6, our approaches maintain high classification accuracies and outperform the other competing approaches. Fig. 7 shows samples of ten categories with high classification accuracies when using 30 training images per category.

6.5 Fifteen Scene Categories

This is a dataset of fifteen natural scene categories introduced in [48]. Each category has 200 to 400 images, and the average image size is about 250×300 pixels. This dataset contains fifteen scenes such as bedroom, kitchen and country scenes as shown in Fig. 8. Following the common experimental settings, we randomly select 100 images per category as training data and use the rest as test data. α and β are set to 0.001 in the experiment. The learned dictionary has 450 items.

We compare our results with K-SVD [11], D-KSVD [16], SRC [6], LLC [12] and other approaches [5], [39], [42], [48], [58], [59]. We evaluate the LLC algorithm with 30 local bases. As Table 8 shows, our approaches

TABLE 7
Recognition results using spatial pyramid features on the Caltech256 dataset.

number of training samples	15	30
Griffin [53]	28.3	34.10
Gemert [58]	-	27.17
Yang [5]	27.73	34.02
K-SVD [11]	25.33	30.62
D-KSVD [16]	27.79	32.67
SRC [6]	27.86	33.33
LLC [12]	25.61	30.43
LC-KSVD1	28.1	32.95
LC-KSVD2	28.9	34.32

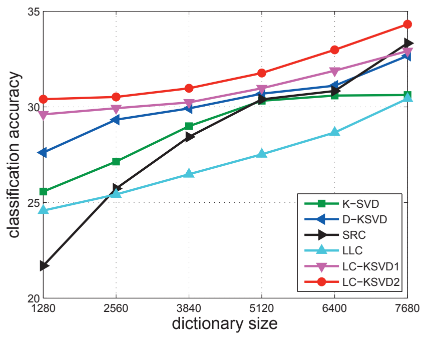


Fig. 6. Recognition results using different approaches with different dictionary sizes on the Caltech256.

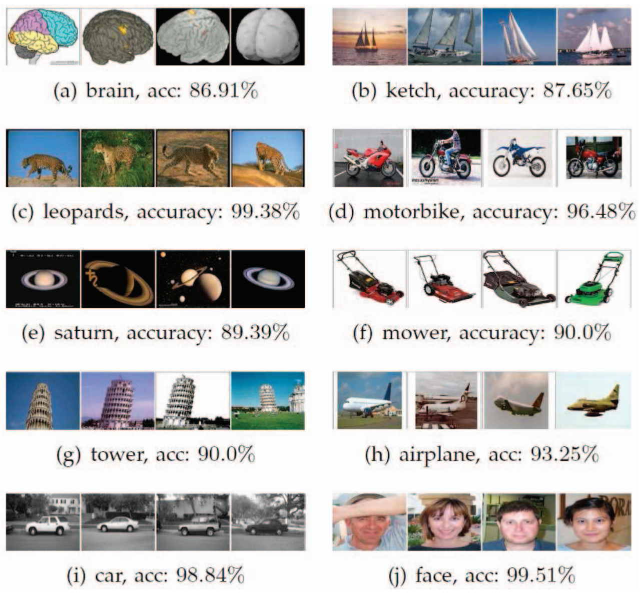

(a) brain, acc: 86.91%
(b) ketch, accuracy: 87.65%
(c) leopards, accuracy: 99.38%
(d) motorbike, accuracy: 96.48%
(e) saturn, accuracy: 89.39%
(f) mower, accuracy: 90.0%
(g) tower, acc: 90.0%
(h) airplane, acc: 93.25%
(i) car, acc: 98.84%
(j) face, acc: 99.51%

Fig. 7. Example images from classes with high classification accuracy from the Caltech256 dataset.

Fig. 8. Ten categories in the Fifteen Scene Category dataset.

outperform the state-of-the-art approaches and the competing dictionary learning approaches such as K-SVD, D-KSVD and LLC [8]. The confusion matrices for the LC-KSVD1 and LC-KSVD2 are shown in Figures 9(a) and 9(b), respectively.

