Domain Adaptive Dictionary Learning

Qiang Qiu[1], Vishal M. Patel[1], Pavan Turaga[2], and Rama Chellappa[1]

[1] Center for Automation Research, UMIACS, University of Maryland, College Park
[2] Arts Media and Engineering, Arizona State University
qiu@cs.umd.edu, {pvishalm,rama}@umiacs.umd.edu, pturaga@asu.edu

Abstract. Many recent efforts have shown the effectiveness of dictionary learning methods in solving several computer vision problems. However, when designing dictionaries, training and testing domains may be different, due to different view points and illumination conditions. In this paper, we present a function learning framework for the task of transforming a dictionary learned from one visual domain to the other, while maintaining a domain-invariant sparse representation of a signal. Domain dictionaries are modeled by a linear or non-linear parametric function. The dictionary function parameters and domain-invariant sparse codes are then jointly learned by solving an optimization problem. Experiments on real datasets demonstrate the effectiveness of our approach for applications such as face recognition, pose alignment and pose estimation.

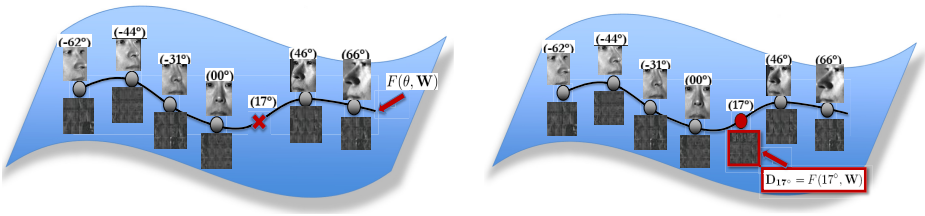
1 Introduction

In recent years, sparse and redundant modeling of signals has received a lot of attention from the vision community [1]. This is mainly due to the fact that signals or images of interest are sparse or compressible in some dictionary. In other words, they can be well approximated by a linear combination of a few elements (also known as atoms) of a redundant dictionary. This dictionary can either be an analytic dictionary such as wavelets or it can be directly trained from data. It has been observed that dictionaries learned directly from data provide better representation and hence can improve the performance of many applications such as image restoration and classification [2].

When designing dictionaries for image classification tasks, we are often confronted with situations where conditions in the training set are different from those present during testing. For example, in the case of face recognition, more than one familiar view may be available for training. Such training faces may be obtained from a live or recorded video sequences, where a range of views are observed. However, the test images can contain conditions that are not necessarily presented in the training images such as a face in a different pose. The problem of transforming a dictionary trained from one visual domain to another without changing signal sparse representations can be viewed as a problem of domain adaptation [3] and transfer learning [4].

Given the same set of signals observed in different visual domains, our goal is to learn a dictionary for the new domain without corresponding observations. We formulate this problem of dictionary transformation in a function learning

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part IV, LNCS 7575, pp. 631–645, 2012.



(a) Example dictionaries learned at known poses with observations.

(b) Domain adapted dictionary at a pose ($\theta = 17^\circ$) associated with no observations.

Fig. 1. Overview of our approach. Consider example dictionaries corresponding to faces at different azimuths. (a) shows a depiction of example dictionaries over a curve on a dictionary manifold which will be discussed later. Given example dictionaries, our approach learns the underlying dictionary function $F(\theta, \mathbf{W})$. In (b), the dictionary corresponding to a domain associated with observations is obtained by evaluating the learned dictionary function at corresponding domain parameters.

framework, i.e., dictionaries across different domains are modeled by a parametric function. The dictionary function parameters and domain-invariant sparse codes are then jointly learned by solving an optimization problem. As shown in Figure 1, given a learned dictionary function, a dictionary adapted to a new domain is obtained by evaluating such a dictionary function at the corresponding domain parameters, e.g., pose angles.

For the case of pose variations, linear interpolation methods have been discussed in [5] to predict intermediate views of faces given a frontal and profile views. These methods essentially apply linear regression on the PCA coefficients corresponding to two different views. In [6], Vetter and Poggio present a method for learning linear transformations from a basis set of prototypical views. Their approach is based on the linear class property which essentially states that if a 3D view of an object can be represented as the weighted sum of views of other objects, its rotated view is a linear combination of the rotated views of the other objects with the same weights [6], [7], [8]. Note that our method is more general than the above mentioned methods in that it is applicable to visual domains other than pose. Second, our method is designed to maintain consistent sparse coefficients for the same signal observed in different domains. Furthermore, our method is based on the recent dictionary learning methods and is able to learn dictionaries that are more general than the ones resulting from PCA.

This paper makes the following contributions

- A general continuous function learning framework is presented for the task of dictionary transformations across domains.
- A simple and efficient optimization procedure is presented that learns dictionary function parameters and domain-invariant sparse codes simultaneously.
- Experiments for various applications, including pose alignment, pose and illumination estimation and face recognition across pose, are presented.

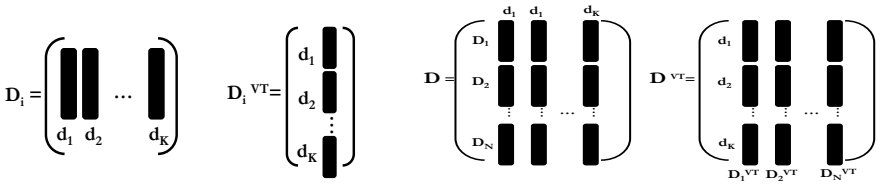
2 Overall Approach

We consider the problem of dictionary transformations in a learning framework, where we are provided with a few examples of dictionaries $\mathbf{D}_i$ with corresponding domain parameter θ_i. Let the parameter space be denoted by Θ, i.e. $\theta_i \in \Theta$. Let the *dictionary space* be denoted $\mathcal{D}$. The problem then boils down to constructing a mapping function $F : \Theta \mapsto \mathcal{D}$. In the simple case where $\Theta = \mathbb{R}$ and $\mathcal{D} = \mathbb{R}^n$, the problem of fitting a function can be solved efficiently using curve fitting techniques [9]. A dictionary of d atoms in $\mathbb{R}^n$ is often considered as an $n \times d$ real matrix or equivalently a point in $\mathbb{R}^{n \times d}$. However, often times there are additional constraints on dictionaries that make the identification with $\mathbb{R}^{n \times d}$ not well-motivated. We present below a few such constraints:

