A General Nonlinear Embedding Framework Based on Deep Neural Network

Yan Huang, Wei Wang, Liang Wang, Tieniu Tan
Center for Research on Intelligent Perception and Computing (CRIPAC)
National Laboratory of Pattern Recognition, Institute of Automation
Chinese Academy of Sciences, Beijing 100190, China
{*yhuang, wangwei, wangliang, tnt*}*@nlpr.ia.ac.cn*

***Abstract*—Recently there has been increasing interest in deep neural network due to its powerful representability in several successful applications such as speech recognition and image classification. In this paper, we propose a general nonlinear embedding framework based on deep neural network which can be utilized to implement a family of dimensionality reduction algorithms. The objective function of our framework consists of two terms: 1) an embedding term transforms the input to a low-dimensional representation with a multilayer network; and 2) a regularization term which computes the reconstruction error of the original input by unrolling the multilayer network to a deep autoencoder. We adopt a layer-by-layer pretraining procedure to obtain good initial weights for the network, and then minimize the objective function by backpropagating derivatives of the two terms. To evaluate the proposed framework, we perform face recognition and digit classification experiments. The experiments demonstrate that the proposed framework achieves better results than the state-of-the-art algorithms. The success of our framework further verifies deep neural network's advantages in representation learning.**

I. Introduction

In real-world applications, it is very common that the data to be analyzed usually has a high dimensionality which leads to the well-known curse of dimensionality in statistical pattern recognition. For analyzing such data efficiently, various dimensionality reduction algorithms—supervised or unsupervised; originating from statistics or geometry theory—have been proposed to uncover the underlying structure of high-dimensional data in low-dimensional spaces ([1], [2], [3]). These techniques have played a very important role in many tasks, such as pattern classification, information visualization and storage of the high-dimensional data ([4], [1], [5]).

However, the problem of "out-of-sample" is a typical limitation commonly existing in previous methods, such as ISOMAP [2], LLE [4] and Laplacian Eigenmap (LE) [6], which can not handle the samples out of the training data well. To solve the out-of-sample problem, some researchers adopt either a linear transform for the linear embedding, such as Locality Preserving Projection (LPP) [1] and Neighborhood Preserving Embedding (NPE) [7], or a kernel trick for the nonlinear embedding, such as kernel PCA [8]. When the high-dimensional data is very complex, linear projections fails to capture the intrinsic manifold of the data due to its limited representability, while kernel-based methods usually need rich experience to choose the kernel function or have to enumerate all kernel functions.

Recently, deep neural network has attracted much attention again since an efficient layer-wise unsupervised learning strategy is proposed to pretrain this kind of deep architectures [5]. It has shown powerful representability and achieved great successes in several fields, such as image classification [9]. Given the advantages mentioned above, a special neural network, called deep autoencoder, has been used to learn low-dimensional representations [5], and a deep neural network for preserving the class neighborhood structure has been proposed in [10].

In this paper, we propose a general nonlinear embedding framework based on deep neural network, which is called *Deep Neural network Embedding* (DNE). Specifically, DNE utilizes deep neural network to learn a nonlinear embedding from a high-dimensional data space to a low-dimensional feature space. The framework aims to minimize two terms: 1) an embedding term computes the objective function of any dimensionality reduction algorithm, by transforming the input to a low-dimensional representation with a multilayer neural network, and preserving the statistical or geometrical properties of the input; and 2) a regularization term computes the reconstruction error of the input by unrolling the multilayer neural network into a deep autoencoder. When a linear activation is used, the regularization term actually imposes an orthogonal constraint on the weight vectors of the network. Similar to [5], considering two adjacent layers as a Restricted Boltzmann Machine, we pretrain the multilayer neural network in a layer-by-layer way. Finally, we minimize the two terms of this framework through backpropagating their derivatives. Within this framework, we implement a family of dimensionality reduction algorithms and achieve better results than the state-of-the-art algorithms.

Several aspects of DNE should be pointed out here:

1) DNE solves the problem of "out-of-sample" through a highly nonlinear transformation.
2) DNE can be applied to any existing dimensionality reduction algorithm as long as it has an explicit objective function.
3) DNE is a flexible multilayer neural network which can handle complex datasets through changing the number of the network layers and the nodes of each layer.