6.6 UCF Sports Action

The UCF sports action dataset [54] contains video sequences which are collected from various broadcast sports channels such as BBC and ESPN. This dataset contains videos covering a wide range of scenarios and viewpoints. There are ten action classes included in this dataset: diving, golfing, kicking, lifting, horse riding, running, skateboarding, swinging-(pommel horse and floor), swinging-(high bar) and walking. Example images from this dataset are show in Fig. 10. α and β are set to 1.0 in the experiment.

Following the common experimental settings, we evaluate our approach using the leave-one-video-out evaluation strategy. We compared with state-of-the-art approaches [49], [54], [60]–[63]. We also evaluate our approach using five-fold cross validation as in [32], [64], where one fold is used for testing and the remaining four folds are used for training. The dictionary size for the leave-one-video-out scheme is 70 while the dictionary size for the five fold cross validation is 50. We compared the results with SRC [6], LLC [12], K-SVD [11] and D-KSVD [16] and [32], [64]. We evaluate the LLC algorithm with 30 local bases.

The detailed comparison results are shown in Table 9. For both evaluation schemes including leave-one-video-

8. The reason why LLC algorithm works better than the original result in the paper is that we use sparse coding (LLC) to encode the spatial pyramid features while the original LLC uses sparse coding to encode sift descriptors for extracting sparse pyramid features. The low-level features are the same for SRC, LLC, K-SVD and D-KSVD.

TABLE 8
Recognition results using spatial pyramid features on the Fifteen scene category dataset.

Method	Accuracy(%)	Method	Accuracy(%)
LC-KSVD1	90.4	Lazebnik [48]	81.4
LC-KSVD2	92.9	Gemert [58]	76.7
SRC [6]	91.8	Yang [5]	80.3
LLC [12]	89.2	Gao [59]	89.7
K-SVD [11]	86.7	lian [42]	86.4
D-KSVD [16]	89.1	Boureau [39]	84.3

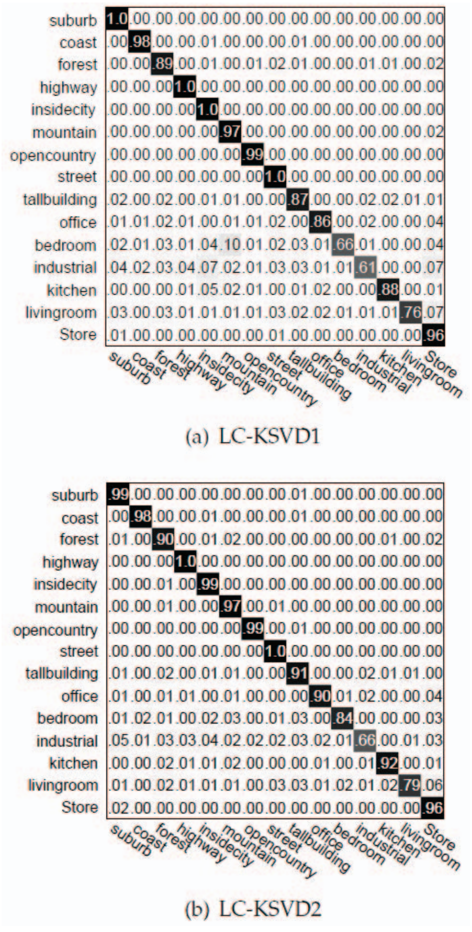


Fig. 9. Confusion matrices on the Fifteen scene category dataset using the dictionary size $K = 450$.