- Subspaces: For the special case of under-complete dictionaries where the matrix is full-rank and thus represents a choice of basis vectors for a d-dimensional subspace in $\mathbb{R}^n$, the dictionary space is naturally considered as a Grassmann manifold $\mathcal{G}_{n,d}$ [10]. The geometry of the Grassmann manifold is studied either as a quotient-space of the special orthogonal group or in terms of full-rank projection matrices, both of which result in non-Euclidean geometric structures.
- Products of subspaces: In many cases, it is convenient to think of the dictionary as a union of subspaces, e.g. a line and a plane. This structure has been utilized in many applications such as generalized PCA (GPCA), sparse subspace clustering [11] etc. In this case, the dictionary-space becomes a subset of the product space of Grassmann manifolds.
- Overcomplete dictionaries: In the most general case one considers an overcomplete set of basis vectors, where each basis vector has unit-norm, i.e. each basis vector is a point on the hypershere $\mathbb{S}^{n-1}$. In this case, the dictionary space becomes a subset of the product-space $\mathbb{S}^{(n-1)\times d}$.

To extend classic multi-variate function fitting to manifolds such as the ones above, one needs additional differential geometric tools. In our case, we propose extrinsic approaches that rely on embedding the manifold into an ambient vector space, perform function/curve fitting in the ambient space, and project the results back to the manifold of interest. This is conceptually simpler, and we find in our experiments that this approach works very well for the problems under consideration. The choice of embedding is in general not unique. We describe below the embedding and the corresponding projection operations for the manifolds of interest describe above.

- Subspaces: Each point in $\mathcal{G}_{n,d}$ corresponds to a d-dimensional subspace of $\mathbb{R}^n$. Given a choice of orthonormal basis vectors for the subspace $\mathbb{Y}$, the $n \times n$ projection matrix given by $\mathbf{P} = \mathbb{Y}\mathbb{Y}^T$ is a unique representation for the subspace. The projection matrix represntation can then be embedded into the ambient vector-space $\mathbb{R}^{n \times n}$. The projection operation $\mathbf{\Pi}$ is given by $\mathbf{\Pi}(\mathbf{M}) = \mathbf{U}\mathbf{U}^\mathbf{T}$, where $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathbf{T}$ is a rank-d SVD of $\mathbf{M}$ [12].
- Products of subspaces: Following the procedure above, each component of the product space can be embedded into a different vector-space and the projected back to the manifold using the corresponding projection operation.



(a) $\mathbf{D_i}$ vs. $\mathbf{D_i}^{VT}$ (b) $\mathbf{D}$ vs. $\mathbf{D}^{VT}$

Fig. 2. The vector transpose (VT) operator over dictionaries

– Overcomplete dictionaries: The embedding from $\mathbb{S}^{n-1}$ to $\mathbb{R}^n$ is given by a vectorial representation with unit-norm. The projection $\mathbf{\Pi} : \mathbb{R}^n \mapsto \mathbb{S}^{n-1}$ is given by $\mathbf{\Pi}(\mathbf{V}) = \frac{\mathbf{V}}{\|\mathbf{V}\|}$, where $\|.\|$ is the standard Euclidean norm. A similar operation on the product-space $\mathbb{S}^{(n-1)\times d}$ can be defined by component-wise projection operations.

In specific examples in the paper, we consider the case of over-complete dictionaries. We adopt the embedding and projection approach described above as a means to exploit the wealth of function-fitting techniques available for vector-spaces. Next, we describe the technique we adopt.

2.1 Problem Formulation

We denote the same set of P signals observed in N different domains as $\{\mathbf{Y_1}, ..., \mathbf{Y_N}\}$, where $\mathbf{Y_i} = [\mathbf{y_{i1}}, ..., \mathbf{y_{iP}}]$, $\mathbf{y_{ip}} \in \mathbb{R}^n$. Thus, $\mathbf{y_{ip}}$ denotes the p^{th} signal observed in the i^{th} domain. In the following, we will use $\mathbf{D_i}$ as the vector-space embedded dictionary. Let $\mathbf{D_i}$ denote the dictionary for the i^{th} domain, where $\mathbf{D_i} = [\mathbf{d_{i1}}...\mathbf{d_{iK}}]$, $\mathbf{d_{ik}} \in \mathbb{R}^n$. We define a *vector transpose* (VT) operation over dictionaries as illustrated in Figure 2. The VT operator treats each individual dictionary atom as a value and then perform the typical matrix transpose operation. Let $\mathbf{D}$ denote the stack dictionary shown in Figure 2b over all N domains. It is noted that $\mathbf{D} = [\mathbf{D^{VT}}]^{\mathbf{VT}}$.

The domain dictionary learning problem can be formulated as (1). Let $\mathbf{X} = [\mathbf{x_1}, ..., \mathbf{x_P}]$, $\mathbf{x_p} \in \mathbb{R}^K$, be the sparse code matrix. The set of domain dictionary $\{\mathbf{D_i}\}_i^N$ learned through (1) enables the same sparse codes $\mathbf{x_p}$ for a signal $\mathbf{y_p}$ observed across N different domains to achieve domain adaptation.

$$\underset{\{\mathbf{D_i}\}_\mathbf{i}^\mathbf{N}, \mathbf{X}}{\arg\min} \sum_i^N \|\mathbf{Y_i} - \mathbf{D_i X}\|_F^2 \;\; s.t.\; \forall p \; \|\mathbf{x}_p\|_o \leq T, \tag{1}$$

where $\|\mathbf{x}\|_o$ counts the number of non-zero values in $\mathbf{x}$. T is a sparsity constant.

