Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 390328, 10 pages
http://dx.doi.org/10.1155/2014/390328

Research Article

Facial Expression Recognition Based on Discriminant Neighborhood Preserving Nonnegative Tensor Factorization and ELM

Gaoyun An,[1] Shuai Liu,[1,2] Yi Jin,[1] Qiuqi Ruan,[1] and Shan Lu[3]

[1] *Institute of Information Science, Beijing Jiaotong University, Beijing 100044, China*
[2] *Baidu Online Network Technology (Beijing) Co. Ltd., Beijing 100085, China*
[3] *School of Information, Renmin University of China, Beijing 100872, China*

Correspondence should be addressed to Gaoyun An; gyan@bjtu.edu.cn

Received 29 May 2014; Revised 28 July 2014; Accepted 12 August 2014; Published 19 October 2014

Academic Editor: Jiuwen Cao

A novel facial expression recognition algorithm based on discriminant neighborhood preserving nonnegative tensor factorization (DNPNTF) and extreme learning machine (ELM) is proposed. A discriminant constraint is adopted according to the manifold learning and graph embedding theory. The constraint is useful to exploit the spatial neighborhood structure and the prior defined discriminant properties. The obtained parts-based representations by our algorithm vary smoothly along the geodesics of the data manifold and have good discriminant property. To guarantee the convergence, the project gradient method is used for optimization. Then features extracted by DNPNTF are fed into ELM which is a training method for the single hidden layer feed-forward networks (SLFNs). Experimental results on JAFFE database and Cohn-Kanade database demonstrate that our proposed algorithm could extract effective features and have good performance in facial expression recognition.

1. Introduction

Facial expression recognition plays an important role in human-computer interaction, and 55% information is transferred by facial expression in face-to-face human communication [1]. Although many methods were proposed, recognizing facial expression is still challenging due to the complex, variable, and subtle facial expressions.

One of the effective methods for facial expression recognition is the subspace-based algorithm [2–5]. It aims to project the samples into a lower dimensional space which preserves the needed information and discards the redundant information. There are many widely used subspace-based algorithms, for example, principle component analysis (PCA) [2], linear discriminant analysis (LDA) [3], neighborhood preserving embedding (NPE) [4], locality-preserving projection (LPP) [6], and singular value decomposition (SVD) [5], and so forth.

Recently, the nonnegative matrix factorization (NMF) was introduced into facial expression recognition [7]. NMF decomposes the face samples into two nonnegative parts: the basis images and the corresponding weights. As many data in bases and weights were degenerated too close to zero, NMF derived the parts-based sparse representations. For facial expression recognition, the localized subtle features, such as the corners of mouth, upward or downward eyebrows, and change of eyes, are critical for the recognition performance. Since NMF yields the parts-based representations, it outperforms the subspace-based models. To further improve NMF, several variants have been presented by introducing different constraints to the objective function. Li et al. put forward a local NMF (LNMF) by adding a local constraint to the basis images [8], to learn a localized, parts-based representation. Hoyer gave a sparse constraint NMF (SNMF) by incorporating sparseness constraint into both the bases and the weights [9]. Cai et al. developed a graph constraint NMF (GNMF) by adding a graph preserving constraint to the weights [10]. Zafeiriou et al. used the discriminant NMF (DNMF) for frontal face verification [11]. Wang et al. extended NMF to PNMF with a PCA constraint and FNMF with a Fisher constraint [12].

For facial expression recognition, NMF and its variants vectorize the samples before factorization, which may lose the local geometric structures. However, the spatial neighborhood relationships within pixels are critical for image representation, understanding, and recognition [13]. Another drawback of NMF is that it could not generate a unique decomposition result. Welling and Weber developed a positive tensor factorization (PTF) algorithm, which handled images as 2D matrices directly [14]. Shashua and Hazan proposed the nonnegative tensor factorization (NTF) which implemented the factorization in the rank-one tensor space [15]. The factorization in the tensor space could preserve the local structures and guarantee the uniqueness of the decomposition.

On the other hand, the choice of classifier plays an important role for recognition. For facial expression recognition, nearest neighbor (NN) and support vector machine (SVM) are the commonly used methods [16]. The sparse representation classifier (SRC) was adopted in [17]. Recently, the extreme learning machine (ELM) was proposed for classification which is a training method for the single hidden layer feed-forward networks (SLFNs) [18]. The conventional methods need long time to converge or may lose the generalization property due to overfitting. However, ELM converges fast and provides good generalization performance. For ELM, the input weights and biases are randomly assigned, and the output weights can be simply calculated by the generalized inverse of the hidden layer output matrix. Therefore it converges extremely fast and obtains an excellent generalization capability. Many variants of ELM were proposed for different applications [19–26], including the Kernel-based ELM [21] and the incremental ELM (I-ELM) [23], which lead to the state-of-the-art results in different applications.

In this paper, we propose a novel facial expression recognition algorithm based on discriminant neighborhood preserving nonnegative tensor factorization (DNPNTF) and ELM. It works well in the rank-one tensor space. The simple ELM is adopted to testify its effectiveness for facial expression recognition [18]. Our algorithm is composed of two stages: feature extraction and classification. Firstly, to extract the discriminant features, a neighborhood preserving constraint form of NTF is used. The constraint is derived according to the manifold learning and graph embedding theory [27–29]. Since the columns of the weighting matrix have a one-to-one correspondence with the columns of the original sample, the discriminant constraint is added to the weighting matrix. With the neighborhood preserving constraint, the obtained parts-based representations vary smoothly along the geodesics of the data manifold and are more discriminant. Secondly, the discriminant features extracted by DNPNTF are fed into ELM classifier to conduct the recognition task.

The rest of this paper is organized as follows. The mathematical notations are given in Section 2. In Section 3, we give the detailed analysis about DNPNTF and its optimization procedure. ELM is introduced in Section 4, and the experiments are given in Section 5. Finally, the conclusions are drawn in Section 6.

2. Basic Algebra and Notations

In this paper, a tensor is represented as $\mathbf{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$, whose order is n. The element of A is denoted by $\mathbf{A}_{i_1 i_2 \cdots i_n} > 0$, where $1 \le i_k \le m_k$, $1 \le k \le n$.

Definition 1 (inner product and tensor product [30]). The inner product of two tensors $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$ is defined as

$$\langle \mathbf{A}, \mathbf{B} \rangle = \sum_{i_1=1,\ldots,i_n=1}^{m_1,\ldots,m_n} \mathbf{A}_{i_1 \cdots i_n} \mathbf{B}_{i_1 \cdots i_n}. \tag{1}$$

The tensor product of two tensors $\mathbf{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$ and $\mathbf{B} \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_q}$ is

$$\mathbf{C} = \mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n \times p_1 \times \cdots \times p_q}. \tag{2}$$

Definition 2 (rank-one tensor [30]). A nth-order tensor $\mathbf{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$ could be represented as a tensor product of n tensors $\mathbf{u}^1 = [u_1^1, u_2^1, \ldots, u_{m_1}^1] \in \mathbb{R}^{m_1}, \ldots, \mathbf{u}^n = [u_1^n, u_2^n, \ldots, u_{m_n}^n] \in \mathbb{R}^{m_n}$ as

$$\mathbf{A} = \mathbf{u}^1 \otimes \mathbf{u}^2 \otimes \cdots \otimes \mathbf{u}^n. \tag{3}$$

Here $\mathbf{A}$ is called rank-one tensor, and $\mathbf{A}_{i_1 i_2 \cdots i_n} = u_{i_1}^1 u_{i_2}^2 \cdots u_{i_n}^n$ $(1 \le i_1 \le m_1, \ldots, 1 \le i_n \le m_n)$.

Definition 3 (mode product [30]). The mode k product of $\mathbf{A} \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_n}$ and $\mathbf{U} \in \mathbb{R}^{m_k \times m_k'}$ is

$$\mathbf{C} = \mathbf{A} \times_k U \in \mathbb{R}^{m_1 \times m_{k-1} \times m_k' \times m_{k+1} \times \cdots \times m_n}, \tag{4}$$

where $\mathbf{C}_{i_1 \cdots i_{k-1} q i_{k+1} \cdots i_n} = \sum_{p=1}^{m_k} \mathbf{A}_{i_1 \cdots i_{k-1} p i_{k+1} \cdots i_n} \cdot U_{p,q}$, $1 \le i_j \le m_j$, $1 \le j \le n$.

