Two-Dimensional Non-negative Matrix Factorization for Face Representation and Recognition

Daoqiang Zhang[1, 2], Songcan Chen[1], and Zhi-Hua Zhou[2, *]

[1] Department of Computer Science and Engineering,
Nanjing University of Aeronautics and Astronautics,
Nanjing 210016, China
{dqzhang, s.chen}@nuaa.edu.cn

[2] National Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210093, China
zhouzh@nju.edu.cn

Abstract. Non-negative matrix factorization (NMF) is a recently developed method for finding parts-based representation of non-negative data such as face images. Although it has successfully been applied in several applications, directly using NMF for face recognition often leads to low performance. Moreover, when performing on large databases, NMF needs considerable computational costs. In this paper, we propose a novel NMF method, namely 2DNMF, which stands for 2-D non-negative matrix factorization. The main difference between NMF and 2DNMF is that the former first align images into 1D vectors and then represents them with a set of 1D bases, while the latter regards images as 2D matrices and represents them with a set of 2D bases. Experimental results on several face databases show that 2DNMF has better image reconstruction quality than NMF under the same compression ratio. Also the running time of 2DNMF is less, and the recognition accuracy higher than that of NMF.

1 Introduction

There is psychological and physiological evidence for parts-based representations in the brain, and certain computational theories of object rely on such representations [11]. For that reason, parts-based learning has received much interest in machine learning, computer vision and pattern recognition [13]. Many parts-based image representation approaches ca be ascribed to a general subspace method, which has been successfully used in many high dimensional data analysis applications. Given a class of image patterns, there are many approaches to construct the subspace. One such method is principal component analysis (PCA) [10], also known as Eigenface method in face recognition [19]. In PCA, any image can be represented as a linear combination of a set of orthogonal bases which form an optimal transform in the sense of reconstruction error. However, due to the holistic nature of the method, PCA cannot extract basis components manifesting localized features [13]. And its two extensions: independent component analysis (ICA) [4], [17] and kernel principal component analysis (KPCA) [18] also have the same problem.

[*] Corresponding author.

Recently a new subspace method called non-negative matrix factorization (NMF) [11] is proposed to learn the parts of objects and images. NMF imposes the non-negativity constraints in its bases and coefficients. Thus NMF learns localized features that can be added together to reconstruct the whole images, because only additive combination, not subtractive cancellations, are allowed in the reconstruction [11], [20], [6], [9], [5]. The localized, parts-based representation is very different from the holistic 'eigenface' of PCA. And due to its parts-based representation property, NMF or its variations have been used to image classification [1], [7], [8], face expression recognition [2], face detection [3], face and object recognition [13], [14], [15]. However, experiments have shown that when used for image compression and recognition tasks, NMF usually has low image reconstruction image quality and low recognition accuracy. Also NMF needs comparatively more computational costs due to the alternate iterations. To try to overcome those problems, many improved algorithms are proposed including local NMF [13] and sparse NMF [9] which impose extra constraints on the bases. But those methods often need even more iteration time to learn the bases, especially for high-dimensional data such as faces.

In this paper, we present a novel NMF method, called 2-Dimensional non-negative matrix factorization (2DNMF) for image representation and recognition. The key difference between 2DNMF and NMF is that the former adopt a novel representation for original images. In traditional NMF, the 2D image matrices must be previously transformed into 1D image vectors. The resulting image vectors usually lead to a high-dimensional image vector space, where it is difficult to find good bases to approximately reconstruct original images. That is also called the 'curse of dimensionality' problem, which is more apparent in small-sample-size cases. Another disadvantage of NMF is that such a matrix-to-vector transform may cause the loss of some structure information hiding in original 2D images. In contrast to the 1D representation of NMF, we adopt a more natural 2D matrix representation in 2DNMF, i.e. representing 2D images with a set of 2D bases. We first apply NMF on column vectors and then row vectors of original images to obtain the corresponding 1D column bases and 1D row bases respectively, and finally compute the outer-product of those two 1D bases as the 2D bases used in 2DNMF. To evaluate the performances of 2DNMF, a series of experiments are performed on several face databases: FERET, UMIST, Yale and AR. The experimental results demonstrate advantages of 2DNMF over NMF on image reconstruction quality at similar compression ratio, computational efficiency and recognition accuracy.

The rest of the paper is organized as follows: Section 2 first introduces NMF method briefly. This is followed by the detailed description of 2DNMF algorithm in Section 3. In Section 4, experimental results are presented for the FERET, UMIST, Yale, and AR face databases to demonstrate the effectiveness of 2DNMF. Finally, we conclude in Section 5.

