Factorization Towards a Classifier

Qiang Chen¹, Shuicheng Yan¹, and Tian-Tsong Ng²

¹ Department of Electrical and Computer Engineering, National University of Singapore, Singapore
² Computer Vision and Image Understanding Department, Institute for Infocomm Research, Singapore
{chenqiang, eleyans}@nus.edu.sg, ttng@i2r.a-star.edu.sg

Abstract

In practice, nonnegative data factorization is often performed for data dimensionality reduction prior to a classification task using a classifier which is effective in low dimensional space such as nearest neighbor classifier. In this work, we propose a novel formulation to learn a multi-class classifier directly through a supervised nonnegative data factorization. This new formulation has the following properties: 1) the nonnegative data matrix is approximated as the product of a nonnegative basis matrix and a coefficient matrix where the nonnegative bases distinctively capture the common characteristics of all classes apart from that specific to individual classes; 2) a regularization term is imposed on nonnegative data factorization so that each datum can be predominantly reconstructed by the common basis vectors and its corresponding class-specific basis vectors; and 3) the coefficient vector for each datum is assumed to be transformed from a mapped kernel space, and the L₂ norm of the class-specific coefficients reveals the relative confidence of classes, which then directly leads to a multi-class classifier. We also present an iterative optimization technique for our formulation and analytically show its convergence property. Extensive experiments on face recognition, head pose estimation, and handwritten digit recognition tasks clearly demonstrate the advantages of the proposed classifier over the conventional two-step approach of nonnegative data factorization followed by a classic classifier.

1. Introduction

In machine learning, data analysis is often applied for finding a data representation suited for the subsequent data clustering or classification tasks. Motivated by the psychological and physiological evidences for parts-based representation of mental objects in the human brain [16], nonnegative data factorization that captures the notion of non-subtractive parts was proposed. In a pioneering work [12], a data matrix X is approximated through nonnegative matrix factorization (NMF) by the product of two nonnegative basis vectors and its corresponding class-specific basis vectors; and 3) the coefficient vector for each datum is assumed to be transformed from a mapped kernel space, and the L₂ norm of the class-specific coefficients reveals the relative confidence of classes, which then directly leads to a multi-class classifier. We also present an iterative optimization technique for our formulation and analytically show its convergence property. Extensive experiments on face recognition, head pose estimation, and handwritten digit recognition tasks clearly demonstrate the advantages of the proposed classifier over the conventional two-step approach of nonnegative data factorization followed by a classic classifier.

Figure 1. Illustration of class-specific data decomposition (pose estimation problem) idea. A face image from the 4-th pose class (from dataset CMU PIE [22]) is decomposed as two parts within: the common bases U₀ and class-specific bases Uᵢ. As can be seen intuitively, the common bases contain the inter-class variations (pose, illumination and identity changes) while the class-specific bases toilet the intra-class differences and capture the discriminative strength. The L₂ norms of corresponding coefficient vectors over different class-specific sub-matrices convey the relative confidences for different classes. The pose exemplar image from the 4-th pose class is best represented by U₀ + U₄.

matrices, i.e., X = UV, where U and V are called the basis matrix and coefficient matrix respectively. The nonnegativity property of NMF distinguishes NMF from many other linear representations such as principal component analysis (PCA) [10] and independent component analysis (ICA) [6]. NMF and its extensions [14, 4, 29, 25] have been shown to be useful for applications including face recognition [14, 7], text mining [27], polyphonic music transcription [21], and collaborative filtering [26].

The conventional NMF and its variants focus on uncovering the inner-structure of data for dimensionality reduction. Lee et al. [12] proposed two multiplicative algorithms for computing NMF. Since the original NMF may produce holistic bases, Li et al. [14] imposed localization constraints in parts-based representation. In [7], Hoyer incorporated the notion of ‘sparseness’ into the standard NMF so the spar-
sity of parts-based representation can be better controlled. The work in [32] generalized the original NMF to kernel NMF (KNMF) in order to extract the additive characteristics in data left unexplored by NMF through a nonlinear mapping to a higher dimensional space. NMF was originally proposed as an unsupervised method. Recently, Yang et al. [30] proposed a general solution for supervised nonnegative graph embedding by integrating the characteristics of both intrinsic and penalty graphs of the data. This supervised method is more powerful for classification tasks but it suffered from high computational cost. Wang et al. [25] further proposed an efficient computation procedure for nonnegative graph multiplicative update rules similar to that used for computing the original NMF.