3. The DNPNTF Algorithm

In this section, we give a detailed description about the proposed DNPNTF algorithm. Instead of converting into vectors, it processes the samples in rank-one tensor space. The objective function of NTF is adopted, which could learn the parts-based representation and have the sparse property. To discover the spatial local geometric structure and the discriminant class-based information, a constraint is added in the objective function according to the manifold learning and graph embedding analysis. To guarantee the convergence, the project gradient method is used.

3.1. The Analysis of DNPNTF. Given a $m \times N$ image database $\mathbf{X}$, it contains N sample images $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{m \times N}$. The dimension of each sample is m $(m = m_1 \times n_2)$. In NTF, the database is organized as a 3rd-order tensor $\mathbf{A} \in \mathbb{R}^{m_1 \times m_2 \times N}$, which is overlapped by $\{\mathbf{x}_i \in \mathbb{R}^{m_1 \times m_2} |_{i=1}^{N}\}$ sequentially. The objective function of NTF is

$$\begin{aligned} &\min_{\mathbf{u}_r, \mathbf{v}_r, \mathbf{w}_r} \left\| \mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r \right\|^2 \\ &\text{s.t.} \quad \mathbf{u}_r, \mathbf{v}_r, \mathbf{z}_r \ge 0, \end{aligned} \tag{5}$$

where $\mathbf{u}_r \in \mathbb{R}^{m_1}$, $\mathbf{v}_r \in \mathbb{R}^{m_2}$, and $\mathbf{z}_r \in \mathbb{R}^N$ $(1 \le r \le R)$ describe the first, second, and third modules of $\mathbf{A}$, respectively. Each sample $\mathbf{x}_i$ is approximated by

$$\mathbf{x}_i \approx \sum_{r=1}^{R} \mathbf{z}_r^i \mathbf{u}_r \mathbf{v}_r^T, \quad (i = 1, \ldots, N). \tag{6}$$

By minimizing (5), the bases $\{\mathbf{u}_r \mathbf{v}_r^T|_{r=1}^R\}$ and the corresponding weights $\{\mathbf{z}_r|_{r=1}^R\}$ are conducted. The inner product of $\{\mathbf{u}_r \mathbf{v}_r^T|_{r=1}^R\}$ and the sample image is calculated to derive the low-dimensional parts-based representation.

To incorporate more properties into NTF, different constraints could be added into the objective function. The constraint form of objective function is

$$\begin{aligned} \min_{\mathbf{u}_r, \mathbf{v}_r, \mathbf{z}_r} \quad & \left\| \mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r \right\|^2 + \alpha F(U, V) + \beta J(Z) \\ \text{s.t.} \quad & \mathbf{u}_r, \mathbf{v}_r, \mathbf{z}_r \ge 0, \quad \forall r, \end{aligned} \tag{7}$$

where $U = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_R] \in \mathbb{R}^{m_1 \times R}$, $V = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_R] \in \mathbb{R}^{m_2 \times R}$, $Z = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_R] \in \mathbb{R}^{N \times R}$. F is the constraint function about $\{U, V\}$ and J is the constraint about Z. α and β are the corresponding positive coefficients. To encode the spatial structure and discriminant class-based information into sparse representations, we propose a constraint function J according to the manifold learning and graph embedding analysis. In NTF, the columns of the weighting matrix have a one-to-one correspondence with the columns of the original image matrix. Therefore, we add the discriminant constraint to $\{\mathbf{z}_r|_{r=1}^R\}$, and J for $\mathbf{z}_r$ is defined as

$$J(Z) = \min \sum \sum_{i \ne j} \sum_{l} \eta_l \left\| \mathbf{z}_i - \mathbf{z}_j \right\|^2 \mathbf{S}_{ij}^l, \tag{8}$$

where $\mathbf{S}_{ij}^l$ denote the graphs with different properties and η_l are the corresponding coefficients. By deriving different $\mathbf{S}_{ij}$, the graph embedding model could have different properties, such as the neighborhood preserving property and the discriminant property.

Now, we discuss the selection of $\mathbf{S}_{ij}^l$. The most commonly used graph is the Laplacian graph, and $\mathbf{S}_{ij}^l$ is calculated in form of the heat kernel function as

$$\mathbf{S}_{ij} = \mathbf{e}^{-\|\mathbf{z}_i - \mathbf{z}_j\|^2 / t}, \quad t \in R. \tag{9}$$

Here, $\mathbf{S}_{ij}$ measures the similarity between a pair of vertices and has neighborhood preserving property. To further incorporate the class-based discriminant information, we derive a universal penalty graph [27], where the similarity matrix $\mathbf{S}_{ij}^p$ is defined as

$$\mathbf{S}_{ij}^p = \begin{cases} 1 \text{ or } \dfrac{1}{k}, & \text{if } (i, j) \in N_k(l_i) \text{ or } N_k(l_j) \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

where $N_k(l_i)$ represents the nearest k pairs of samples between class l_i and the other classes. The purpose of the penalty graph was to separate marginal samples between different classes.

Now, the objective function of constrained NTF becomes

$$\begin{aligned} \min_{\mathbf{u}_r, \mathbf{v}_r, \mathbf{z}_r} \quad & \left\| \mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r \right\|^2 + \lambda \left\| \mathbf{z}_i - \mathbf{z}_j \right\|^2 \mathbf{S}_{ij} - \sigma \left\| \mathbf{z}_i - \mathbf{z}_j \right\|^2 \mathbf{S}_{ij}^p \\ \text{s.t.} \quad & \mathbf{u}_r, \mathbf{v}_r, \mathbf{z}_r \ge 0, \quad \forall r. \end{aligned} \tag{11}$$

By solving the generalized eigenvalue decomposition problem, the graph embedding criterion in (11) can be calculated as

$$\begin{aligned} & \frac{1}{2} \lambda \left\| \mathbf{z}_i - \mathbf{z}_j \right\|^2 \mathbf{S}_{ij} - \frac{1}{2} \sigma \left\| \mathbf{z}_i - \mathbf{z}_j \right\|^2 \mathbf{S}_{ij}^p \\ & \quad = \frac{1}{2} \lambda \sum_{i,j} tr \left[\left(\mathbf{z}_i - \mathbf{z}_j \right) \left(\mathbf{z}_i - \mathbf{z}_j \right)^T \right] \mathbf{S}_{ij} \\ & \qquad - \frac{1}{2} \sigma \sum_{i,j} tr \left[\left(\mathbf{z}_i - \mathbf{z}_j \right) \left(\mathbf{z}_i - \mathbf{z}_j \right)^T \right] \mathbf{S}_{ij}^p \\ & \quad = \frac{1}{2} tr \left[\lambda \sum_{i,j} \left[\left(\mathbf{z}_i - \mathbf{z}_j \right) \left(\mathbf{z}_i - \mathbf{z}_j \right)^T \right] \mathbf{S}_{ij} \right. \\ & \qquad \left. - \sigma \sum_{i,j} \left[\left(\mathbf{z}_i - \mathbf{z}_j \right) \left(\mathbf{z}_i - \mathbf{z}_j \right)^T \right] \mathbf{S}_{ij}^p \right] \\ & \quad = tr \left[\lambda \sum_{i,j} \left(\mathbf{z}_i \mathbf{S}_{ij} (\mathbf{z}_i)^T - \mathbf{z}_i \mathbf{S}_{ij} \left(\mathbf{z}_j \right)^T \right) \right. \\ & \qquad \left. - \sigma \sum_{i,j} \left(\mathbf{z}_i \mathbf{S}_{ij}^p (\mathbf{z}_i)^T - \mathbf{z}_i \mathbf{S}_{ij}^p \left(\mathbf{z}_j \right)^T \right) \right] \\ & \quad = tr \left[\lambda \sum_{i,j} \left(\mathbf{z}_i E_{ij} (\mathbf{z}_i)^T - \mathbf{z}_i E_{ij} \left(\mathbf{z}_j \right)^T \right) \right. \\ & \qquad \left. - \sigma \sum_{i,j} \left(\mathbf{z}_i E_{ij}^p (\mathbf{z}_i)^T - \mathbf{z}_i E_{ij}^p \left(\mathbf{z}_j \right)^T \right) \right] \\ & \quad = tr \left[\lambda \sum_{i,j} \mathbf{z}_i L_{ij} \left(\mathbf{z}_j \right)^T - \sigma \sum_{i,j} \mathbf{z}_i L_{ij}^p \left(\mathbf{z}_j \right)^T \right] \\ & \quad = \lambda tr \left(Z^T L Z \right) - \sigma tr \left(Z^T L^p Z \right). \end{aligned} \tag{12}$$