2 Non-negative Matrix Factorization

The key ingredient of NMF is the non-negativity constraints imposed on the two factors, and the non-negativity constraints are compatible with the intuitive notion of combining parts to form a whole. Because a part-based representation can naturally deal with partial occlusion and some illumination problems, it has received much attention recently. Assume that the image database is represented as an $n \times m$ matrix ***V***, each column of which contains *n* non-negative pixel values of one of the *m* face images. In order to compress data or reduce the dimensionality, NMF finds two non-negative matrix factors ***W*** and ***H*** such that

$$V_{i\mu} \approx (WH)_{i\mu} = \sum_{a=1}^{r} W_{ia} H_{a\mu} \tag{1}$$

Here the *r* columns of ***W*** are called NMF bases, and the columns of ***H*** are its combining coefficients. The dimensions of ***W*** and ***H*** are $n \times r$ and $r \times m$ respectively. The rank *r* of the factorization is usually chosen such that $(n+m)r<nm$, and hence the compression or dimensionality reduction is achieved. The compression ratio of NMF is easily gotten as $nm/(nr+mr)$.

To find an approximate factorization $\boldsymbol{V} \approx \boldsymbol{W}\boldsymbol{H}$, a cost function is needed to quantify the quality of the approximation. NMF uses the divergence measure as the objective function

$$D(A \| B) = \sum_{i,j} \left(A_{ij} \log \frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij} \right) \tag{2}$$

NMF factorization is a solution to the following optimization problem:

Problem 1 [12]. *Minimize* $D(V \| WH)$ *with respect to* W *and* H *, subject to the constraints* $W, H \geq 0$.

In order to obtain ***W*** and ***H***, a multiplicative update rule is given in [11] as follows:

$$W_{ia} = W_{ia} \sum_{\mu=1}^{m} \frac{V_{i\mu}}{(WH)_{i\mu}} H_{a\mu} \tag{3a}$$

$$W_{ia} = \frac{W_{ia}}{\sum_{j=1}^{n} W_{ja}} \tag{3b}$$

$$H_{a\mu} = H_{a\mu} \sum_{i=1}^{n} W_{ia} \frac{V_{i\mu}}{(WH)_{i\mu}} \tag{3c}$$

The pseudo-code for computing the bases ***W*** and coefficients ***H*** following the above iterative procedure is given in **Algorithm 1**.

```
Algorithm 1: NMF
Input: $n \times m$ matrix $V$, each column of which denotes the
aligned image vector, and rank $r$
```

```
Output: $n \times r$ matrix $W$ and $r \times m$ matrix $H$
1. Obtain initial values for $W$ and $H$, and set $k \leftarrow 1$
2. While not convergent
3.      update the bases $W$ using Eqs. (3a) and (3b)
4.      update the coefficients $H$ using Eq. (3c)
5.      $k \leftarrow k+1$
6. EndWhile
```

3 2-D Non-negative Matrix Factorization

3.1 2DNMF Algorithm

Let $p \times q$ matrices $\boldsymbol{A}_k$, k=1, 2, …, m, denote original training images. In traditional NMF, a 2D image is first transformed into a 1D vector, and then the image databases are represented with an $n \times m$ matrix $\boldsymbol{V}$, each column of which contains $n=pq$ non-negative pixel values of one of the m face images. In 2DNMF, however, we never transform the 2D images into its corresponding 1D vectors. Instead we will use a more straightforward way which views an image as a 2D matrix.

The procedure of 2DNMF method consists of two successive stages. At first we align the m training images into a $p \times qm$ matrix $\boldsymbol{X} = [\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_m]$, where each $\boldsymbol{A}_k$ denotes one of the m face images. Similar to NMF, 2DNMF first finds $p \times d$ non-negative matrix $\boldsymbol{L}$ and $d \times qm$ non-negative matrix $\boldsymbol{H}$ such that

$$X \approx LH \tag{4}$$

Here $\boldsymbol{L}$ and $\boldsymbol{H}$ are the bases and combining coefficients respectively. For convenience, we divide $\boldsymbol{H}$ into m $d \times q$ sub-matrices as $\boldsymbol{H} = [\boldsymbol{H}_1, \boldsymbol{H}_2, \ldots, \boldsymbol{H}_m]$, where $\boldsymbol{H}_k$ denotes the coefficients of the image $\boldsymbol{A}_k$. Since each column of $\boldsymbol{X}$ corresponds to a column of original images, we also call $\boldsymbol{L}$ as *column bases*. Thus the k-th image $\boldsymbol{A}_k$ can be written as a weighted sum of the column bases $\boldsymbol{L}$ as follows:

$$A_k \approx LH_k, \quad k = 1, 2, \ldots, m \tag{5}$$

The column bases $\boldsymbol{L}$ can be obtained by solving the following optimization problem:

Problem 2a. *Minimize* $D(X \,||\, LH)$ *with respect to* L *and* H, *subject to the constraints* $L, H \geq 0$.