For a classification task, data is often decomposed by NMF for dimensionality reduction before going through a classifier, e.g., nearest neighbor classifier, which is effective in low dimensional space. However, such a two-step procedure has two issues concerning in test data onto the nonnegative bases and in classifier generalizability. A test sample is generally projected on the nonnegative bases by minimizing the least squared reconstruction error. While such projection is straight-forward, it does not guarantee that the test data coefficients are nonnegative and hence violate the non-subtractive notion of a parts-based representation. As the nonnegative coefficients of training data are used for classifier learning, the appearance of negative test data coefficients amounts to a discrepancy between the training and testing distribution. As a result, the classifier model may not be generalized to the deviant test data and this can lead to degraded classification performance. To ensure nonnegativity, test coefficients can be computed with the update rules similar to that of NMF. However, such procedure is significantly more computationally expansive than least squared projection and hence may be impractical for many real-world applications.

In this work, we propose a novel multi-class classifier learning algorithm which directly learns a classifier through a supervised nonnegative data factorization process. Therefore, we name our method "Factorization towards A Classifier" (FAC). Unlike the previous works, we bypass the intermediate step of dimensionality reduction and directly transfer test samples to the final class-specific coefficients. To achieve this end, we derive a variant of NMF with the following characteristics: 1) the nonnegative data matrix is approximated as the product of a nonnegative basis matrix $U$ and a coefficient matrix $V$ where the nonnegative bases distinctively capture the common characteristics of all classes apart from those specific to individual classes; 2) a regularization term is imposed on nonnegative data factorization so that each datum is predominantly reconstructed by the common basis matrix and its corresponding class-specific basis matrix. 3) we further assume that the coefficient vector of a datum is transformed from a mapped kernel space. Through this formulation, the $l_2$ norm of the class-specific coefficients reveals the relative confidence of the datum’s class and hence it directly leads to a multi-class classifier. We also present an iterative optimization technique for our formulation and analytically show its convergence property.

The main contributions of this paper are two-fold. First, we design a supervised nonnegative data factorization formulation which is class-specific. Fig. 1 shows this class-specific data decomposition idea. Second, we derive a multi-class classifier directly from this formulation. Classification is obtained directly as the coefficient vector naturally reveals the class confidence of a test sample. This way of using NMF is not only different from the conventional two-step procedure, it outperforms the latter as shown in our experiments.

2. Learning a Classifier by Nonnegative Data Factorization

In this section, we formulate a multi-class classifier within the framework of nonnegative data decomposition. In this work, we utilize the following rules and notations to facilitate presentation. For any matrix $A$, $A_i$ represents its $i$-th row vector, $a_i$ its $i$-th column vector, and $A_{ij}$ the element of $A$ at row $i$ and column $j$. $A^i$ denotes the sub-matrix of $A$ with column vectors related to class $i$. $X = [x_1, x_2, ..., x_N]$ represents the training data matrix where $x_i \in \mathbb{R}^m$ is the $i$-th datum’s feature descriptor and $N$ is the total number of training samples. Here, we assume that $X$ is nonnegative. The corresponding class label vector is denoted as $L = [l_1, l_2, ..., l_N] \in \mathbb{R}^N$, where $l_i \in [1, N_c]$ and $N_c$ is the total number of classes.

2.1. Motivations

Most conventional nonnegative data factorization algorithms aim to obtain an informative and common nonnegative low-dimensional vector space. In this work, we pursue a different data decomposition strategy. As shown in Subsection 2.3, we decompose $X$ in a supervised manner and impose a structure on the basis matrix $U$ so that $U$ consists of two parts which respectively correspond to the shared feature space and the class-specific feature spaces. In this case, $U = [U^0, U^1, U^2, ..., U^N_c] \in \mathbb{R}^{m \times n}$, where $U^i \in \mathbb{R}^{m \times n_i}$ and $\sum_{i=0}^{N_c} n_i = n$. $U^0$ represents the common basis sub-matrix which captures the common characteristics of all classes and $U^i, 1 \leq i \leq N_c$, are the basis sub-matrices that captures the specific characteristics of class $i$. For the basis matrix $U$ to have such a structure, we constrain the coefficient matrix $V$ so that a strong correspondence between the data coefficients and the data labels can be forged, as shown in Subsection 2.4. Note that, once the samples
from class $i$ can be reconstructed by the common basis sub-matrix $U^0$ and predominantly the class $i$ basis sub-matrix $U^i$, then the coefficient vectors would be inherently class indicative and hence we achieve our goal of producing a classifier directly from the decomposition process.

### 2.2. Coefficient Vector

In this work, the coefficient vector for each datum is assumed to be linearly transformed from the datum’s feature vector in a mapped kernel feature space, i.e., we relate a coefficient vector to its corresponding datum. This setup is not only impose a structure on the coefficient vector, most importantly it turns the nonnegative data factorization formulation into a classifier. As we make the coefficient vector indicative of the datum’s class label, we effectively turn the coefficient vector into a classification decision. As a result, the mapping from a datum to its class-indicative coefficient vector is essentially a classifier embedded in the nonnegative data factorization formulation.