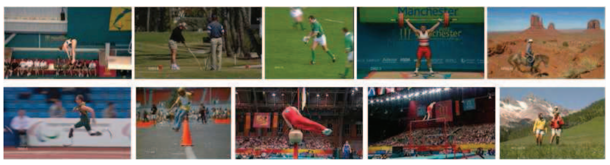
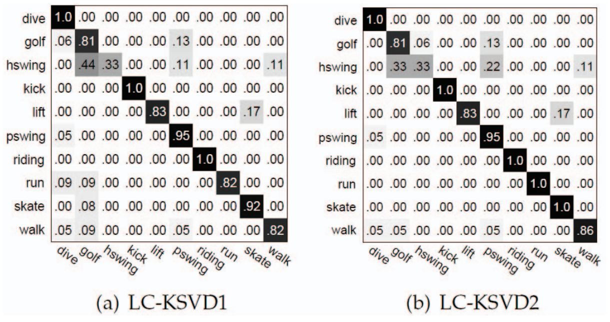


Fig. 10. The UCF sports action dataset.

TABLE 9
Recognition results using action bank features on the UCF sports dataset.

Method	evaluation	Accuracy(%)
Rodriguez [54]	leave-one-out	69.2
Yeffet [60]	leave-one-out	79.3
Wu [61]	leave-one-out	91.3
Kovashka [62]	leave-one-out	87.3
Le [63]	leave-one-out	86.5
Sadanand [49]	leave-one-out	95.0
Qiu [32]	five-fold-cross	83.6
Yao [64]	five-fold-cross	86.6
SRC [6]	five-fold-cross	90.4
LLC [12]	five-fold-cross	87.5
K-SVD [11]	five-fold-cross	86.8
D-KSVD [16]	five-fold-cross	88.1
LC-KSVD1	five-fold-cross	88.6
LC-KSVD2	five-fold-cross	91.2
LC-KSVD1	leave-one-out	95.7
LC-KSVD2	leave-one-out	95.7



	dive	golf	hswing	kick	lift	pswing	riding	run	skate	walk
dive	1.0	.00	.00	.00	.00	.00	.00	.00	.00	.00
golf	.06	.81	.00	.00	.00	.13	.00	.00	.00	.00
hswing	.00	.44	.33	.00	.00	.11	.00	.00	.00	.11
kick	.00	.00	.00	1.0	.00	.00	.00	.00	.00	.00
lift	.00	.00	.00	.00	.83	.00	.00	.00	.17	.00
pswing	.05	.00	.00	.00	.00	.95	.00	.00	.00	.00
riding	.00	.00	.00	.00	.00	.00	1.0	.00	.00	.00
run	.09	.09	.00	.00	.00	.00	.00	.82	.00	.00
skate	.00	.08	.00	.00	.00	.00	.00	.00	.92	.00
walk	.05	.09	.00	.00	.00	.05	.00	.00	.00	.82

(a) LC-KSVD1

	dive	golf	hswing	kick	lift	pswing	riding	run	skate	walk
dive	1.0	.00	.00	.00	.00	.00	.00	.00	.00	.00
golf	.00	.81	.06	.00	.00	.13	.00	.00	.00	.00
hswing	.00	.33	.33	.00	.00	.22	.00	.00	.00	.11
kick	.00	.00	.00	1.0	.00	.00	.00	.00	.00	.00
lift	.00	.00	.00	.00	.83	.00	.00	.00	.17	.00
pswing	.05	.00	.00	.00	.00	.95	.00	.00	.00	.00
riding	.00	.00	.00	.00	.00	.00	1.0	.00	.00	.00
run	.00	.00	.00	.00	.00	.00	.00	1.0	.00	.00
skate	.00	.00	.00	.00	.00	.00	.00	.00	1.0	.00
walk	.05	.05	.00	.00	.00	.05	.00	.00	.00	.86

(b) LC-KSVD2

Fig. 11. Confusion matrices on the UCF dataset using the five-fold-cross-validation scheme.