Problem 2a can be solved by performing NMF algorithm on $\boldsymbol{X}$ with rank d. We also call the first stage for computing column bases $\boldsymbol{L}$ as *Column NMF*.

The second stage of 2DNMF involves of computing the row bases. Fom Eq. (5), we construct a new $q \times dm$ matrix $H' = [H_1^T, H_2^T, \ldots, H_m^T]$. Similarly, 2DNMF seeks a $q \times g$ non-negative matrix $\boldsymbol{R}$ and a $g \times dm$ non-negative matrix $\boldsymbol{C}$ such that

$$H' \approx RC \tag{6}$$

Here ***R*** and ***C*** are the bases and combining coefficients respectively. And we also divide ***C*** into m $g\times d$ sub-matrices as ***C*** = [***C***$_1$, ***C***$_2$, …, ***C***$_m$], where ***C***$_k$ denotes the coefficients of the matrix H_k^T. Because the columns of H' contains the row information of original images, we call ***R*** as *row bases*. Thus H_k^T is formulated as a weighted sum of the row bases ***R*** as follows:

$$H_k^T \approx RC_k, \quad k = 1, 2, ..., m \tag{7}$$

The row bases ***R*** can be obtained by solving the following optimization problem:

Problem 2b. *Minimize* $D(H' \| RC)$ *with respect to* R *and* C*, subject to the constraints* $R, C \geq 0$.

Similarly, problem 2b can be solved by performing NMF algorithm on H' with rank g. And we call the second stage for computing row bases ***R*** as *Row NMF*.

By now we have obtained the $p\times d$ dimensional column bases ***L*** and the $q\times g$ dimensional row bases ***R***. By substituting Eq. (7) into Eq. (6), we get

$$A_k \approx LC_k^T R^T, \quad k = 1, 2, ..., m \tag{8}$$

Let ***L*** = [***l***$_1$, ***l***$_2$, …, ***l***$_d$], ***R*** = [***r***$_1$, ***r***$_2$, …, ***r***$_g$], and define the 2D bases of 2DNMF as the outer product between the column base l_i and the row base r_j as follows:

$$E_{ij} = l_i \cdot r_j^T, (1 \leq i \leq d, 1 \leq j \leq g) \tag{9}$$

Let $D^k = C_k^T$, k=1, 2, …, m, then Eq. (8) turns into

$$A_k \approx LD^k R^T = \sum_{i=1}^{d} \sum_{j=1}^{g} D_{ij}^k E_{ij} \tag{10}$$

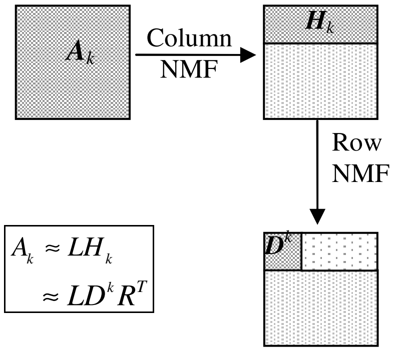

Fig. 1. An illustration for the 2DNMF algorithm

It is easy to verify that the 2D bases $\boldsymbol{E}_{ij}$ s have the following properties:

1) $\boldsymbol{E}_{ij}$ is a 2D matrix with the same size of original image, i.e. $p \times q$.
2) The *intrinsic dimensionality* of E_{ij} is 1.
3) Any training image $\boldsymbol{A}_k$ can be approximately represented with a weighted sum of 2D bases $\boldsymbol{E}_{ij}$ s.

Figure 1 gives a simple illustration for the procedure of 2DNMF. For a $p \times q$ dimensional image $\boldsymbol{A}_k$, it experiences the Column NMF operation (Eq. (5)) and the Row NMF operation (Eq. (7)) successively. If the ranks d and g are chosen such that $d < p$ and $g < q$, then figure 1 indicates that we can approximately represent the original image $\boldsymbol{A}_k$ with much smaller matrix D^k.