The rest of the paper is organized as follows. In Section II, we review related work to the proposed DNE. In Section III, we introduce the formulation of DNE and develop several

dimensionality reduction algorithms under the framework of DNE. The experimental results are shown in Section IV. We conclude the paper in Section V.

II. Related Work

Principal Component Analysis (PCA) [11] is one of the most popular linear dimensionality reduction techniques, which finds projection directions that keep the maximal variance of the original data. Linear Discriminant Analysis (LDA) [12], as a well known supervised technique, uses class labels to find a linear subspace which is optimal for discrimination. To overcome the limitations of LDA, Marginal Fisher Analysis (MFA) [3] is proposed to characterize the intraclass compactness and interclass separability. Recently, numerous nonlinear methods have been proposed to handle a large amount of complex nonlinear data in real world. ISOMAP [2] is proposed to preserve the pairwise geodesic distance by taking into account the distribution of the neighboring data points. The methods mentioned above are all considered globally, which are sensitive to outliers. Locally Linear Embedding (LLE) [4] first considers to fulfill the task of dimensionality reduction by preserving the local property of the data. In LLE, each sample is reconstructed as a linear combination of its nearest neighbors, which is preserved in the projected space. Laplacian Eigenmaps (LE) [6] weights local pairwise distance in the projected space using the corresponding pairwise distance in the high-dimensional data space, to retain the local structure of each sample. Although these nonlinear local methods can capture the intrinsic structure of high dimensional data well, they have a common weakness that the extension to out-of-sample examples needs to recompute all the samples. Neighborhood Preserving Embedding (NPE) [7] and Locality Preserving Projection (LPP) [1] as the linear approximations to LLE and LE are proposed to handle the out-of-sample problem, respectively.

Since the resurgence of deep neural network in 2006, several studies have been conducted to use deep neural network for dimensionality reduction. Deep autoencoder (DAE) [5] minimizes the reconstruction error of the original input to obtain a low-dimensional representation, which exactly provides a nonlinear extension of PCA. Through extending DAE to the multimodal field, shared representations are extracted for both image and text in [9]. By adding noise to the highest hidden layer of DAE, the binary codes have been learned for the input, which are used for efficient information retrieval [13]. The most similar work to our DNE is the nonlinear extension of neighborhood component analysis (NCA) [14][10], which uses a deep neural network as the transformation function and preserves the class neighborhood structure by exploiting the class label information. It should be noted that our DNE serves as a general platform to implement both unsupervised and supervised dimensionality reduction algorithms while nonlinear NCA only implements a supervised algorithm. Additionally, we utilize a regularization term [15] to avoid trivial solutions.

III. DNE for Dimensionality Reduction

The existing linear or nonlinear dimensionality reduction algorithms have proposed various objective functions from the viewpoints of preserving the statistical or geometrical properties of the input. However, these methods generally lack a powerful transformation function. As we know, deep neural network can approximate arbitrarily complex linear or nonlinear functions, which is a good choice to transform complex high-dimensional data into low-dimensional representations. Based on the above considerations, we propose *Deep Neural network Embedding* (DNE) which utilizes deep neural network for nonlinear embedding. Furthermore, inspired by traditional PCA, LDA, ISOMAP, LE, LLE and MFA, we employ DNE as a general platform to implement several corresponding algorithms for dimensionality reduction.

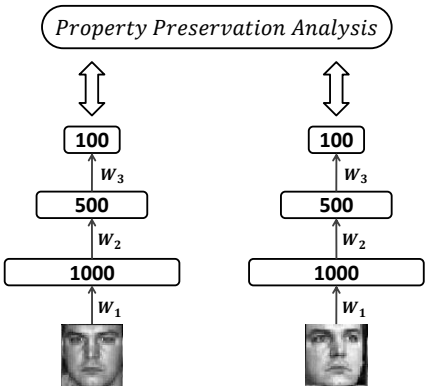


Fig. 1. The proposed Deep Neural network Embedding (DNE) framework. $\{W_1, W_2, W_3\}$ are the weights of the network, the bottom face images are the high-dimensional inputs, the top layers with 100 nodes are the desired low-dimensional representations and the number in each middle layer represents the number of learned features. "Property Preservation Analysis" aims to preserve various properties of the data by imposing constraints on the learned low-dimensional representation space.