And the final objective function of DNPNTF is

$$\begin{aligned} \min_{\mathbf{u}_r, \mathbf{v}_r, \mathbf{z}_r} \quad & \left\| \mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r \right\|^2 + \lambda tr \left(Z^T L Z \right) - \sigma tr \left(Z^T L^p Z \right) \\ \text{s.t.} \quad & \mathbf{u}_r, \mathbf{v}_r, \mathbf{z}_r \ge 0, \quad \forall r, \end{aligned} \tag{13}$$

where $\lambda > 0$ and $\sigma > 0$ to make sure (13) should be nonnegative.

3.2. Projected Gradient Method of DNPNTF. The most popular approach to minimize NMF or NTF is the multiplicative update method. However, it cannot ensure the convergence of the constraint forms of NMF or NTF. In this paper, the projected gradient method is used to solve DNPNTF.

The objective function of DNPNTF can be stated as

$$D_{\text{obj}}(\mathbf{A}\|\mathbf{UVZ}) = D(\mathbf{A}\|\mathbf{UVZ}) + \varsigma D(\mathbf{Z}), \tag{14}$$

where ς is a positive constant. The goal of (14) is to find $\{\mathbf{U}, \mathbf{V}\}$ and $\mathbf{Z}$ by solving the following problem:

$$\begin{aligned} &\min_{\mathbf{U},\mathbf{V},\mathbf{Z}} \quad D_{\text{obj}}(\mathbf{A}\|\mathbf{UVZ}) \\ &\text{s.t.} \quad \mathbf{u}_{i,k} \geq 0, \quad \mathbf{v}_{k,j} \geq 0, \quad \mathbf{z}_{i,j} \geq 0 \quad \forall k, i, j. \end{aligned} \tag{15}$$

To find the optimal solution, (15) is divided into three subproblems: first, we fix $\mathbf{V}$ and $\mathbf{Z}$ and update $\mathbf{U}$ to arrive at the conditional optimal value of the subminimization problem; second, we fix $\mathbf{U}$ and $\mathbf{Z}$ and update $\mathbf{V}$; last, we fix $\mathbf{U}$ and $\mathbf{V}$ and update $\mathbf{Z}$. Three functions are defined as $f_{\mathbf{V},\mathbf{Z}}(\mathbf{U}) = D_{\text{obj}}(\mathbf{A}\|\mathbf{UVZ})$, $f_{\mathbf{U},\mathbf{Z}}(\mathbf{V}) = D_{\text{obj}}(\mathbf{A}\|\mathbf{UVZ})$, and $f_{\mathbf{U},\mathbf{V}}(\mathbf{Z}) = D_{\text{obj}}(\mathbf{A}\|\mathbf{UVZ})$. The update rules are defined as

$$\begin{aligned} \mathbf{U}^{(t+1)} &= \left[\mathbf{U}^{(t)} - \alpha_t \nabla f_{\mathbf{V},\mathbf{Z}}\left(\mathbf{U}^{(t)}\right)\right], \\ \mathbf{V}^{(t+1)} &= \left[\mathbf{V}^{(t)} - \alpha_t \nabla f_{\mathbf{U},\mathbf{Z}}\left(\mathbf{V}^{(t)}\right)\right], \\ \mathbf{Z}^{(t+1)} &= \left[\mathbf{Z}^{(t)} - \alpha_t \nabla f_{\mathbf{U},\mathbf{V}}\left(\mathbf{Z}^{(t)}\right)\right]. \end{aligned} \tag{16}$$

Now the task is calculating $\nabla f_{\mathbf{V},\mathbf{Z}}(\mathbf{U}^{(t)})$, $\nabla f_{\mathbf{U},\mathbf{Z}}(\mathbf{V}^{(t)})$, and $\nabla f_{\mathbf{U},\mathbf{V}}(\mathbf{Z}^{(t)})$.

3.2.1. The Calculation of $\mathbf{U}$ and $\mathbf{V}$. Firstly, we discuss the calculation of $\nabla f_{\mathbf{V},\mathbf{Z}}(\mathbf{U}^{(t)})$ and $\nabla f_{\mathbf{U},\mathbf{Z}}(\mathbf{V}^{(t)})$. The objective function could be written as

$$\begin{aligned} f &= \frac{1}{2}\left\|\mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r\right\|^2 \\ &\quad + \frac{1}{2}\lambda tr\left(\mathbf{Z}^T L \mathbf{Z}\right) - \frac{1}{2}\sigma tr\left(\mathbf{Z}^T L^p \mathbf{Z}\right) \\ &= \frac{1}{2}\left\langle \mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r, \right. \\ &\qquad \left. \mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r \right\rangle \\ &\quad + \frac{1}{2}\lambda \sum_{r=1}^{R} \mathbf{z}_r^T L \mathbf{z}_r - \frac{1}{2}\sigma \sum_{r=1}^{R} \mathbf{z}_r^T L^p \mathbf{z}_r. \end{aligned} \tag{17}$$

The differential of f is

$$\begin{aligned} d\left(f_{\mathbf{u}_s}\right) &= \left\langle \mathbf{A} - \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r, \right. \\ &\qquad \left. -d(\mathbf{u}_s) \otimes \mathbf{v}_s \otimes \mathbf{z}_s \right\rangle + 0 - 0 \\ &= \left\langle \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r, \right. \\ &\qquad \left. d(\mathbf{u}_s) \otimes \mathbf{v}_s \otimes \mathbf{z}_s \right\rangle - \left\langle \mathbf{A}, d(\mathbf{u}_s) \otimes \mathbf{v}_s \otimes \mathbf{z}_s \right\rangle. \end{aligned} \tag{18}$$

And the partial differential for $\mathbf{u}_s^p$ $(1 \leq p \leq m_1)$ is

$$\begin{aligned} \frac{\partial f}{\partial \mathbf{u}_s^p} &= \left\langle \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{v}_r \otimes \mathbf{z}_r, \mathbf{e}^p \otimes \mathbf{v}_s \otimes \mathbf{z}_s \right\rangle \\ &\quad - \left\langle \mathbf{A}, \mathbf{e}^p \otimes \mathbf{v}_s \otimes \mathbf{z}_s \right\rangle, \end{aligned} \tag{19}$$

where the pth element in $\mathbf{e}^p \in \mathbb{R}^{m_1}$ is 1 and others are 0. That is, $(\mathbf{e}^p)_p = 1$ and $(\mathbf{e}^p)_{k \neq p} = 0$. According to Definition 1, for any order tensors $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{R}^{a_1 \times a_2 \times \cdots \times a_n}$, $\mathbf{B}_1, \mathbf{B}_2 \in \mathbb{R}^{b_1 \times b_2 \times \cdots \times b_n}$, there is $\langle \mathbf{A}_1 \otimes \mathbf{B}_1, \mathbf{A}_2 \otimes \mathbf{B}_2 \rangle = \langle \mathbf{A}_1, \mathbf{A}_2 \rangle \langle \mathbf{B}_1, \mathbf{B}_2 \rangle$. Then (19) could be written as