A kernel mapping function $\phi$ maps a datum $x_i$ into a higher or even infinite dimensional feature space: $\phi : x \in \mathbb{R}^m \rightarrow F, dim(F) \gg m$ [18], and then the coefficient vector $v_i$ is linearly transformed from $\phi(x_i) \in F$ by multiplying a projection matrix $H$. Formally, we can obtain the coefficient matrix $V$ from the data matrix $X$ as follows:

$$v_i = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_k \end{bmatrix} \phi(x_i) = H \phi(x_i), \quad \text{or} \quad V = H\Psi(X),$$

where $\Psi(X) = [\phi(x_1), \phi(x_2), ..., \phi(x_N)]$. Furthermore, without losing generality, we can assume that the row vectors of the projection matrix $H$ lie within the span of $\{\phi(x_i)\}$, namely, $H = P\Psi(X)^T$, where $P \in \mathbb{R}^{k \times N}$. Then Eqn. (1) becomes

$$v_i = P\Psi(X)^T \phi(x_i), \quad \text{or} \quad V = P\Psi(X)^T \Psi(X) = PK,$$

where $K = \Psi(X)^T \Psi(X) \in \mathbb{R}^{N \times N}$. Note that $K$ is a Gram matrix whose entries are $K_{ij} = \phi(x_i)^T \phi(x_j) = k(x_i, x_j)$, where $k(\cdot, \cdot)$ is the kernel function associated with the mapping function $\phi(\cdot)$. There exists some popular kernel functions including Gaussian Kernel, Polynomial Kernel and Sigmoid Kernel [18]. For these popular kernels, if the data matrix $X$ is nonnegative, the corresponding kernel matrix shall also be nonnegative.

Note that the relationship between the feature vector $x_i$ and corresponding coefficient vector $v_i$ may be nonlinear. Instead of trying to fit it into a nonlinear model, we map the feature space by a nonlinear transformation $\phi(x)$. This “kernel trick” [18], which allows us to work in the mapped kernel space without knowing the mapping explicitly, can naturally handle the nonlinear problem.

### 2.3. Objective for Nonnegative Data Factorization

Nonnegative data factorization decomposes a nonnegative data matrix $X$ into the product of a basis matrix $U$ and a coefficient matrix $V$. As shown in Subsection 2.2, the coefficient matrix is assumed to be further decomposed as the product of a transformation matrix $P$ and a kernel matrix $K$, i.e., $V = PK$. So the objective function for nonnegative data factorization can be formulated as follows:

$$\min_{U,P} ||X - UPK||^2, \quad \text{s.t.} \quad U, P \geq 0, \quad \text{(3)}$$

The goal of nonnegative data factorization is to minimize the data reconstruction error. The learned projective matrix $P$ is critical in the process of classification as it produces the coefficients for the class-specific basis sub-matrices $U^i, 1 \leq i \leq N$, and the $l_2$ norm of the class-specific coefficients reveals the relative confidence of a class.

### 2.4. Objective for Class-specific Coefficient Matrix

Intuitively, the formulation in Eqn. (3) does not guarantee that the sample data $x_i$ from the class $j$ should only be reconstructed by the specific basis matrix $U^j$ and $U^j$. We thus impose a regularization term to constrain the coefficient matrix $V = PK$ as follows:

$$\min_{P} ||(PK) \circ B||^2, \quad \text{(4)}$$

where $\circ$ is the Hadamard (elementwise) product of two matrices and $B = [b_1, b_2, ..., b_N] \in \mathbb{R}^{N \times N}$ as an indicator matrix of data labels. For a datum $x_i$ from class $j$, the elements in $b_i$ corresponding to $U^0$ and $U^j$ are set to 0 while others are set to 1. In this way, the $l_2$ norm of $(PK) \circ B$ can be considered as the approximate measure of the classification error for the whole training data. Note that, in this work, the element-wise product operator has lower priority than the matrix multiplication operator.

### 2.5. Unified Formulation

We combine Eqn. (3) and Eqn. (4) into a unified objective function as,

$$\min_{U,P} ||X - UPK||^2 + \lambda ||(PK) \circ B||^2, \quad \text{s.t.} \quad U, P \geq 0, \quad \text{(5)}$$

where $\lambda$ is a tradeoff parameter to balance these two terms. We notice that for classification purpose, it is too restrictive to impose the nonnegative constraint on $P$, so we relax this constraint into a constant lower and negative bound. Suppose $P^* = P - E$, where $P$ still satisfies the nonnegative property and $E \in \mathbb{R}^{N \times N}$ is a constant positive matrix (all elements of $E$ are initialized to 1 for all experiments). We set $P^*$ as the projective matrix instead. Besides, this formulation is ill-posed, as the objective function has a tendency
to drive the $P - E$ to be zero, a problem also encountered in Fisher-NMF [24]. To overcome this problem, it is common to limit the basis (column) vectors of $U$ to be of unit length, namely,

$$\|u_i\| = 1, \quad i = 1, 2, \ldots, n.$$  \hfill (6)

This extra constraint makes the optimization problem more complicated, and in this work we choose to merge the basis length normalization into the coefficient matrix and get the final objective function as,

$$\min_{U, P} \|X - U(P - E)K\|^2 + \lambda\|Q(P - E)K \circ B\|^2,$n

s.t. $U, P \geq 0$,

where $Q = \text{diag}\{\|u_1\|, \|u_2\|, \ldots, \|u_n\|\}$ with elements $\|u_i\|$ at the diagonal.