We propose to model domain dictionaries $\mathbf{D_i}$ through a parametric function in (2), where $\boldsymbol{\theta_i}$ denotes a vector of domain parameters, e.g., view point angles, illumination conditions, etc., and $\mathbf{W}$ denotes the dictionary function parameters.

$$\mathbf{D_i} = F(\boldsymbol{\theta_i}, \mathbf{W}) \tag{2}$$

Applying (2) to (1), we formulate the domain dictionary function learning as (3).

$$\underset{\mathbf{W},\mathbf{X}}{\arg\min} \sum_{i}^{N} \|\mathbf{Y_i} - F(\boldsymbol{\theta_i}, \mathbf{W})\mathbf{X}\|_F^2 \quad s.t.\ \forall p\ \|\mathbf{x}_p\|_o \leq T. \tag{3}$$

Once a dictionary is estimated it is projected back to the dictionary-space by the projection operation described earlier.

2.2 Domain Dictionary Function Learning

We first adopt power polynomials to model $\mathbf{D_i^{VT}}$ in Figure 2a through the following dictionary function $F(\boldsymbol{\theta_i}, \mathbf{W})$,

$$F(\boldsymbol{\theta_i}, \mathbf{W}) = w_0 + \sum_{s=1}^{S} w_{1s}\theta_{is} + ... + \sum_{s=1}^{S} w_{ms}\theta_{is}^m \tag{4}$$

where we assume S-dimensional domain parameter vectors and an m^{th}-degree polynomial model. For example, given $\boldsymbol{\theta_i}$ a 2-dimensional domain parameter vector, a quadratic dictionary function is defined as,

$$F(\boldsymbol{\theta_i}, \mathbf{W}) = w_0 + w_{11}\theta_{i1} + w_{12}\theta_{i2} + w_{21}\theta_{i1}^2 + w_{22}\theta_{i2}^2$$

Given $\mathbf{D_i}$ contains K atoms and each dictionary atom is in the $\mathbb{R}^n$ space, as $\mathbf{D_i^{VT}} = F(\boldsymbol{\theta_i}, \mathbf{W})$, it can be noted from Figure 2 that w_{ms} is a nK-sized vector. We define the function parameter matrix $\mathbf{W}$ and the domain parameter matrix $\boldsymbol{\Theta}$ as

$$\mathbf{W} = \begin{bmatrix} w_0^{(1)} & w_0^{(2)} & w_0^{(3)} & ... & w_0^{(nK)} \\ w_{11}^{(1)} & w_{11}^{(2)} & w_{11}^{(3)} & ... & w_{11}^{(nK)} \\ & & & . & \\ & & & . & \\ & & & . & \\ w_{mS}^{(1)} & w_{mS}^{(2)} & w_{mS}^{(3)} & ... & w_{mS}^{(nK)} \end{bmatrix} \qquad \boldsymbol{\Theta} = \begin{bmatrix} 1 & 1 & 1 & ... & 1 \\ \theta_{11} & \theta_{21} & \theta_{31} & ... & \theta_{N1} \\ & & & . & \\ & & & . & \\ & & & . & \\ \theta_{1S}^m & \theta_{2S}^m & \theta_{3S}^m & ... & \theta_{NS}^m \end{bmatrix}$$

Each row of $\mathbf{W}$ corresponds to the nK-sized w_{ms}^T, and $\mathbf{W} \in \mathbb{R}^{(mS+1)\times nK}$. N different domains are assumed and $\boldsymbol{\Theta} \in \mathbb{R}^{(mS+1)\times N}$. With the matrix $\mathbf{W}$ and $\boldsymbol{\Theta}$, (4) can be written as,

$$\mathbf{D^{VT}} = \mathbf{W^T}\boldsymbol{\Theta} \tag{5}$$

where $\mathbf{D^{VT}}$ is defined in Figure 2b. Now dictionary function learning formulated in (3) can be written as,

$$\underset{\mathbf{W},\mathbf{X}}{\arg\min} \|\mathbf{Y} - [\mathbf{W^T}\boldsymbol{\Theta}]^{\mathbf{VT}}\mathbf{X}\|_F^2 \quad s.t.\ \forall p\ \|\mathbf{x}_p\|_o \leq T \tag{6}$$

where $\mathbf{Y}$ is the stacked training signals observed in different domains as illustrated in Figure 3. With the objective function defined in (6), the dictionary function learning can be performed as described below:

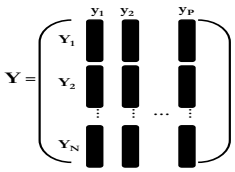


Fig. 3. The stack P training signals observed in N different domains

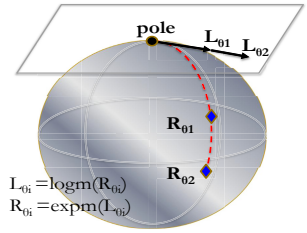


Fig. 4. Illustration of exponential maps *expm* and inverse exponential maps *logm* [12]

Step 1: Obtain the sparse coefficients X and $[\mathbf{W^T\Theta}]^{\mathbf{VT}}$ via any dictionary learning method, e.g., K-SVD [13].
Step 2: Given the domain parameter matrix $\mathbf{\Theta}$, the optimal dictionary function can be obtained as [14],

$$\mathbf{W} = [\mathbf{\Theta\Theta^T}]^{-\mathbf{1}}\mathbf{\Theta}[[[\mathbf{W^T\Theta}]^{\mathbf{VT}}]^{\mathbf{VT}}]^{\mathbf{T}}. \tag{7}$$

Step 3: Sample the dictionary function at desired parameters values, and project it to the dictionary-space using an appropriate projection operation.