A. The Formulation of DNE

The proposed DNE consists of two terms: an embedding term which preserves the statistical or geometrical properties of the data in the low-dimensional space, and a regularization term which minimizes the reconstruction error of the original input. In this section, we first consider a deep neural network as a multilayer encoder to obtain the low-dimensional representations of the input data, and define an embedding term with these representations. Then by unrolling the multilayer encoder, we obtain a deep autoencoder with a symmetric multilayer decoder, and define a regularization term with the reconstruction error from this autoencoder. Finally, we combine the embedding term and the regularization term as a general objective function for our DNE.

Embedding term: Taking a four-layer neural network in Fig. 1 for example as a multilayer encoder, which contains an input layer and three hidden layers. The input layer denoted as face images represents a high-dimensional input, the third hidden layer with 100 nodes represents the desired 100-dimensional feature representation and the number in each middle layer is the number of learned features.

Let $X = \{\mathbf{x}_i : \mathbf{x}_i \in \mathbb{R}^{n\times 1}\}_{i=1,2,\cdots,N}$ denote the set of input data, and $W = \{W_i\}_{i=1,2,3}$ for the weights of the encoder network. The multilayer encoder defines a mapping $f(\cdot|W) : \mathbb{R}^{n\times 1} \rightarrow \mathbb{R}^{d\times 1}(d < n)$ which transforms a high-dimensional input $\mathbf{x}$ to a low-dimensional representation $f(\mathbf{x})$:

$$f(\mathbf{x}) = W_3^T h(W_2^T h(W_1^T \mathbf{x})) \quad (1)$$

where $h(\mathbf{x})$ is a sigmoid function $1/(1+e^{-\mathbf{x}})$. Note that we omit all the bias terms for simplicity.

Utilizing the low-dimensional representations of the input, we define an embedding term as $L(X, f(\cdot|W))$. The embedding term imposes constraints on the low-dimensional representation space $f(X)$ to preserve the statistical or geometrical properties of the data. Various constraints have been formulated into the objective functions in previous dimensionality reduction algorithms. Therefore, inspired by these objective functions, the embedding term can implement a family of dimensionality reduction algorithms. For example, the embedding term L can be defined as the objective function of LE to preserve the local structure of the data space [6]:

$$L = \sum_{ij} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}} \|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|^2 \quad (2)$$

where $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$ obtained from the encoder are respectively the low-dimensional representations of $\mathbf{x}_i$ and $\mathbf{x}_j$, $e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/t}$ is the weight computed in the data space, which penalizes the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ in the projected space, and t is a tuning parameter.

Regularization term: Unrolling the four-layer encoder network, we obtain a deep autoencoder with a symmetric multilayer decoder. The multilayer decoder also defines a mapping $g(\cdot|W^T) : \mathbb{R}^{d\times 1} \rightarrow \mathbb{R}^{n\times 1} (d < n)$ which reconstructs the original input from the low-dimensional representation $f(\mathbf{x})$. When using binary-valued data as the input, the reconstructed output $g(\mathbf{x})$ is

$$g(\mathbf{x}) = h(W_1^T h(W_2^T h(W_3^T f(\mathbf{x})))) \quad (3)$$

When using real-valued data as the input, the reconstructed output $g(\mathbf{x})$ is

$$g(\mathbf{x}) = W_1^T h(W_2^T h(W_3^T f(\mathbf{x}))) \quad (4)$$

Similar to previous methods [3] [1] which usually exploit a constraint to avoid a trivial solution, such as an orthogonal constraint $WW^T = I$, we introduce a regularization term denoted as $E(X, g(\cdot|W^T))$ which computes the reconstruction error of the network E as follows:

$$E = \|g(f(X)) - X\|^2 \quad (5)$$

When considering a two-layer encoder and $h(\mathbf{x})$ as the identity, the proposed regularization term exactly imposes an orthogonal constraint on the rows of the weight matrix W [15]. To illustrate this, we rewrite Eqn 5 as follows:

$$\begin{aligned} E &= \left\|WW^T X - X\right\|^2 \\ &= tr\left[(WW^T - I)^T C_X (WW^T - I)\right] \\ &= tr\left\|(WW^T - I) E_X D^{1/2}\right\|_F^2 \end{aligned} \quad (6)$$

where C_X is the covariance matrix of the input X, the columns of E_X are eigenvectors of C_X, D is a diagonal matrix of eigenvalues of C_X, and I is the identity matrix. As we can see, minimizing E is equal to impose $WW^T = I$ on the weight matrix.

