Non-Linear Dictionary Learning with Partially Labeled Data

Ashish Shrivastava, Vishal M. Patel, Rama Chellappa

Center for Automation Research, UMIACS, University of Maryland, College Park, MD 20742

Abstract

While recent techniques for discriminative dictionary learning have demonstrated tremendous success in image analysis applications, their performance is often limited by the amount of labeled data available for training. Even though labeling images is difficult, it is relatively easy to collect unlabeled images either by querying the web or from public datasets. Using the kernel method, we propose a non-linear discriminative dictionary learning technique which utilizes both labeled and unlabeled data for learning dictionaries in the high-dimensional feature space. Furthermore, we show how this method can be extended for ambiguously labeled classification problem where each training sample has multiple labels and only one of them is correct. Extensive evaluation on existing datasets demonstrate that the proposed method performs significantly better than state of the art dictionary learning approaches when unlabeled images are available for training.

Keywords: Weakly-supervised learning, semi-supervised learning, kernel methods, dictionary learning, classification.

Email addresses: ashish@umiacs.umd.edu (Ashish Shrivastava), pvishalm@umiacs.umd.edu (Vishal M. Patel), rama@umiacs.umd.edu (Rama Chellappa)

1. Introduction

Sparse and redundant signal representations have recently gained much interest in computer vision field [34], [12], [27]. This is partly due to the fact that signals or images of interest are often sparse with respect to some dictionary. These dictionaries can be either analytic or they can be learned directly from the data. In fact, it has been observed that learning a dictionary directly from data often leads to improved results in many practical applications such as classification and restoration [34], [22], [6].

While dictionaries are often trained to obtain good reconstruction, training supervised dictionaries with a specific discriminative criterion has also been considered. For instance, linear discriminant analysis (LDA)-based basis selection and feature extraction algorithm for classification using wavelet packets was proposed by Etemand and Chellappa [14] in the late nineties. Recently, similar algorithms for simultaneous sparse signal representation and discrimination have also been proposed [25], [15], [24][38], [17], [18], [36], [21], [37].

Sparse representation and dictionary learning methods for unsupervised learning have also been proposed. In [33], a method for simultaneously learning a set of dictionaries that optimally represent each cluster is proposed. To improve the accuracy of sparse coding, this approach was later extended by adding a block incoherence term in their optimization problem [23]. Some of the other sparsity motivated clustering and subspace clustering methods include [13], [8].

The performance of a supervised classification algorithm is often depen-

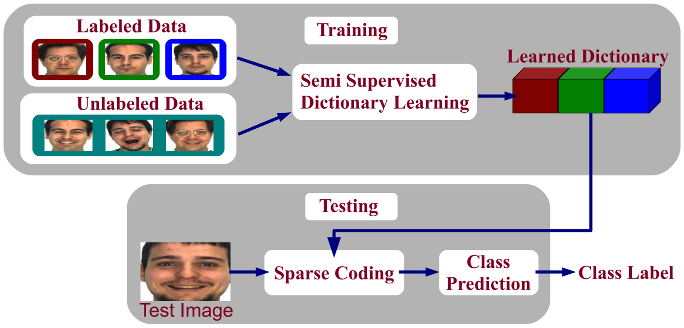


Figure 1: Block diagram illustrating semi-supervised dictionary learning.

dent on the quality and diversity of training images, which are mainly hand-labeled. However, labeling images is expensive and time consuming due to the significant human effort involved. On the other hand, one can easily obtain large amounts of unlabeled images from public image datasets like Flickr or by querying image search engines like Bing. This has motivated researchers to develop semi-supervised algorithms, which utilize both labeled and unlabeled data for learning classifier models. Such methods have demonstrated improved performance when the amount of labeled data is limited. See [4] for an excellent survey of recent efforts on semi-supervised learning.

Two of the most popular methods for semi-supervised learning are Co-Training [2] and Semi-Supervised Support Vector Machines (S3VM) [32]. Co-Training assumes the presence of multiple views for each feature and uses the confident samples in one view to update the other. However, in applications such as image classification, one often has just a single feature vector and hence it is difficult to apply Co-Training. S3VM considers the labels of the unlabeled data as additional unknowns and jointly optimizes over the classifier parameters and the unknown labels in the SVM framework

[3].

Using the kernel trick, several methods have been proposed in the literature that exploit sparsity of data in the high dimensional feature space. In these methods, a preselected Mercer kernel is used to map the input data onto a features space where dictionaries are trained. It has been shown that such non-linear dictionaries can provide better discrimination than their linear counterparts [20], [30], [19].

Motivated by the success of non-linear dictionary learning methods [20], [30], we propose a novel method to learn kernel discriminative dictionaries for classification in a semi-supervised manner. Fig. 1 shows the block diagram of the proposed approach which uses both labeled and unlabeled data. While learning a dictionary, we maintain a probability distribution over class labels for each unlabeled data. The discriminative part of the cost is made proportional to the confidence over the assigned label of the participating training sample. This makes the proposed method robust to label assignment errors.

This paper makes the following contributions[1]:

1. We propose a discriminative dictionary learning method that utilizes both labeled and unlabeled data.
2. Using the kernel trick, we extend the formulation for learning linear dictionaries with labeled and unlabeled data to the non-linear case. An efficient optimization procedure is proposed for solving this non-linear dictionary learning problem.
3. We show how the proposed method can be extended to ambiguously

[1] Preliminary version of this work appeared in [31]. Items 2 and 3 are extensions to [31].

labeled data where each training sample has multiple labels and only one of them is correct.

In our previous work [30], we developed a supervised non-linear discriminative dictionary learning method for image classification. The method proposed in this paper is different from [30] in that it is a general non-linear semi-supervised dictionary learning method. The methods proposed for learning dictionaries form ambiguously labeled data [7] are also different from the one proposed in this paper. Specifically, in [7] two linear methods are proposed - one based on soft decision rules and the other based on hard decision rules. In contrast to linear reconstructive dictionary leaning methods in [7] and [38], we propose a general discriminative non-linear kernel dictionary learning method for semi-supervised learning.

The rest of the paper is organized as follows. In Section 2, we formulate the problem of non-linear dictionary learning with partially labeled data. The optimization of the proposed framework is presented in Section 3. Experimental results are presented in Section 4 and Section 5 concludes the paper with a brief summary and discussion.

2. Problem Formulation

In this section, we formulate the optimization problem for learning discriminative dictionaries with partially labeled data. We first present the linear formulation. We then extend it to the non-linear case.

2.1. Linear Dictionary Learning with Partially Labeled Data

Let $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N] \in \mathbb{R}^{d \times N}$ be the data matrix where d is the dimension of each data sample $\mathbf{y}_i$ and N is the total number of training samples. We assume that the data is partially labeled and denote the label of the i^{th} sample by l_i. When the sample $\mathbf{y}_i$ is not labeled, we set l_i to 0, i.e., $l_i \in \{0, 1, \ldots C\}$, where C is the total number of classes.

Our goal is to learn a dictionary $\mathbf{D} \in \mathbb{R}^{d \times K}$, where K is the number of unit norm atoms. We represent this dictionary as the concatenation of all the classes' dictionary, i.e. $\mathbf{D} \triangleq [\mathbf{D}_1 | \ldots | \mathbf{D}_C]$ such that each $\mathbf{D}_c \in \mathbb{R}^{d \times K_c}$ can represent the c^{th} class data well while not economically representing the other class data. Here, K_c is the number of atoms in dictionary $\mathbf{D}_c$, and hence, $K = \sum_{c=1}^{C} K_c$. Enforcing each $\mathbf{D}_c$ to represent only its own class c improves the discriminative capability of the learned dictionary. We represent each sample $\mathbf{y}_i$ by sparse linear combination of dictionary $\mathbf{D}$'s atoms and represent the sparse coefficient of the i^{th} sample by $\mathbf{x}_i$. Furthermore, we denote the coefficient matrix for all the samples by $\mathbf{X}$, i.e., $\mathbf{X} \triangleq [\mathbf{x}_1, \ldots, \mathbf{x}_N]$.