$$\begin{aligned} \frac{\partial f}{\partial \mathbf{u}_s^p} &= \sum_{r=1}^{R} \left(\langle \mathbf{u}_r, \mathbf{e}^p \rangle \langle \mathbf{v}_r \otimes \mathbf{z}_r, \mathbf{v}_s \otimes \mathbf{z}_s \rangle \right) \\ &\quad - \left(\sum_{k=1,q=1,i=1}^{m_1,m_2,N} A_{kqi} (\mathbf{e}^p)_k \mathbf{v}_s^q \mathbf{z}_s^i \right) \\ &= \sum_{r=1}^{R} \left(\langle \mathbf{u}_r, \mathbf{e}^p \rangle \langle \mathbf{v}_r \otimes \mathbf{z}_r \rangle \langle \mathbf{v}_s \otimes \mathbf{z}_s \rangle \right) \\ &\quad - \left(\sum_{q=1,i=1}^{m_2,N} A_{pqi} (\mathbf{e}^p)_p \mathbf{v}_s^q \mathbf{z}_s^i + \sum_{k \neq p} A_{kqi} (\mathbf{e}^p)_k \mathbf{v}_s^q \mathbf{z}_s^i \right) \\ &= \sum_{r=1}^{R} \left(\langle \mathbf{u}_r, \mathbf{e}^p \rangle \langle \mathbf{v}_r, \mathbf{v}_s \rangle \langle \mathbf{z}_r, \mathbf{z}_s \rangle \right) \\ &\quad - \left(\sum_{q=1,i=1}^{m_2,N} A_{pqi} \mathbf{v}_s^q \mathbf{z}_s^i + 0 \right) \\ &= \sum_{r=1}^{R} \left(\left(\mathbf{u}_r^p (\mathbf{e}^p)_p + \sum_{k \neq p} \mathbf{u}_r^k (\mathbf{e}^p)_k \right) \langle \mathbf{v}_r, \mathbf{v}_s \rangle \langle \mathbf{z}_r, \mathbf{z}_s \rangle \right) \\ &\quad - \sum_{q=1,i=1}^{m_2,N} A_{pqi} \mathbf{v}_s^q \mathbf{z}_s^i \\ &= \sum_{r=1}^{R} \left(\mathbf{u}_r^p \langle \mathbf{v}_r, \mathbf{v}_s \rangle \langle \mathbf{z}_r, \mathbf{z}_s \rangle \right) - \sum_{q=1,i=1}^{m_2,N} A_{pqi} \mathbf{v}_s^q \mathbf{z}_s^i. \end{aligned} \tag{20}$$

According to (16), the update rule for $\mathbf{u}_s^p$ is

$$\begin{aligned}\mathbf{u}_s^p &= \mathbf{u}_s^p - \mu\left(\mathbf{u}_s^p\right)\frac{\partial f}{\partial \mathbf{u}_s^p}\\ &= \mathbf{u}_s^p - \mu\left(\mathbf{u}_s^p\right)\left(\sum_{r=1}^{R}\left(\mathbf{u}_r^p\left\langle \mathbf{v}_r,\mathbf{v}_s\right\rangle\left\langle \mathbf{z}_r,\mathbf{z}_s\right\rangle\right)\right.\\ &\qquad \left.-\sum_{q=1,i=1}^{m_2,N} A_{pqi}\mathbf{v}_s^q\mathbf{z}_s^i\right).\end{aligned} \tag{21}$$

To confirm the nonnegative of $\mathbf{u}_s^p$, $\mu(\mathbf{u}_s^p)$ is set to be

$$\mu\left(\mathbf{u}_s^p\right) = \frac{\mathbf{u}_s^p}{\sum_{r=1}^{R}\left(\mathbf{u}_r^p\left\langle \mathbf{v}_r,\mathbf{v}_s\right\rangle\left\langle \mathbf{z}_r,\mathbf{z}_s\right\rangle\right)}. \tag{22}$$

Then the update function is

$$\begin{aligned}\mathbf{u}_s^p &= \frac{\mathbf{u}_s^p\left(\sum_{q=1,i=1}^{m_2,N} A_{pqi}\mathbf{v}_s^q\mathbf{z}_s^i\right)}{\sum_{r=1}^{R}\left(\mathbf{u}_r^p\left\langle \mathbf{v}_r,\mathbf{v}_s\right\rangle\left\langle \mathbf{z}_r,\mathbf{z}_s\right\rangle\right)}\\ &= \frac{\mathbf{u}_s^p\left(\mathbf{v}_s^T A_{p;;}\mathbf{z}_s\right)}{\sum_{r=1}^{R}\left(\mathbf{u}_r^p\left(\mathbf{v}_r^T\mathbf{v}_s\right)\left(\mathbf{z}_r^T\mathbf{z}_s\right)\right)}\\ &= \frac{\mathbf{u}_s^p\left(\mathbf{v}_s^T A_{p;;}\mathbf{z}_s\right)}{\left(U_{p;}\left(\left(V^T\mathbf{v}_s\right)\odot\left(Z^T\mathbf{z}_s\right)\right)\right)},\end{aligned} \tag{23}$$

where $U_{p;} \in \mathbb{R}^{1\times R}$ represents the pth row of $U = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_R]$, $\odot$ is the matrix Hadamard product, $A_{p;;} \in \mathbb{R}^{m_2\times N}$ represents the matrix which fixes the first module of A, and traversals are the other two modules. It is defined as

$$\left(A_{p;;}\right)_{qi} = \mathrm{A}_{pqi}, \quad 1 \le \forall q \le m_2, 1 \le \forall i \le N. \tag{24}$$

Similarly, the update rule of the qth element of $\mathbf{v}_s$ ($\mathbf{v}_s^q$, $1 \le s \le R, 1 \le q \le m_2$) is

$$\begin{aligned}\mathbf{v}_s^q &= \frac{\mathbf{v}_s^q\sum_{p=1,i=1}^{m_1,N}\left(A_{pqi}\mathbf{u}_s^p\mathbf{z}_s^i\right)}{\sum_{r=1}^{R}\left(\mathbf{v}_r^q\left\langle \mathbf{u}_r,\mathbf{u}_s\right\rangle\left\langle \mathbf{z}_r,\mathbf{z}_s\right\rangle\right)}\\ &= \frac{\mathbf{v}_s^q\left(\mathbf{u}_s^T A_{;q;}\mathbf{z}_s\right)}{\left(V_{q;}\left(\left(U^T\mathbf{u}_s\right)\odot\left(Z^T\mathbf{z}_s\right)\right)\right)},\end{aligned} \tag{25}$$

where $V_{q;} \in \mathbb{R}^{1\times R}$ represents the pth row of $V = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_R] \in \mathbb{R}^{m_2\times R}$, $\odot$ is the matrix Hadamard product, $A_{;q;} \in \mathbb{R}^{m_1\times N}$ represents the matrix which fixes the second module of A, and traversals are the other two modules. It is defined as

$$\left(A_{;q;}\right)_{pi} = \mathrm{A}_{pqi}, \quad 1 \le \forall p \le m_1, 1 \le \forall i \le N. \tag{26}$$

Now, $\mathbf{v}_r$ and $\mathbf{u}_r$ are calculated.