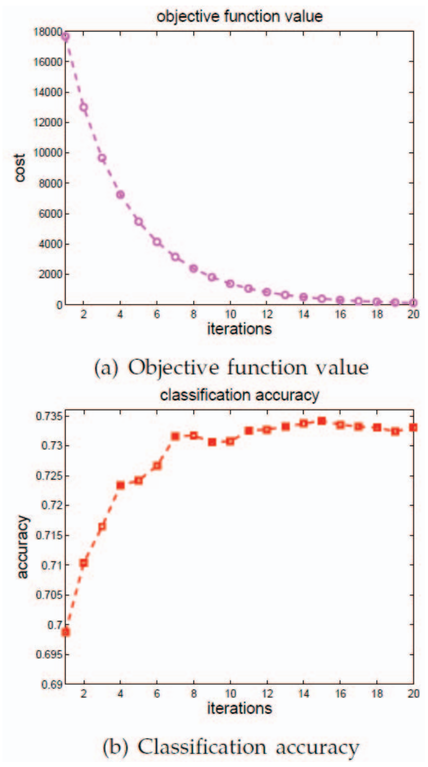
out and five-fold cross validation, our results are better than the other state-of-the-art approaches and the competing dictionary learning approaches. Fig. 11 shows the confusion matrices for both the LC-KSVD1 and the LC-KSVD2 approaches.

6.7 Evaluation of Incremental Dictionary Learning

We evaluate the incremental dictionary learning approach on the Caltech101 dataset. We randomly select 30 images per category as training data, and evaluate the incremental learning approach using different dictionary sizes $K \in \{510, 1020, 1530, 2040, 2550, 3060\}$. We compared the experimental results with those from the 'batch' version LC-KSVD2 in terms of classification accuracy and memory consumption.

Fig. 12 shows the optimization process of incremental dictionary learning for $\hat{T} = 20$ iterations using $K = 3060$. We record the objective function value and evaluate the recognition performance with the learned dictionary on the test data. As expected, the accuracy increases as the objective function value decreases on the whole.

Fig. 13(a) shows that the classification performance of incremental dictionary learning is comparable to LC-KSVD2. For memory consumption comparison, we learn a dictionary of size $K = 510$ given different sizes of training data. Fig. 13(b) shows that the incremental dictionary learning uses nearly constant memory while the batch process (LC-KSVD2) consumes memory linear



(a) Objective function value

(b) Classification accuracy

Fig. 12. The optimization process of the objective function for incremental dictionary learning on the Caltech101 dataset with 20 iterations.

in the training data size. This means that the incremental dictionary learning is feasible for a limited machine memory situation when given a large scale training dataset.

7 Conclusions

We proposed a new dictionary learning approach, label consistent K-SVD algorithm, for sparse coding. Our main contribution lies in explicitly integrating the 'discriminative' sparse codes and a single predictive linear classifier into the objective function for dictionary learning. Additionally, the solution to the new objective function is efficiently achieved by simply employing the original K-SVD algorithm. Unlike most existing dictionary learning approaches that rely on iteratively solving sub-problems in order to approximate a global solution, our approach is able to learn the dictionary, discriminative coding parameters and classifier parameters simultaneously. We also propose an incremental dictionary learning approach with the label consistency constraint for the situation of limited memory resources.

The experimental results show that our approach yields very good classification results on six well-known public datasets with only one simple linear classifier, which is unlike some competing approaches learning multiple classifiers for categories to gain discrimination between classes. Our approach outperforms recently proposed methods including D-KSVD [16], SRC [6] and LLC [12], especially when the number of training samples is small.