### 2.6. Multi-class Classifier

After obtaining the proper $U$ and $P$, we can naturally derive a multi-class classifier. Given a test sample $y$, we first map $y$ into the kernel feature space, then the learned transformation matrix $P$ is used to get the coefficient $v$:

$$v = (P - E)\Psi(X)^T\phi(y) = (P - E)k(X, y),$$  \hfill (8)

where $k(x_i, y) = [k(x_1, y), k(x_2, y), \ldots, k(x_N, y)]^T$. After getting $v_i$, the $l_2$ norm of the class-specific coefficients reveals the relative confidence of the test sample belonging to a class, and the classification can be performed as follows,

$$\arg \max_{i \in \{1, 2, \ldots, n\}} \|v^i\|,$$  \hfill (9)

where $v^i$ is the sub-vector of $v$ corresponding to the sub-matrix $U^i$.

In next section, we present a convergence provable updating procedure which is used for optimizing the proposed objective function of nonnegative data factorization.

### 3. Multiplicative Iterative Solution

Most iterative procedures for solving high-order optimization problem transform the original intractable problem into a set of tractable sub-problems and ensure a convergence to a local optimum. Our proposed iterative procedure also follows this philosophy and optimizes $U$ and $P$ alternately.

#### 3.1. Optimize $U$ for Given $P$

If we suppose $P$ is given, to integrate the nonnegative constraint into the objective function, we can set $\Upsilon^u_{ab}$ as the Lagrange multiplier for the constraint $U_{ab} \geq 0$ and define the matrix $\Upsilon^u = [\Upsilon^u_{ab}]$. Then the Lagrangian $\mathcal{L}(U)$ with respect to $U$ is defined as,

$$\mathcal{L}(U) = \|X - U(P - E)K\|^2 + \lambda\|(Q(P - E)K \circ B\|^2 + Tr(\Upsilon^u U^T)$$  \hfill (10)

where $\text{Tr}(\cdot)$ and $(\cdot)$ are respectively the trace and the transpose of a matrix. Since $Q$ is a diagonal matrix of size $n \times n$, the Eqn. (10) turns to,

$$\mathcal{L}(U) = \text{Tr}(X X^T) - 2\text{Tr}(X K^T (P - E) U^T) + \lambda\text{Tr}(\Upsilon^u U)$$

$$+ \text{Tr}(U(P - E) K K^T (P - E) U^T) + \text{Tr}(\Upsilon^u U^T) + \lambda\text{Tr}((Q(P - E) K \circ B)(B^T \circ (K^T (P - E)^T) Q^T).$$

Let $P' = P - E$ and $Y = ((P' K) \circ B)(B^T \circ (K^T P' T)) \circ I$, where $I$ is an identity matrix. By applying the equality of $\text{Tr}(Q C Q^T) = \text{Tr}(U(C \circ I) U^T)$, Eqn. (11) becomes

$$\mathcal{L}(U) = \text{Tr}(X X^T) - 2\text{Tr}(X K^T P' T U^T) + \lambda\text{Tr}(U' Y U) + \text{Tr}(\Upsilon^u U^T).$$

By setting the derivation of $\mathcal{L}(U)$ with respect to $U$ as zero, we have

$$\frac{\partial \mathcal{L}}{\partial U} = -2XX^T P' T + 2U' K K^T P' T + 2\lambda U Y + \Upsilon^u.$$  \hfill (13)

According to KKT [8] condition of $\Upsilon^u_{ab} U_{ab} = 0$, we get the following equation,

$$- (X K^T P' T)_{ab} + \lambda(U Y)_{ab} + (U P K K^T P' T)_{ab} = 0.$$  \hfill (14)

Separate the negative and positive term, then we get,

$$X K^T E^T + U PK K^T P' T + U E K K^T E^T$$

$$- U PK K^T E^T - U E K K^T P' T)_{ab}$$

$$- (X K^T P' T)_{ab} + \lambda(U Y_+ - U Y)_{ab} = 0;$$  \hfill (15)

where

$$Y_+ = ((PK) \circ B)(B^T \circ (K^T P')) \circ I$$

$$+ ((EK) \circ B)(B^T \circ (K^T E^T)) \circ I$$

$$Y = ((EK) \circ B)(B^T \circ (K^T P')) \circ I$$

$$+ ((PK) \circ B)(B^T \circ (K^T E^T)) \circ I.$$  \hfill (16)

