Design of Non-Linear Discriminative Dictionaries for Image Classification

Ashish Shrivastava, Hien V. Nguyen, Vishal M. Patel, and Rama Chellappa

UMIACS, University of Maryland, College Park, MD, USA

Abstract. In recent years there has been growing interest in designing dictionaries for image classification. These methods, however, neglect the fact that data of interest often has non-linear structure. Motivated by the fact that this non-linearity can be handled by the kernel trick, we propose learning of dictionaries in the high-dimensional feature space which are simultaneously reconstructive and discriminative. The proposed optimization approach consists of two main stages- coefficient update and dictionary update. We propose a kernel driven simultaneous orthogonal matching pursuit algorithm for the task of sparse coding in the feature space. The dictionary update step is performed using an efficient approximate KSVD algorithm in feature space. Extensive experiments on image classification demonstrate that the proposed non-linear dictionary learning method is robust and can perform significantly better than many competitive discriminative dictionary learning algorithms.

1 Introduction

Sparse and redundant signal representations have recently drawn much interest in vision and image processing fields [1]. This is due in part to the fact that objects and images of interest can be sparse or compressible in some basis of a dictionary. We say a signal $\mathbf{x}$ is sparse in dictionary $\mathbf{D}$ when it can be well represented as $\mathbf{x} = \mathbf{D}\boldsymbol{\alpha}$, where $\boldsymbol{\alpha}$ is the sparse representation vector and $\mathbf{D}$ is a dictionary that contains atoms as its columns. The dictionary $\mathbf{D}$ can be analytic such as a redundant Gabor dictionary or it can be trained directly from data. It has been observed that learning a dictionary directly from training data rather than using a predetermined dictionary usually leads to better representation and hence can provide improved results in many practical image processing applications such as restoration and classification [1], [2], [3]. Two of the most well-known algorithms for learning a dictionary are the method of optimal directions (MOD) [4] and the KSVD algorithm [5]. While these approaches are purely generative, the design of discriminative dictionaries has also gained a lot of interest in recent years. Linear discriminant analysis (LDA) based basis selection and feature extraction algorithm for classification using wavelet packets was originally proposed in [6]. More recently, many other methods have shown significant improvements over purely reconstructive dictionaries, e.g. [7], [8], [9], [10]. One of the major advantages of learning dictionaries which are simultaneously

reconstructive and discriminative is that they are known to be less sensitive to noise.

In many practical applications, data of interest lies on a non-linear structure. Linear dictionary learning methods such as MOD and KSVD are almost always inadequate for representing these nonlinear data. In [11], Nguyen *et al.* address this issue by learning dictionaries in the high dimensional feature space. Using kernel methods, they developed dictionary learning algorithms that take into account the nonlinear structure of data. They showed that their non-linear dictionary learning methods yield representations that are more compact than kernel PCA (Principal Component Analysis) and are able to handle the non-linearity better than their linear counterparts.

Several other methods have also been proposed that essentially exploit the non-linear structure of data by sparse coding in the feature space [12], [13]. In [14], Yuan and Yan propose a multi-task joint sparse representation for visual recognition. Their method is formulated as the solution to the problem of multi-task least squares regression problem with $\ell_{1,2}$ mixed-norm regularization. In [13], Zhang *et al.* propose a kernel version of the sparse representation-based classification algorithm which was originally proposed for robust face recognition [15].

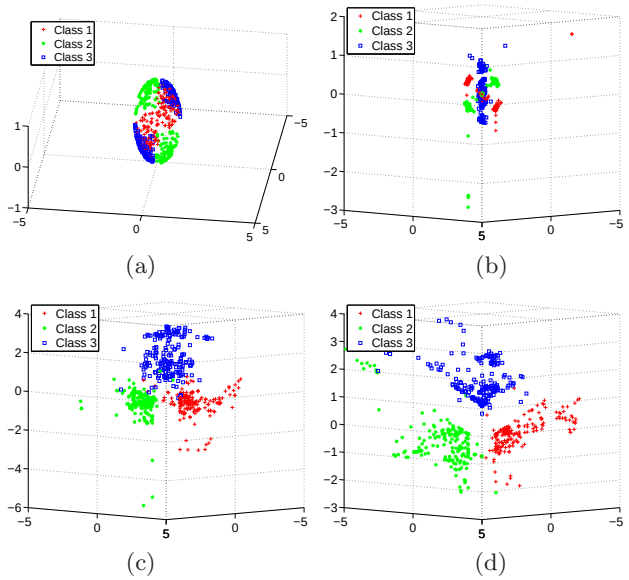


Fig. 1. A synthetic example showing the significance of the proposed method for classification. (a) Synthetic data which consists of linearly non separable 3D points on a sphere. Different classes are represented by different colors. (b) Sparse coefficients from KSVD projected onto learned SVM (Support Vector Machine) hyperplanes. (c) Sparse coefficients from a non-linear dictionary projected onto learned SVM hyperplanes. (d) Sparse coefficients from the proposed method projected onto learned SVM hyperplanes.

The optimization approach presented in [11] is purely generative. It does not explicitly contain the discrimination term which is important for many classification tasks. Using the kernel trick, when the data is transformed into a high dimensional feature space, the data from different classes may still overlap. Hence, by learning generative dictionaries for different classes in the feature space, one may not be able to capture the internal structure of the data in each class. This may lead to poor performance in classification. Motivated by this fact, we propose a method for designing dictionaries in the feature space which are simultaneously reconstructive and discriminative.

Figure 1 presents an important comparison in terms of discriminative power of our approach with a few other methods. A scatter plot of the sparse coefficients obtained using different approaches show that our method is able to learn the underlying non-linear sparsity of data as well provide more discriminative representation.
This paper makes the following contributions:

- A non linear coefficient update has been proposed which provides discriminative capability to the dictionary learning algorithm.
- A novel non-linear dictionary update algorithm based on approximate KSVD has been presented.
- These two stages provide the framework of non-linear dictionary learning to classify data in feature space.