3.2.2. The Calculation of Z. Then we discuss the calculation of $\nabla f_{\mathbf{U},\mathbf{V}}(\mathbf{Z}^{(t)})$. The differential of f along $\mathbf{z}_s$ ($\forall s, 1 \le s \le R$) is

$$\begin{aligned}d\left(f_{\mathbf{z}_s}\right) &= \left\langle \mathbf{A} - \sum_{r=1}^{R}\mathbf{u}_r\otimes\mathbf{v}_r\otimes\mathbf{z}_r, -d\left(\mathbf{z}_s\right)\otimes\mathbf{u}_s\otimes\mathbf{v}_s\right\rangle\\ &\quad + \frac{1}{2}d\left(\lambda\mathbf{z}_s^T L\mathbf{z}_s\right) - \frac{1}{2}d\left(\sigma\mathbf{z}_s^T L^p\mathbf{z}_s\right)\\ &= \left\langle \sum_{r=1}^{R}\mathbf{u}_r\otimes\mathbf{v}_r\otimes\mathbf{z}_r, d\left(\mathbf{z}_s\right)\otimes\mathbf{u}_s\otimes\mathbf{v}_s\right\rangle\\ &\quad - \left\langle \mathbf{A}, d\left(\mathbf{z}_s\right)\otimes\mathbf{u}_s\otimes\mathbf{v}_s\right\rangle + d\left(\frac{1}{2}\lambda\mathbf{z}_s^T\left(D-S\right)\mathbf{z}_s\right)\\ &\quad - d\left(\frac{1}{2}\sigma\mathbf{z}_s^T\left(D^p - S^p\right)\mathbf{z}_s\right).\end{aligned} \tag{27}$$

For $(\lambda\mathbf{z}_s^T(D-S)\mathbf{z}_s)/2$, the partial differential for $\mathbf{z}_s^i$ is

$$\begin{aligned}\frac{\partial\left((1/2)\lambda\mathbf{z}_s^T\left(D-S\right)\mathbf{z}_s\right)}{\partial\mathbf{z}_s^i} &= \frac{\partial\left(\mathbf{z}_s\right)}{\partial\mathbf{z}_s^i}\frac{\partial\left((1/2)\lambda\mathbf{z}_s^T\left(D-S\right)\mathbf{z}_s\right)}{\partial\mathbf{z}_s}\\ &= \left(\mathbf{e}^i\right)^T\lambda\left(D-S\right)\mathbf{z}_s,\end{aligned} \tag{28}$$

where the ith element in $\mathbf{e}^i \in \mathbb{R}^N$ is 1 and others are 0. That is, $(\mathbf{e}^i)_i = 1$ and $(\mathbf{e}^i)_{k\ne i} = 0$. Then the partial differential for $\mathbf{z}_s^i$ is

$$\begin{aligned}\frac{\partial f}{\partial\mathbf{z}_s^i} &= \left\langle \sum_{r=1}^{R}\mathbf{u}_r\otimes\mathbf{v}_r\otimes\mathbf{z}_r, \mathbf{u}_s\otimes\mathbf{v}_s\otimes\mathbf{e}^i\right\rangle\\ &\quad - \left\langle \mathbf{A}, \mathbf{u}_s\otimes\mathbf{v}_s\otimes\mathbf{e}^i\right\rangle + \lambda\left(\mathbf{e}^i\right)^T\left(D-S\right)\mathbf{z}_s\\ &\quad - \sigma\left(\mathbf{e}^i\right)^T\left(D^p - S^p\right)\mathbf{z}_s\\ &= \sum_{r=1}^{R}\left\langle \mathbf{u}_r,\mathbf{u}_s\right\rangle\left\langle \mathbf{v}_r,\mathbf{v}_s\right\rangle\mathbf{z}_r^i - \sum_{p=1,q=1}^{m_1,m_2} A_{pqi}\mathbf{u}_s^p\mathbf{v}_s^q\\ &\quad + \left(\lambda D_{ii}\mathbf{z}_s^i - \lambda\sum_{k=1}^{N}S_{ik}\mathbf{z}_s^k\right) - \left(\sigma D_{ii}^p\mathbf{z}_s^i - \sigma\sum_{k=1}^{N}S_{ik}^p\mathbf{z}_s^k\right).\end{aligned} \tag{29}$$

According to (16), the update rule for $\mathbf{z}_s^i$ is

$$\begin{aligned}\mathbf{z}_s^i &= \mathbf{z}_s^i - \mu\left(\mathbf{z}_s^i\right)\frac{\partial f}{\partial\mathbf{z}_s^i}\\ &= \mathbf{z}_s^i - \mu\left(\mathbf{z}_s^i\right)\left(\sum_{r=1}^{R}\left\langle \mathbf{u}_r,\mathbf{u}_s\right\rangle\left\langle \mathbf{v}_r,\mathbf{v}_s\right\rangle\mathbf{z}_r^i + \lambda D_{ii}\mathbf{z}_s^i\right.\\ &\qquad + \sigma\sum_{k=1}^{N}S_{ik}^p\mathbf{z}_s^k - \sum_{p=1,q=1}^{m_1,m_2}A_{pqi}\mathbf{u}_s^p\mathbf{v}_s^q\\ &\qquad \left.-\lambda\sum_{k=1}^{N}S_{ik}\mathbf{z}_s^k - \sigma D_{ii}^p\mathbf{z}_s^i\right).\end{aligned} \tag{30}$$

The update step $\mu(\mathbf{z}_s^i)$ is set as

$$\mu\left(\mathbf{z}_s^i\right) = \frac{\mathbf{z}_s^i}{\left(\sum_{r=1}^{R}\left\langle \mathbf{u}_r, \mathbf{u}_s\right\rangle \left\langle \mathbf{v}_r, \mathbf{v}_s\right\rangle \mathbf{z}_r^i + \lambda D_{ii}\mathbf{z}_s^i + \sigma\sum_{k=1}^{N} S_{ik}^p \mathbf{z}_s^k\right)}. \tag{31}$$

And the final update rule of $\mathbf{z}_s^i$ is

$$\begin{aligned}\mathbf{z}_s^i &= \mathbf{z}_s^i \frac{\sum_{p=1,q=1}^{m_1,m_2} A_{pqi}\mathbf{u}_s^p\mathbf{v}_s^q + \lambda\sum_{k=1}^{N} S_{ik}\mathbf{z}_s^k + \sigma D_{ii}^p\mathbf{z}_s^i}{\sum_{r=1}^{R}\left\langle \mathbf{u}_r, \mathbf{u}_s\right\rangle \left\langle \mathbf{v}_r, \mathbf{v}_s\right\rangle \mathbf{z}_r^i + \lambda D_{ii}\mathbf{z}_s^i + \sigma\sum_{k=1}^{N} S_{ik}^p\mathbf{z}_s^k} \\ &= \mathbf{z}_s^i \frac{\mathbf{u}_s^T A_{:,:i}\mathbf{v}_s + \lambda S_{i,}\mathbf{z}_s + \sigma D_{ii}^p\mathbf{z}_s^i}{Z_{i,}\left(\left(U^T\mathbf{u}_s\right)\odot\left(V^T\mathbf{v}_s\right)\right) + \lambda D_{ii}\mathbf{z}_s^i + \sigma\lambda S_{i,}^p\mathbf{z}_s},\end{aligned} \tag{32}$$

where $Z_{i,} \in \mathbb{R}^{1\times R}$ represents the ith row of $Z = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_R]$; $S_{i,} \in \mathbb{R}^{1\times N}$ and $S_{i,}^p \in \mathbb{R}^{1\times N}$ represent the ith row of S and $\widetilde{S}^p$, respectively; $\odot$ is Hadamard product, $A_{:,:i} \in \mathbb{R}^{m_1\times m_2}$, and is defined as

$$\left(A_{:,:i}\right)_{pq} = A_{pqi}, \quad 1 \le \forall p \le m_1, 1 \le \forall q \le m_2. \tag{33}$$

Now $\mathbf{u}_r$, $\mathbf{v}_r$, and $\mathbf{z}_r$ in the objective function are all calculated.

4. Extreme Learning Machine

ELM is proposed by Huang et al. [18] for SLFNs. Unlike the traditional feedforward neural network training methods, such as the gradient-descent method, the standard optimization method, and the least-square based method, ELM need not tune the hidden layer of SLFNs which may cause learning complicated and inefficient. It could reach the smallest training error and have better generalization performance. The learning speed of ELM is fast, and the parameters have not to be tuned manually. In our proposed algorithm, the extracted features by DNPNTF are fed into ELM for classification.