Demonstrated by [15], the following updating rule monotonously decreases the objective function. Thereinto, $M = K K^T$.

$$U_{ab} \leftarrow U_{ab} \frac{(X K^T P' T + U P M E^T + U E M P' T + \lambda U Y_+_+)_{ab}}{(X K^T E^T + U P M P' T + U E M E^T + \lambda U Y_+)}.$$  \hfill (17)

#### 3.2. Optimize $P$ for Given $U$

Suppose that $U$ is given, to integrate the nonnegative constraint into the objective function, we can set $\Upsilon^p_{ab}$ as the Lagrange multiplier for the constraint $P_{ab} \geq 0$ and define the matrix $\Upsilon^p = [\Upsilon^p_{ab}]$. Then the Lagrangian $\mathcal{L}(P)$ with respect to $P$ is defined as,

$$\mathcal{L}(P) = \|X - U(P - E)K\|^2 + \lambda\|(P - E) K \circ B\|^2 + \text{Tr}(\Upsilon^p P^T)$$  \hfill (18)
The derivative of \( \mathcal{L}(P) \) with respect to \( P \) can be calculated as,
\[
\frac{\partial \mathcal{L}}{\partial P} = -2U^T XK^T + 2U^T UPKKT^T - 2U^T UEKKT^T + 2\lambda((PK)c B)K^T - ((EK)c B)K^T)
\]
where \( P = ((P - E)K) \circ B \) and \( G(F) = \hat{F}\hat{F}^T \). We have
\[
\frac{\partial G(F)}{\partial P_{ab}} = Tr[(2\hat{F})^T \frac{\partial \hat{F}}{\partial P_{ab}}]
\]
\[
= Tr[2((P - E)K)c B)^T(0 - E)K)c B]
\]
where \( \lambda((P - E)K) \circ B = (J_{ab}K) \circ B, J_{ab} \) is the single-entry matrix with 1 at \( (a, b) \) and zero elsewhere. Thus we can further have
\[
\frac{\partial G(F)}{\partial P_{ab}} = 2(((P - E)K)c B)K^T)
\]
Using matrix notation, this derivative can be written as
\[
\frac{\partial G(F)}{\partial P} = 2(((P - E)K)c B)K^T)
\]
Demonstrated by [15], the following updating rule monotonically decreases the objective function. Thereinto, \( M = KKT^T \).
\[
P_{ab} \leftarrow P_{ab} \frac{(U^T XK + U^T UEMT + \lambda((PK)c B)B)^T)_{ab}}{(U^T UPM + \lambda((PK)c B)B)^T)_{ab}} \]
\]
The matrices \( U \) and \( P \) are optimized alternately until they converge. The detailed procedure is listed in Algorithm 1, where \( T_{max} \) is set as 1000 and \( \varepsilon \) is set as 10^{-5} in this work.

**Algorithm 1.** Factorization towards A Classifier (FAC) Algorithm

1. **Input:** Nonnegative data matrix \( X = [x_1, x_2, \cdots, x_N] \)
and the corresponding class label \( I_i \) for \( x_i \):
2. **Output:** Subspace basis matrix \( U \) and the learnt transformation matrix \( P \);
3. For \( t = 0, 1, 2, \ldots, T_{max} \):
   1. For given \( P = P^t \), the basis matrix \( U \) is updated as Eqn.(17) and we set:
   \[
P_i := \|u_i\|_P \times p_i, \forall i,
   \]
   \[
E_i := \|u_i\|_{E_i} \times e_i, \forall i,
   \]
   \[
u_i := u_i/\|u_i\|_2, \forall i.
   \]
   2. For given \( U = U^t \), the transformation matrix \( P \) is updated as Eqn.(20).
   3. If \( \|U^{t+1} - U^t\| < \varepsilon \) and \( \|P^{t+1} - P^t\| < \varepsilon \), then break.
4. Output \( U = U^t \) and \( P = P^t \).

**4.1. Implementation Details**

Most NMF related algorithms [13, 33, 23] cannot guarantee to achieve the global optimum, and parameter initialization is crucial for a good factorization. Generally, the parameter matrices are randomly initialized. As FAC is supervised and its objective function is relatively more complicated than those of other NMF related algorithms, we utilize a modified supervised NMF-like method for parameter matrix initialization. First, the general NMF is performed,
\[
\min_{U,V} \|X - UV\|_F, \ s.t. \ U,V \geq 0,
\]
by initializing \( v_i \) such that its elements in connection to \( U^0 \) and \( U^1 \) (\( l_i \) is the label index of \( v_i \)) are assigned a random value while other elements are set to zero. It is worth noting that \( V_{ij} \) remains as zero if being initialized as zero in the NMF updating process, and thus the coefficients of \( v_i \) not corresponding to \( U^0 \) and \( U^1 \) shall remain as zeros. After we obtain \( U \) and \( V \) from NMF, the elements of the constant matrix \( E \) is set to the absolute value of \( \min(VK^{-1}) \), while \( U \) and \( VK^{-1} + E \) are then set as the initial matrices for the basis matrix \( U \) and projection matrix \( P \) of FAC. The experiment results show that Algorithm 1 with good initialization usually converges to a local minimum within small number of iterations.