The pseudo-code for computing the column bases $\boldsymbol{L}$, row bases $\boldsymbol{R}$, and the 2D bases $\boldsymbol{E}_{ij}$ s is given in part of **Algorithm 2**.

```
Algorithm 2: 2DNMF
Input: $p \times q$ matrices $\{A_k\}_{k=1}^{m}$, and rank $d$, $g$
Output: $p \times d$ column bases $L$, $q \times g$ row bases $R$, $d \times g$ matrices $\{D^k\}_{k=1}^{m}$ and 2D bases $E_{ij}$
1. Align the m training images into a $p \times qm$ matrix $X = [A_1, A_2, \ldots, A_m]$
2. Perform Column NMF on $X$ with rank $d$: $X \approx LH$, obtaining column bases $L$ and coefficients matrix $H = [H_1, H_2, \ldots, H_m]$
3. Construct $q \times dm$ matrix $H' = [H_1^T, H_2^T, \ldots, H_m^T]$
4. Perform Row NMF on $H$ with rank $g$: $H' \approx RC$, obtaining row bases $R$ and coefficients matrix $C = [C_1, C_2, \ldots, C_m]$
5. For $k$ = 1 to $m$
6.         $D^k \leftarrow C_k^T$
7. EndFor
8. For $i$ = 1 to $d$, for $j$ =1 to $g$
9.         $E_{ij} \leftarrow l_i \cdot r_j^T$
10. EndFor
```

3.2 2DNMF-Based Image Compression

Suppose we have learned the 2D bases $E_{ij} = l_i \cdot r_j^T, (1 \le i \le d, 1 \le j \le g)$, from the training images $\boldsymbol{A}_k$, k=1, 2, …, m. According to Eq. (10) each training image $\boldsymbol{A}_k$ can be approximately reconstructed as

$$\hat{A}_k = \sum_{i=1}^{d} \sum_{j=1}^{g} D_{ij}^k E_{ij} \tag{11}$$

Here $\hat{A}_k$ denotes the reconstructed image corresponding to image $\boldsymbol{A}_k$.

In Eq. (11), the coefficients D^k are obtained by performing Column NMF (Eq. (5)) and the Row NMF (Eq. (7)) successively. For a test image $\boldsymbol{A}$ which is not contained in the training sets, the values for the coefficients D are unknown. However, we can approximately compute $D \approx L^+ AR$ (L^+ is the generalized inverse of L), and then use $\hat{A} \approx LDR^+$ (R^+ is the generalized inverse of R) to get the reconstructed image.

In this paper, we measure the quality of the reconstructed image using the peak signal-to-noise ratio (PSNR), which is defined as follows:

$$PSNR = 10\log 10\left(\frac{255^2}{\frac{1}{pq}\sum_{i,j}\left(A(i,j) - B(i,j)\right)^2} \right) \tag{12}$$

Here $\boldsymbol{A}$ denotes the original image, $\boldsymbol{B}$ denotes the reconstructed image, and p, q is the size of the image.

Equations (10) and (11) indicate that the $p \times d$ matrix $\boldsymbol{L}$, $q \times g$ matrix $\boldsymbol{R}$ and the $d \times g$ matrix D^k, k=1, 2, …, m, can be used to reconstruct the original m $p \times q$ matrices $\boldsymbol{A}_k$, k=1, 2, …, m. The memories required for storing $\boldsymbol{L}$, $\boldsymbol{R}$ and D^k, k=1, 2, …, m, are $pd+qg+mdg$. So if $pd+qg+mdg < mpq$, the compression or dimensionality reduction is achieved. And it is easy to compute the compression ratio of 2DNMF as $mpq\,/\,(pd+qg+mdg)$.

3.3 2DNMF-Based Face Recognition

It contains two steps when using 2DNMF for recognition. One is the feature extraction step and the other is classification step.

In feature extraction step, we first project each training image $\boldsymbol{A}_k$ into the bilinear space as a feature matrix $F_k = L^T A_k R$, for k=1, 2, …, m, which are then used as the comparing prototypes. A test or query face image $\boldsymbol{A}$ to be classified is represented by its projection onto the space as $F_A = L^T AR$.

In classification step, we calculate the distance based on Frobenius norm between the query and each prototype as follows:

$$d(F_A, F_k) = \left\| F_A - F_k \right\|_F \tag{13}$$

And the query is classified to the class to which the closest prototype belongs.