In order to deal with unlabeled data, we introduce a probability matrix $\mathbf{P} \in \mathbb{R}^{C \times N}$ such that each column of $\mathbf{P}$ represents the class distribution of the corresponding data sample. In other words, $(c, i)^{\text{th}}$ element P_{ci} of $\mathbf{P}$ denotes the probability of the i^{th} sample belonging to class c. Hence, by definition,

$$\begin{aligned}
&P_{ci} = 1 \text{ if } \mathrm{y}_i \text{ is labeled with one class and } l_i = c. \\
&P_{ci} = 0 \text{ if } \mathrm{y}_i \text{ is labeled with one class and } l_i \neq c. \\
&0 \leq P_{ci} \leq 1 \text{ if } \mathbf{y}_i \text{ is unlabeled or ambiguously labeled.} \qquad (1)
\end{aligned}$$

We denote the probability of all the samples belonging to class c by a diagonal matrix $\mathbf{P}_c \in \mathbb{R}^{N \times N}$ such that $\mathbf{P}_c(i, i) = P_{ci}$ and the non-diagonal elements of $\mathbf{P}_c$ are set equal to zeros. Also, we define a matrix $\mathbf{Q}_c \triangleq 1 - \mathbf{P}_c$

to denote the probability of all the samples not belonging to the c^{th} class. Furthermore, we define $\mathbf{P}_c^{sqrt}$ and $\mathbf{Q}_c^{sqrt}$ the square root of $\mathbf{P}_c$ and $\mathbf{Q}_c$, respectively, i.e., $\mathbf{P}_c = \mathbf{P}_c^{sqrt}\mathbf{P}_c^{sqrt}$ and $\mathbf{Q}_c = \mathbf{Q}_c^{sqrt}\mathbf{Q}_c^{sqrt}$. The Frobenius norm and the sparsity promoting ℓ_1 norm of a matrix $\mathbf{A}$ are denoted as $\|\mathbf{A}\|_F$ and $\|\mathbf{A}\|_1$, respectively.

Equipped with these notations, we formulate the dictionary learning problem as one of optimizing

$$\mathcal{J}_0(\mathbf{D}, \mathbf{X}, \mathbf{P}) = \mathcal{F}_0(\mathbf{Y}, \mathbf{D}, \mathbf{X}, \mathbf{P}) + \mathcal{H}(\mathbf{X}, \mathbf{P}) + \lambda_1 \|\mathbf{X}\|_1, \tag{2}$$

where,

$$\begin{aligned}\mathcal{F}_0(\mathbf{Y}, \mathbf{D}, \mathbf{X}, \mathbf{P}) = {} & \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \\ & + \tau_1 \sum_{c=1}^{C} \|\big(\mathbf{Y} - \mathbf{D}_c\mathbf{X}^c\big)\mathbf{P}_c^{sqrt}\|_F^2 \\ & + \tau_2 \sum_{c=1}^{C} \|\mathbf{D}_c\mathbf{X}^c\mathbf{Q}_c^{sqrt}\|_F^2,\end{aligned} \tag{3}$$

$$\mathcal{H}(\mathbf{X}, \mathbf{P}) = \lambda_2\big(tr(S_w(\mathbf{X}, \mathbf{P}) - S_b(\mathbf{X}, \mathbf{P}))\big) + \eta\|\mathbf{X}\|_F^2, \tag{4}$$

and $\mathbf{X}^c$ is the coefficient matrix corresponding to the cth class. Here, the first term of $\mathcal{F}_0$ encourages $\mathbf{D}$ to be a good representative of the data matrix $\mathbf{Y}$ without needing any label information. The second term of $\mathcal{F}_0$ enforces that the c^{th} class dictionary $\mathbf{D}_c$ represents well those samples which are likely to belong to class c. Note that $\mathbf{P}_c^{sqrt}$ is a diagonal matrix and hence the contribution of each sample in this part of the cost is proportional to the probability of it having come from the c^{th} class. The third part of $\mathcal{F}_0$ enlarges the reconstruction error of those samples which are less likely to have come from the c^{th} class. The parameters τ_1 and τ_2 control the discriminative capability of the learned dictionary.

The second term $\mathcal{H}$ of $\mathcal{J}_0$ in (2) makes the sparse coefficients of samples discriminative by decreasing the trace of within-class scatter matrix

$$S_w = \sum_{c=1}^{C} \sum_{i:l_i=c} (\mathbf{x}_i - \mathbf{m}_c)(\mathbf{x}_i - \mathbf{m}_c)^T$$

and increasing the trace of between-class scatter matrix

$$S_b = \sum_{c=1}^{C} N_c(\mathbf{m}_c - \mathbf{m})(\mathbf{m}_c - \mathbf{m})^T,$$

where $\mathbf{m}_c$ is the average of the c^{th} class coefficients, $\mathbf{m}$ is the average of all the coefficients and N_c is the number of samples in class c. However, when the label information is available in the form of probability matrix, these scatter matrices can be defined as follows

$$\begin{aligned} S_w(\mathbf{X}, \mathbf{P}) &= \sum_{c=1}^{C} (\mathbf{X} - \mathbf{M}_c)\mathbf{P}_c(\mathbf{X} - \mathbf{M}_c)^T \\ &= \sum_{c=1}^{C} (\mathbf{X} - \mathbf{X}\mathbf{E}_c)\mathbf{P}_c(\mathbf{X} - \mathbf{X}\mathbf{E}_c)^T, \end{aligned} \tag{5}$$

where $\mathbf{E}_c \in \mathbb{R}^{N \times N}$ has N repeated column and each of them, denoted by $\mathbf{e}_c$, has the following form,

$$\mathbf{e}_c(i) = \frac{P_{ci}}{w_c}, \quad \text{where } w_c = \sum_{i=1}^{N} P_{ci}, \tag{6}$$

and

$$S_b(\mathbf{X}, \mathbf{P}) = \sum_{c=1}^{C} w_c(\mathbf{X}\mathbf{e}_c - \mathbf{X}\mathbf{b})(\mathbf{X}\mathbf{e}_c - \mathbf{X}\mathbf{b})^T, \tag{7}$$

where, $\mathbf{b}(i) = \frac{1}{N}, \forall i = 1, \ldots, N$. Note that $\mathbf{X}\mathbf{e}_c$ is the average of the c^{th} class coefficients and $\mathbf{X}\mathbf{b}$ is the average of all the coefficients.

In (4), $tr(.)$ denotes the matrix trace operator and an elastic term $\|\mathbf{X}\|_F^2$ is added to make the cost with respect to $\mathbf{X}$ convex and stable. Similar formulations have been used in [35, 14]. The last term of $\mathcal{J}_0$ enforces the sparsity of coefficients. Finally, λ_1, λ_2 and η are the parameters controlling sparsity of coefficients, discriminability of sparse codes and elastic term, respectively.

2.2. Non-Linear Dictionary Learning

Let $\mathbf{\Phi} : \mathbb{R}^d \rightarrow G$ be a non-linear mapping from d-dimensional space into a dot product space G. Dictionary learning algorithm can be formulated in the feature space by writing $\mathbf{D} = \mathbf{\Phi}(\mathbf{Y})\mathbf{A}$, where $\mathbf{A} \in \mathbb{R}^{N \times K}$ is a matrix with K columns [20], [30]. By changing the columns of $\mathbf{A}$, we can learn the dictionary atoms in the feature space. Hence, the columns of $\mathbf{A}$ are referred to as atoms and denoted by $\mathbf{a}_k$, with $k = 1, \ldots, K$. The k^{th} atom in the feature space can be written as $\mathbf{\Phi}(\mathbf{Y})\mathbf{a}_k$. In order to enforce unit norm constraint on the atoms in the feature space, $\mathbf{a}_k \boldsymbol{\mathcal{K}} \mathbf{a}_k$ should be equal to 1 for all k. Also, we define $\mathbf{A}$ as the concatenation of C matrices, one for each class, i.e., $\mathbf{A} = [\mathbf{A}_1 | \ldots | \mathbf{A}_C]$. Next, we can change $\mathcal{F}_0$ and denote it by $\mathcal{F}$ such that,

$$\begin{aligned}\mathcal{F}(\mathbf{Y},\mathbf{A},\mathbf{X},\mathbf{P}) = &\|\mathbf{\Phi}(\mathbf{Y}) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}\mathbf{X}\|_F^2 \\ &+ \tau_1 \sum_{c=1}^{C} \|\big(\mathbf{\Phi}(\mathbf{Y}) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{X}^c\big)\mathbf{P}_c^{sqrt}\|_F^2 \\ &+ \tau_2 \sum_{c=1}^{C} \|\big(\mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{X}^c\big)\mathbf{Q}_c^{sqrt}\|_F^2. \end{aligned} \tag{8}$$