2 Problem Formulation

We represent the data matrix as $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_C] = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{d \times N}$ where $\mathbf{x}_i \in \mathbb{R}^d$ is a d dimensional data sample, $\mathbf{X}_c \in \mathbb{R}^{d \times N_c}$ is the matrix of data samples in c^{th} class, and $\mathbf{y} = [y_1, \ldots, y_N]$, where y_i is the class label of data sample $\mathbf{x}_i$ and C is the number of classes. We denote the number of samples in c^{th} class by N_c and total number of training samples by N, i.e. $N = N_1 + \cdots + N_C$. Linear dictionary $\mathbf{D}$ is denoted as the concatenation of C sub-dictionaries $\mathbf{D} = [\mathbf{D}_1 \mathbf{D}_2 \ldots \mathbf{D}_C] \in \mathbb{R}^{d \times K}$ where $\mathbf{D}_c \in \mathbb{R}^{d \times K_c}$ is the dictionary for class c and K_c is the number of atoms in this dictionary. Here, $K (= K_1 + \cdots + K_C)$ is the total number of atoms in the dictionary $\mathbf{D}$. Similarly, the coefficient matrix is denoted by $\mathbf{\Gamma} = \begin{bmatrix} \mathbf{\Gamma}_1 \\ \vdots \\ \mathbf{\Gamma}_C \end{bmatrix} = [\boldsymbol{\gamma}_1, \ldots \boldsymbol{\gamma}_N] \in \mathbb{R}^{K \times N}$, where $\mathbf{\Gamma}_c \in \mathbb{R}^{K_c \times N}$ is the coefficient matrix corresponding to the c^{th} class and $\boldsymbol{\gamma}_i$ is the coefficient vector for the i^{th} data sample.

In the following sub-sections, we first present our formulation for designing a linear discriminative dictionary which, then, sets the ground for non-linear discriminative dictionary.

2.1 Linear discriminative dictionary learning model

In the linear case, we propose the following discriminative dictionary learning model

$$\hat{\mathbf{D}}, \hat{\mathbf{\Gamma}} = \min_{\mathbf{D},\mathbf{\Gamma}} J(\mathbf{D}, \mathbf{\Gamma}; \mathbf{X}, \mathbf{y}) \text{ subject to } \|\boldsymbol{\gamma}_i\|_0 \leq T_0, \tag{1}$$

where the ℓ_0 norm $\|\boldsymbol{\gamma}\|_0$ counts the number of nonzero elements in the representation $\boldsymbol{\gamma}$ and T_0 is a sparsity measure. The objective function J is designed such that it captures discrimination as well as representation. It imposes Fisher type of discrimination on the sparse coefficients and enforces separability among dictionary atoms of different classes. It is defined as follows

$$J(\mathbf{D}, \mathbf{\Gamma}; \mathbf{X}, \mathbf{y}) = \sum_{c=1}^{C} \|\mathbf{X}_c - \mathbf{D}_c\mathbf{\Gamma}_c\|_F^2 - \lambda_1 F_1(\mathbf{D}, \mathbf{X}) + \lambda_2 F_2(\mathbf{D}), \tag{2}$$

Here, $\|.\|_F$ denotes the Frobenius norm. The first term in Eq. (2) reduces the representation error. The second term essentially ensures that the sparse coding coefficients have small within-class scatter but large between-class scatter. Mathematically, it is defined as follows

$$F_1(\mathbf{D}, \mathbf{X}) = trace(\mathbf{D}^T\mathbf{S}_b\mathbf{D} - \mathbf{D}^T\mathbf{S}_w\mathbf{D}), \tag{3}$$

where $\mathbf{S}_b = \frac{1}{N}\sum_1^C N_c(\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^T$ and $\mathbf{S}_w = \frac{1}{N}\sum_{c=1}^C \sum_{\substack{i=1 \\ y_i=c}}^N (\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^T$ are the between-class and within-class data scatter matrices, respectively. Here, $\mathbf{m}_c = \frac{1}{N_c}\sum_{\substack{i=1 \\ y_i=c}}^N \mathbf{x}_i$ is the mean of c^{th} class data samples and $\mathbf{m} = \frac{1}{N}\sum_{i=1}^N \mathbf{x}_i$ is the mean of all the data samples. With this, F_1 can be rewritten as

$$F_1(\mathbf{D}, \mathbf{X}) = \sum_{k=1}^{K} \mathbf{d}_k^T\mathbf{S}_b\mathbf{d}_k - \sum_{k=1}^{K} \mathbf{d}_k^T\mathbf{S}_w\mathbf{d}_k. \tag{4}$$

Where, $\mathbf{d}_k$ is the k^{th} atom of dictionary $\mathbf{D}$. Finally, the third term in (2) is defined such that it enforces dissimilarity among atoms of different classes

$$F_2(\mathbf{D}) = \sum_{\substack{c=1 \\ j\neq c}}^{C} \|\mathbf{D}_c^T\mathbf{D}_j\|_F^2, \tag{5}$$

where $\mathbf{D}_c$ refers to the set of those atoms which belong to class c. This enforces dissimilarity between the atoms of different classes.

2.2 Non-linear discriminative dictionary (NLDD)

Let $\mathbf{\Phi} : \mathbb{R}^N \rightarrow G$ be a non-linear mapping from $\mathbb{R}^N$ into a higher dimensional feature space G. Since the feature space G can be very high dimensional, in the kernel methods, Mercer kernels are usually employed to carry out the mapping

implicitly. A Mercer kernel is a function $\tau(\mathbf{x}_1, \mathbf{x}_2)$ that for all data $\{\mathbf{x}_i\}$ gives rise to a positive semidefinite matrix $\boldsymbol{\mathcal{K}}(i,j) = \tau(\mathbf{x}_i, \mathbf{x}_j)$. It can be shown that using τ instead of dot product in input space corresponds to mapping the data with some mapping $\boldsymbol{\Phi}$ into a feature space G. That is, $\tau(\mathbf{x}_i, \mathbf{x}_j) = \langle \boldsymbol{\Phi}(\mathbf{x}_i), \boldsymbol{\Phi}(\mathbf{x}_j) \rangle$. Some commonly used kernels include polynomial kernels $\tau(\mathbf{x}, \mathbf{y}) = \langle (\mathbf{x}, \mathbf{y} \rangle + c)^d$ and Gaussian kernels $\tau(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{c})$, where c and d are the parameters. We now show how the discriminative dictionary learning framework (1) can be kernelized.