4 Experimental Results

In this section, we experimentally evaluate the performance of 2DNMF with NMF and local NMF (LNMF) [13] on several face databases. All our experiments are carried out on a P4 1.7 GHz PC machine with 512M memory. For all the three algorithms, the convergence condition is

$$\max_{i,j}\left(\left|W_{ij}^{new}\right|-\left|W_{ij}^{old}\right|\right)\Big/\max_{i,j}\left(W_{ij}^{old}\right)<0.01 \tag{14}$$

Where W_{ij}^{new} is the value of the bases at the current iteration step while W_{ij}^{old} denotes the value at the last iteration. The maximum iteration steps for NMF and LNMF are both set to 100.

The parameters d (number of column bases) and g (number of row bases) in 2DNMF are both set to 20 in all experiments, if without explicit explanations. The value of r (number of bases) in NMF and LNMF is chosen such that NMF, LNMF and 2DNMF have similar compression ratio in most cases, except for the Yale database where we assure r=16 at least. For example, if we set d=g=20 on the training set (m=200, p=q=60, n=pq=3600) of the FERET database, the compression ratio of 2DNMF will be mpq / (pd+qg+mdg)=8.74. And then we adjust the value of r to make the compression ratio of NMF and LNMF near to 8.74. Here we choose r=22, and the corresponding compression ratio of NMF and LNMF is nm/ (nr+mr)= 8.61.

4.1 Datasets

We use the following four face databases in our experiments: FERET, UMIST, Yale and AR [16] face database. Table 1 summarizes the statistics of the four datasets (the

Table 1. Statistics of four face databases

Datasets	Size	Dimension	# of classes
FERET[1]	400(200)	60×60	200
UMIST[2]	575(375)	112×92	20
Yale[3]	165 (75)	100×100	15
AR[4]	1400(700)	66×48	100

values in bracket indicate the size of training set). Note that for FERET and AR we only use a subset of the whole datasets. More detailed description of the four face databases can be obtained through browsing the face databases websites, whose linked address are given at the bottom of this page.

4.2 Learning the Bases

In this subsection, we compute the bases of NMF and 2DNMF from the training set. For comparison, we also compute the bases of PCA and its extension 2DPCA [21].

[1] http://www.itl.nist.gov/iad/humanid/feret/feret_master.html
[2] http://images.ee.umist.ac.uk/danny/database.html
[3] http://cvc.yale.edu/projects/yalefaces/yalefaces.html
[4] http://rvl1.ecn.purdue.edu/~aleix/aleix_face_DB.html

We do the experiment on the FERET face databases, and 200 **fa** face images are used as the training set.

Fig. 2 plots parts of the bases gotten from the four methods respectively. Fig. 2 (a) and (b) are plotted using the method in [21]. For NMF the first 16 columns of matrix

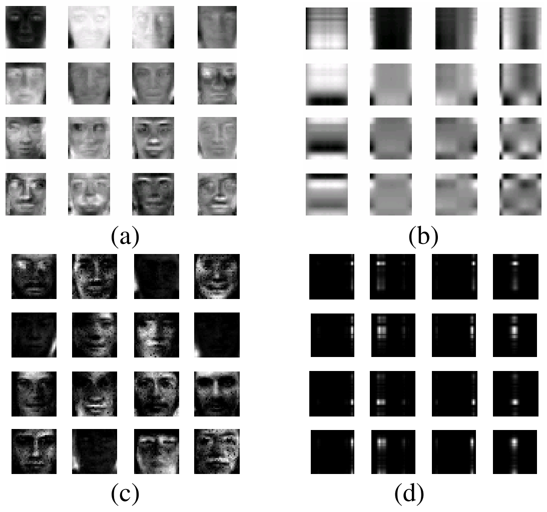

Fig. 2. Bases obtained from PCA (a), 2DPCA (b), NMF (c) and 2DNMF (d) respectively. (a) and (b) are plotted using the method in [21]. For NMF the first 16 columns of matrix $\boldsymbol{W}$ are retransformed to matrix for plotting, while for 2DNMF the 16 bases $\boldsymbol{E}_{ij}$ ($1 \le i \le 4$, $1 \le j \le 4$) are directly plotted as images.

$\boldsymbol{W}$ are retransformed to matrix for plotting, while for 2DNMF the 16 bases $\boldsymbol{E}_{ij}$ ($1 \le I \le 4$, $1 \le j \le 4$) are directly plotted as images.

From Fig. 2, we can see that both the bases of PCA and 2DPCA are global. Compared with PCA, 2DPCA possess of some strip or block like structures. The reason for that is 2DPCA is essentially a kind of line-based PCA [21]. However, the bases of 2DPCA cannot yet reflect any local or part-based features. On the other hand, Fig. 2(c) indicates that although NMF is a part-based algorithm, its bases still take on some holistic property similar to PCA. In contrast we notice from Fig. 2(d) that the bases of 2DNMF are much sparser than those of NMF. It is worth noting that although the base of 2DNMF is sparse, it has no parts-based (like eye, mouth, etc. in face image) features any more due to the essence of 2D methods. That is, 2DNMF is essentially a kind of line-based NMF, so what 2DNMF really learns are some parts of 'line'.