As we will see later, each of the terms in $\mathcal{F}$ containing $\mathbf{\Phi}(\mathbf{Y})$ can be written in terms of the dot products $\mathbf{\Phi}(\mathbf{Y})^T\mathbf{\Phi}(\mathbf{Y})$. This allows us to use the kernel trick by writing $\mathbf{\Phi}(\mathbf{Y})^T\mathbf{\Phi}(\mathbf{Y}) = \boldsymbol{\mathcal{K}}(\mathbf{Y},\mathbf{Y}) \in \mathbb{R}^{N\times N}$, where, $\boldsymbol{\mathcal{K}}$ is the kernel matrix whose $(i,j)^{\text{th}}$ element measures the similarity between $\mathbf{y}_i$ and $\mathbf{y}_j$ by means of a mercer kernel function denoted by $\kappa(\mathbf{y}_i,\mathbf{y}_j) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Some commonly used kernels include polynomial kernels

$$\kappa(\mathbf{y}_i,\mathbf{y}_j) = (\mathbf{y}_i^T\mathbf{y}_j + a)^b$$

and Gaussian kernels

$$\kappa(\mathbf{y}_i^T\mathbf{y}_j) = \exp\Big(\frac{\|\mathbf{y}_i - \mathbf{y}_j\|^2}{c}\Big),$$

where a, b and c are the parameters of the kernel functions. The overall cost for the non-linear dictionary learning can be written as follows

$$\mathcal{J}(\mathbf{A},\mathbf{X},\mathbf{P}) = \mathcal{F}(\mathbf{Y},\mathbf{A},\mathbf{X},\mathbf{P}) + \mathcal{H}(\mathbf{X},\mathbf{P}) + \lambda_1\|\mathbf{X}\|_1. \tag{9}$$

Having proposed the formulation for learning non-linear dictionaries with partially labeled data, we describe our approach to optimize the cost in (9).

3. Optimization of the Proposed Formulation

Our optimization problem is to minimize the cost in (9) with respect to dictionary $\mathbf{A}$, sparse coefficient matrix $\mathbf{X}$ and probability matrix $\mathbf{P}$,

$$\begin{aligned} \hat{\mathbf{A}}, \hat{\mathbf{X}}, \hat{\mathbf{P}} &= \arg\min_{\mathbf{A},\mathbf{X},\mathbf{P}} \mathcal{J}(\mathbf{A}, \mathbf{X}, \mathbf{P}) \\ \text{subject to} \qquad \mathbf{a}_k^T \boldsymbol{\mathcal{K}} \mathbf{a}_k &= 1, \quad \forall k = 1, \ldots, K. \end{aligned} \tag{10}$$

Equation (10) is jointly non-convex in all the three variable. Hence, we resort to optimizing one variable at a time, while keeping the other two fixed.

3.1. Optimization of the Dictionary $\mathbf{A}$

When the coefficient matrix $\mathbf{X}$ and the probability matrix $\mathbf{P}$ are fixed, we optimize $\mathbf{A}$ one class at a time. To optimize the c^{th} class dictionary, we write the cost $\mathcal{J}$ with respect to $\mathbf{A}_c$ as

$$\begin{aligned} \mathcal{J}_{\mathbf{A}_c} = &\|\boldsymbol{\Phi}(\mathbf{Y}) - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{X}^c - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{A}_o\mathbf{X}^o\|_F^2 \\ &+ \tau_1 \|\big(\boldsymbol{\Phi}(\mathbf{Y}) - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{X}^c\big)\mathbf{P}_c^{sqrt}\|_F^2 \\ &+ \tau_2 \|\big(\boldsymbol{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{X}^c\big)\mathbf{Q}_c^{sqrt}\|_F^2, \end{aligned} \tag{11}$$

where, $\mathbf{Y}_o$ and $\mathbf{A}_o$ denote the other class (i.e. not c) data matrix and dictionary, respectively. $\mathbf{X}^o$ denotes the coefficient matrix corresponding to $\mathbf{A}_o$. These matrices are defined as,

$$\mathbf{Y}_o \triangleq [\mathbf{Y}_1, \ldots, \mathbf{Y}_{c-1}, \mathbf{Y}_{c+1}, \ldots, \mathbf{Y}_C], \tag{12}$$

$$\mathbf{A}_o \triangleq [\mathbf{A}_1, \ldots, \mathbf{A}_{c-1}, \mathbf{A}_{c+1}, \ldots, \mathbf{A}_C], \tag{13}$$

$$\mathbf{X}^o \triangleq [\mathbf{X}^{1^T}, \ldots, \mathbf{X}^{c-1^T}, \mathbf{X}^{c+1^T}, \ldots, \mathbf{X}^{C^T}]^T, \tag{14}$$

where $\mathbf{Y}_c \in \mathbb{R}^{d \times N_c}$ is part of the data matrix consisting of samples from the c^{th} class. To update $\mathbf{A}$, we solve the following optimization problem for all $c = 1, \dots, C$,

$$\hat{\mathbf{A}}_c = \arg\min_{\mathbf{A}_c} \mathcal{J}_{\mathbf{A}_c} \tag{15}$$

$$\text{subject to} \quad \mathbf{a}_k^T \boldsymbol{\mathcal{K}} \mathbf{a}_k = 1, \quad \forall k = 1, \dots, K_c, \tag{16}$$

where $\mathbf{a}_k$ is the k^{th} columns of $\mathbf{A}_c$.

Next, we optimize one atom at a time while keeping the others fixed. The cost with respect to $\mathbf{a}_k$ can be written as

$$\begin{aligned} \mathcal{J}_{\mathbf{a}_k} = \|\mathbf{Z}_c^{\boldsymbol{\Phi}} - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{a}_k\mathbf{x}^k\|_F^2 + \tau_1 \|(\mathbf{U}_c^{\boldsymbol{\Phi}} - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{a}_k\mathbf{x}^k)\mathbf{P}_c^{sqrt}\|_F^2 \\ + \tau_2 \|\boldsymbol{\Phi}(\mathbf{Y})(\mathbf{a}_k\mathbf{x}^k + \mathbf{W}_c)\mathbf{Q}_c^{sqrt}\|_F^2, \end{aligned} \tag{17}$$

where,

$$\begin{aligned} \mathbf{W}_c &:= \sum_{j \neq k} \mathbf{A}_c(:, j)\mathbf{X}^c(j, :), \\ \mathbf{Z}_c^{\boldsymbol{\Phi}} &:= \boldsymbol{\Phi}(\mathbf{Y}) - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{A}_o\mathbf{X}^o - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{W}_c, \quad \text{and} \\ \mathbf{U}_c^{\boldsymbol{\Phi}} &:= \boldsymbol{\Phi}(\mathbf{Y}) - \boldsymbol{\Phi}(\mathbf{Y})\mathbf{W}_c. \end{aligned} \tag{18}$$