Given a training set $\mathbf{X} = \{(\mathbf{x}_i, t_i) \mid \mathbf{x}_i \in R^n, t_i \in R^n, i = 1, 2, \ldots, N\}$, where $\mathbf{x}_i$ is the $n \times 1$ input feature vector and t_i is a $m\times 1$ target vector. ELM with K hidden nodes and activation function $g(x)$ is modeled as

$$\sum_{i=1}^{K}\beta_i g\left(\mathbf{x}_j\right) = \sum_{i=1}^{K}\beta_i g\left(w_i\cdot\mathbf{x}_j + b_i\right) = o_j, \quad j = 1, \ldots, N, \tag{34}$$

where $w_i = (w_{i1}, w_{i2}, \ldots, w_{id})^T$ represents the input weight, which is the ith neuron in the hidden layer and the input layer; $\beta_i = (\beta_{i1}, \beta_{i2}, \ldots, \beta_{im})^T$ is the weight vector between the ith hidden neuron and the output layer; o_j is the target vector of the jth input data. In training step, ELM aims to approximate N training samples with zero error, which means $\sum_{j=1}^{N}\|o_j - t_j\| = 0$. Then there exist β_i, w_j, and b_j satisfying that

$$\sum_{i=1}^{K}\beta_i g\left(w_i\cdot\mathbf{x}_j + b_i\right) = t_j, \quad j = 1, \ldots, N. \tag{35}$$

Equation (35) can be reformulated compactly as

$$H\beta = T, \tag{36}$$

where

$$\begin{aligned}&H\left(w_1, \ldots, w_K, b_1, \ldots, b_K, x_1, \ldots, x_N\right) \\ &\quad = \begin{pmatrix} g\left(w_1\cdot x_1 + b_1\right) & \cdots & g\left(w_K\cdot x_1 + b_K\right) \\ \vdots & \ddots & \vdots \\ g\left(w_1\cdot x_N + b_1\right) & \cdots & g\left(w_K\cdot x_N + b_K\right)\end{pmatrix}_{N\times K}, \\ &\beta = \begin{pmatrix}\beta_1^T \\ \vdots \\ \beta_K^T\end{pmatrix}_{K\times m}, \qquad T = \begin{pmatrix}t_1^T \\ \vdots \\ t_N^T\end{pmatrix}_{N\times m}.\end{aligned} \tag{37}$$

H is called the hidden layer output matrix of the neural network, and the ith column of H is the ith hidden neuron output with respect to inputs $x_1, x_2, \ldots, x_N$. It is proved by Huang et al. [18] that weights and biases need not be adjusted and can be arbitrarily given. Therefore, the output weights could be determined by finding the least-square solution $\widehat{\beta}$ as

$$H\beta = T \Longrightarrow \widehat{\beta} = H^{\dagger}T, \tag{38}$$

where $H^{\dagger}$ is the Moore-Penrose generalized inverse of matrix H. Furthermore, the smallest training error can be obtained by $\widehat{\beta}$ as

$$\left\|H\widehat{\beta} - T\right\| = \left\|HH^{\dagger}T - T\right\| = \min_{\beta}\left\|H\beta - T\right\|. \tag{39}$$

As analyzed by Huang, ELM could obtain a good generalization performance with a dramatically increased learning speed by solving (39).

5. Experiments

In this section, we apply DNPNTF via ELM to facial expression recognition. We compare DNPNTF with NMF [7], DNMF [11], and NTF [9] and give the experimental results by employing ELM, NN, SVM [16], and SRC [17]. Two facial expression databases are used: the JAFFE database [31] and the Cohn-Kanade database [32]. Raw facial images are cropped according to the position of eyes and normalized to 32×32 pixels. Figure 1 shows an example of the original face image and the corresponding cropped image. According to the rank-one tensor theory, gray level images are encoded in tensor space.

Since the results of ELM may vary during each different execution, we repeat the execution for 5 times and take the average value as the final result. It is proved by theory analysis and experiments that the classification performance of ELM is affected by the hidden activation function and the number of hidden nodes [23]. However, in this paper we just focus on the application of ELM to facial expression recognition. The activation function used in our algorithm is a simple

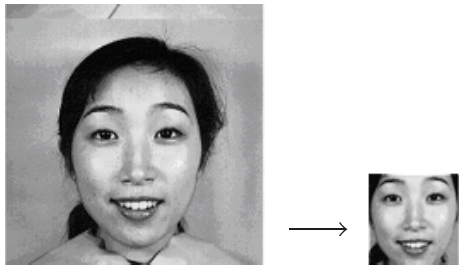


Figure 1: Illustration of the preprocess for original images.

sigmoidal function. The number of hidden nodes is set to be the same as the number of facial expression classes (e.g., 7 for the JAFFE database and 6 for the Cohn-Kanade database). For SVM, the radial basis function (RBF) is used, which is $\exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2) \cdot \gamma$ that is set to be 3 as an empirical value. For SRC, "Homotopy" algorithm is used to solve the minimization of ℓ_1 norm constraint.

5.1. Experiments on JAFFE Database. The JAFFE database [31] is an expression database which contains 213 static facial images captured from 10 Japanese females. Each person poses 2 to 4 examples for each of the 6 prototypic expressions (anger, disgust, fear, happiness, sadness, and surprise) plus the natural face. To evaluate the algorithms, we randomly partition all images into 10 groups, with roughly 70 samples in each group. We take any 9 groups for training and calculate the recognition rates with the remaining one. We repeat it for all the 10 possible choices. Finally, the average result over 10 times' testing was taken.

The average recognition rates of different feature extraction algorithms are shown in Figure 2, where the vertical axis represents the correct recognition rate in percentage and the horizontal axis represents the corresponding dimensions (from 1 to 120). Here, only the NN classifier is used. In the lower range of dimensions, the recognition rates of DNPNTF are similar to other algorithms. This is because DNPNTF extracts the parts-based sparse representations, and only a few features could be generated for recognition in the low range of dimensions. In the higher range of dimensions, DNPNTF outperforms the others. With the increase of the extracted parts-based features, DNPNTF could achieve good recognition performance. Since different constraints were added, the improved versions of NMF, including DNMF and NTF, outperform the conventional NMF.

The top recognition rates of different algorithms with corresponding dimensions are illustrated in Table 1. NMF achieves the highest rate at a low dimension, while DNMF achieves the highest rate at a high dimension. Although more dimensions are needed, DNPNTF achieves the highest recognition rate compared with others. This is because the constraints about manifold structure and discriminant information are considered, which are critical for classification.

Figure 3 shows the basis images obtained on the JAFFE database by NMF, NTF, and DNPNTF. Based on the principle of NMF, the face images are represented by combining multiple basis images with addition only, and the basis images

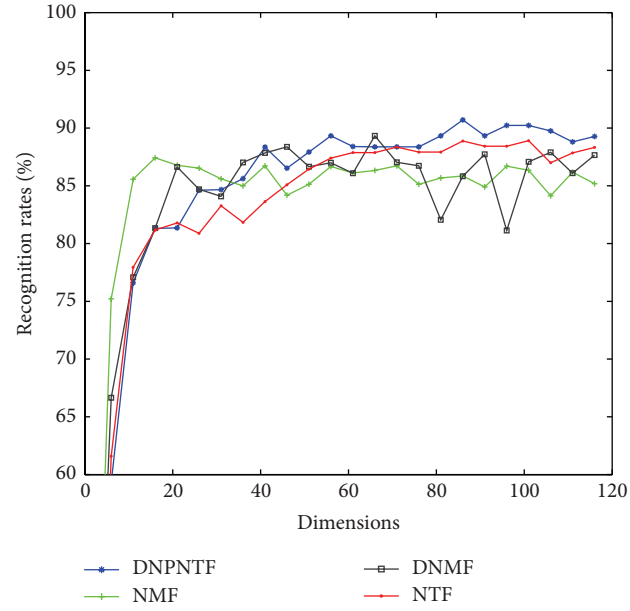


Figure 2: The recognition rate versus dimension curves achieved on the JAFFE database.