TABLE I. THE SIMILARITY MATRIXES OF THE DEVELOPED DNES.

Methods	Similarity matrix	Index set
DNE-PCA	$S_{ij} = -1/N$	$j : \mathbf{x}_j \in X$
DNE-LDA	$S_{ij} = 1/n_{c_i}$	$j \in V(c_i)$
DNE-ISOMAP	$S_{ij} = (-\frac{1}{2} H D_G^2 H)_{ij}$	$j : \mathbf{x}_j \in X$
DNE-LE	$S_{ij} = (M + M^T - M^T M)_{ij}$	$j \in N_k(i)$
DNE-LLE	$S_{ij} = e^{-\|\|\mathbf{x}_i - \mathbf{x}_j\|\|^2/t}$	$j \in N_k(i)$
DNE-MFA	$S_{ip} = 1$, $S_{iq} = -1$	$p \in V_{k_1}(c_i)$, $q \in V_{k_2}(\bar{c_i})$

Objective function: After computing the embedding term L for every two samples and the regularization term E for each sample, we combine them together as a general objective function Φ for the proposed DNE framework:

$$\Phi = L(X, f(\cdot|W)\,) + \lambda E(X, g(\cdot|W^T)\,) \quad (7)$$

where λ is a tuning parameter which balances the embedding term L and the regularization term E.

B. The Training of DNE

A two-step procedure is exploited to efficiently train the weights W of the proposed DNE, which consists of a layer-by-layer pretraining and a final fine-tuning.

Pretraining: Before introducing the pretraining procedure, we would like to recall the conception of Restricted Boltzmann Machine (RBM) [5], which plays an important role in pretraining. A RBM consists of a visible layer and a hidden layer. Each node in the visible layer is connected to each node in the hidden layer, and values of these nodes are all binary-valued. The energy function of this model is defined as follows:

$$F(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^T \mathbf{w}\mathbf{h} - \mathbf{c}\mathbf{v} - \mathbf{b}\mathbf{h} \quad (8)$$

where $\mathbf{v}$ and $\mathbf{h}$ are respectively the visible and hidden nodes, $\mathbf{w}$ is the weight matrix between visible nodes and hidden nodes, $\mathbf{c}$ and $\mathbf{b}$ are respectively the visible biases and hidden biases.

RBM cannot handle real-valued data well because its visible nodes are binary-valued, while Gaussian restricted Boltzmann machine (GRBM) contains real-valued visible nodes and has the following energy function:

$$F(\mathbf{v}, \mathbf{h}) = \sum_i \frac{(v_i - c_i)^2}{2\sigma_i^2} - \sum_i \sum_j \frac{v_i}{\sigma_i} w_{ij} h_j - \sum_j b_j h_j \quad (9)$$

Where $\{w_{ij}, c_i, b_j\}$ are model parameters, σ_i is the standard deviation of the Gaussian noise for visible node i. Based on the energy function, the joint probability distribution of all the nodes is defined as:

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-F(\mathbf{v}, \mathbf{h})) \quad (10)$$

where Z is a normalization factor that scales $P(\mathbf{v}, \mathbf{h})$ to [0,1]. The RBM parameters $\{\mathbf{w}, \mathbf{b}, \mathbf{c}\}$ can be trained by minimizing the negative log-likelihood $-\sum_{\mathbf{h}} \log P(\mathbf{v}, \mathbf{h})$ via stochastic gradient descend. Although exact gradients are intractable, they

can be approximated by Contrastive Divergence (CD) [16], which has been proved to be efficient in practice.