We will use the following model for the dictionary in the feature space

$$\boldsymbol{\Phi}(\mathbf{D}) = \boldsymbol{\Phi}(\mathbf{X})\mathbf{A},$$

where $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K] \in \mathbb{R}^{N \times K}$ and $\boldsymbol{\Phi}(\mathbf{X}) = [\boldsymbol{\Phi}(\mathbf{x}_1)\boldsymbol{\Phi}(\mathbf{x}_2) \ldots \boldsymbol{\Phi}(\mathbf{x}_N)]$. This model provides adaptivity via modification of the matrix $\mathbf{A}$ [16],[11]. First note that $\|\boldsymbol{\Phi}(\mathbf{x}_i) - \boldsymbol{\Phi}(\mathbf{X})\mathbf{A}_i\boldsymbol{\gamma}_i\|_2^2$ can be kernelized as follows,

$$\|\boldsymbol{\Phi}(\mathbf{x}_i) - \boldsymbol{\Phi}(\mathbf{X})\mathbf{A}_i\boldsymbol{\gamma}_i\|_2^2 = \boldsymbol{\mathcal{K}}(\mathbf{x}_i, \mathbf{x}_i) + \boldsymbol{\gamma}_i^T\mathbf{A}_i^T\boldsymbol{\mathcal{K}}\mathbf{A}_i\boldsymbol{\gamma}_i - 2\boldsymbol{\gamma}_i\mathbf{A}_i\boldsymbol{\mathcal{K}}(\mathbf{X}, \mathbf{x}_i) \tag{6}$$

Next, the term $\mathbf{d}_k^T\mathbf{S}_b\mathbf{d}_k$ in (2) can be written in the feature space as

$$(\boldsymbol{\Phi}(\mathbf{X})\mathbf{a}_k)^T\Big(\frac{1}{N}\sum_{c=1}^{C} N_c(\mathbf{m}_c^{feat} - \mathbf{m}^{feat})(\mathbf{m}_c^{feat} - \mathbf{m}^{feat})^T\Big)\boldsymbol{\Phi}(\mathbf{X})\mathbf{a}_k, \tag{7}$$

where $\mathbf{m}_c^{feat} = \frac{1}{N_c}\sum_{\substack{i=1 \\ y_i=c}}^{N} \boldsymbol{\Phi}(\mathbf{x}_i)$ and $\mathbf{m}^{feat} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\Phi}(\mathbf{x}_i)$ are the mean vectors in the feature space. Equation (7) can be simplified as, $\mathbf{a}_k^T\mathbf{S}_b^{ker}\mathbf{a}_k$, where

$$\mathbf{S}_b^{ker} = \frac{1}{N}\sum_{c=1}^{C} N_c(\mathbf{m}_c^{ker} - \mathbf{m}^{ker})(\mathbf{m}_c^{ker} - \mathbf{m}^{ker})^T, \tag{8}$$

$\mathbf{m}_c^{ker} = \frac{1}{N_c}\sum_{\substack{i=1 \\ y_i=c}}^{N} \boldsymbol{\mathcal{K}}(\mathbf{X}, \mathbf{x}_i)$ and $\mathbf{m}^{ker} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\mathcal{K}}(\mathbf{X}, \mathbf{x}_i)$. Similarly, $\mathbf{d}_k^T\mathbf{S}_w\mathbf{d}_k$ (2) can be kernelized as $\mathbf{a}_k^T\mathbf{S}_w^{ker}\mathbf{a}_k$, where

$$\mathbf{S}_w^{ker} = \frac{1}{N}\sum_{c=1}^{C}\sum_{\substack{i=1 \\ y_i=c}}^{N} (\boldsymbol{\mathcal{K}}(\mathbf{X}, \mathbf{x}_i) - \mathbf{m}_c^{ker})(\boldsymbol{\mathcal{K}}(\mathbf{X}, \mathbf{x}_i) - \mathbf{m}_c^{ker})^T. \tag{9}$$

Finally, to kernelize the term in (5), we observe that the dot product of any two dictionary atoms in feature space can be written as

$$(\boldsymbol{\Phi}(\mathbf{X})\mathbf{a}_i)^T(\boldsymbol{\Phi}(\mathbf{X})\mathbf{a}_j) = \mathbf{a}_i^T\boldsymbol{\mathcal{K}}\mathbf{a}_j. \tag{10}$$

Equipped with the above notations, the main problem can be formally stated as follows

$$\hat{\mathbf{A}}, \hat{\boldsymbol{\Gamma}} = \min_{\mathbf{A}, \boldsymbol{\Gamma}} J(\mathbf{A}, \boldsymbol{\Gamma}; \mathbf{X}, \mathbf{y}) \quad \text{subject to } \|\boldsymbol{\gamma}_i\|_0 \leq T_0, \forall i \in \{1, \ldots, N\}, \tag{11}$$

where

$$J(\mathbf{A},\mathbf{\Gamma};\mathbf{X},\mathbf{y}) = \sum_{c=1}^{C} \|\mathbf{\Phi}(\mathbf{X}_c) - \mathbf{\Phi}(\mathbf{X})\mathbf{A}_c\mathbf{\Gamma}_c\|_F^2 - \lambda_1 F_1(\mathbf{A},\mathbf{S}_b^{ker},\mathbf{S}_w^{ker}) + \lambda_2 F_2(\mathbf{A},\boldsymbol{\mathcal{K}}), \tag{12}$$

$$F_1(\mathbf{A},\mathbf{S}_b^{ker},\mathbf{S}_w^{ker}) = \sum_{k=1}^{K} \mathbf{a}_k^T\mathbf{S}_b^{ker}\mathbf{a}_k - \sum_{k=1}^{K} \mathbf{a}_k^T\mathbf{S}_w^{ker}\mathbf{a}_k, \tag{13}$$

$$F_2(\mathbf{A},\boldsymbol{\mathcal{K}}) = \sum_{\substack{c=1 \\ j\neq c}}^{C} \|\mathbf{A}_c^T\boldsymbol{\mathcal{K}}\mathbf{A}_j\|_F^2. \tag{14}$$

Here, $\mathbf{A}_c$ correspond to the atoms corresponding to class c. As the cost function (12) is jointly non-convex in $\mathbf{A}$ and $\mathbf{\Gamma}$, we adopt alternating optimization between coefficients $\mathbf{\Gamma}$ and $\mathbf{A}$. In what follows, we described these steps in detail.