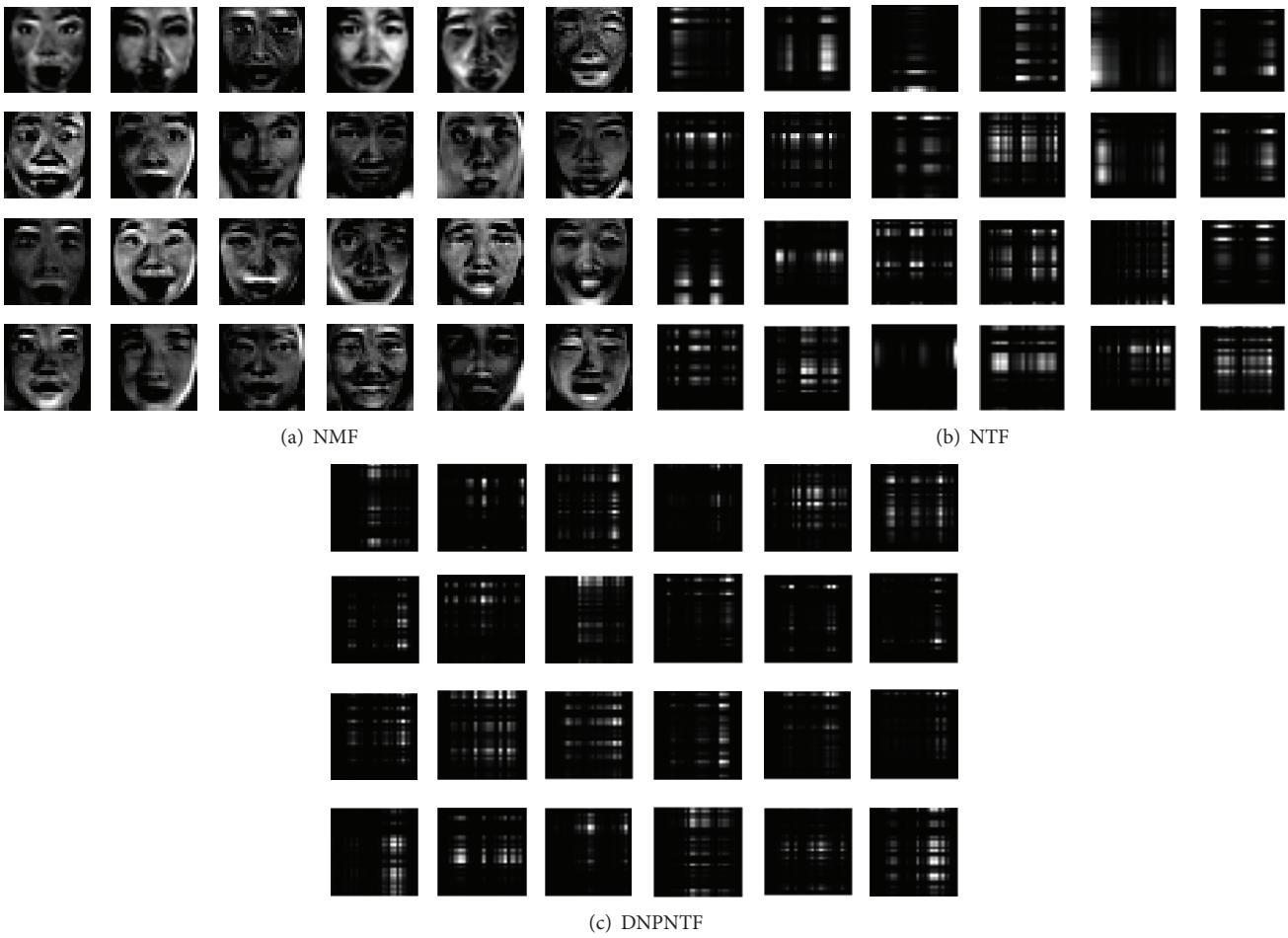
Table 1: The top recognition rate and the corresponding dimension achieved on the JAFFE database.

	DNPNTF	NMF	DNMF	NTF
Recognition rates (%)	90.71	87.43	89.33	88.91
Dimensions	98	16	66	101

are expected to represent facial parts. In this database, the basis images calculated by NMF are not sparse. NTF and DNPNTF which execute in the tensor space could generate parts-based sparse representations. Since more constraints were adopted, DNPNTF generate sparser basis images which reflect distinct features for recognition.

Then we give the experiments to prove the effectiveness of DNPNTF via ELM. The average recognition rates of DNPNTF with ELM, NN, SVM, and SRC are given in Figure 4, where the vertical axis represents the correct recognition rate in percentage and the horizontal axis represents the corresponding dimensions (from 1 to 120). ELM and SRC achieve better recognition performance compared with NN and SVM, and ELM achieves the highest recognition rate. The top recognition rates with the corresponding dimensions are given in Table 2.

5.2. Experiments on Cohn-Kanade Database. The Cohn-Kanade database [32] consists of a large amount of image sequences starting from natural face and ending with the peak of the corresponding expression. 104 subjects with different ages, genders, and races are instructed to pose a series of 23 facial displays, including the 6 prototypic expressions. In our experiments, for every image sequence, we take 2 to 8 continuous frames near the peak expression as the static samples. We use the face images of all subjects. We partition the subject to 3 exclusive groups, and in each group,



(a) NMF (b) NTF

(c) DNPNTF

Figure 3: Basis images obtained on the JAFFE database.

Table 2: The top recognition rate of different classifiers achieved on the JAFFE database.

	ELM	NN	SRC	SVM
Recognition rates (%)	94.39	90.71	93.54	91.03
Dimensions	55	98	63	101

Table 3: The top recognition rate and the corresponding dimensions achieved on the Cohn-Kanade database.

	DNPNTF	NMF	DNMF	NTF
Recognition rates (%)	63.61	58.17	58.83	59.72
Dimensions	101	76	26	120

for each of the prototypic expression, we select 100 samples; that is, there are 600 samples in each group and the size of the total set is 1800. During the experiment, we adopt the leave-one-group-out strategy and 3-fold cross-validation: each time two groups are taken as training set and the remaining group is left for testing. This procedure is repeated for 3 times.

The average recognition rates of different algorithms on the Cohn-Kanade database are shown in Figure 5, where the vertical axis represents the correct recognition rate in percentage and the horizontal axis represents the corresponding dimensions (from 1 to 120). Here, only the NN classifier is used. Table 3 shows the top recognition rates with the corresponding dimensions. The recognition rates obtained on the Cohn-Kanade database are lower than those obtained on the JAFFE database. It can be explained that the experiments on the Cohn-Kanade database are person-independent, which are more difficult than the person-dependent experiments on the JAFFE database. From Figure 5, we can see that the performance of DNPNTF is superior to others with nearly all dimensions. Its recognition rates improve with the increase of dimensions.

Lastly, we give the experiments about different classifiers on the Cohn-Kanade database. The average recognition rates of DNPNTF via ELM, NN, SVM, and SRC are shown in Figure 6, and the top recognition rates are given in Table 4. SVM and SRC achieve better performance compared with NN. ELM achieves the best recognition accuracy among all tested algorithms on almost all dimensions. It means ELM could use the information contained in extracted features better than other classifiers.

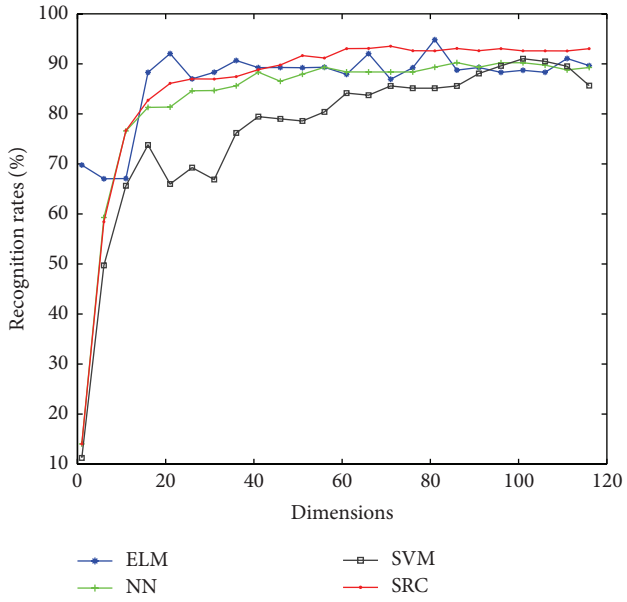


Figure 4: The recognition rate versus dimension curves of DNPNTF by using different classifiers on the JAFFE database.