Writing $\mathcal{J}_{\mathbf{a}_k}$ in kernel form (and ignoring the terms independent of $\mathbf{a}_k$), we get

$$\begin{aligned} \mathcal{J}_{\mathbf{a}_k} = \operatorname{tr}[\mathbf{x}^k\mathbf{x}^{k^T}\mathbf{a}_k^T\boldsymbol{\mathcal{K}}\mathbf{a}_k - 2\mathbf{a}_k^T\big(\boldsymbol{\mathcal{K}} - \boldsymbol{\mathcal{K}}\mathbf{A}_o\mathbf{X}^o - \boldsymbol{\mathcal{K}}\mathbf{W}_c\big)\mathbf{x}^{k^T}] \\ + \tau_1 \operatorname{tr}[\mathbf{a}_k^T\boldsymbol{\mathcal{K}}\mathbf{a}_k\mathbf{x}^k\mathbf{P}_c\mathbf{x}^{k^T} - 2\mathbf{a}_k^T\big(\boldsymbol{\mathcal{K}} - \boldsymbol{\mathcal{K}}\mathbf{W}_c\big)\mathbf{P}_c\mathbf{x}^{\mathbf{k}^T}] \\ + \tau_2 \operatorname{tr}[\mathbf{a}_k^T\boldsymbol{\mathcal{K}}\mathbf{a}_k\mathbf{x}^{\mathbf{k}}\mathbf{Q}_c\mathbf{x}^{\mathbf{k}^T} + 2\mathbf{a}_k^T\boldsymbol{\mathcal{K}}\mathbf{W}_c\mathbf{Q}_c\mathbf{x}^{k^T}]. \end{aligned} \tag{19}$$

To optimize, $\mathscr{J}_{\mathbf{a}_k}$, subject to $\mathbf{a}_k\boldsymbol{\mathcal{K}}\mathbf{a}_k = 1$, we write the Lagrange function as

$$\mathcal{L}(\mathbf{a}_k, \gamma) = \mathscr{J}_{\mathbf{a}_k} + \gamma(\mathbf{a}_k\boldsymbol{\mathcal{K}}\mathbf{a}_k - 1), \tag{20}$$

where, γ is a Lagrange multiplier. Next, we take the derivative of $\mathcal{L}(.)$ with respect to $\mathbf{a}_k$ and set it equal to zero

$$\begin{aligned}\alpha.\boldsymbol{\mathcal{K}}\mathbf{a}_k =& [\boldsymbol{\mathcal{K}} - \boldsymbol{\mathcal{K}}\mathbf{A}_o\mathbf{X}^o - \boldsymbol{\mathcal{K}}\mathbf{W}_c]\mathbf{x}^{k^T} \\ &+ \tau_1[\boldsymbol{\mathcal{K}} - \boldsymbol{\mathcal{K}}\mathbf{W}_c]\mathbf{P}_c\mathbf{x}^{\mathbf{k}^T} - \tau_2\boldsymbol{\mathcal{K}}\mathbf{W}_c\mathbf{Q}_c\mathbf{x}^{k^T},\end{aligned} \tag{21}$$

where α is a scalar constant. Denoting the right hand side of the above equation by $\boldsymbol{\mathcal{K}}\mathbf{v}$, we get $\alpha.\mathbf{a}_k = \mathbf{v}$, where,

$$\mathbf{v} \triangleq [\mathbf{I} - \mathbf{A}_o\mathbf{X}^o - \mathbf{W}_c]\mathbf{x}^{k^T} + \tau_1[\mathbf{I} - \mathbf{W}_c]\mathbf{P}_c\mathbf{x}^{k^T} - \tau_2\mathbf{W}_c\mathbf{Q}_c\mathbf{x}^{k^T},$$

and along with the constraint $\mathbf{a}_k^T\boldsymbol{\mathcal{K}}\mathbf{a}_k = 1$, we choose the dual variable γ, and hence α, such that the condition is satisfied. In other words,

$$\mathbf{a}_k = \frac{\mathbf{v}}{\|\mathbf{v}\|_2}. \tag{22}$$

3.2. Optimization of the Coefficient Matrix $\mathbf{X}$

With the fixed dictionary $\mathbf{A}$, and the probability matrix $\mathbf{P}$, the cost in (9) can be re-written with respect to $\mathbf{X}$ as,

$$\begin{aligned}\mathscr{J}_{\mathbf{X}} =& \mathcal{F}_1(\mathbf{X}) + \tau_1\mathcal{F}_2(\mathbf{X}) + \tau_2\mathcal{F}_3(\mathbf{X}) + \\ &\lambda_2\mathcal{H}_1(\mathbf{X}) + \lambda_2\mathcal{H}_2(\mathbf{X}) + \eta\|\mathbf{X}\|_F^2 + \lambda_1\|\mathbf{X}\|_1,\end{aligned} \tag{23}$$

where,

$$
\begin{aligned}
&\mathcal{F}_1 = \|\mathbf{\Phi}(\mathbf{Y}) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}\mathbf{X}\|_F^2,\\
&\mathcal{F}_2 = \sum_{c=1}^{C} \|\big(\mathbf{\Phi}(\mathbf{Y}) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{X}^c\big)\mathbf{P}_c^{sqrt}\|_F^2,\\
&\mathcal{F}_3 = \sum_{c=1}^{C} \|\big(\mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{X}^c\big)\mathbf{Q}_c^{sqrt}\|_F^2,\\
&\mathcal{H}_1 = tr[\sum_{c=1}^{C}(\mathbf{X} - \mathbf{X}\mathbf{E}_c)\mathbf{P}_c(\mathbf{X} - \mathbf{X}\mathbf{E}_c)^T], \quad \text{and}\\
&\mathcal{H}_2 = -tr[\sum_{c=1}^{C} w_c(\mathbf{X}\mathbf{e}_c - \mathbf{X}\mathbf{b})(\mathbf{X}\mathbf{e}_c - \mathbf{X}\mathbf{b})^T].
\end{aligned}
$$

The problem of updating $\mathbf{X}$ can be written as

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}} \mathcal{J}_{\mathbf{X}}. \tag{24}$$

In order to minimize $\mathcal{J}_{\mathbf{X}}$ with respect to $\mathbf{X}$, we use the Iterative Projection Method (IPM) that minimizes a cost consisting of a convex term with an additional ℓ_1 regularizer [26, 35]. IPM is an iterative algorithm that computes the derivative of all the terms except the ℓ_1 part of the cost and takes a gradient descent step at each iteration. Followed by this gradient descent at each iteration, the values of $\mathbf{X}$ are soft thresholded [26]. The required derivative of all the terms in (23) can be computed as follows

$$\frac{\partial \mathcal{F}_1}{\partial \mathbf{X}} = 2\mathbf{A}^T\boldsymbol{\mathcal{K}}\mathbf{A}\mathbf{X} - 2\mathbf{A}^T\boldsymbol{\mathcal{K}}, \tag{25}$$

$$\frac{\partial \mathcal{F}_2}{\partial \mathbf{X}^c} = \frac{\partial}{\partial \mathbf{X}^c} tr[\mathbf{A}_c^T\boldsymbol{\mathcal{K}}\mathbf{A}_c\mathbf{X}^c\mathbf{P}_c\mathbf{X}^{\mathbf{c}T} - \mathbf{A}_c^T\boldsymbol{\mathcal{K}}\mathbf{P}_c\mathbf{X}^{cT}] \tag{26}$$

$$= 2\mathbf{A}_c^T\boldsymbol{\mathcal{K}}\mathbf{A}_c\mathbf{X}^c\mathbf{P}_c - 2\mathbf{A}_c^T\boldsymbol{\mathcal{K}}\mathbf{P}_c, \tag{27}$$

$$\frac{\partial \mathcal{F}_3}{\partial \mathbf{X}^c} = 2\mathbf{A}_c^T\boldsymbol{\mathcal{K}}\mathbf{A}_c\mathbf{X}^c\mathbf{Q}_c. \tag{28}$$

Note that $\mathcal{H}_1(\mathbf{X}) = \sum_{c=1}^{C} tr[\mathbf{X}^T\mathbf{S}_c\mathbf{X}]$, where $\mathbf{S}_c := (\mathbf{I} - \mathbf{E}_c)\mathbf{P}_c(\mathbf{I} - \mathbf{E}_c)^T$. Hence,

$$\frac{\partial \mathcal{H}_1}{\partial \mathbf{X}} = \sum_{c=1}^{C} 2\mathbf{X}\mathbf{S}_c. \tag{29}$$

Similarly,

$$\frac{\partial \mathcal{H}_2}{\partial \mathbf{X}} = -\frac{\partial}{\partial \mathbf{X}} \sum_{c=1}^{C} tr[\mathbf{X}\mathbf{T}_c\mathbf{X}^T] \tag{30}$$

$$= -\sum_{c=1}^{C} 2\mathbf{X}\mathbf{T}_c, \tag{31}$$

where, $\mathbf{T}_c := w_c(\mathbf{e}_c - \mathbf{b})(\mathbf{e}_c - \mathbf{b})^T$.