**4.2. Datasets and Baselines**

**4.2.1 Datasets**

We use four public available face datasets, i.e., ORL [3], YALE-B [5], CMU PIE [22] and FRGC V1.0 [20] for face recognition tasks. All images are aligned by fixing the location of the two eyes. For face pose estimation task, PIE
database is used. We select images with 10 poses, 3 random illumination and 59 persons for evaluation, and all images are normalized to a size of 32-by-32 pixels. USPS [9] handwritten digit database is used for handwritten digit recognition.

Each data point of these datasets is normalized to have a unit $l_2$ norm. Five random splits of the training set and testing set are generated for each database according to the descriptions above, and the reported result is the mean value of five rounds of experiment.

4.2.2 Baselines and Setups

Several popular dimensionality reduction algorithms are evaluated for comparison: 1) principal component analysis (PCA) [10], 2) the standard nonnegative matrix factorization (NMF) [12], 3) localized NMF (LNNMF) [14], 4) linear discriminant analysis (LDA) [19], and 5) marginal fisher analysis (MFA) [28]. The algorithms (1-3) are unsupervised, while LDA and MFA are supervised. For classification tasks, we first transform the training and testing images into the same learnt dimension-reduced subspace, and then nearest neighbor classifier is used for final classification. The test image is assigned to the class to which the nearest training sample belongs.

For PCA, we reduce the data dimension such that the subspace contains $95\%, 96\%, \ldots, 100\%$ energy of the training set and we report the best result among these settings. For NMF and LNNMF, the dimension of the subspace is tuned to $k \in \{6 \times 6, 7 \times 7, \ldots, 13 \times 13\}$ for the face recognition and pose estimation tasks and $k \in \{4 \times 4, 5 \times 5, \ldots, 8 \times 8\}$ for the handwritten recognition task. As a common practice [19], before performing LDA and MFA, PCA is used to reduce the feature dimension by retaining $93\%, 95\%, 97\%$ energy of the whole dataset with a lower-bound of $N - N_c$ dimension, where $N_c$ is the class number. For MFA, the number of nearest kindred neighbors of each point $k_1$ is set to $\{2, 3, 4, 5\}$ and we choose $k_2$, the number of the closest out-of-class sample pairs for each class, to be between 20 and $8N_c$ intervals at 20 [28]. The best performance of the multiple settings is herein reported.

We also compare FAC with SVM [1, 2] on these three tasks. The Gaussian radial basis function is used as the kernel function for SVM. The upper bound $C$ and the kernel parameter $\delta^2$ play an important role in the performance of SVM. Note that the same choice of $\delta^2$ as FAC is not optimal for SVM (confirmed by experiments). So we vary these two parameters in the range of $\{10^{-2}, 10^{-1}, 1, 10^1, 10^2, 10^3\}$ respectively to select the optimal values which give the best classification performance. We use LIBSVM [2] for the experiments.

For FAC, the Gaussian Kernel $\exp\left( -\frac{|x-y|^2}{2\sigma^2} \right)$ is used and the parameter $\delta$ in the Gaussian kernel is set to $\{0.8, 0.9, 1.0, 1.1, 1.2\}$, the number of the class-specific basis vectors is set to $\{3, 4, 5\}$, the number of the common basis vectors is set to 5 or 10, and the parameter $\lambda$ is set to 10. Similarly, we report the best result among all the parameter combinations.

4.3. Bases Visualization

As aforementioned, the coefficient vector of a datum derived from FAC is class indicative, and the $l_2$ norm of the coefficients corresponding to a class-specific basis sub-matrix reveals the confidence of that particular class. Fig. 2 depicts the data distribution over different basis vectors to illustrate this property in an head pose estimation example. To draw this figure, we set the column number of each class-specific basis matrix as 2 and that of common basis matrix as 3. In total, 870 testing samples (10 classes and 87 data each class from the CMU PIE dataset) are plotted. We can see that the data distribution over common bases (left figure) are nearly random, while the data shown on class-specific bases (right four figures) are easily separable. It is observed that the data belonging to a target class are generally farther away from the origin, while the data from other classes are nearer to the origin point. Thus the $l_2$ norm of the coefficients corresponding to a class-specific basis sub-matrix conveys the confidence of a datum belonging to that class.

4.4. Classification Capability

We present the classification results for three typical classification problems, i.e., face recognition, head pose estimation, and handwritten digit recognition in the following subsections.