3 Computing Coefficients

To compute coefficients, we fix $\mathbf{A}$ and find the best atoms indexed by I on which we project the data. Hence, we solve the following optimization problem

$$\begin{aligned} \mathbf{\Gamma}^* = \arg\min_{\mathbf{\Gamma}} &\|\mathbf{\Phi}(\mathbf{X}) - \mathbf{\Phi}(\mathbf{X})\mathbf{A}\mathbf{\Gamma}\|_F^2 - \lambda_1 F_1(\mathbf{A},\mathbf{S}_b^{ker},\mathbf{S}_w^{ker}) + \lambda_2 F_2(\mathbf{A}) \\ &\text{subject to} \quad \|\boldsymbol{\gamma}_i\|_0 \leq T_0, \qquad \forall i \in \{1,\ldots,N\}. \end{aligned} \tag{15}$$

In what follows, we describe how the well-known Simultaneous Orthogonal Matching Pursuit (SOMP) algorithm [17] can be extended to solve (15).

3.1 Supervised SOMP

Given a fixed dictionary $\mathbf{D}$ and examples $\mathbf{X}$, SOMP approximates all these samples at once using different linear combinations of the dictionary elements, while balancing the error in approximating the data against the total number of atoms that are used. It is a greedy algorithm that essentially solves the following optimization problem

$$\mathbf{\Gamma}^* = \arg\min_{\mathbf{\Gamma}} \|\mathbf{X} - \mathbf{D}\mathbf{\Gamma}\|_F^2 \quad \text{s. t.} \quad \|\mathbf{\Gamma}\|_{\text{row-0}} \leq T_0, \tag{16}$$

The SOMP algorithm can be extended to the supervised case where it includes the second and third terms of the cost function (2). The supervised SOMP can be obtained by changing the atom selection stage as follows

$$m = \arg\max_{p\in U} \sum_{\substack{i=1 \\ y_i=c}}^{N} |\langle \mathbf{r}_i^{(t-1)}, \mathbf{d}_p\rangle| + \lambda_1(\mathbf{d}_p^T\mathbf{S}_b\mathbf{d}_p - \mathbf{d}_p^T\mathbf{S}_w\mathbf{d}_p) - \lambda_2 \sum_{\substack{j=1 \\ j\neq c}}^{C} \|\mathbf{D}_j^T\mathbf{d}_p\|_2^2. \tag{17}$$

where m is the index of the selected atom at the current iteration and $\mathbf{r}_i^{(t-1)}$ is the residual for i^{th} sample at $(t-1)^{\text{th}}$ iteration. In the next section, we show how this supervised SOMP algorithm can be kernelized to solve (15).

3.2 Supervised Kernel SOMP (KSOMP)

Note that residue for the i^{th} sample in the feature space is

$$res_i = \mathbf{\Phi}(\mathbf{x}_i) - \mathbf{\Phi}(\mathbf{X})\mathbf{A}_I\boldsymbol{\gamma}_i^R, \tag{18}$$

where $\mathbf{A}_I$ is the set of selected elements indexed by I and $\boldsymbol{\gamma}_i^R$ is the corresponding coefficient vector. Here, superscript R denotes that the vector has been reduced to length of I so that the remaining elements correspond to the atoms in $\mathbf{A}_I$. As will be evident later, this residue always appear as a dot product with $\mathbf{\Phi}(\mathbf{X})$. Hence we define,

$$\begin{aligned}\mathbf{r}_i &= \mathbf{\Phi}(\mathbf{X})^T\big(\mathbf{\Phi}(\mathbf{x}_i) - \mathbf{\Phi}(\mathbf{X})\mathbf{A}_I\boldsymbol{\gamma}_i^R\big) &(19)\\ &= \boldsymbol{\mathcal{K}}(\mathbf{X},\mathbf{x}_i) - \boldsymbol{\mathcal{K}}(\mathbf{X},\mathbf{X})\mathbf{A}_I\boldsymbol{\gamma}_i^R &(20)\end{aligned}$$

The projection of the residue on the dictionary atom $\mathbf{a}_p$ can be computed as,

$$\langle\mathbf{\Phi}(\mathbf{X})\mathbf{a}_p, res_i\rangle = \mathbf{a}_p^T\Big(\boldsymbol{\mathcal{K}}(\mathbf{X},\mathbf{x}_i) - \boldsymbol{\mathcal{K}}(\mathbf{X},\mathbf{X})\mathbf{A}_I\boldsymbol{\gamma}_i^R\Big) = \langle\mathbf{r}_i,\mathbf{a}_p\rangle. \tag{21}$$

Equation 21 is the counter part of the dot product of candidate atom and residual in the feature space. Combining (21), with the definitions of scatter matrices in (8) and (9) we can write the atom selection stage of supervised kernel SOMP as

$$m = \arg\max_{p\in U}\sum_{\substack{i=1\\y_i=c}}^{N}|\langle\mathbf{r}_i^{(t-1)},\mathbf{a}_p\rangle| + \lambda_1(\mathbf{a}_p^T\mathbf{S}_b^{ker}\mathbf{a}_p - \mathbf{a}_p^T\mathbf{S}_w^{ker}\mathbf{a}_p) + \lambda_2\sum_{\substack{j=1\\j\neq c}}^{C}\|\mathbf{A}_j^T\boldsymbol{\mathcal{K}}\mathbf{a}_p\|_2^2. \tag{22}$$