Because each base of 2DNMF can be generated using a p-dimensional column base and a q-dimensional row base, its storing cost (p+q) is much less than that of NMF base (which is pq). Thus we can use much more sparse-distributed 2D bases to represent original image, which will be further discussed in the next subsection.

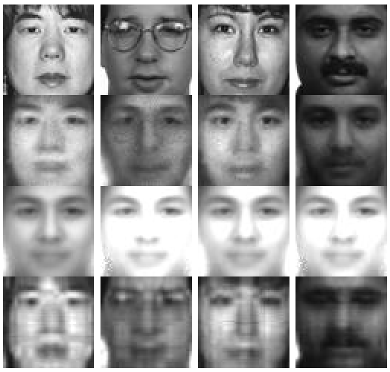

Fig. 3. Some reconstructed training images on FERET database. First row: original images. Second row: images gotten by NMF. Third row: images gotten by LNMF. Bottom row: images gotten by 2DNMF.

Table 2. Comparisons of the performances (PSNR / compression ratio / recognition accuracy) of NMF, LNMF and 2DNMF on four datasets

Data sets	NMF	LNMF	2DNMF
FERET	20.89 / 8.6/ 0.68	7.74 / 8.6/ 0.73	**23.50** / 8.7/ **0.79**
UMIST	21.34 / 24.1/ 0.71	8.08 / 24.1/ 0.67	**24.12 /** 25.1/ **0.76**
Yale	22.03 / 4.7/ 0.69	7.16 / 4.7/ 0.68	**23.98** / 22.1/ **0.81**
AR	19.50 / 8.0/ 0.49	5.00 / 8.0/ 0.62	**20.68 /** 7.9/ **0.64**

4.3 Image Compression and Reconstruction

In this subsection, we will compare the compression performances of NMF, LNMF and 2DNMF. We carry out experiments on the training set of the four datasets listed in table 1. That is, we only use the training images to learn the bases of NMF, LNMF and 2DNMF respectively. After that we reconstruct the training images or a new image not appearing in the training sets (test images) using the corresponding methods discussed in subsection 3.2.

Table 2 gives the PSNR values of NMF, LNMF and 2DNMF on the four datasets, and the corresponding compression ratios are also given in the brackets of the table. Table 2 shows that 2DNMF has the highest PSNR values on all the four datasets under nearly the same compression ratio except for the Yale database, where even when the compression ratio of 2DNMF is 5 times of that of NMF and LNMF, the former still achieves the best performance.

Fig. 3 shows parts of the reconstructed training images using NMF, LNMF and 2DNMF respectively on the FERET database. From Fig. 3, although the reconstructed images of NMF and LNMF smoother than that of 2DNMF, they don't resemble the

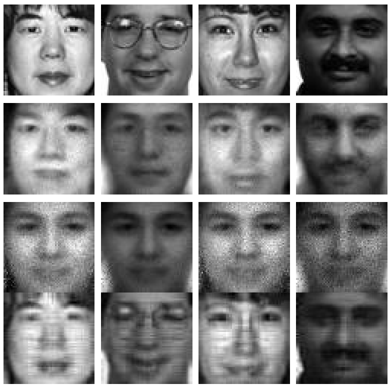

Fig. 4. Some reconstructed test images on FERET database using the same bases as those in figure 3. First row: original test images. Second row: images gotten by NMF. Third row: images gotten by LNMF. Bottom row: images gotten by 2DNMF.

original images any more. That phenomenon is especially severe for LNMF, because it imposes additional constrains on the bases which are useful for recognition but not for reconstruction, as clearly shown in Fig. 3. On the other hand, although there exist some stripping artifacts arising from the use of 2D bases of outer products of 1D row and column bases, 2DNMF reconstruct the original image more faithfully than the other two methods.

Fig. 4 shows some reconstructed test images of the three methods on FERET database using the same bases as those in Fig. 3. Similar as in Fig. 3, the reconstructed images of 2DNMF are most faithful compared with those of NMF and LNMF.

4.4 Face Recognition

In this subsection, we use the learned bases of NMF, LNMF and 2DNMF in section 4.2 for recognition. The detailed method for recognition has been discussed in section 3.3, and the recognition accuracy, which is defined as the percentage of correctly recognized images in test images, is used as the performance measure.