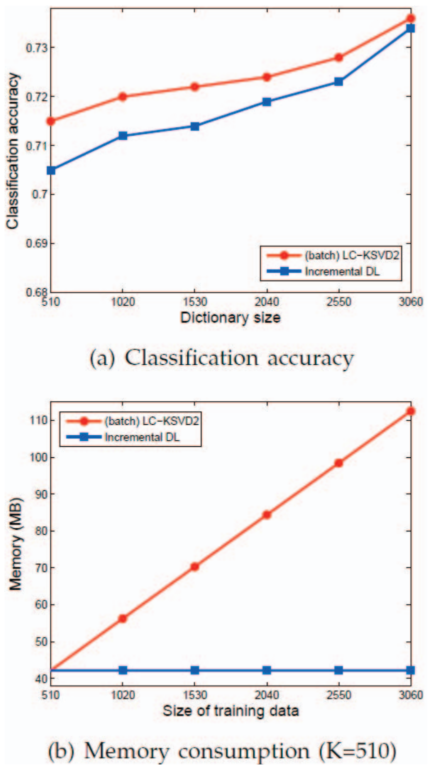


Fig. 13. The classification and memory consumption comparisons on the Caltech101 dataset between LC-KSVD2 and incremental dictionary learning approach.

Possible future work includes extending our approach to learn category-structure aware dictionaries, which encourage input signals from highly related categories to share visual patterns (i.e. dictionary items). It will make the classification of signals more 'close' to their true identities. The sparse representations of signals of the same category are defined to be strictly equal in variable Q, which corresponds to the single model of a feature distribution for one category. A more flexible approach is to consider the multi-modal distributions of signals of the same category when defining Q.

Acknowledgments

This work was funded, in part, by Army Research MURI project (contract number: W911NF-09-10383).

References

[1] M. Elad and M. Aharon, "Image denosing via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Processing*, vol. 54, no. 12, pp. 3736–3745, 2006.

[2] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun, "Efficient learning of sparse representations with an energy-based model," *Proc. Conf. Neural Information Processing Systems*, 2006.

[3] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *Journal of Machine Learning Research 11*, pp. 19–60, 2010.

[4] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image superresolution as sparse representation of raw patches," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[5] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[6] J. Wright, M. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Analysis and Machine Intellegence*, vol. 31, no. 2, pp. 210–227, 2009.

[7] D. Bradley and J. Bagnell, "Differential sparse coding," *Proc. Conf. Neural Information Processing Systems*, 2008.

[8] J. Yang and M. Yang, "Top-down visual saliency via joint crf and dictionary learning," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012.

[9] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictioanry learning," *IEEE Trans. Pattern Analysis and Machine Intellegence*, vol. 34, no. 4, pp. 791–804, 2012.

[10] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," *Proc. Conf. Neural Information Processing Systems*, 2006.

[11] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionries for sparse representation," *IEEE Trans. on Signal Processing*, vol. 54, no. 1, pp. 4311–4322, 2006.

[12] J. Wang, J. Yang, K. Yu, F. Lv, T. huang, and Y. Gong, "Locality-constrained linear coding for image classification," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[13] G. Karol and Y. LeCun, "Learning fast approximations of sparse coding," *Proc. Int'l Conf. Machine Learning*, 2010.

[14] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Discriminative learned dictionaries for local image analysis," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[15] D. Pham and S. Venkatesh, "Joint learning and dictionary construction for pattern recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[16] Q. Zhang and B. Li, "Discriminative k-svd for dictionary learning in face recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[17] K. Engan, S. Aase, and J. Husϕy, "Frame based signal compression using method of optimal directions (mod)," *Proc. IEEE Int'l Symp. Circuits and Systems*, 1999.

[18] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach, "Proximal methods for sparse hierarchical dictionary learning," *Proc. Int'l Conf. Machine Learning*, 2010.

[19] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," *Proc. Int'l Conf. Machine Learning*, 2007.

[20] K. Kavukcuoglu, M. Ranzato, R. Fergus, and Y. LeCun, "Learning invariant features through topographic filter maps," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[21] M. Yang, L. Zhang, X. Feng, and D. Zhang, "Fisher discrimination dictionary learning for sparse representation," *Proc. IEEE Int'l Conf. Computer Vision*, 2011.

[22] W. Zhang, A. Surve, X. Fern, and T. Dietterich, "Learning non-redundant codebooks for classifying complex objects," *Proc. Int'l Conf. Machine Learning*, 2009.