Pretraining [5] is an unsupervised learning strategy which treats adjacent two layers as a RBM and trains RBMs layer-by-layer, to obtain good initial weights that are close to the optimum. Taking Fig. 1 as an example, we pretrain the weight W_1 by regarding the input layer (the input face image) and the first hidden layer with 1000 nodes as a GRBM. When pretraining the weight W_2, we regard the first hidden layer and the second layer with 500 nodes as a RBM, and the input to this RBM is the output of the GRBM. In the same way, we pretrain all the weights by training the RBMs in a bottom-up manner. In this procedure, it should be noticed that DNE can exploit a large amount of unlabeled data to find good initial weights when performing supervised dimensionality reduction with labeled data.

Fine-tuning: After the pretraining procedure, we fine-tune the weights through backpropagating derivatives of the general objective function to achieve better performance.

C. Algorithms for Dimensionality Reduction

In this section, inspired by several well-known algorithms, namely PCA, LDA, ISOMAP, LE, LLE and MFA, we utilize DNE as a general platform to develop associated dimensionality reduction algorithms: DNE-PCA, DNE-LDA, DNE-ISOMAP, DNE-LE, DNE-LLE and DNE-MFA. Similar to [3], we unify the six algorithms and rewrite the general objective function in Eqn 7 as follows:

$$\Phi = \sum_{i,j} S_{ij} \|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|^2 + \lambda \sum_i \|g(f(\mathbf{x}_i)) - \mathbf{x}_i\| \quad (11)$$

where S_{ij} is the similarity between the input $\mathbf{x}_i$ and $\mathbf{x}_j$, $f(\mathbf{x}_i)$ and $g(f(\mathbf{x}_i))$ are respectively the low-dimensional representation and the reconstruction of $\mathbf{x}_i$, λ is a ratio which balances the embedding term and the regularization term.

In the following, by changing the definition of similarity matrix S, we utilize the DNE framework to implement PCA, LDA, ISOMAP, LE, LLE and MFA. We give a summary of these similarity matrixes in Table I.

DNE-PCA: The goal of PCA is to find projection directions which maximize the variance of the data. To be consistent with the goal, DNE-PCA tries to remove projection directions which minimize the variance, and defines the similarity matrix as $S_{ij} = -1/N$, where N is the total number of input samples.

It should be noticed that DNE-PCA is different from deep autoencoder (DAE) [5] which is often regarded as a nonlinear extension of PCA. DNE-PCA explicitly considers to maximize the variance in the projected low-dimensional space while DAE does not.

DNE-LDA: LDA aims to find directions which maximize the interclass variance and minimize the intraclass variance. DNE-LDA follows to define the similarity matrix as $S_{ij} = 1/n_{c_i}$, $j \in V(c_i)$, where c_i is the class that sample x_i belongs to, n_{c_i} is the number of samples whose class is c_i, and $V(c_i)$ is the index set of class c_i. Thus, by adding the regularization term, the objective function of DNE-LDA follows Eqn 11.

We notice that DNE-LDA relaxes two assumptions arisen in traditional LDA: 1) the distribution of data of each class is Gaussian; 2) the number of classes is bigger than the number of projection dimensions.

DNE-ISOMAP: In order to preserve the geodesic distances of the data points when they are projected to the low-dimensional space, we define the similarity matrix as $S_{ij} = (-\frac{1}{2} H D_G^2 H)_{ij}$ [3], where D_G is geodesic distance matrix [2], $H = I - \frac{1}{N} e e^T$, and e is a N-dimensional unit vector. Reconstruction error is also employed as the regularization term in the objective function of DNE-ISOMAP.

DNE-LLE: The assumption made by LLE is that the local property of the high-dimensional data is preserved in the projected low-dimensional space, which is followed by DNE-LLE. Let $N_k(i)$ be the index set of k nearest neighbors of sample $\mathbf{x}_i$, M be the local reconstruction weight matrix: $\sum_{j \in N_k(i)} M_{ij} = 1$, $M_{ij} = 0$ if $j \notin N_k(i)$ [4]. So the similarity matrix of DNE-LLE is defined as $S_{ij} = (M + M^T - M^T M)_{ij}$ if $i \neq j$; 0 otherwise [3]. The similarity matrix and the reconstruction error are respectively utilized to compute the embedding term and the regularization term in Eqn 11.