The first experiment is recognizing test images on four face databases without occlusion and noise and the result is given in part of table 2. Clearly, 2DNMF obtains the best accuracy in that case on all the datasets. The second experiment is to test the recognition accuracy of the three methods under partial occlusions or noises. Fig. 5 gives some examples of different levels of occluded images and noisy images (here only the ‘Salt & Pepper’ noise is considered). Fig. 6 gives the averaged (20 times) results of the three methods on FERET database when occlusions and noises are considered. From Fig. 6, 2DNMF outperforms NMF and LNMF in all cases. And because LNMF learns more localized parts than NMF, it achieves better result than NMF.

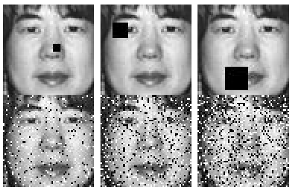

Fig. 5. Some examples of randomly occluded images and noisy images

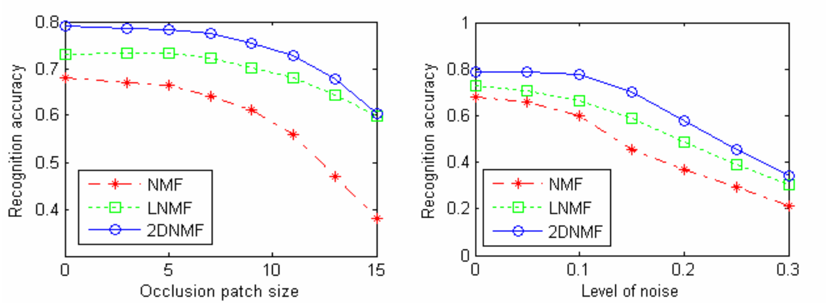

Fig. 6. Comparison of recognition accuracy of NMF, LNMF and 2DNMF under different size of occlusions (left) and different level of noises (right)

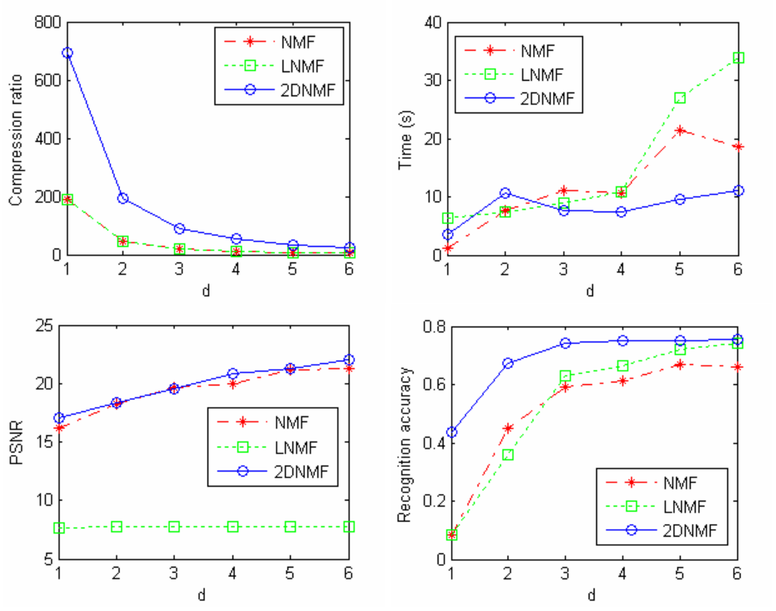

Fig. 7. Comparisons of performances of NMF, LNMF and 2DNMF under different number of bases. For NMF and LNMF, number of bases is *d***d*, while for 2DNMF the both numbers of row and column bases are 2*d*.

4.5 The Effect of Number of Bases

In this subsection, we evaluate the effect of the number of bases used in NMF, LNMF and 2DNMF respectively. For NMF and LNMF, we set the number of bases $p = d*d$, and varies d from 1 to 6. For 2DNMF, we set the both the numbers of row and column bases d=g=2*d, and also varies d from 1 to 6. The results on FERET database are shown in Fig. 7. From Fig. 7, we see that when d is relatively small (e.g. d<4), the compression ratio of 2DNMF is much greater than those of NMF and LNMF, and the recognition of 2DNMF is apparently higher than those of NMF and LNMF. As the value of d increases, the differences on compression ratio and recognition accuracy among the three methods reduce, while the difference on execution time begins to increase. In nearly all cases, 2DNMF achieves much better performance than the other two methods. Remember the number of bases of NMF and LNMF are $d*d$ and the number of bases of 2DNMF is $2d+2d=4d$. Especially, when d=4, all the three methods use the same number of bases. However, from Fig. 7, we know that 2DNMF still achieves better performance than NMF and LNMF.