[23] L. Yang, R. Jin, R. Sukthankar, and F. Jurie, "Unifying discriminative visual codebook genearation with classifier training for object category recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[24] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce, "Discriminative sparse image models for class-specific edage detection and image interpretation," *Proc. European Conf. Computer Vision*, 2008.

[25] I. Ramirez, P. Sprechmann, and G. Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[26] N. Zhou, Y. Shen, J. Peng, and J. Fan, "Learning inter-related visual dictionay for object recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012.

[27] F. Perronnin, "Universal and adapted vocabularies for generic visual categorization," *IEEE Trans. Pattern Analysis and Machine Intellegence*, vol. 30, no. 7, pp. 1243–1256, 2008.

[28] R. sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos, "Positive definite dictionary learning for region covariances," *Proc. IEEE Int'l Conf. Computer Vision*, 2011.

[29] J. Winn, A. Criminisi, and T. Minka, "Object categorization by learned universal visual dictionary," *Proc. IEEE Int'l Conf. Computer Vision*, 2005.

[30] J. Liu and M. Shah, "Learning human actions via information maximization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[31] S. Lazebnik and M. Raginsky, "Supervised learning of quantizer codebooks by information loss minimization," *IEEE Trans. Pattern Analysis and Machine Intellegence*, vol. 31, no. 7, pp. 1294–1309, 2009.

[32] Q. Qiu, Z. Jiang, and R. Chellappa, "Sparse dictionary-based representation and recognition of action attributes," *Proc. IEEE Int'l Conf. Computer Vision*, 2011.

[33] B. Fulkerson, A. Vedaldi, and S. Soatto, "Localizing objects with smart dictionaries," *Proc. European Conf. Computer Vision*, 2008.

[34] N. Slonim and N. Tishby, "Agglomerative information bottleneck," *Proc. Conf. Neural Information Processing Systems*, 1999.

[35] A. Krause and V. Cevher, "Submodular dictionary selection for sparse representation," *Proc. Int'l Conf. Machine Learning*, 2010.

[36] Z. Jiang, G. Zhang, and L. Davis, "Submodular dictionary learning for sparse coding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012.

[37] J. Yang, K. Yu, and T. Huang, "Supervised translation-invariant sparse coding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[38] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," *Proc. Conf. Neural Information Processing Systems*, 2009.

[39] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[40] K. Huang and S. Aviyente, "Sparse representation for signal classification," *Proc. Conf. Neural Information Processing Systems*, 2007.

[41] X. Lian, Z. Li, C. Wang, B. Lu, and L. Zhang, "Probabilistic models for supervised dictinary learning," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[42] X. Lian, Z. Li, B. Lu, and L. Zhang, "Max-margin dictionary learning for multiclass image categorization," *Proc. European Conf. Computer Vision*, 2010.

[43] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proc. of IEEE*, 2009.

[44] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[45] G. Golub, P. Hansen, and D. O'leary, "Tikhonov regularization and total least squares," *SIM J. Matrix Anal. Appl.*, vol. 21, no. 1, pp. 185–194, 1999.

[46] E. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[47] N. Murata, "Statistical study on on-line learning," *On-line Learning in Neural Networks*, 1999.

[48] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007.

[49] S. Sadanand and J. Corso, "Action bank: A high-level representation of activity in video," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012.

[50] A. Georghiades, P. Belhumeur, and D. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Analysis and Machine Intellegence*, vol. 23, no. 6, pp. 643–660, 2001.

[51] A. Martinez and R. Benavente, "The ar face database," *CVC Technical Report 24*, 1998.

[52] L. FeiFei, R. Fergus, and P. Perona, "Learning generative visual models from few training samples: An incremental bayesian appoach tested on 101 object categories," *CVPR Workshop on Generative Model Based Vision*, 2004.

[53] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," *CIT Technical Report 7694*, 2007.