4.4.1 Face Recognition Task

We first conduct the experiments on a face recognition task. Four datasets are used, including ORL, Yale-B, PIE, and FRGC. The comparison results of different algorithms are
The performances of PCA, NMF, and LNMF seem to be generally better than those of LDA, MFA, and the proposed FAC. It shows that supervised learning algorithms are generally superior to the unsupervised ones in this task. The NMF-related algorithms, e.g., standard NMF and LNMF, may give suitable feature representation for data, but their potentials are limited by not considering the label information.

2. The proposed FAC algorithm performs generally better than all other algorithms on all four datasets. NMF and LNMF generally perform better than PCA as the non-subtractive parts-based representation is more suited to this task.

3. On dataset with large number of classes, e.g., FRGC which contains 275 subjects, the FAC shows to be much better than all other algorithms. A possible explanation is that FAC is natural in handling multi-class classification problem.

4.4.2 Pose Estimation Task
We also evaluate the classification capability of FAC compared with other algorithms on a head pose estimation task. Head pose estimation is an important component for a practical multi-view face recognition system [31]. It is essentially a regression problem, and here we simplify this task as a classification problem to categorize the face images into classes of pre-defined pose. The experiments are performed on PIE dataset, which contains 13 poses, 68 subjects and 43 different illumination conditions. We select the faces of 59 subjects with all illumination changes. For every subject, we randomly select 3 illuminations for each of the 10 poses (exclude C02, C14, C27). To avoid confusing the pose estimation problem with face recognition, we separate the data by subject identity, and the images from 30 subjects are used for training and the other 29 subjects are used for testing.

The detailed results are listed in Table 2, from which we can observe that FAC again performs the best among all evaluated algorithms. Another observation is that the supervised dimensionality reduction algorithms, LDA and MFA, do not outperform those unsupervised algorithms too much. A possible explanation is that the discrepancy between the training and testing data due to the difference in subject identity limits the generalization capability of the supervised algorithms like LDA and MFA.

4.4.3 Handwritten Digit Recognition Task
The final task for evaluating FAC is handwritten digit recognition [11, 17]. As we do not always write the same character in exactly the same way, it is generally very difficult to build a robust general handwriting recognition system.

The detailed results on handwritten digit recognition on USPS dataset are listed in Table 3, from which we can see that FAC gets much higher accuracy than all the other unsupervised or supervised algorithms. It is worth noting that PCA gets the second highest score, and both NMF and LNMF get higher scores than LDA and MFA. A possible explanation is that the distribution of the training data is different from that of the testing data, and the derived dimension-reduced feature representations in a supervised manner easily overfit to the training data and hence could not generalize well on the testing data compared to those unsupervised methods.

4.4.4 A Comparison of SVM and FAC
Three classification tasks, i.e. face recognition (denoted as “Face”), pose estimation (denoted as “Pose”), digit recognition (denoted as “Digit”) are conducted for comparing SVM and FAC. Dataset preparation is the same as previous experiments. All the data are first normalized to unit length before classification.

The detailed results are listed on Table 4, from which it can be shown that FAC outperforms SVM by 2.1%, 7.51%, 11.48%, 5.08%, 0.43% and 0.07% respectively on the six experiments. We can observe that the results for SVM on Face(Yale-B) and Face(PIE) are worse than FAC’s by a larger margin. In these cases, few samples are provided for training therefore SVM has difficulties finding the optimal hyperplane, and this implies that our FAC has a much stronger generalizability compared with SVM in the case of small sample size.

Acknowledgment
This research was partially supported by IDMPO-MDA project number NRF20071DM-1DM002-069 on Live Spaces: Place Oriented Embodied Media (POEM) and by AcRF Tier-1 Grant of R-263-000-464-112, Singapore.

References
Table 1. Face Recognition accuracies (%) of different algorithms on four datasets, ORL, Yale-B, PIE, and FRGC. The first column lists the algorithms to evaluate, and other columns show the average values and standard deviations (in the parentheses) of the results from five random splits of the datasets.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>ORL</th>
<th>Yale-B</th>
<th>PIE</th>
<th>FRGC</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>86.4 (±2.04)</td>
<td>52.82 (±0.95)</td>
<td>61.39 (±0.75)</td>
<td>77.35 (±0.27)</td>
</tr>
<tr>
<td>NMF</td>
<td>88.4 (±2.27)</td>
<td>81.02 (±0.49)</td>
<td>81.45 (±0.39)</td>
<td>83.16 (±0.56)</td>
</tr>
<tr>
<td>LNMF</td>
<td>88.8 (±2.28)</td>
<td>75.99 (±0.44)</td>
<td>83.35 (±0.59)</td>
<td>79.17 (±0.98)</td>
</tr>
<tr>
<td>PCA+LDA</td>
<td>94.4 (±1.29)</td>
<td>87.80 (±0.59)</td>
<td>92.17 (±0.52)</td>
<td>87.75 (±0.49)</td>
</tr>
<tr>
<td>PCA+MFA</td>
<td>95.4 (±1.43)</td>
<td>88.25 (±0.99)</td>
<td>91.15 (±0.23)</td>
<td>85.99 (±1.09)</td>
</tr>
<tr>
<td>FAC</td>
<td>95.4 (±1.51)</td>
<td>88.53 (±0.51)</td>
<td>93.08 (±0.27)</td>
<td>90.52 (±0.63)</td>
</tr>
</tbody>
</table>