2.3 Non-linear Dictionary Function Models

Till now, we only assume power polynomials for the dictionary model. In this section, we discuss non-linear dictionary functions. We only focus on linearizeable functions, and a general Newton's method based approach to learn a non-linear dictionary function is presented in Algorithm 2 in Appendix A.

Linearizeable Models. There are several well-known linearizeable models, such as the Cobb-Douglass model, the logistic model, etc. We use the Cobb-Douglass model as the example to discuss in detail how dictionary function learning can be performed over these linearizable models.

The Cobb-Douglass model is written as,

$$\mathbf{D_i^{VT}} = F(\boldsymbol{\theta_i}, \mathbf{W}) = w_0 \exp(\sum_{s=1}^{S} w_{1s}\theta_{is} + ... + \sum_{s=1}^{S} w_{ms}\theta_{is}^m) \tag{8}$$

The logarithmic transformation yields,

$$\log(\mathbf{D_i^{VT}}) = \log(w_0) + \sum_{s=1}^{S} w_{1s}\theta_{is} + ... + \sum_{s=1}^{S} w_{ms}\theta_{is}^m$$

As the right side of (8) is in the same linear form as (4), we can define the corresponding function parameter matrix $\mathbf{W}$ and the domain parameter matrix $\mathbf{\Theta}$ as discussed. The dictionary function learning is written as,

$$\underset{\mathbf{W},\mathbf{X}}{\arg\min} \|\mathbf{Y} - [\exp(\mathbf{W^T\Theta})]^{\mathbf{VT}}\mathbf{X}\|_F^2 \quad s.t.\ \forall p\ \|\mathbf{x}_p\|_o \leq T.$$

Through any dictionary learning methods, we obtain $[[\exp(\mathbf{W^T\Theta})]^{\mathbf{T}}]^{\mathbf{VT}}$ and $\mathbf{X}$. Then, the dictionary function is obtained as,

$$\mathbf{W} = [\mathbf{\Theta\Theta^T}]^{-\mathbf{1}}\mathbf{\Theta}[\log([[\exp(\mathbf{W^T\Theta})]^{\mathbf{VT}}]^{\mathbf{VT}})]^{\mathbf{T}}.$$

2.4 Domain Parameter Estimation

Given a learned dictionary function $F(\boldsymbol{\theta}, \mathbf{W})$, the domain parameters $\boldsymbol{\theta_y}$ associated with an unknown image $\mathbf{y}$, e.g., pose (azimuth, altitude) or light source directions (azimuth, altitude), can be estimated using Algorithm 1.

It is noted that we adopt the following strategy to represent the domain parameter vector $\boldsymbol{\theta}$ for each pose in a linear space: we first obtain the rotation matrix $\mathbf{R}_\theta$ from the azimuth and altitude of a pose; we then compute the inverse

Input: a dictionary function $F(\theta, \mathbf{W})$, an image $\mathbf{y}$, domain parameter matrix $\mathbf{\Theta}$
Output: an S-dimensional domain parameter vector $\boldsymbol{\theta_y}$ associated with $\mathbf{y}$
begin
1. Initialize with mean domain parameter vector: $\boldsymbol{\theta_y} = \text{mean}(\mathbf{\Theta})$;
2. Estimate $\theta^{(s)}$, the s^{th} value in $\boldsymbol{\theta_y}$;
for $s \leftarrow 1$ **to** S **do**
3. Obtain the value range to estimate $\theta^{(s)}$
$\theta^{(s)}_{min} = \min$ (s^{th} row of $\mathbf{\Theta}$) ;
$\theta^{(s)}_{max} = \max$ (s^{th} row of $\mathbf{\Theta}$) ;
$\theta^{(s)}_{mid} = (\theta^{(s)}_{min} + \theta^{(s)}_{max})/2$;
4. Estimate $\theta^{(s)}$ via a search for the parameters to best represent $\mathbf{y}$.
repeat
$\boldsymbol{\theta_{min}} \leftarrow$ replace the s^{th} value of $\boldsymbol{\theta_y}$ with $\theta^{(s)}_{min}$;
$\boldsymbol{\theta_{max}} \leftarrow$ replace the s^{th} value of $\boldsymbol{\theta_y}$ with $\theta^{(s)}_{max}$;
$\mathbf{x_{min}} \leftarrow \min_{\mathbf{x}} |\mathbf{y} - F(\boldsymbol{\theta_{min}}, \mathbf{W})|_2^2, \quad s.t. |\mathbf{x}|_o \leq$ t (sparsity) ;
$\mathbf{x_{max}} \leftarrow \min_{\mathbf{x}} |\mathbf{y} - F(\boldsymbol{\theta_{max}}, \mathbf{W})|_2^2, \quad s.t. |\mathbf{x}|_o \leq$ t (sparsity) ;
$\mathbf{r_{min}} \leftarrow \mathbf{y} - F(\boldsymbol{\theta_{min}}, \mathbf{W})\mathbf{x_{min}}$;
$\mathbf{r_{max}} \leftarrow \mathbf{y} - F(\boldsymbol{\theta_{max}}, \mathbf{W})\mathbf{x_{max}}$;
if $\mathbf{r_{min}} \leq \mathbf{r_{max}}$ **then**
$\theta^{(s)}_{max} = \theta^{(s)}_{mid}$;
else
$\theta^{(s)}_{min} = \theta^{(s)}_{mid}$;
end
$\theta^{(s)}_{mid} = (\theta^{(s)}_{min} + \theta^{(s)}_{max})/2$;
until $|\theta^{(s)}_{max} - \theta^{(s)}_{min}| \leq threshold$;
$\theta^{(s)} \leftarrow \theta^{(s)}_{mid}$;
end
7. return $\boldsymbol{\theta_y}$;
end