DNE-LE: In order to preserve the local structure of the data, DNE-LE uses $S_{ij} = e^{-||\mathbf{x}_i - \mathbf{x}_j||^2/t}$ as the similarity matrix, where t is a tuning parameter [6]. By combining the embedding term and the regularization term together, the objective function of DNE-LE also follows the proposed general objective function.

DNE-MFA: Similar to MFA which characterizes the interclass separability and intraclass compactness, we define the similarity matrix as: $S_{ip} = 1$ if $p \in V_{k_1}(c_i)$, and $S_{iq} = -1$ if $q \in V_{k_2}(\bar{c}_i)$, where $V_{k_1}(c_i)$ is the index set of k_1 nearest neighbors whose classes are c_i, and $V_{k_2}(\bar{c}_i)$ is the index set of k_2 nearest neighbors whose classes are not c_i, Thus, the objective function of DNE-MFA follows the formulation in Eqn 11.

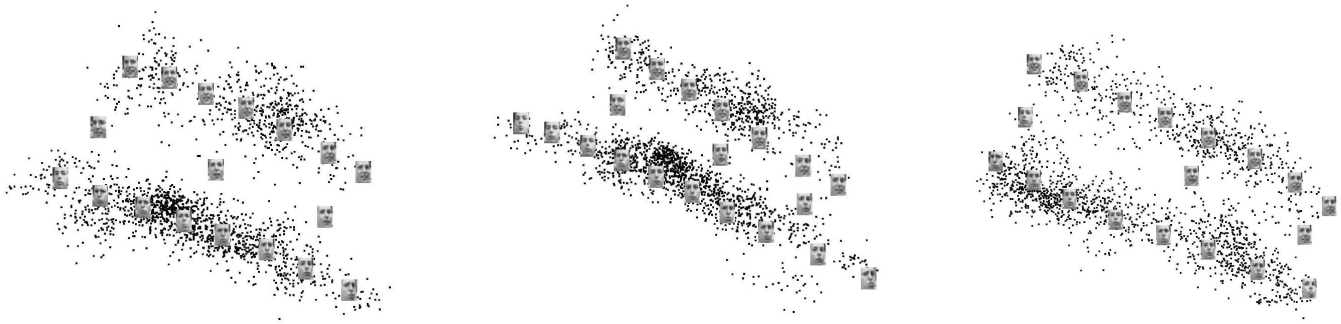
IV. Experimental Results

In this section, we will apply the proposed DNEs (DNE-PCA, DNE-LDA, DNE-ISOMAP, DNE-LE, DNE-LLE, DNE-MFA) to two applications. One is face recognition and the other is digit classification. Before this, we present face manifold visualization.

A. Manifold Visualization

It is widely believed that high-dimensional data can be efficiently represented by their intrinsic low-dimensional manifold. Various manifolds have been discovered by some dimensionality reduction algorithms, such as [4] and [1]. In the following, we show that the proposed algorithms (DNEs) are also able to detect meaningful manifolds of high-dimensional face images. The experimental dataset is a face image set which is also used in [4]. The dataset contains 1,965 gray-level face images, each one is taken from sequential frames of a small video and represented by a 560-dimensional vector.

All the DNEs are utilized to generate two-dimensional manifolds for the face images. Due to limited space, we just select three representative manifolds for illustration in Fig.



(a) DNE-LE (b) DNE-LLE (c) DNE-ISOMAP

Fig. 2. Two-dimensional embedding of face images by DNE-LE, DNE-LLE and DNE-ISOMAP, respectively.

2, which are respectively obtained by DNE-LE, DNE-LLE, and DNE-ISOMAP. All of them apply a 560-1000-2 neural network. In the manifolds, each face is embedded as a two-dimensional point, and several representative faces are shown next to the corresponding data points. As can be seen in Fig. 2 (a), there exist two clusters of data points corresponding to different facial expressions, e.g., happy expressions in the top cluster and abnormal expressions in the bottom cluster. Among each cluster, the face images distribute evenly due to their large variations and the facial expressions and viewpoints are varying smoothly along the tube-like manifold. Between the two clusters, there exist a small amount of face images with normal expressions, which account for the expression transition. Similarly, we can observe two clusters of face images with different facial expressions in all three manifolds because DNE-LE, DNE-LLE and DNE-ISOMAP all aim to preserve the properties of high-dimensional data during embedding, which implicitly impose clustering constraints on the data. However, among these three manifolds, the shapes of clusters are quite different, which can be attributed to the various preserved properties.