3.3. Optimization of the Probability Matrix $\mathbf{P}$

With the fixed dictionary $\mathbf{A}$, and the coefficient matrix $\mathbf{X}$, the cost in (9) can be re-written with respect to $\mathbf{P}$ as,

$$\begin{aligned}\mathcal{J}_{\mathbf{P}} =&\tau_1 \sum_{c=1}^{C}\sum_{i=1}^{N} P_{ci}\|\mathbf{\Phi}(\mathbf{y}_i) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{x}_i^c\|_2^2 + \tau_2 \sum_{c=1}^{C}\sum_{i=1}^{N}(1-P_{ci})\|\mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{x}_i^c\|_2^2 \\ &+ \lambda_2 \sum_{c=1}^{C}\sum_{i=1}^{N} P_{ci}\|\mathbf{x}_i - \mathbf{m}_c\|_2^2 - \lambda_2 \sum_{c=1}^{C} N_c\|\mathbf{m}_c - \mathbf{m}\|_2^2. \end{aligned} \tag{32}$$

We can solve the above problem by optimizing for the class probabilities for the i^{th} sample $\mathbf{p}_i$ independently, where $\mathbf{p}_i = [P_{1i}, \ldots, P_{Ci}]^T$, provided that $\mathbf{m}_c$ does not change much with each update. Hence, the cost with respect to $\mathbf{p}_i$ is given by

$$\mathcal{J}_{\mathbf{p}_i} = \mathbf{p}_i^T\mathbf{v}_i, \tag{33}$$

where the cth element of $\mathbf{v}_i$ is given by,

$$\begin{aligned}\mathbf{v}_i(c) =&\tau_1\|\mathbf{\Phi}(\mathbf{y}_i) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{x}_i^c\|_2^2 \\ &- \tau_2\|\mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{x}_i^c\|_2^2 + \lambda_2\|\mathbf{x}_i - \mathbf{m}_c\|_2^2. \end{aligned} \tag{34}$$

The goal is to, minimize $\mathcal{J}_{\mathbf{p}_i}$ subject to $\mathbf{p}_i^T\mathbf{1} = 1, \mathbf{p}_i \geq 0$. To minimize a linear cost subject to linear constraints is a linear programming (LP) optimization problem whose solution is on one of the vertices. In other words, the element of $\mathbf{p}_i$ corresponding to minimum value in $\mathbf{v}_i$ would be 1 and other elements would be zeros. This is to say that each sample will be assigned to a fixed class rather than a class distribution. Hence, instead of solving this LP, we compute the probability of each sample based on the reconstruction error e_{ci} of the i^{th} sample on the c^{th} class dictionary, defined as

$$\begin{aligned} e_{ci} &= \|\mathbf{\Phi}(\mathbf{y}_i) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{x}_i^c\|_2^2 \\ &= \boldsymbol{\mathcal{K}}(\mathbf{y}_i, \mathbf{y}_i) + (\mathbf{x}_i^c)^T\mathbf{A}\boldsymbol{\mathcal{K}}\mathbf{A}\mathbf{x}_i^c - \boldsymbol{\mathcal{K}}(\mathbf{y}_i, \mathbf{Y})\mathbf{A}_c\mathbf{x}_i^c, \end{aligned} \tag{35}$$

where $\mathbf{x}_i^c$ is the sparse coefficient of the i^{th} sample corresponding to dictionary $\mathbf{A}_c$. Now, the probability of the i^{th} sample belonging to the c^{th} class can be defined as

$$P_{ci} = \begin{cases} \frac{\exp\{-\frac{e_{ci}}{\sigma}\}}{\sum_{c=1}^{C}\exp\{-\frac{e_{ci}}{\sigma}\}} & \text{if} \quad \frac{\exp\{-\frac{e_{ci}}{\sigma}\}}{\sum_{c=1}^{C}\exp\{-\frac{e_{ci}}{\sigma}\}} > \theta, \\ 0 & \text{otherwise.} \end{cases} \tag{36}$$

Here, σ is a parameter that controls how sharp the probability distributions are. Furthermore, we want to add only those samples which are quite confident about its class and remove the ones that have similar probability of having come from multiple classes. This is achieved by setting the probability of those samples to zero which are less than a certain parameter θ. Furthermore, instead of updating $\mathbf{P}$ at each iteration, we skip a few iteration(s) (typically $1-5$) before updating the probability matrix. This gives some time for the learned dictionary to converge before adding more samples. The proposed method for learning dictionary is summarized in Algorithm 1.

3.4. Dictionary Learning with Ambiguously Labeled Data

In many practical situations there might be multiple labels available for each training sample. For example, given a picture with multiple faces and a caption specifying who are in the picture, the reader may not know which face goes with the names in the caption. The problem of learning identities

Algorithm 1: Algorithm for learning non-linear dictionary $\mathbf{A}$ by solving (9).

```
Input: Training Data $\mathbf{Y}$, Partial Labels $l_i, \forall i = 1, \dots N$, Kernel
       Function $\kappa$.
Output: Dictionary $\mathbf{A}$.
Initialize Dictionary $\mathbf{A}$, sparse Coefficient matrix $\mathbf{X}$ and Probability
matrix $\mathbf{P}$.
$itr = 0$
repeat
    $itr = itr + 1$
    Update sparse coefficient matrix $\mathbf{X}$ by solving (24).
    if $mod(itr,\ skipItr)$=0 then
        Update Probability matrix $\mathbf{P}$ using (36)
    end
    for $c = 1, \dots, C$ do
        for $k = 1, \dots, K_c$ do
            Update atom $\mathbf{a}_k$ using (22).
        end
    end
until convergence or maximum iterations ;
return $\mathbf{A}$.
```

where each example is associated with multiple labels, when only one of which is correct is often known as ambiguously labeled learning [11].

This ambiguously labeled data can be easily handled using the proposed formulation by giving equal probabilities to each of the given class for that sample. For example, if a sample $\mathbf{y}_i$ has labels $1, 4, 5, 7$, we can set $\mathbf{P}(c, i) = 0.25$, for $c = 1, 4, 5, 7$. However, a major challenge in handling such ambiguously labeled data is to learn an initial dictionary [7]. For the cases where data is either unambiguously labeled or completely unlabeled, we can use the unambiguously labeled data to learn an initial dictionary for each class. However, when each sample has multiple labels, we first need to cluster the data into different classes to make sure that the learned dictionary for each class is not influenced by the samples of the other classes.

Let $\mathbf{y}_i$ have multiple labels denoted by the set L_i and the number of ambiguous labels be denoted by $C_i \triangleq |L_i|$. In order to assign one cluster label to $\mathbf{y}_i$, we learn C_i dictionaries, one for each ambiguous class label, using all the samples excluding $\mathbf{y}_i$. While learning the c^{th} class dictionary $\mathbf{D}_{ci}$, where $c \in L_i$, for the i^{th} sample, we use all the samples excluding $\mathbf{y}_i$ and with at least one class label as c. Let the set of these samples be denoted by $\mathbf{Y}_{ci}$. We learn a dictionary $\mathbf{D}_{ci}$ with the data matrix $\mathbf{Y}_{ci}$ using the KSVD algorithm [1] for each $c \in L_i$. The reconstruction error of $\mathbf{y}_i$ is computed on $\mathbf{D}_{ci}$ as follows,

$$r_{ci} = \|\mathbf{y}_i - \mathbf{D}_{ci}\mathbf{x}\|_2, \tag{37}$$

where, $\mathbf{x} = (\mathbf{D}_{ci}^T \mathbf{D}_{ci})^{-1} \mathbf{D}_{ci}^T \mathbf{y}_i$. Next, $\mathbf{y}_i$ is assigned to the cluster c with the minimum reconstruction error r_{ci}. These steps are summarized in Algorithm 2.