Algorithm 1. Domain parameters estimation

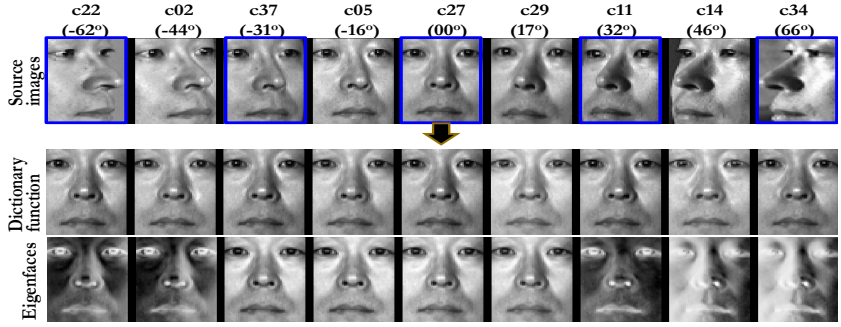


Fig. 5. Frontal face alignment. For the first row of source images, pose azimuths are shown below the camera numbers. Poses highlighted in blue are known poses to learn a linear dictionary function (m=4), and the remaining are unknown poses. The second and third rows show the aligned face to each corresponding source image using the linear dictionary function and Eigenfaces respectively.

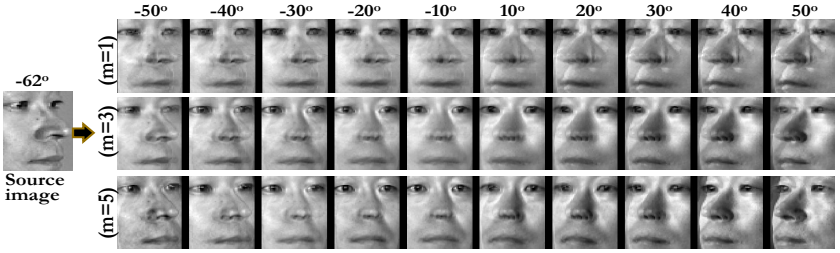
exponential map of the rotation matrix $\text{logm}(\mathbf{R}_\theta)$ as shown in Figure 4. We denote $\boldsymbol{\theta}$ using the upper triangular part of the resulting skew-symmetric matrix [12]. The exponential map operation in Figure 4 is used to recover the azimuth and altitude from the estimated domain parameters. We represent light source directions in the same way.

3 Experimental Evaluation

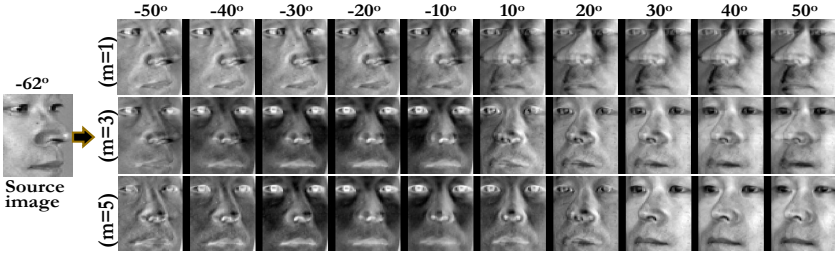
We conduct our experiments using two public face datasets: the CMU PIE dataset [15] and the Extended YaleB dataset [16]. The CMU PIE dataset consists of 68 subjects in 13 poses and 21 lighting conditions. In our experiments we use 9 poses which have approximately the same camera altitude, as shown in the first row of Figure 5. The Extended YaleB dataset consists of 38 subjects in 64 lighting conditions. All images are in 64×48 size. We will first evaluate the basic behaviors of dictionary functions through pose alignment. Then we will demonstrate the effectiveness of dictionary functions in face recognition and domain estimation.

3.1 Dictionary Functions for Pose Alignment

Frontal Face Alignment In Figure 5, we align different face poses to the frontal view. We learn for each subject in the PIE dataset a linear dictionary function $F(\boldsymbol{\theta}, \mathbf{W})$ (m=4) using 5 out of 9 poses. The training poses are highlighted in blue in the first row of Figure 5. Given a source image $\mathbf{y_s}$, we first estimate the domain parameters $\boldsymbol{\theta_s}$, i.e., the pose azimuth here, by following Algorithm 1. We then obtain the sparse representation $\mathbf{x_s}$ of the source image as $\min_{\mathbf{x_s}} \|\mathbf{y_s} - F(\boldsymbol{\theta_s}, \mathbf{W})\mathbf{x_s}\|_2^2,\ s.t.\ \|\mathbf{x_s}\|_o \leq T$ (sparsity level) using any pursuit methods such as OMP [17]. We specify the fontal pose azimuth (00^o) as the



(a) Pose synthesis using a linear dictionary function



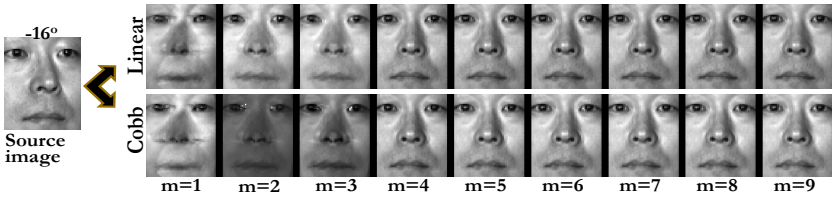
(b) Pose synthesis using Eigenfaces

Fig. 6. Pose synthesis using various degrees of dictionary polynomials. All the synthesized poses are unknown to learned dictionary functions and associated with no actual observations. m is the degree of a dictionary polynomial in (4).