Finally, after selecting I_c atoms of the c^{th} class, the coefficients of the i^{th} data sample can be computed as,

$$\begin{aligned}\boldsymbol{\gamma}_i^R &= \Big((\mathbf{\Phi}(\mathbf{X})\mathbf{A}_{I_c})^T(\mathbf{\Phi}(\mathbf{X})\mathbf{A}_{I_c})\Big)^{-1}\big((\mathbf{\Phi}(\mathbf{X})\mathbf{A}_{I_c})^T\mathbf{\Phi}(x_i)\big)\\ &= \Big(\mathbf{A}_{I_c}^T\boldsymbol{\mathcal{K}}\mathbf{A}_{I_c}\Big)^{-1}\Big(\mathbf{A}_{I_c}^T\boldsymbol{\mathcal{K}}(\mathbf{X},\mathbf{x}_i)\Big). &(23)\end{aligned}$$

The supervised kernel SOMP algorithm to compute coefficients $\mathbf{\Gamma}$ is summarized in Algorithm 1.

4 Dictionary update

When $\mathbf{\Gamma}$ is fixed, we ignore F_1 and F_2 and solve the following optimization problem to update the dictionary

$$\mathbf{A}^* = \arg\min_{\mathbf{A}}\|\mathbf{\Phi}(\mathbf{X}) - \mathbf{\Phi}(\mathbf{X})\mathbf{A}\mathbf{\Gamma}\|_F^2. \tag{24}$$

This update can be computed efficiently by utilizing approximate kernel-KSVD algorithm [16] in the feature space.

Algorithm 1: supervised KSOMP

Input: $\mathbf{A}, \mathbf{h}, \mathcal{K}, \mathbf{y}, T_0$.
Output: $\mathbf{\Gamma}$
Initialization: residual $(\forall i,)\mathbf{r}_i = \mathcal{K}(:,i)$, Set $(\forall c), I_c = [\,]$, $\mathbf{\Gamma} = \mathbf{0}^{K\times N}$, $\mathbf{Q}_c = [\,]$
for $t = 1, \dots T_0$ **do**
  for $c = 1, \dots C$ **do**
    $U = \{p = 1, \dots, K\} \backslash I_c$
    compute m using Eq. 22
    Set $I_c = I_c \cup m$
    Compute $\boldsymbol{\gamma}_i^R$ using Eq 23 $\quad \forall i,$ such that $y_i = c.$
    Set $\mathbf{r}_i = \mathcal{K}(:,i) - \mathcal{K}\mathbf{A}_{I_c}\boldsymbol{\gamma}_i^R$ $\quad \forall i,$ such that $y_i = c$
    $\mathbf{\Gamma}(I_c, i) \leftarrow \boldsymbol{\gamma}_i^R,$ $\quad \forall i,$ such that $y_i = c$
  end
end
return $\mathbf{\Gamma}$

4.1 Approximate kernel KSVD

The following optimization problem is solved for each $\mathbf{a}_k$,

$$\mathbf{a}_k^* = \arg\min_{\mathbf{a}_k} \|\mathbf{\Phi}(\mathbf{X}) - \mathbf{\Phi}(\mathbf{X})\Big(\sum_{i\neq k} \mathbf{a}_i\boldsymbol{\gamma}_T^i + \mathbf{a}_k\boldsymbol{\gamma}_T^k\Big)\|_F^2 \tag{25}$$

$$= \arg\min_{\mathbf{a}_k} \|\mathbf{\Phi}(\mathbf{X})\mathbf{E}_k - \mathbf{\Phi}(\mathbf{X})\mathbf{a}_k\boldsymbol{\gamma}_T^k\|_F^2, \tag{26}$$

where $\mathbf{E}_k = \mathbf{I} - \sum_{i\neq k} \mathbf{a}_i\boldsymbol{\gamma}_T^i$. Furthermore, we need to consider only those samples which use $\mathbf{a}_k$. To do this we define the index set $I_k = \{j | 1 \leq j \leq K, \boldsymbol{\gamma}_T^k(j) \neq 0\}$ and a matrix $\mathbf{\Omega}_k \in \mathbb{R}^{N\times|I_k|}$ with 1's in the $(I_k(j), j)^{\text{th}}$ entry and 0's elsewhere.

Algorithm 2: Dictionary update stage using approximate kernel KSVD

Input: kernel matrix $\mathcal{K} \in \mathbb{R}^{N\times N}$, input labels $\mathbf{y} \in \mathbb{R}^{1\times N}$ initial dictionary $\mathbf{A}_0 \in \mathbb{R}^{N\times K}$, coefficients matrix $\mathbf{\Gamma} \in \mathbb{R}^{K\times N}$.
Output: $\mathbf{A}$
Initialization: $\mathbf{A} \leftarrow \mathbf{A}_0$,
for $k = 1, \dots K$ **do**
  $J = \{j \mid 1 \leq j \leq N, \mathbf{\Gamma}(k,j) \neq 0\}$
  $\mathbf{I}^R = \mathbf{I}(:, J)$
  $\boldsymbol{\gamma}_k^R = (\mathbf{\Gamma}(k, J))^T$
  $\mathbf{a}_k = \mathbf{I}^R\boldsymbol{\gamma}_k^R - \mathbf{A}\mathbf{\Gamma}_J\boldsymbol{\gamma}_k^R$
  $\mathbf{a}_k = \frac{\mathbf{a}_k}{\sqrt{\mathbf{a}_k^T\mathcal{K}\mathbf{a}_k}}$
  $\boldsymbol{\gamma}_k^R = (\mathbf{I}^R)^T\mathcal{K}\mathbf{a}_k - (\mathbf{A}\mathbf{\Gamma}_J)^T\mathcal{K}\mathbf{a}_k$
  $\mathbf{\Gamma}(k, J) = (\boldsymbol{\gamma}_k^R)^T$
  $\mathbf{A}(:,k) = \mathbf{a}_k$
end
return $\mathbf{A}$