B. Face Recognition

In order to evaluate the six DNEs quantitatively, we first perform face recognition experiments and compare the results of DNEs with those of PCA [11], LDA [12], LPP [1], MFA [3] and DAE [5]. We do not compare the results of ISOMAP and LLE because of their "out of sample" problems. LPP is compared because it is the linear extension of LE and is very popular in face recognition.

TABLE II. EXPERIMENTAL RESULTS ON THE PIE DATASET.

Method	Dimension	Error Rate
PCA [11]	150	20.6%
LDA [12]	67	5.7%
LPP [1]	110	4.6%
MFA [3]	85	2.6%
DAE [5]	70	3.5%
DNE-PCA	70	**3.3%**
DNE-LDA	70	**1.8%**
DNE-ISOMAP	70	**3.1%**
DNE-LE	70	**3.1%**
DNE-LLE	70	**3.3%**
DNE-MFA	70	**1.6%**

In this experiment, we use the CMU PIE database [17], which contains 41,368 face images from 68 persons. The face images in this dataset are collected under 13 different poses, 43 different illumination conditions, and with 4 different expressions. 170 face images are used for each person, 85 for training and 85 for testing.

Similar to the data preprocessing in [1], we project all the face images to a PCA subspace to reduce noise. In this step, we preserve 98% of the energy and obtain a 157-dimensional feature for each image. The architecture of DNE in this experiment is similar to Fig. 1, the input layer and three hidden layers have respectively 157, 200, 70 nodes (i.e., a 157-200-70 DNE). We employ the pretraining procedure to initialize the network weights, and the fine-tuning procedure to tune the weights better. Then, the trained network is utilized as transformation to project the test data to the low-dimensional representations. Finally, the obtained low-dimensional representations are classified by a nearest neighborhood classifier.

The recognition results are listed in Table II, which are evaluated by misclassified error rate. We select the reduced dimensions of traditional PCA, LDA, LPP and MFA when they achieve the lowest error rate. The reduced dimensions of our DNEs are not very sensitive in a large range, and the experimental results remain stable when the dimensions are larger than 70, so we select 70 as the reduced dimension for all the DNEs. As we can see from Table II, our six DNEs generally perform better than the corresponding traditional algorithms, which indicates that deep neural network has an ability of modeling complex transformation. Through considering the relationships between the samples by preserving the statistical or geometrical properties, our six DNEs all achieve better results than DAE. Although both DAE and DNE-PCA are two nonlinear extensions of PCA based on deep neural network, DNE-PCA achieves a lower error rate than DAE on this dataset, which demonstrates the effectiveness of DNE. By fully exploiting class label information, two supervised algorithms DNE-LDA (1.8%) and DNE-MFA (1.6%) achieve the best performance. Therefore, DNE-LDA is not restricted to the assumption that the distribution of data is a Gaussian.

The selection of parameter λ in our experiment do not has a significant impact on the results. We select two algorithms for illustration, namely the unsupervised DNE-LE and supervised DNE-MFA, and show their plots of error rate versus parameter

λ in Fig 3. As we can see, the error rate is not sensitive to the parameter λ in a large range.

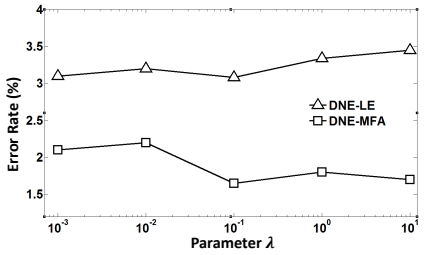


Fig. 3. The plots of error rate versus parameter λ on the PIE dataset by DNE-LE and DNE-MFA.

C. Digit Classification

In this section, we further apply the DNEs to the application of digit image classification, and compare their results to the same algorithms in the last experiment. We re-implement all the comparison algorithms.