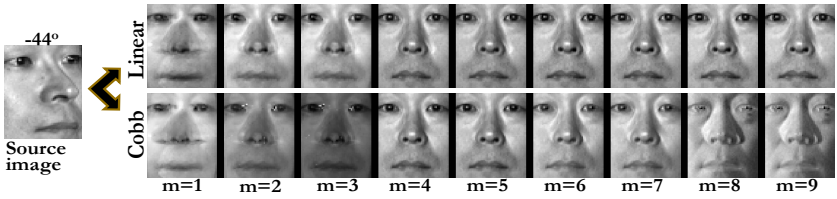
parameter for the target domain $\boldsymbol{\theta_t}$, and obtain the frontal view image $\mathbf{y_t}$ as $\mathbf{y_t} = F(\boldsymbol{\theta_t}, \mathbf{W})\mathbf{x_s}$. The second row of Figure 5 shows the aligned frontal view images to the respective poses in the first row. These aligned frontal faces are close to the actual image, i.e., c27 in the first row. It is noted that images with poses c02, c05, c29 and c14 are unknown poses to the learned dictionary function.

For comparison, we learn Eigenfaces for each of the 5 training poses and obtain adapted Eigenfaces at 4 unknown poses using the same function fitting method in our framework. We then project each source image (mean-subtracted) on the respective eignefaces and use frontal Eigenfaces to reconstruct the aligned image shown in the third row of Figure 5. Our method of jointly learning the dictionary function parameters and domain-invariant sparse codes in (6) significantly outperforms the Eigenfaces approach, which fails for large pose variations.

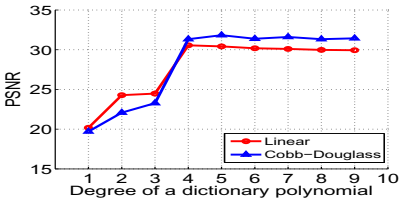
Pose Synthesis. In Figure 6, we synthesize new poses at any given pose azimuth. We learn for each subject in the PIE dataset a linear dictionary function $F(\boldsymbol{\theta}, \mathbf{W})$ using all 9 poses. In Figure 6a, given a source image $\mathbf{y_s}$ in a profile pose (-62^o), we first estimate the domain parameters $\boldsymbol{\theta_s}$ for the source image, and sparsely decompose it over $F(\boldsymbol{\theta_s}, \mathbf{W})$ for its sparse representation $\mathbf{x_s}$. We specify every 10^o pose azimuth in $[-50^o, 50^o]$ as parameters for the target domain $\boldsymbol{\theta_t}$, and obtain a synthesized pose image $\mathbf{y_t}$ as $\mathbf{y_t} = F(\boldsymbol{\theta_t}, \mathbf{W})\mathbf{x_s}$. It is noted that none of the target poses are associated with actual observations. As shown in Figure 6a, we obtain reasonable synthesized images at poses with no observations. We observe improved synthesis performance by increasing the value of



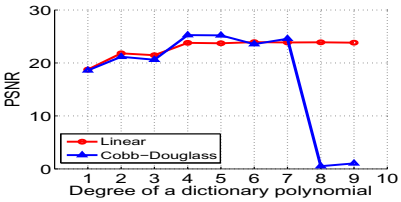
(a) Pose c05 frontal alignment



(b) Pose c02 frontal alignment



(c) Pose c05 alignment PSNR

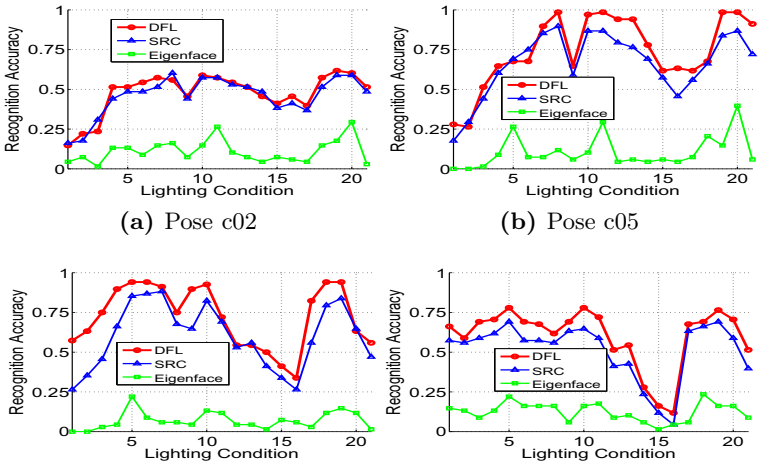


(d) Pose c02 frontal PSNR

Fig. 7. Linear vs. non-linear dictionary functions. m is the degree of a dictionary polynomial in (4) and (8).

m, i.e., the degree of a dictionary polynomial. In Figure 6b, we perform curve fitting over Eigenfaces as discussed. The proposed dictionary function learning framework exhibits better synthesis performance.

Linear vs. Non-linear. In Figure 7, we conduct the same frontal face alignment experiments discussed above. Now we learn for each subject both a linear and a nonlinear Cobb-Douglass dictionary function discussed in Section 2.3. As a Cobb-Douglass function is linearizeable, various degrees of polynomials are experimented for both linear and nonlinear dictionary function learning. As shown in Figure 7a and Figure 7c, the nonlinear Cobb-Douglass dictionary function exhibits better reconstruction while aligning pose c05, which is also indicated by the higher PSNR values. However, in Figure 7b and 7d, we notice that the Cobb-Douglass dictionary function exhibits better alignment performance only when $m \leq 7$, and then the performance drops dramatically. Therefore, a linear dictionary function is a more robust choice over a nonlinear Cobb-Douglass dictionary function; however, at proper configurations, a nonlinear Cobb-Douglass dictionary function outperforms a linear dictionary function.



(a) Pose c02 **(b)** Pose c05

(c) Pose c29 **(d)** Pose c14

Fig. 8. Face recognition accuracy on the CMU PIE dataset. The proposed method is denoted as DFL in color red.