Next, we define the reduced matrix $\mathbf{E}_k^R \triangleq \mathbf{E}_k\mathbf{\Omega}_k$, which consists of only those columns that use $\mathbf{a}_k$. Similarly, we reduce the length of the coefficient vector γ_k and define a new column vector as $\boldsymbol{\gamma}_k^R = (\boldsymbol{\gamma}_T^k\mathbf{\Omega}_k)^T$. With this the optimization problem can be rewritten as

$$\mathbf{a}_k^* = \arg\min_{\mathbf{a}_k} \|\mathbf{\Phi}(\mathbf{X})\mathbf{E}_k^R - \mathbf{\Phi}(\mathbf{X})\mathbf{a}_k(\boldsymbol{\gamma}_k^R)^T\|_F^2 \quad \text{s.t.} \quad \|\mathbf{\Phi}(\mathbf{X})\mathbf{a}_k\|_2 = 1. \tag{27}$$

We use alternate optimization (fixing one variable and differentiating with respect to the other one) to compute $\mathbf{a}_k$ and $\boldsymbol{\gamma}_k^R$

$$\mathbf{a}_k = \frac{\mathbf{E}_k^R\boldsymbol{\gamma}_k^R}{\sqrt{(\mathbf{E}_k^R\boldsymbol{\gamma}_k^R)^T\boldsymbol{\mathcal{K}}\mathbf{E}_k^R\boldsymbol{\gamma}_k^R}}, \qquad \boldsymbol{\gamma}_k^R = (\mathbf{E}_k^R)^T\boldsymbol{\mathcal{K}}\mathbf{a}_k. \tag{28}$$

We summarize the procedure for approximate kernel KSVD in Algorithm 2.

Algorithm 3: Non-linear discriminative dictionary (NLDD) learning

Input: Training Data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, class labels $\mathbf{y} = [y_1, \ldots, y_N]$, sparsity level T_0, parameters λ_1, λ_2.
Output: $\mathbf{A} = [\mathbf{A}_1, \ldots, \mathbf{A}_C]$
Step 1: (*Initialization*) Initialize each column of $\mathbf{A}$ with 1 at a random location. Compute kernel matrix $\boldsymbol{\mathcal{K}}$, data scatter matrices $\mathbf{S}_b^{ker}$ and $\mathbf{S}_w^{ker}$.
Step 2: Compute sparse coefficients using Algorithm 1.
Step 3: Using sparse coefficients from step 2, update $\mathbf{A}$ using Algorithm 2 .
Step 4: Repeat steps 2 and 4 for pre-specified number of iterations.
Step 5: Remove those columns of $\mathbf{A}$ which were not used in last iteration.
return A

The complete NLDD algorithm of dictionary learning has been summarized in Algorithm 3

5 Classification

Once the dictionary has been learned, we use one of the following two methods of classification depending on the number of training samples per class.

Case 1: (Large number of training samples per class) When we have a large number of training samples per class (e.g digit recognition, gender recognition), we compute the per class reconstruction errors as follows

$$\epsilon_c(\mathbf{x}_t) = \|\mathbf{\Phi}(\mathbf{x}_t) - \mathbf{\Phi}(\mathbf{X})\mathbf{A}_c\boldsymbol{\gamma}_t^*\|_2 \qquad \text{subject to} \quad \|\gamma_t\|_0 = T_0 \tag{29}$$

where $\mathbf{A}_c$ is the matrix corresponding to class c, $\mathbf{x}_t$ is the test sample, and $\boldsymbol{\gamma}_t^*$ is the corresponding sparse coefficient computed using the kernel OMP algorithm [11] which solves the following problem

$$\boldsymbol{\gamma}_t^* = \min_{\boldsymbol{\gamma}} \|\mathbf{x}_t - \mathbf{\Phi}(\mathbf{X})\mathbf{A}_c\boldsymbol{\gamma}\|_2 \qquad \text{subject to} \quad \|\gamma_t\|_0 = T_0. \tag{30}$$

Once the reconstruction errors are computed, the classification is done as follows

$$\text{class of } \mathbf{x}_t = \arg\min_c \epsilon_c(\mathbf{x}_t). \tag{31}$$

Case 2: (Small number of training samples per class) When the number of training samples per class is relatively small, one class is not expressive enough for a given test sample. Hence, while computing the sparse coefficients, we use the whole dictionary $\mathbf{D}$ and solve the following optimization problem,

$$\boldsymbol{\gamma}_t^* = \min_{\boldsymbol{\gamma}} \|\boldsymbol{\Phi}(\mathbf{x}_t) - \boldsymbol{\Phi}(\mathbf{X})\mathbf{A}\boldsymbol{\gamma}\|_2 \qquad \text{subject to} \quad \|\gamma_t\|_0 = T_0. \tag{32}$$

Now the reconstruction error for class c is computed by using elements of $\boldsymbol{\gamma}_t^*$ which correspond to the c^{th} class. This can be written as,

$$\epsilon_c(\mathbf{x}_t) = \|\boldsymbol{\Phi}(\mathbf{x}_t) - \boldsymbol{\Phi}(\mathbf{X})\mathbf{A}_c\delta_c(\boldsymbol{\gamma}_t^*)\|_2 \tag{33}$$

where $\delta_c(.)$ is a characteristic function that selects the coefficients corresponding to class c. Classification based on only reconstruction error may be misleading in cases where two classes have very similar reconstruction errors. In such cases, we make the decision in favor of the class which gets the biggest contribution from the coefficient vector. To quantify this, we define a quantity $w_c(\boldsymbol{\gamma}_t^*) \triangleq \frac{\|\delta_c(\boldsymbol{\gamma}_t^*)\|_1}{\|\boldsymbol{\gamma}_t^*\|_1}$. The final classification is then done as

$$\text{class of } \mathbf{x}_t = \arg\min_c(\epsilon_c(\mathbf{x}_t) - \eta\, w_c(\boldsymbol{\gamma}_t^*)), \tag{34}$$

where η is a constant that measures the importance of coefficient based classification.