Algorithm 2: Algorithm for clustering ambiguously labeled data into C clusters.

```
Input: Training Data $\mathbf{Y}$, Partial Labels $L_i$, $\forall i = 1, \ldots N$.
Output: Cluster labels $h_i \in \{1, \ldots, C\}$ for each sample $\mathbf{y}_i$, for all
        $i = 1, \ldots, N$.
for $i = 1, \ldots, N$ do
    for $j = 1, \ldots, C_i$ do
        $c = L_i(j)$
        Collect all the samples except $\mathbf{y}_i$ with at least one class label as
        $c$ into data matrix $\mathbf{Y}_{ci}$.
        Learn dictionary $\mathbf{D}_{ci}$ with $\mathbf{Y}_{ci}$ using KSVD algorithm.
        $\mathbf{x} = (\mathbf{D}_{ci}^T \mathbf{D}_{ci})^{-1} \mathbf{D}_{ci}^T \mathbf{y}_i$.
        $r_{ci} = \|\mathbf{y}_i - \mathbf{D}_{ci}\mathbf{x}\|_2$.
    end
    Cluster label $h_i = \arg\min_{c \in L_i} r_{ci}$
end
return $h_i, \forall i = 1, \ldots, N$.
```

For each class, an initial dictionary $\mathbf{D}_c^{(0)}$ is learned with samples in the c^{th} cluster using the KSVD algorithm. Finally, initial non-linear dictionary $\mathbf{A}_c^{(0)}$ is computed using $\mathbf{D}_c^{(0)}$ as

$$\mathbf{A}_c^{(0)} = \text{pinv}(\mathbf{Y})\mathbf{D}_c^{(0)}, \tag{38}$$

where pinv($\mathbf{Y}$) is the pseudo-inverse of the data matrix $\mathbf{Y}$.

3.5. Classification

Having learned the non-linear dictionary $\mathbf{A}$, we classify a given test sample $\mathbf{y}_t$ by first computing its sparse code $\mathbf{x}_t$ by solving the following optimization problem,

$$\begin{aligned} \mathbf{x}_t &= \arg\min_{\mathbf{x}} \|\mathbf{\Phi}(\mathbf{y}_t) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1 && (39) \\ &= \arg\min_{\mathbf{x}} \Big(\kappa(\mathbf{y}_t, \mathbf{y}_t) + \mathbf{x}^T\mathbf{A}^T\boldsymbol{\mathcal{K}}(\mathbf{Y}, \mathbf{Y})\mathbf{A}\mathbf{x} \\ &\qquad - 2\boldsymbol{\mathcal{K}}(\mathbf{y}_t, \mathbf{Y})\mathbf{A}\mathbf{x} + \lambda\|\mathbf{x}\|_1\Big). && (40) \end{aligned}$$

The above problem in (40) is solved using the IPM. Next, to determine the class of the test sample, we compute the reconstruction error for each class as

$$\begin{aligned} r_c &= \|\mathbf{\Phi}(\mathbf{y}_t) - \mathbf{\Phi}(\mathbf{Y})\mathbf{A}_c\mathbf{x}^c\|_2^2 && (41) \\ &= \kappa(\mathbf{y}_t, \mathbf{y}_t) + (\mathbf{x}^c)^T\mathbf{A}_c^T\boldsymbol{\mathcal{K}}(\mathbf{Y}, \mathbf{Y})\mathbf{A}_c\mathbf{x}^c \\ &\qquad - 2\boldsymbol{\mathcal{K}}(\mathbf{y}_t, \mathbf{Y})\mathbf{A}_c\mathbf{x}^c. && (42) \end{aligned}$$

Finally, the test sample is assigned the class corresponding to the minimum reconstruction error as

$$\text{class of } \mathbf{y}_t = \arg\min_c r_c. \tag{43}$$

4. Experimental Results

To illustrate the effectiveness of our method, we present experimental results on some of the publicly available databases such as the USPS digit dataset [16], the Kimia's object dataset [29] and TV LOST dataset [10, 9] that consists of cropped face images from TV series 'LOST'. A comparison with other existing object recognition methods in [35] suggests that the discriminative dictionary learning algorithm known as Fisher Discriminant Dictionary Learning (FDDL) is among the best dictionary-based method for classification. Hence, we use FDDL and a semi-supervised dictionary learning algorithm S2D2 [31] to compare the performance on semi-supervised experiments. We also compare our method with that of Support Vector Machines (SVM) as well as a semi-supervised extension of SVM known as (S3VM) [32]. Also, we compare our method with recently proposed Pseudo Multi-view Automatic Feature Decomposition for Co-training (PMC) method [5]. In all of our experiments, λ is set equal to 0.05 and η is set equal to 0.001. The number of iterations are set to a maximum value of 30. All the other parameters are set using cross-validation separately for each experiment. For big training datasets, they can be optimized on a small validation dataset to reduce training time. In our experiments, we optimized the sparsity parameter over the set $\{0.01, 0.05, 0.1, 0.5\}$. The discriminative parameters τ_1 and τ_2 were optimized over the set $\{0.1, 1, 5, 10\}$. We skipped a few iterations when updating $\mathbf{P}$ to ensure the convergence of the cost function. This allows dictionary atoms to converge before using them to compute the probability matrix. Furthermore, the parameter σ controls the sharpness of probability distribution. Although, this can be computed in each iteration as the average reconstruction error as was done in [7], we set this equal to 1 for simplicity. If the probability distributions appear very flat, we reduce it to a smaller value.

4.1. Digit Recognition

The USPS digit dataset [16] consists of gray images of hand written digits from 0 to 9. This dataset contains 7291 training samples and 2007 test samples. From the training data, four samples from each class are randomly chosen as the labeled samples and the rest of the training data is used as the unlabeled data. The original images are of size 16×16 which forms the feature vector of dimension 256. We added a maximum of 10 unlabeled samples per class at each iterations. For this experiment we used polynomial kernel of degree 4, and set sparsity parameter $\lambda_1 = 0.01$. Furthermore, to avoid low confidence samples we set $\theta = 0.5$.

We compare the recognition accuracies of the proposed method with other methods in Table 1. The parameters τ_1 and τ_2 were set equal to 10 and 0.1, respectively, for this dataset. Observe that the proposed method outperforms the other methods by more than 5%. The major difference between S2D2 and the proposed method is the use of non-linear kernel. This confirms the importance of non-linear kernels in dictionary learning methods. The improvement in performance compared to SVM and FDDL is due to the fact that we utilize the unlabeled data for updating dictionaries in the training stage. Being supervised techniques, the performance of SVM and FDDL reduces when the available labeled samples are small. Unlike S3VM which assigns hard labels to the unlabeled data points at each iteration, the proposed method assigns only a soft probability of class for each unlabeled

Algorithms	Accuracy(%)
SVM	74.47
S3VM [32]	75.61
FDDL [35]	79.24
PMC [5]	79.78
S2D2 [31]	85.61
Proposed Method	**90.60**

Table 1: Recognition accuracy for the proposed method on USPS Digits dataset.

data.The reason why the proposed method performs better than S3VM is because the soft assignment approach is more robust to labeling errors when compared to the hard assignment.

Pre-Images of the learned dictionary atoms: Recall that the k^{th} atom of the learned non-linear dictionary is represented as $\mathbf{\Phi}(\mathbf{Y})\mathbf{a}_k$ with respect to the base $\mathbf{\Phi}(\mathbf{Y})$ in the feature space G. Since G is large, and possibly of infinite dimension, we visualize the pre-image [28] of dictionary atoms. The pre-image of a dictionary atom $\mathbf{\Phi}(\mathbf{Y})\mathbf{a}_k$ is obtained by seeking a vector $\mathbf{d}_k$ in input space $\mathbb{R}^d$ that minimizes the cost function $\|\mathbf{\Phi}(\mathbf{d}_k) - \mathbf{\Phi}(\mathbf{Y})\mathbf{a}_k\|_2$. Due to various noise effects and the generally non-invertible mapping $\mathbf{\Phi}$, pre-image does not always exist. However, an approximated pre-image can be reconstructed without venturing into feature space using techniques described in [28]. In Fig. 2, we show the pre-images of some of the learned dictionary atoms from each class.