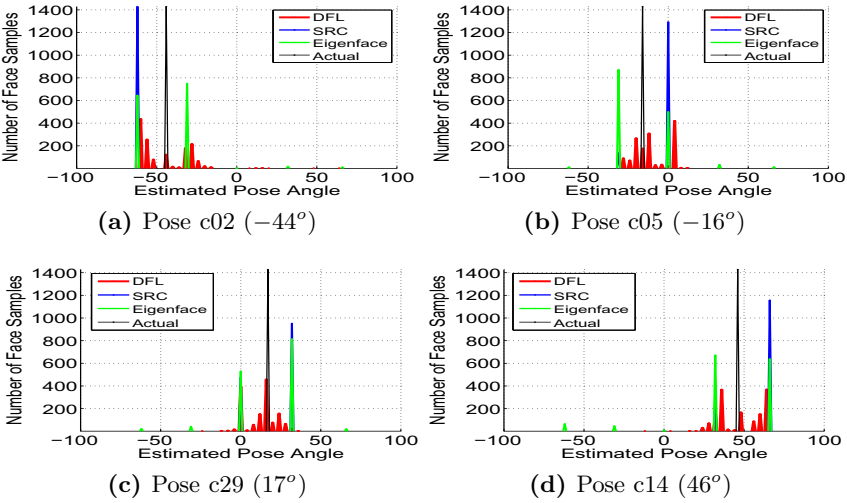
3.2 Dictionary Functions for Classification

Two face recognition methods are adopted for comparisons: Eigenfaces [18] and SRC [19]. Eigenfaces is a benchmark algorithm for face recognition. SRC is a state of the art method to use sparse representation for face recognition. We denote our method as the Dictionary Function Learning (DFL) method. For a fair comparison, we adopt exactly the same configurations for all three methods, i.e., we use 68 subjects in 5 poses c22, c37, c27, c11 and c34 in the PIE dataset for training, and the remaining 4 poses for testing.

For the SRC method, we form a dictionary from the training data for each pose of a subject. For the proposed DFL method, we learn from the training data a dictionary function across pose for each subject. In SRC and DFL, a testing image is classified using the subject label associated with the dictionary or the dictionary function respectively that gives the minimal reconstruction error. In Eigenfaces, a nearest neighbor classifier is used. In Figure 8, we present the face recognition accuracy on the PIE dataset for different testing poses under each lighting condition. The proposed DFL method outperforms both Eigenfaces and SRC methods for all testing poses.

3.3 Dictionary Functions for Domain Estimation

Pose Estimation. As described in Algorithm 1, given a dictionary function, we can estimate the domain parameters associated with an unknown image, e.g., view point or illumination. It can be observed from the face recognition experiments discussed above that the SRC and eigenfaces methods can also estimate the domain parameters based on the domain associated with each dictionary



(a) Pose c02 (-44^o) **(b)** Pose c05 (-16^o)

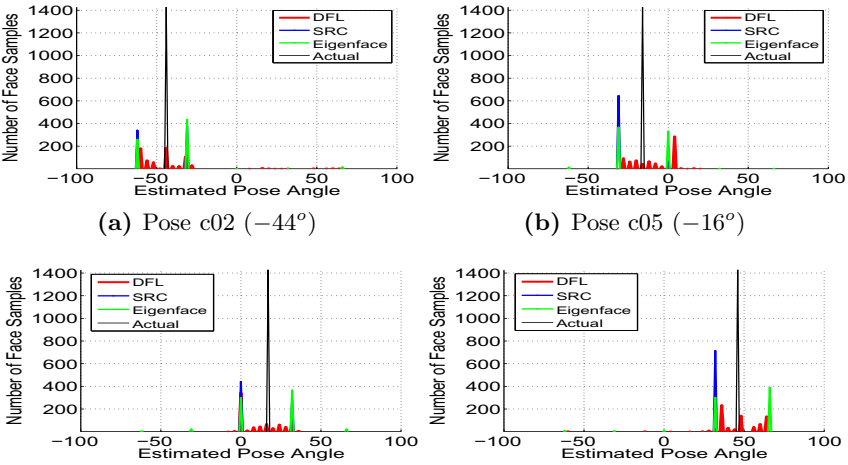
(c) Pose c29 (17^o) **(d)** Pose c14 (46^o)

Fig. 9. Pose azimuth estimation histogram (*known* subjects). Azimuths estimated using the proposed dictionary functions (red) spread around the true values (black).

or each training sample. However, the domain estimation accuracy using such recognition methods is limited by the domain discretization steps present in the training data. We perform pose estimation along with the classification experiments above. We have 4 testing poses and each pose contains 1428 images (68 subjects in 21 lighting conditions). Figure 9 shows the histogram of pose azimuth estimation. We notice that poses estimated from Eigenfaces and SRC methods are limited to one of the 5 training pose azimuths, i.e., -62^o (c22), -31^o (c37), 00^o (c27), 32^o (c11) and 66^o (c34). As shown in Figure 9, the proposed DFL method enables a more accurate pose estimation, and poses estimated through the DFL method are distributed in a continuous region around the true pose.

To demonstrate that a dictionary function can be used for domain estimation for unknown subjects, we use the first 34 subjects in 5 poses c22, c37, c27, c11 and c34 in the PIE dataset for training, and the remaining 34 subjects in the rest 4 poses for testing. We learn from the training data a dictionary function across pose over the first 34 subjects. As shown in Figure 10, the proposed DFL method provides a more accurate continuous pose estimation.

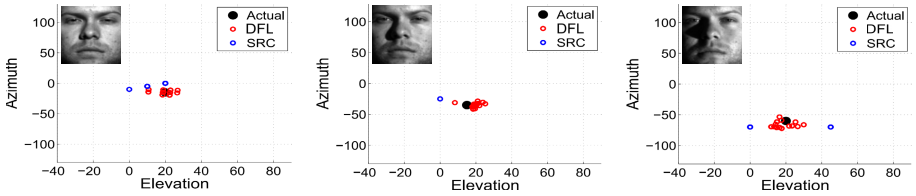
Illumination Estimation. In this set of experiments, given a face image in the Extended YaleB dataset, we estimate the azimuth and elevation of the single light source direction. We randomly select 50% (32) of the lighting conditions in the Extended YaleB dataset to learn a dictionary function across illumination over all 34 subjects. The remaining 32 lighting conditions are used for testing. For the SRC method and for each training illumination condition, we form a dictionary from the training data using all 34 subjects. We perform illumination estimation in a similar way as pose estimation. Figure 11a, 11b, and 11c show the illumination estimation for several example lighting conditions. The proposed DFL method provides reasonable estimation to the actual light source directions.