6 Experiments and results

In this section, we present several experimental results demonstrating the effectiveness of the proposed dictionary learning method for classification tasks. In particular, we present classification results on the AR face dataset [18], the extended Yale B face dataset [19][20], and the USPS digits [21] dataset. The comparison with other existing discriminative dictionary learning methods for image classification in [7] suggests that Fisher discrimination-based dictionary learning (FDDL) algorithm is among the best. Hence, we treat it as state-of-the-art and use it as a bench mark for comparisons in this paper. We also compare our method with that of kernel PCA (KPCA) [22] and kernel LDA (KLDA) [23]. In all of our experiments, we set the sparsity at test time to approximately 10% of the dictionary size. The dictionary size is chosen according to the size of available training data. When available data samples per class are relatively small (e.g. AR face recognition), we set the dictionary size same as the number of training samples. Conversely, when we have a large number of training samples per class, we limit the dictionary size to 100. The discriminative parameters λ_1 and λ_2 were experimentally selected so that they provide the best results. We set λ_1 and λ_2 equal to 0.7 and 0.4, respectively for all the experiments. For all the face recognition experiments we use the Gaussian kernel with $\sigma = 1.6$ and for digit recognition experiments we use the polynomial kernel of degree 4.

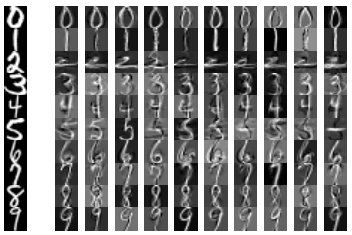
6.1 Digit Recognition

In the first set of experiments, we evaluate the performance of our method on the USPS digit dataset and compare it with some recent state-of-the-art methods namely, KSVD, FDDL [7], and kernel based methods KPCA, KLDA, kernel KSVD [11]. This dataset consists of 7291 training and 2007 test images. In this experiment, we randomly pick 200 training samples per class and 100 test samples per class. The results are shown in Table 1. It can be seen from the table that our method performs the best on this experiment with the USPS dataset.

Algorithm	KSVD	FDDL	KPCA	KLDA	ker KSVD	NLDD
Accuracy (%)	96.10	97.00	96.30	96.90	96.90	**97.50**

Table 1. Recognition accuracy for the proposed method, compared to competing ones for digit recognition.

Pre-images of learned atoms: Recall that the k^{th} kernel dictionary atom is represented by $\mathbf{\Phi}(\mathbf{X})\mathbf{a}_k$, where $\mathbf{a}_k \in \mathbb{R}^N$ is the representation of the kernel dictionary atom with respect to the base $\mathbf{\Phi}(\mathbf{X})$ in the feature space G. The pre-image of $\mathbf{\Phi}(\mathbf{X})\mathbf{a}_k$ is obtained by seeking a vector in the input space $\mathbf{d}_k \in mathbbR^N$ that minimizes the cost function $\|\mathbf{\Phi}(\mathbf{d}_k) - \mathbf{\Phi}(\mathbf{X})\mathbf{a}_k\|^2$. Due to various noise effects and the generally non-invertible mapping $\mathbf{\Phi}$, the exact pre-image does not always exist. However, the approximated pre-image can be reconstructed without venturing into the feature space using the techniques described in [23].

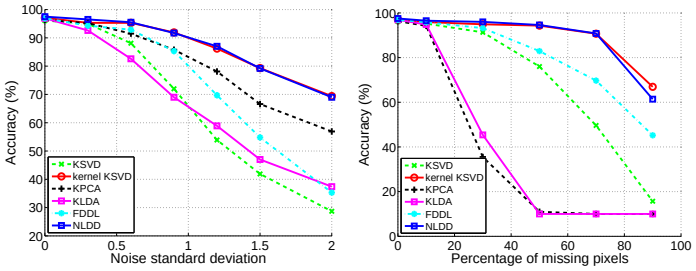


(a) (b)

Fig. 2. Pre-images of the learned dictionary atoms corresponding to the NLDD method. (a) class examples. (b) preimages of 10 dictionary atoms from each class.

Figure 2 shows the pre-images of the learned atoms corresponding the NLDD method. Note that our method is able to capture the internal common structure of data while maintaining the discriminative capability.

Robustness of NLDD: In this section, we evaluate the performance of the proposed method in the presence of various degradations such as missing pixels and noise. From each class, we randomly select 200 images for training and 100 images for testing. The first experiment presents the results for the situation where the test samples are corrupted by random Gaussian noise with different standard deviations as shown in Figure 3(a). The results obtained when pixels are randomly removed from the test images are shown in Figure 3(b).



(a) (b)

Fig. 3. Analysis of algorithms with noisy test data. (a) Accuracy with Gaussian noise. (b) Accuracy with missing pixels.

In both experiments, Kernel-based dictionary learning methods such as kernel KSVD and NLDD perform much better than the other methods. As the distortion level increases the performance difference between kernel dictionaries and linear dictionaries become more dramatic. This demonstrates that non-linear dictionary learning can prove significantly better in some scenarios. Experiments described later show that having discriminative power along with non-linearity in dictinary learning can provide extra performance for classification.

6.2 AR Face Recognition

The AR dataset consists of 126 individuals with frontal faces captured in two sessions with different illuminations, expressions and occlusions. We follow the experimental setup of [7] and choose 50 male subjects and 50 female subjects with lighting and expression variations. The 7 images per subject from session 1 were used for training and 7 images with same lighting and expressions from session 2 were used for testing. The dimension of the images was reduced to 300 using PCA.

Datset	SRC	SVM	DKSVD	FDDL	KPCA	KLDA	ker KSVD	NLDD
AR Face	88.8	87.1	85.4	92.0	83.86	92.14	92.57	**93.71**
Yale B	90.0	88.8	75.3	91.9	88.09	92.02	91.84	**94.62**

Table 2. Recognition accuracy for the proposed method, compared to competing ones for AR and Yale B face recognition.