[54] M. Rodriguez, J. Ahmed, and M. Shah, "A spatio-temporal maximum average correlation height filter for action recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[55] H. Zhang, A. Berg, M. Maire, and J. Malik, "Svm-knn: Discriminative nearest neighbor classification for visual category recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006.

[56] O. Boiman, E. Shechtman, and M. Irani, "In defense of nearest-neighor based image classification," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[57] P. Jain, B. Kullis, and K. Grauman, "Fast image search for learned metrics," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[58] J. Gemert, J. Geusebroek, C. Veenman, and A. Smeulders, "Kernel codebooks for scene categorization," *Proc. European Conf. Computer Vision*, 2008.

[59] S. Gao, I. Tsang, L. Chia, and P. Zhao, "Laplacian sparse coding for image classification," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

[60] L. Yeffet and L. Wolf, "Local trinary patterns for human action recognition," *Proc. IEEE Int'l Conf. Computer Vision*, 2009.

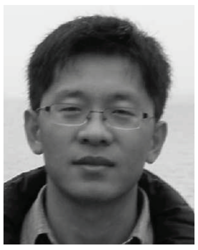
[61] X. Wu, D. Xu, L. Duan, and J. Luo, "Action recognition using context and appearance distribution features," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2011.

[62] A. Kovashka and K. Grauman, "Learning a hierarchy of discriminative spacetime neighborhood features for human action recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

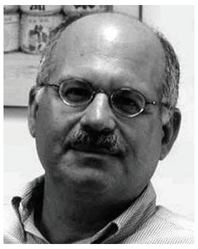
[63] Q. Le, W. Zou, S. Yeung, and A. Ng, "Learning hierarchical invariant spatiotemporal features for action recognition with independent subspace analysis," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2011.

[64] A. Yao, J. Gall, and L. V. Gool, "A hough transform-based voting framework for action recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010.

Zhuolin Jiang received the PhD degree in Computer Science from the South China University of Technology in 2010. He is currently working as a researcher in the Noah's Ark Lab of Huawei. He was an assistant research scientist in the Institute for Advanced Computer Studies at the University of Maryland from 2011 to 2012. His research interests include Computer Vision, Pattern Recognition and Machine Learning, specifically on human action detection and recognition, object detection and tracking, video content analysis and retrieval, sparse coding and sparse dictionary learning. He is a member of the IEEE.



Zhe Lin received the BEng degree in Automatic Control from the University of Science and Technology of China in 2002 and the MS degree in Electrical Engineering from the Korea Advanced Institute of Science and Technology in 2004, and the PhD degree in Electrical and Computer Engineering from the University of Maryland, College Park, in 2009. He has been a research intern at Microsoft Live Labs Research. He is currently working as a research scientist at the Advanced Technology Labs, Adobe Systems Incorporated, San Jose, California. His research interests include object detection and recognition, content-based image and video retrieval, human motion tracking and activity analysis. He is a member of the IEEE.



Larry S. Davis received the BA degree from Colgate University in 1970 and the MS and PhD degrees in computer science from the University of Maryland in 1974 and 1976, respectively. From 1977 to 1981, he was an assistant professor in the Department of Computer Science at the University of Texas, Austin. He returned to the University of Maryland as an associate professor in 1981. From 1985 to 1994, he was the director of the University of Maryland Institute for Advanced Computer Studies. He is currently a professor in the institute and in the Computer Science Department, as well as the chair of the Computer Science Department. He was named a fellow of the IEEE in 1997. He is known for his research in computer vision and high-performance computing. He has published more than 100 papers in journals and has supervised more than 20 PhD students. He is an associate editor of the International Journal of Computer Vision and an area editor for Computer Models for Image Processing: Image Understanding. He has served as the program or general chair for most of the fields major conferences and workshops, including the Fifth International Conference on Computer Vision, the 2004 Computer Vision and Pattern Recognition Conference, the 11th International Conference on Computer Vision. He is a fellow of the IEEE.