Performance in the presence of missing and noisy pixels: To further evaluate the robustness of the proposed method, we computed the recog-

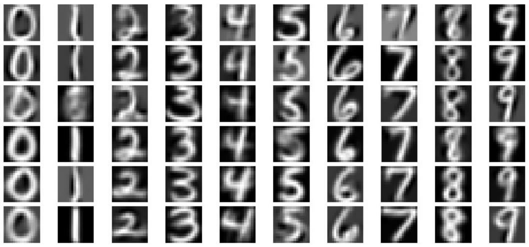
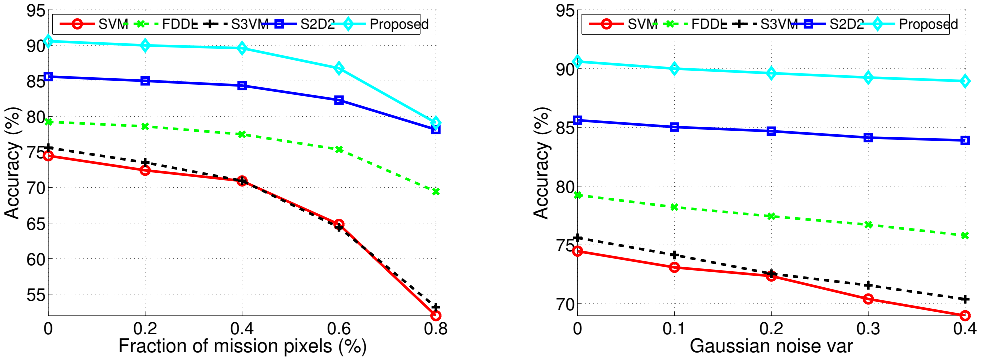


Figure 2: Pre-images of the learned atoms of USPS digits. Columns show the learned dictionary atoms for each class.

nition performance of the proposed method when pixels in the image are either missing or corrupted by noise. In the missing data experiment, we set pixels at random locations to zero for test images in the digit recognition application. The number of corrupted pixels was varied and we plot the corresponding accuracy in Fig. 3(a). Note that the recognition accuracy falls as expected when the amount of missing pixels is increased. But the fall in accuracy is much lower for the proposed technique when compared to the other methods. This clearly demonstrates the improved robustness of the proposed method compared to the competing methods. Similarly to study the robustness of our method in the presence of noise, we added independent and identically distributed Gaussian noise to the pixels. We varied the variance of the added noise and compute the recognition accuracy for all the methods. The results are shown in Fig. 3(b). We observed a similar improvement in robustness of the proposed technique.

4.2. Object Recognition

In the next set of experiments, we use Kimia's object dataset [29] which has 18 object categories each with 12 binary shapes. We randomly chose



(a) (b)

Figure 3: Accuracy for two kinds of corruption for digit recognition. (a) accuracy vs missing data. (b) Accuracy vs noise variance.

six images per class for training and the remaining six for testing. Furthermore, we randomly picked four images per class as the labeled data and the remaining two as the unlabeled data. Each image was resized to 16×16 and intensity values were used as features. The classification rates for all the algorithms are compared in Table 2. We see that the proposed method performs better than the other methods. In this experiment we used polynomial kernel of degree 2. We set sparsity parameter $\lambda_1 = 0.5$, $\tau_1 = 0.1$ and $\tau_2 = 1$. Furthermore, to avoid low confidence samples we set $\theta = 0.5$. These results clearly demonstrate that the performance of discriminative dictionary learning methods can be improved significantly by using unlabeled data, when the available labeled data is limited. Furthermore, the use of nonlinear kernel can improve the performance of dictionary learning methods for classification.

Caltech101 object recognition: The Caltech101 dataset contains 102 ob-

Algorithms	Accuracy(%)
SVM	84.26
S3VM [32]	84.26
FDDL [35]	86.11
PMC [5]	88.89
S2D2 [31]	87.96
Proposed Method	**92.59**

Table 2: Recognition accuracy for the proposed method, compared to competing ones for shape recognition.

ject categories and each category has about 40 to 80 images downloaded from Internet. We randomly selected 10 labeled and 10 unlabeled training images from each category to evaluate the proposed algorithm. To evaluate our method on this dataset, we used spatial pyramid features [17]. For each image, dense SIFT descriptors were extracted from 16 × 16 patches, separated by 6 pixels. To train the codebook for spatial pyramid, standard k-means clustering with k = 1024 was used. Finally, the dimension of spatial pyramid features were reduced to 3000 dimensions by PCA. The results of our comparison are provided in Table 3. As can be seen from this table, the proposed method compares favorably even on the large dataset.

4.3. Ambiguously Labeled Data

In order to test our algorithm on ambiguously labeled data we chose the TV LOST dataset as used by [7]. This dataset consists of face images from TV series ‘LOST’. In original dataset, there are 1122 registered face images corresponding to a total of 14 subjects, each containing from 18 to 204 images.

Algorithms	Accuracy(%)
SVM	60.8
FDDL [35]	61.1
PMC [5]	58.4
S3VM [32]	65.6
Proposed Method	**66.4**

Table 3: Recognition accuracy for the proposed method on Caltech101 dataset.

In our experiment, we followed the same setting as [7] and chose 12 subjects with at least 25 face images per subject. For each subject, first 25 images were selected to evaluate our method. Each image was resized to 30×30 pixels, and histogram-equalized intensities were used as features. This experiment was conducted under transductive setting, meaning all the data was available at training time. We ambiguously labeled 85% of the data and remaining 15% of the data was correctly labeled. For each ambiguously labeled sample, we assigned one correct label and 3 randomly chosen incorrect class labels. We compare our method with the Convex Learning from Partial Labels (CLPL) presented in [9], and various dictionary learning-based methods proposed in [7]. DLHD [7] clusters training data into various clusters based on the reconstruction error, and then learn dictionary for each cluster. DLSD [7] assigns a soft label to each sample based on the the reconstruction error and learns a dictionary for each class based on the assigned soft labels. Equally-weighted K-SVD [7] learns a dictionary using K-SVD for each class by giving equal weight to each ambiguous class. We compare our method with the other methods in Table 4. We use a polynomial kernel of degree 4 and set sparsity

Algorithms	Accuracy(%)
CLPL [9]	78.53
Equally-Weighted K-SVD[7]	81.67
DLHD [7]	86.17
DLSD [7]	86.63
Proposed Method	**88.33**

Table 4: Recognition accuracy for the proposed method, compared to competing ones for TV LOST dataset.

parameter $\lambda_1 = 0.05$. Furthermore, discriminative parameters τ_1, and τ_2 are set equal to 1 and 0.1, respectively. In order to visualize the dictionary atoms, we plot pre-images of the dictionary atoms for each class in Figure 4. As we can see the learned dictionary atoms capture the variations present in each class. Furthermore, we analyze the convergence of our algorithm. In Figure 5, we display the probability matrices at the start, end and intermediate iterations. We can clearly visualize how the label accuracy improves over iterations. We also plot the total cost over iterations in Figure 6. As can be seen from this figure, our cost decreases with increase in iterations.