The MNIST dataset[1] is used in this experiment. Each digit image has a size of 28 $\times$ 28 pixels. Because of computational limitation, we randomly select 1,000 digit images for each digit, 500 for training and 500 for testing. To make fair comparisons, we directly perform classification on the raw data with a nearest neighborhood classifier and take its error rate 6.5% as a baseline.

In this experiment, we employ a (28$\times$28)-500-200-30 neural network as the architecture of DNEs. After pretraining and fine-tuning procedures, we utilize the trained network to obtain low-dimensional representations for all the test digit images. Finally, we also adopt a nearest neighborhood classifier to classify the low-dimensional representations. The classification results of all the algorithms are also measured by the error rate which are shown in Table III. From the table, we can obtain similar results to the face recognition experiment. DNE-PCA, DNE-LDA, DNE-LE and DNE-MFA outperform their corresponding traditional algorithms by 1.2%, 6.8%, 2.9% and 5.0%, respectively. Due to the large variations of digit images, the similarities computed from the original data space can not reflect the real relationships between samples. So the three unsupervised method: DNE-ISOMAP, DNE-LE and DNE-LLE perform similarly. The similarities of the two supervised algorithms DNE-MFA and DNE-LDA are only related to the class label information and they achieve the relatively better results with the error rates of 4.5% and 4.8%, respectively. The experimental results remain stable when the reduced dimensions are larger than 30.

V. Conclusion and Future Work

This paper has proposed a nonlinear embedding framework named *Deep Neural network Embedding* (DNE). Regarding DNE as a general platform, we have developed six dimensionality reduction algorithms inspired by PCA, LDA, ISOMAP, LE, LLE and MFA. In the face recognition and digit classification experiments, DNE has achieved better results than the state-of-the-art algorithms. In the future, we will propose more powerful algorithms by designing new embedding terms which preserve both local and global prosperities of the data.

[1]The MNIST dataset is available at: *http://yann.lecun.com/exdb/mnist/*.

TABLE III. Experimental results on the MNIST dataset

Method	Dimension	Error Rate
PCA [11]	55	6.2%
LDA [12]	9	11.6%
LPP [1]	55	7.9%
MFA [3]	45	9.5%
DAE [5]	30	5.3%
DNE-PCA	30	**5.0%**
DNE-LDA	30	**4.8%**
DNE-ISOMAP	30	**5.1%**
DNE-LE	30	**5.0%**
DNE-LLE	30	**5.1%**
DNE-MFA	30	**4.5%**

Acknowledgments

This work is jointly supported by National Natural Science Foundation of China (61175003, 61135002, 61202328), Hundred Talents Program of CAS, National Basic Research Program of China (2012CB316300).

References

[1] X. He, S. Yan, Y. Hu, P. Niyogi, and H. J. Zhang, "Face recognition using laplacianfaces," *IEEE TPAMI*, 2005.

[2] J. Tenenbaum, V. Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, 2000.

[3] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Li, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE TPAMI*, 2007.

[4] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, 2000.

[5] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, 2006.

[6] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," *NIPS*, 2002.

[7] X. He, D. Cai, S. Yan, and H. Zhang, "Neighborhood preserving embedding," *ICCV*, 2005.

[8] H. Hoffmann, "Kernel pca for novelty detection," *Pattern Recognition*, 2007.

[9] N. Srivastava and R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," *NIPS*, 2012.

[10] R. Salakhutdinov and G. Hinton, "Learning a non-linear embedding by preserving class neighbourhood structure," *AI and Statistics*, 2007.

[11] M. Turk and A. Pentland, "Face recognition using eigenfaces," *CVPR*, 1991.

[12] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE TPAMI*, 1997.

[13] R. Salakhutdinov and G. E. Hinton, "Semantic hashing," *IJAR*, 2009.

[14] J. Goldberger, S. T. Roweis, and G. E. Hinton, "Neighbourhood components analysis," *NIPS*, 2004.

[15] Q. V. Le, A. Karpenko, J. Ngiam, and A. Ng, "Ica with reconstruction cost for efficient overcomplete feature learning," *NIPS*, 2011.

[16] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, 2002.

[17] T. Sim, S. Baker, and M. Bsat, "The cmu pose, illumination, and expression (pie) database," *IEEE AFGR*, 2002.