(a) Pose c02 (-44^o) **(b)** Pose c05 (-16^o)

(c) Pose c29 (17^o) **(d)** Pose c14 (46^o)

Fig. 10. Pose azimuth estimation histogram (*unknown* subjects). Azimuths estimated using the proposed dictionary functions (red) spread around the true values (black).



(a) Lighting condition f40 **(b)** Lighting condition f45 **(c)** Lighting condition f51

Fig. 11. Illumination estimation in the Extended YaleB face dataset

4 Conclusion

We have presented a general dictionary function learning framework to transform a dictionary learned from one domain to the other. Domain dictionaries are modeled by a parametric function. The dictionary function parameters and domain-invariant sparse codes are then jointly learned by solving an optimization problem with a sparsity constraint. Extensive experiments on real datasets demonstrate the effectiveness of our approach on applications such as pose alignment, pose and illumination estimation and face recognition. The proposed framework can be generalized for non-linearizeable dictionary functions, however, further experimental evaluations are to be performed.

Acknowledgment. This work was supported by a MURI grant N00014-10-1-0934 from the Office of Naval Research.

A A Nonlinear Dictionary Function Learning Algorithm

Input: signals in N different domains $\{\mathbf{Y_i}\}_{\mathbf{i=1}}^{\mathbf{N}}$, domain parameter matrix $\boldsymbol{\Theta}$
Output: dictionary function $\mathbf{W}$
begin
  Initialization:
  1. Create the stack signal $\mathbf{Y}$ and initialize $\mathbf{D}$ from $\mathbf{Y}$ using K-SVD;
  2. Initialize $\mathbf{W}$ with random values ;
  repeat
    3. Compute current residuals: $\mathbf{R} \leftarrow \mathbf{D} - \mathbf{F}(\boldsymbol{\Theta}, \mathbf{W})$;
    4. Compute the row vector of derivatives w.r.t. $\mathbf{W}$ evaluated at $\boldsymbol{\Theta}$ $\mathbf{P} \leftarrow \nabla \mathbf{F}(\boldsymbol{\Theta}, \mathbf{W})$;
    5. Learn the linear dictionary function $\mathbf{B}$ using $\mathbf{R} = \mathbf{PB}$
    6. Update the dictionary function parameters: $\mathbf{W} \leftarrow \mathbf{W} + \lambda \mathbf{B}$
  until *convergence*;
  7. return $\mathbf{W}$;
end

Algorithm 2. A general method for nonlinear dictionary function learning

References

1. Wright, J., Ma, Y., Mairal, J., Sapiro, G., Huang, T., Yan, S.: Sparse representation for computer vision and pattern recognition. Proceedings of the IEEE 98, 1031–1044 (2010)
2. Rubinstein, R., Bruckstein, A., Elad, M.: Dictionaries for sparse representation modeling. Proceedings of the IEEE 98, 1045–1057 (2010)
3. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.: A theory of learning fromdifferent domains. Machine Learning 79, 151–175 (2010)
4. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowledge and Data Engineering 22, 1345–1359 (2010)
5. Gong, S., McKenna, S.J., Psarrou, A.: Dynamic vision from images to face recognition. Imperial College Press (2000)
6. Vetter, T., Poggio, T.: Linear object classes and image synthesis from a single example image. PAMI 19, 733–742 (1997)
7. Beymer, D., Shashua, A., Poggio, T.: Example-based image analysis and synthesis. Artificial Intelligence Laboratory A.I. Memo No. 1431 19 (1993)
8. Beymer, D., Poggio, T.: Face recognition from one example view. Artificial Intelligence Laboratory A.I. Memo No. 1536 19 (1995)
9. Lancaster, P., Salkauskas, K.: Curve and surface fitting (1990)
10. Edelman, A., Arias, T.A., Smith, S.T.: The geometry of algorithms with orthogonality constraints. SIAM J. Matrix Analysis and Applications 20, 303–353 (1999)
11. Elhamifar, E., Vidal, R.: Sparse subspace clustering. In: CVPR (2009)
12. Turaga, P., Veeraraghavan, A., Srivastava, A., Chellappa, R.: Statistical Analysis on Manifolds and its Applications to Video Analysis. In: Schonfeld, D., Shan, C., Tao, D., Wang, L. (eds.) Video Search and Mining. SCI, vol. 287, pp. 115–144. Springer, Heidelberg (2010)

13. Aharon, M., Elad, M., Bruckstein, A.: k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans. on Signal Process. 54, 4311–4322 (2006)
14. Machado, L., Leite, F.S.: Fitting smooth paths on riemannian manifolds. Int. J. Appl. Math. Stat. 4, 25–53 (2006)
15. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination, and expression (PIE) database. PAMI 25, 1615–1618 (2003)
16. Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J.: From few to many: Illumination cone models for face recognition under variable lighting and pose. PAMI 23, 643–660 (2001)
17. Pati, Y.C., Rezaiifar, R., Krishnaprasad, P.S.: Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In: Asilomar Conf. on Signals, Systems, and Computers (1993)
18. Turk, M., Pentland, A.: Face recognition using eigenfaces. In: CVPR (1991)
19. Wright, J., Yang, A., Ganesh, A., Sastry, S., Ma, Y.: Robust face recognition via sparse representation. PAMI 31, 210–227 (2009)