5. Conclusion

We proposed a method that utilizes unlabeled and ambiguously labeled training data for learning non-linear discriminative dictionaries. The proposed method iteratively estimates the confidence of unlabeled samples belonging to each of the classes and uses it to refine the learned dictionaries. Experiments using various publicly available datasets demonstrate the im-

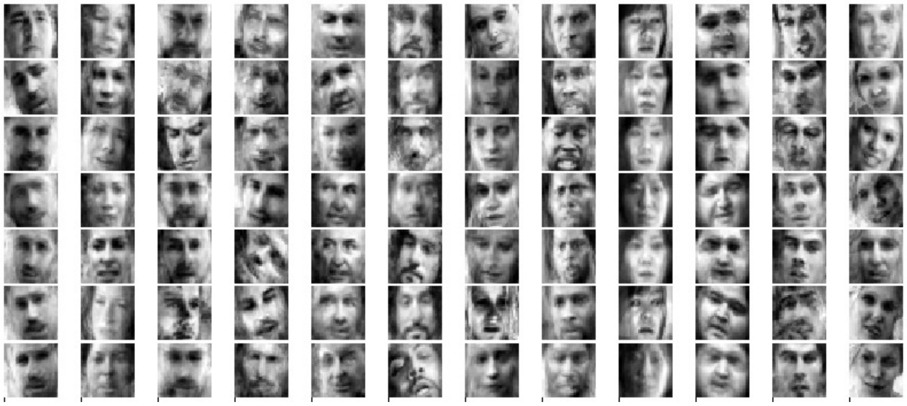
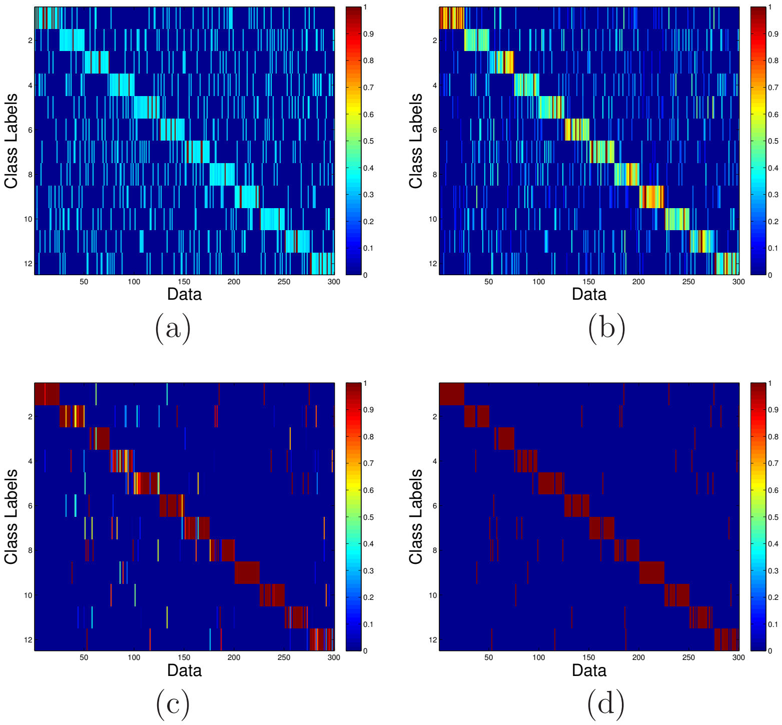


Figure 4: Pre-images of dictionary atoms for TV LOST dataset.



(a)

(b)

(c)

(d)

Figure 5: Convergence of probability matrices for TV LOST dataset. Figures (a), (b), (c), (d) show the probability matrix $\mathbf{P}$ at intermediate iterations.

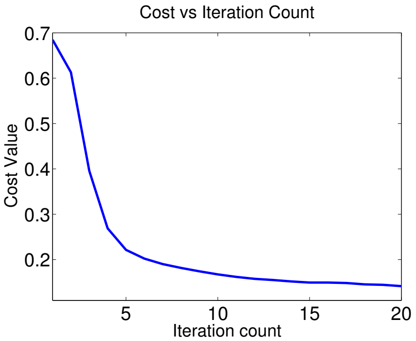


Figure 6: Convergence of cost over iterations for TV LOST dataset

proved accuracy and robustness to noise and missing information of the proposed method compared to state-of-the-art dictionary learning techniques.

Acknowledgment

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

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

[2] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT. ACM, 1998.

[3] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[4] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised learning*. Adaptive computation and machine learning. MIT Press, 2006.

[5] M. Chen, K. Q. Weinberger, and Y. Chen. Automatic feature decomposition for co-training. In *IEEE International Conference on Machine Learning (ICML)*, 2011.

[6] Y. Chen, V. M. Patel, P. J. Phillips, and R. Chellappa. Dictionary-based face recognition from video. In *European Conference on Computer Vision (ECCV)*, 2012.

[7] Y. Chen, V. M. Patel, Jaishanker K. Pillai, Rama Chellappa, and P. Jonathon Phillips. Dictionary learning from ambiguously labeled data. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[8] Y. Chen, C. S. Sastry, V. M. Patel, P. J. Phillips, and R. Chellappa. In-plane rotation and scale invariant clustering using dictionaries. *IEEE Transactions on Image Processing*, 22(6):2166–2180, June 2013.

[9] T. Cour, B. Sapp, C. Jordan, and B. Taskar. Learning from ambiguously labeled images. *Journal of Machine Learning (JMLR)*, 2011.

[10] T. Cour, B. Sapp, and B. Taskar. Annotated faces on tv dataset.

[11] T. Cour, B. Sapp, and B. Taskar. Learning from partial labels. *Journal of Machine Learning Research*, 12:1225–1261, 2011.

[12] M. Elad, M.A.T. Figueiredo, and Y. Ma. On the role of sparse and redundant representations in image processing. *Proceedings of the IEEE*, 98(6):972 –982, june 2010.

[13] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

[14] K. Etemand and R. Chellappa. Separability-based multiscale basis selection and feature extraction for signal and image classification. *IEEE Transactions on Image Processing*, 7(10):1453–1465, Oct. 1998.

[15] K. Huang and S. Aviyente. Sparse representation for signal classification. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[16] J. J. Hull. A database for handwritten text recognition research. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16:550–554, May 1994.

[17] Z. Jiang, Z. Lin, and L. S. Davis. Learning a discriminative dictionary for sparse coding via label consistent K-SVD. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[18] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):791–804, April 2012.

[19] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa. Sparse embedding: a framework for sparsity promoting dimensionality reduction. In *European Conference on Computer Vision (ECCV)*, 2012.

[20] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa. Design of non-linear kernel dictionaries for object recognition. *IEEE Transactions on Image Processing*, 22(12):5123–5135,, Dec. 2013.

[21] V. M. Patel and R. Chellappa. Sparse representations, compressive sensing and dictionaries for pattern recognition. In *Asian Conference on Pattern Recognition (ACPR)*, 2011.

[22] V. M. Patel, T. Wu, S. Biswas, P. J. Phillips, and R. Chellappa. Dictionary-based face recognition under variable lighting and pose. *IEEE Transactions on Information Forensics and Security*, 7(3):954–965, June 2012.

[23] I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[24] M. Ranzato, F. Haung, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[25] F. Rodriguez and G. Sapiro. Sparse representations for image classification: Learning discriminative and reconstructive non-parametric dictionaries. *Tech. Report, University of Minnesota*, Dec. 2007.

[26] L. Rosasco, A. Verri, M. Santoro, S. Mosci, and S. Villa. Iterative projection methods for structured sparsity regularization. *MIT-CSAIL-TR-2009-050, CBCL-282*, 2009.

[27] R. Rubinstein, A.M. Bruckstein, and M. Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045 –1057, june 2010.

[28] B. Scholkopf and A. J. Smola. *Learning with Kernels, Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[29] T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing shock graphs. In *IEEE International Conference on Computer Vision (ICCV)*, 2001.

[30] A. Shrivastava, H. V. Nguyen, V. M. Patel, and R. Chellappa. Design of non-linear discriminative dictionaries for image classification. In *Asian Conference on Computer Vision*, 2012.

[31] A. Shrivastava, J. K. Pillai, V. M. Patel, and R. Chellappa. Learning discriminative dictionaries with partially labeled data. In *IEEE International Conference on Image Processing (ICIP)*, 2012.

[32] V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear svms. In *ACM SIGIR*, 2006.

[33] P. Sprechmann and G. Sapiro. Dictionary learning and sparse coding for unsupervised clustering. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2010.

[34] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T.S. Huang, and S. Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031 –1044, june 2010.

[35] M. Yang, L. Zhang, X. Feng, and D. Zhang. Fisher discrimination dictionary learning for sparse representation. In *IEEE International Conference on Computer Vision (ICCV)*, 2011.

[36] M. Yang, X. F. L. Zhang, and D. Zhang. Fisher discrimination dictionary learning for sparse representation. In *IEEE International Conference on Computer Vision (ICCV)*, 2011.

[37] G. Zhang, Z. Jiang, and L. S. Davis. Online semi-supervised discriminative dictionary learning for sparse representation. In *Asian Conference on Computer Vision*, 2012.

[38] Q. Zhang and B. Li. Discriminative k-svd for dictionary learning in face recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2010.