Table 2. Head pose estimation accuracies (%) of different algorithms on PIE dataset. The detailed results on three individual poses are also listed for different algorithms. The results include the average values and standard deviations (in the parentheses) of the results from five random splits of the dataset.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Pose 1 (C05)</th>
<th>Pose 2 (C07)</th>
<th>Pose 9 (C34)</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>94.94 (±2.89)</td>
<td>76.55 (±2.89)</td>
<td>87.59 (±2.98)</td>
<td>85.08 (±0.46)</td>
</tr>
<tr>
<td>NMF</td>
<td>94.94 (±1.31)</td>
<td>74.02 (±5.30)</td>
<td>85.06 (±1.15)</td>
<td>82.74 (±1.40)</td>
</tr>
<tr>
<td>LNMF</td>
<td>92.87 (±3.76)</td>
<td>71.03 (±5.88)</td>
<td>85.98 (±2.49)</td>
<td>83.95 (±0.64)</td>
</tr>
<tr>
<td>PCA+LDA</td>
<td>94.94 (±2.89)</td>
<td>76.09 (±3.19)</td>
<td>86.90 (±2.09)</td>
<td>84.94 (±0.65)</td>
</tr>
<tr>
<td>PCA+MFA</td>
<td>94.02 (±1.26)</td>
<td>81.15 (±1.92)</td>
<td>88.97 (±5.30)</td>
<td>86.90 (±0.92)</td>
</tr>
<tr>
<td>FAC</td>
<td>95.63 (±0.96)</td>
<td>88.74 (±1.89)</td>
<td>93.79 (±2.52)</td>
<td>91.95 (±0.56)</td>
</tr>
</tbody>
</table>

Table 3. Handwritten digit recognition accuracies (%) of different algorithms on USPS dataset. The detailed results on three representative digits are also listed for different algorithms. The listed accuracies are the average values and standard deviations (in the parentheses) of the results from five random splits of the dataset.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Digit '1'</th>
<th>Digit '4'</th>
<th>Digit '8'</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>99.89 (±0.10)</td>
<td>92.87 (±1.23)</td>
<td>91.53 (±0.87)</td>
<td>96.13 (±0.11)</td>
</tr>
<tr>
<td>NMF</td>
<td>99.71 (±0.21)</td>
<td>93.13 (±1.48)</td>
<td>90.29 (±0.94)</td>
<td>95.11 (±0.31)</td>
</tr>
<tr>
<td>LNMF</td>
<td>99.75 (±0.16)</td>
<td>94.25 (±0.76)</td>
<td>92.65 (±0.97)</td>
<td>96.53 (±0.26)</td>
</tr>
<tr>
<td>PCA+LDA</td>
<td>97.71 (±0.79)</td>
<td>94.47 (±0.80)</td>
<td>88.29 (±1.96)</td>
<td>93.40 (±0.28)</td>
</tr>
<tr>
<td>PCA+MFA</td>
<td>97.71 (±0.51)</td>
<td>100.00 (±0.00)</td>
<td>88.51 (±2.57)</td>
<td>93.49 (±0.55)</td>
</tr>
<tr>
<td>FAC</td>
<td>97.78 (±0.24)</td>
<td>98.29 (±0.21)</td>
<td>95.82 (±0.43)</td>
<td>98.07 (±0.10)</td>
</tr>
</tbody>
</table>

Table 4. Classification accuracies (%) of Support Vector Machine and proposed FAC on three tasks, i.e. face recognition(Face), pose estimation(Pose), digit recognition(Digit). The first column lists the algorithms to evaluate, and other columns show the average values and standard deviations (in the parentheses) of the results from five random splits of the datasets.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Face (ORL)</th>
<th>Face (Yale-B)</th>
<th>Face (PIE)</th>
<th>Face (FRGC)</th>
<th>Pose (PIE)</th>
<th>Digit (USPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>93.30 (±1.04)</td>
<td>81.02 (±0.38)</td>
<td>81.50 (±0.33)</td>
<td>85.46 (±0.69)</td>
<td>91.52 (±1.00)</td>
<td>98.00 (±0.17)</td>
</tr>
<tr>
<td>FAC</td>
<td>95.40 (±1.51)</td>
<td>88.53 (±0.51)</td>
<td>93.08 (±0.27)</td>
<td>90.52 (±0.63)</td>
<td>91.95 (±0.56)</td>
<td>98.07 (±0.10)</td>
</tr>
</tbody>
</table>