The results are shown in Table 2 which shows around 2% improvement over FDDL and more than 1% improvement over kernel KSVD-based classification. Since the number of training images per class is only 7 we use the whole dictionary (case 2) for classification. Sparsity at test time was set to 23 and η in (34) was set equal to 0.5. As can be seen from the table, our kernel-based discriminative dictionary learning method provides the best results. Here, SRC stands for sparse representation-based classifier and DKSVD for discriminative KSVD.

6.3 Extended Yale B Face Recognition

In this experiment we evaluate our algorithm on the extended Yale B dataset which has 38 subjects and about 64 images per subject with various illumination conditions. We follow the experimental set up as considered in [7]. We randomly select 20 images per subject for training and the rest for testing. Dimension of all the images have been reduced to 300 using PCA. Dictionary size of each class was set to 20 and at the test time, we use sparsity $T0 = 35$ and $\eta = 0.5$ for classification.

As shown in Table 2, NLDD performs the best and shows an improvement of about 3% over other competitive methods. This experiment shows that even in the presence of extreme illumination, our method is able to provide reasonable recognition performance.

6.4 Gender Recognition

In the final set of experiments, we evaluate the performance of our method on a two class problem of gender recognition. We choose 50 male subjects and 50 female subjects of the AR face database. We choose 14 faces per subject from both sessions. We train our algorithm, with first 25 males subjects and 25 female subjects and test our method with the remaining 25 male and 25 female subjects. The feature dimension was reduced to 300 using PCA. Results are presented in Table 3.

Algo.	SRC	SVM	DKSVD	FDDL	KPCA	KLDA	ker KSVD	NLDD
Acc. (%)	93.0	92.4	86.1	95.4	94.57	94.57	95.07	**95.71**

Table 3. Recognition accuracy for the proposed method, compared to competing ones for gender recognition recognition.

Since we have enough training samples per class, we classify based on the reconstruction error from each class (case 1). In this experiment, the sparsity was set equal to 30 for training as well as testing. As shown in the table, our method performance favorably over some of the competitive methods.

7 Discussions and future work

We have proposed an approach for learning discriminative and reconstructive dictionaries in a high dimensional features space. The proposed algorithm consists of two steps that iteratively update sparse coefficients and dictionary atoms. Sparse coefficients are updated using a variant of SOMP algorithm and the dictionary atoms are updated using an efficient kernel KSVD algorithm. Various experiments on popular face and digit recognition data sets have shown that our method is robust and can perform significantly better than many existing dictionary based recognition algorithms.

Acknowledgement

This work was partially supported by an ONR grant N00014-12-1-0124.

References

1. Rubinstein, R., Bruckstein, A., Elad, M.: Dictionaries for sparse representation modeling. Proceedings of the IEEE **98** (2010)
2. Chen, Y.C., Patel, V.M., Phillips, P.J., Chellappa, R.: Dictionary-based face recognition from video. European Conference on Computer Vision (2012)
3. Patel, V.M., Wu, T., Biswas, S., Philips, P.J., Chellappa, R.: Dictionary-based face recognition under variable lighting and pose. IEEE Transactions on Information Forensics and Security **7** (2012) 954–965
4. Engan, K., Aase, S.O., Husoy, J.H.: Method of optimal directions for frame design. ICASSP (1999)
5. Aharon, M., Elad, M., Bruckstein, A.M.: The K-SVD: an algorithm for designing of overcomplete dictionaries for sparse representation. IEEE Trans. Signal Process. **54** (2006)
6. Etemand, K., Chellappa, R.: Separability-based multiscale basis selection and feature extraction for signal and image classification. TIP (1998)
7. Yang, M., Zhang, L., Feng, X., Zhang, D.: Fisher discrimination dictionary learning for sparse representation. In: ICCV. (2011)
8. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Supervised dictionary learning. In: NIPS. (2009)
9. Ramirez, I., Sprechmann, P., Sapiro, G.: Classification and clustering via dictionary learning with structured incoherence and shared features. CVPR (2010)
10. Patel, V.M., Chellappa, R.: Sparse representations, compressive sensing and dictionaries for pattern recognition. ACPR (2011)
11. Nguyen, H.V., Patel, V.M., Nasrabadi, N.M., Chellappa, R.: Kernel dictionary learning. In: ICASSP. (2012)
12. Gao, S., Tsang, I.W., Chia, L.T.: Kernel sparse representation for image classification and face recognition. ECCV (2010)
13. Zhang, L., Zhou, W.D., Chang, P.C., Liu, J., Yan, Z., Wang, T., Li, F.Z.: Kernel sparse representation-based classifier. IEEE Trans. Signal Process. (2011)
14. Yuan, X.T., Yan, S.: Visual classification with multi-task joint sparse representation. CVPR (2010)
15. Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y.: Robust face recognition via sparse representation. PAMI **31** (Feb. 2009)
16. Rubinstein, R., Zibulevsky, M., Elad, M.: Double sparsity: Learning sparse dictionaries for sparse signal approximation. IEEE Trans. Signal Process. **58** (2010)
17. Tropp, J.A., Gilbert, A.C., Strauss, M.J.: Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. Signal Processing **86** (2006)
18. Martinez, A., Benavente, R.: The ar face database. CVC Technical Report No. 24, (1998)
19. Georghiades, A., Belhumeur, P., Kriegman, D.: From few to many: Illumination cone models for face recognition under variable lighting and pose. PAMI (2001)
20. Lee, K., Ho, J., Kriegman, D.: Acquiring linear subspaces for face recognition under variable lighting. IEEE Trans. Pattern Anal. Mach. Intelligence **27** (2005)
21. : Usps handwriten digit database. (`http://www-i6.informatik.rwth-aachen.de/~keysers/usps.html`)
22. Schölkopf, B., Smola, A., Müller, K.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput. **10** (1998)
23. Scholkopf, B., Smola, A.J.: Learning With Kernels, Support Vector Machines, Optimization, and Beyond. MIT Press (2001)