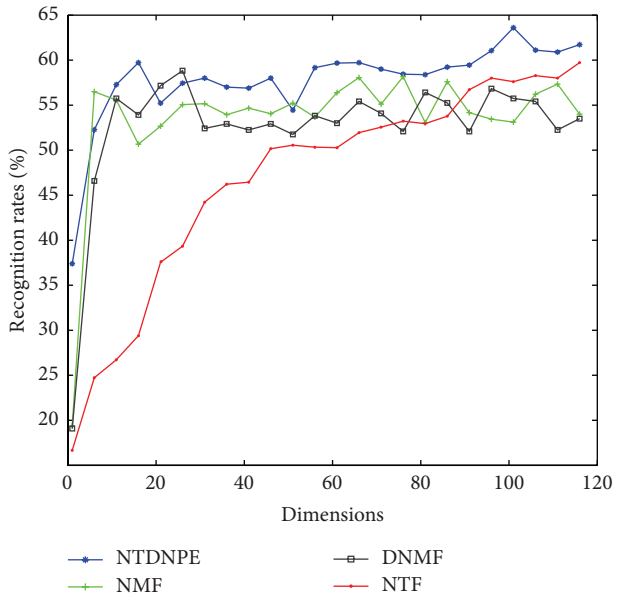


Figure 5: The recognition rate versus dimension curves achieved on the Cohn-Kanade database.

6. Conclusions

In this paper, a novel DNPNTF algorithm with the application to facial expression recognition was proposed, which adopts ELM as the classifier. To incorporate the spatial information and the discriminant class information, a discriminant constraint is added to the objective function according to the manifold learning and graph embedding theory. To guarantee the convergence, the project gradient method is used for optimization. Theoretical analysis and experimental results demonstrate that DNPNTF could achieve better performance compared with NTF, NMF, and its variant. Then the discriminant features generated by DNPNTF are fed into ELM to learn an optimal model for recognition. In our experiments, DNPNTF via ELM achieves higher recognition rate compared with NN, SVM, and SRC.

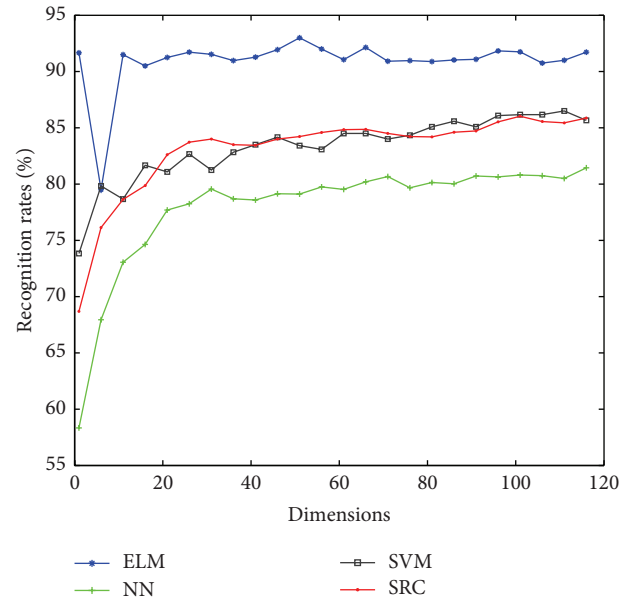


Figure 6: The recognition rate versus dimension curves of DNPNTF by using different classifiers on the Cohn-Kanade database.

Table 4: The top recognition rate and the corresponding dimensions achieved on the Cohn-Kanade database.

	ELM	NN	SRC	SVM
Recognition rates (%)	86.00	63.61	73.00	72.06
Dimensions	51	101	116	106

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported partly by the National Natural Science Foundation of China (61370127, 61472030), the Fundamental Research Funds for the Central Universities (2013JBM020 and 2014JBZ004), and Beijing Higher Education Young Elite Teacher Project (YETP0544).

References

[1] A. Mehrabian, "Communication without words," *Psychology Today*, vol. 2, no. 4, pp. 53–56, 1968.

[2] A. J. Calder, A. M. Burton, P. Miller, A. W. Young, and S. Akamatsu, "A principal component analysis of facial expressions," *Vision Research*, vol. 41, no. 9, pp. 1179–1208, 2001.

[3] W. S. Yanmbor, "Analysis of PCA-based and Fisher discriminant-based image recognition algorithms, Computer Science Department, Colorado State University, Fort Collins, Colo, USA, M.S. Thesis," Tech. Rep. CS-00-103, 2000.

[4] X. F. He, D. Cai, S. C. Yan, and H.-J. Zhang, "Neighborhood preserving embedding," in *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV '05)*, pp. 1208–1213, October 2005.

[5] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 586–591, June 1991.

[6] X. F. He and P. Niyogi, "Locality preserving projections," in *Proceedings of the Advances Neural Information Processing Systems*, pp. 153–160, 2003.

[7] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 21, pp. 788–791, 1999.

[8] S. Z. Li, X. W. Hou, H. J. Zhang, and Q. S. Cheng, "Learning spatially localized, parts-based representation," in *Proceeding of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I207–I212, December 2001.

[9] P. O. Hoyer, "Non-negative sparse coding," in *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, pp. 557–565, 2002.

[10] D. Cai, X. He, X. Wu, and J. Han, "Non-negative matrix factorization on manifold," in *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM '08)*, pp. 63–72, Pisa, Italy, December 2008.

[11] S. Zafeiriou, A. Tefas, I. Buciu, and I. Pitas, "Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 683–695, 2006.

[12] Y. Wang, Y. Jia, C. Hu, and M. Turk, "Non-negative matrix factorization framework for face recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 19, no. 4, pp. 495–511, 2005.

[13] T. Hazan, S. Polak, and A. Shashua, "Sparse image coding using a 3D non-negative tensor factorization," in *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV '05)*, vol. 1, pp. 50–57, October 2005.

[14] M. Welling and M. Weber, "Positive tensor factorization," *Pattern Recognition Letters*, vol. 22, no. 12, pp. 1255–1261, 2001.

[15] A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*, pp. 793–800, Bonn, Germany, August 2005.

[16] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[17] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[18] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.

[19] S.-J. Wang, H.-L. Chen, W.-J. Yan, Y.-H. Chen, and X. Fu, "Face recognition and micro-expression recognition based on discriminant tensor subspace analysis plus extreme learning machine," *Neural Processing Letters*, vol. 39, no. 1, pp. 25–43, 2014.

[20] X. Chen, W. Liu, J. Lai, Z. Li, and C. Lu, "Face recognition via local preserving average neighborhood margin maximization and extreme learning machine," *Soft Computing*, vol. 16, no. 9, pp. 1515–1523, 2012.

[21] G. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[22] G. Huang, "An insight into extreme learning machines: random neurons, random features and kernels," *Cognitive Computation*, 2014.

[23] G.-B. Huang and L. Chen, "Enhanced random search based incremental extreme learning machine," *Neurocomputing*, vol. 71, no. 16–18, pp. 3460–3468, 2008.

[24] J. Cao, T. Chen, and J. Fan, "Fast online learning algorithm for landmark recognition based on BoW framework," in *Proceedings of the 9th IEEE Conference on Industrial Electronics and Applications*, Hangzhou, China, 2014.

[25] J. Cao and L. Xiong, "Protein sequence classification with improved extreme learning machine algorithms," *BioMed Research International*, vol. 2014, Article ID 103054, 12 pages, 2014.

[26] J. Cao, Z. Lin, G. Huang, and N. Liu, "Voting based extreme learning machine," *Information Sciences*, vol. 185, no. 1, pp. 66–77, 2012.

[27] S. C. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: a general framework for dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, 2007.

[28] D. Liang, J. Yang, Z. L. Zheng, and Y. C. Chang, "A facial expression recognition system based on supervised locally linear embedding," *Pattern Recognition Letters*, vol. 26, no. 15, pp. 2374–2389, 2005.

[29] X. F. He and P. Niyogi, "Locality preserving projections," in *Proceedings of Advances Neural Information Processing Systems*, pp. 153–160, 2003.

[30] W. H. Greub, *Multilinear Algebra*, Springer, New York, NY, USA, 2nd edition, 1978.

[31] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expressions with Gabor wavelets," in *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 200–205, 1998.

[32] T. Kanade, J. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," in *Proceedings of the 4th IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 46–53, Grenoble, France, 2000.