5 Conclusions

In this paper, we have proposed a new method, 2-D non-negative matrix factorization (2DNMF), for face representation and recognition. This work is aimed to improve the performance of original non-negative matrix factorization (NMF) in the following aspect: reducing the computational costs, enhancing the image reconstruction quality and improving the recognition accuracy with or without occlusions and noises. We achieved our goal through using a novel image representation method, i.e. using 2D bases instead of traditional 1D bases. Experimental results on four face databases convince our claim that 2DNMF improves NMF on the above three aspects.

The number of bases (d and g) of 2DNMF is set by hand in advance, and if we change the values each time, we have to re-execute the whole algorithm, which will be very inconvenient in practice. We will investigate how to choose the values automatically like in PCA. Another future work is to investigate further improving the recognition accuracy of 2DNMF. But how to use the learned bases and feature vectors via NMF for further analysis such as recognition is still an open problem. We also encounter that problem in 2DNMF, and maybe we have to integrate 2DNMF with other method for better recognition accuracy.

Acknowledgements

The authors would like to thank the anonymous referees for their helpful comments and suggestions. This work was supported by the Jiangsu Postdoctoral Research Foundation, funds from Shanghai Key Laboratory of Intelligent Information Processing, National Science Foundation of China under Grant No. 60473035 and the Jiangsu Science Foundation under Grant No. BK2004001 and BK2005122. Portions of the research in this paper use the FERET database of facial images collected under the FERET program.

References

1. Buchsbaum, G., Bloch, O.: Color categories revealed by non-negative matrix factorization of Munsell color spectra. Vision Research 42 (2002) 559-563
2. Buciu, I., Pitas,I.: Application of non-negative and local non-negative matrix factorization to facial expression recognition. In: ICPR, Cambridge, 2004
3. Chen, X., Gu, L., Li, S.Z., Zhang, H.J.: Learning representative local features for face detection. In: CVPR, Hawaii, 2001
4. Comon, P.: Independent component analysis- a new comcept? Signal Processing 36 (1994) 287-314
5. Donoho, D., Stodden, V.: When does non-negative matrix factorization give a correct decomposition into parts? In: NIPS, 2004
6. Ge, X., Iwata, S.: Learning the parts of objects by auto-association. Neural Networks 15 (2002) 285-295
7. Guillamet, D., Bressan, M., Vitria, J.: A weighted non-negative matrix factorization for local representation. In: CVPR, Hawaii, 2001
8. Guillamet, D., Vitria, J., Schiele, B.: Introducing a weighted non-negative matrix factorization for image classification. Pattern Recognition Letters 24 (2003) 2447-2454
9. Hoyer, P.O.: Non-negative matrix factorization with sparseness constraints. Journal of machine Learning Research 5 (2004) 1457-1469
10. Jolliffe, I.T.: Principal Component Analysis. Springer-Verlag, New York, 1986
11. Lee D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature 401(1999) 788-791
12. Lee D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: NIPS 13 (2001) 556–562
13. Li, S.Z., Hou, X.W., Zhang, H.J., Cheng, Q.S.: Learning spatially localized, parts-based representation. In: CVPR, Hawaii, 2001.
14. Liu, W., Zheng, N.: Learning sparse features for classification by mixture models Pattern Recognition Letters 25 (2004) 155-161
15. Liu, W., Zheng, N.: Non-negative matrix factorization based methods for object recognition. Pattern Recognition Letters 25 (2004) 893-897
16. Martinez, A., Benavente, R.: The ar face database (Technical Report CVC Tech. Report No. 24).
17. Plumbley, M.D., Oja, E.: A 'nonnegative PCA' algorithm for independent component analysis. IEEE Trans. On Neural Networks 15 (1) (2004) 66-76
18. Scholkopf, B., Smola, A.J., Muller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10 (1998) 1299-1319
19. Turk, M., Pentland, A.: Eigenfaces for recognition. J. Cognitive Neuroscience 3 (1) (1991) 71-86
20. Wild, S., Curry, J., Dougherty, A.: Improving non-negative matrix factorizations through structured initialization. Pattern Recognition 37 (11) (2004) 2217-2232
21. Zhang, D.Q., Chen, S.C., Liu, J.: Representing image matrices: Eigenimages vs. Eigenvectors. In: ISNN, Chongqing, China, 2005.