# Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction

Anonymous ICCV submission

Paper ID 657

## Abstract

*Autoencoder and its deep version as traditional dimensionality reduction methods have achieved great success via the powerful representability of neural network. However, they just use each instance to reconstruct itself and ignore to explicitly model the relationships between instances. In this paper, we propose a generalized autoencoder, which extends the traditional autoencoder in two aspects: 1) each instance $x_i$ is used to reconstruct a set of instances $\{x_j\}$ rather than itself. 2) The reconstruction error of each instance ($||x_j - x'_i||^2$) is weighted by a relational function of $x_i$ and $x_j$. Hence, the generalized autoencoder captures the structure of the data space through minimizing the weighted distances between reconstructed instances and the original ones. The generalized autoencoder provides a general neural network framework for dimensionality reduction. In addition, we propose a multilayer architecture of the generalized autoencoder called deep generalized autoencoder to handle highly complex datasets. Finally, to evaluate the proposed methods, we perform extensive experiments on three datasets. The experiments demonstrate that the proposed methods achieve the state-of-the-art performance.*

## 1. Introduction

Real-world data, such as images of faces and digits, usually has a high dimensionality which leads to the well-known curse of dimensionality in statistical pattern recognition. Moreover, high dimensional data largely increases the computational cost of the existing systems. However, high dimensional data usually lies in a lower dimensional space, so-called "intrinsic dimensionality space". Various methods of dimensionality reduction have been proposed to discover the underlying structure of the low-dimensional space, which play a very important role in many tasks, such as information retrieval [7], pattern classification [10], information visualization and the storage of high-dimensional data [18].

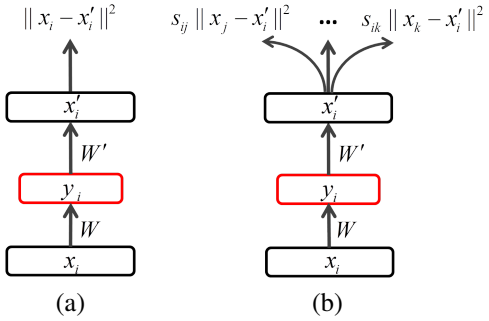



Figure 1. Traditional autoencoder and the proposed generalized autoencoder. (a) In the traditional autoencoder, $x_i$ only needs to reconstruct itself and the reconstruction error $||x_i - x'_i||^2$ just measures the distance between $x_i$ and $x'_i$. (b) In the generalized autoencoder, $x_i$ needs to reconstruct a set of instances $\{x_j, x_k, ...\}$. Each reconstruction error $s_{ij}||x_j - x'_i||^2$ measures a weighted distance between $x_j$ and $x'_i$.

Traditional autoencoder is an artificial neural network method for dimensionality reduction, which aims to learn a compressed representation for an input through minimizing its reconstruction error [14]. Fig. 1(a) shows the architecture of an autoencoder. More recently, several autoencoder variants have been proposed to learn more robust representations, such as sparse autoencoder [9], denoising autoencoder [19] and contractive autoencoder [12]. However, these methods have a common problem that they just use each instance to reconstruct itself and ignore to explicitly model the relationships between instances, which limits them to effectively exploit the structure of the data space, e.g. neighborhood structure. Preserving the global or local structure of the original data space is very important for dimensionality reduction methods [20].

In this paper, we propose a generalized autoencoder to model the relationship among data instances and preserve the structure of the original data space. The architecture of a generalized autoencoder is shown in Fig. 1(b), which differs from the traditional autoencoder in two aspects. On the

one hand, the generalized autoencoder learns a compressed representation $y_i$ for an instance $x_i$, and builds the relationship between $x_i$ and a set of instances $\{x_j, x_k...\}$ by using $y_i$ to reconstruct each instance of this set, not just $x_i$ itself. On the other hand, the generalized autoencoder imposes a relational function $s_{ij}$ on the reconstruction error $||x_j - x_i'||^2$. The stronger the relationship between $x_i$ and $x_j$, the larger the weight $s_{ij}$ and the more similar between $x_i'$ and $x_j$. Hence, the generalized autoencoder captures the structure of the data space through minimizing the weighted distances between reconstructed instances and the original ones. When replacing the sigmoid function in the generalized autoencoder with a linear transformation, we find that the linear generalized autoencoder and the linearization extension of graph embedding [20] have a similar objective function. It indicates that the generalized autoencoder can also be a general framework for dimensionality reduction by defining different reconstructed instances and different weights for reconstruction errors. Inspired by existing dimensionality reduction ideas, we derive six generalized autoencoders including both supervised and unsupervised methods. Furthermore, we propose a deep generalized autoencoder which extends generalized autoencoder to multiple layers.

Several contributions of the proposed generalized autoencoder are listed below:

1. The generalized autoencoder provides a general framework for developing new dimensionality reduction algorithms through defining the set of reconstructed instances and the reconstruction weights.

2. The generalized autoencoder can solve the problem of "out-of-sample" which traditional nonlinear methods often encounter. Moreover, different from the kernel extensions, such as kenel PCA [6], which heavily depend on the choice of the kernels, the generalized autoencoder does not need to choose a specific nonlinear transformation.

3. The deep generalized autoencoder is a flexible multilayer neural network which can handle various complex datasets through changing the number of the layers and the nodes of each layer.

The rest of the paper is organized as follows. We review the related work in Section 2 and introduce the formulation of generalized autoencoder with its connection to graph embedding in Section 3. In Section 4, we illustrate deep generalized autoencoder. The experimental results on three datasets are presented in Section 5. Finally, we discuss the proposed method and conclude the paper in Section 6 and 7, respectively.

## 2. Related Work

In this section, we briefly review the existing literature that closely relates to the generalized autoencoder and dimensionality reduction.

**Autoencoders** More recently, several autoencoder variants are proposed to learn robust features. Sparse autoencoder imposes a sparsity constraint on hidden representation to model the sparse coding of primary visual cortex [9]. Denoising autoencoder transforms a corrupted input into a hidden representation, and then tries to recover original input from this representation [19]. Contractive autoencoder adds a regularization term that is the sum of squares of all partial derivatives of the hidden representations with respect to the input, which penalizes the sensitivity of the hidden representation to the input [12]. We will discuss the connection between the proposed generalized autoencoder and denoising autoencoder in Section 6.

**Dimensionality Reduction** Traditionally, Principal Component Analysis (PCA) is one of the most popular linear dimensionality reduction techniques [11]. It projects the original data onto its principal directions with the maximal variance. Linear Discriminant Analysis (LDA) as a popular supervised technique is used to find a linear subspace which is optimal for discrimination when the class labels are provided [2]. To overcome the limitations of LDA, Marginal Fisher Analysis (MFA) is proposed to characterize the intraclass compactness and interclass separability [20]. However, there exists a large amount of complex nonlinear data in real world that linear techniques can not handle well. Accordingly, some nonlinear methods have been proposed in the recent literature. ISOMAP [16] retains the pairwise geodesic distance by taking into account the distribution of the neighboring data points. Locally Linear Embedding (LLE) [13] opens the way to local nonlinear techniques which generally have the local-preserving property. In LLE, each datapoint is a linear combination of its nearest neighbors. The linear relationship is preserved in the projected space. Laplacian Eigenmaps (LE) [1] weigh the local pairwise distance in the projected space with the corresponding distance in the original space. 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) [3] and Locality Preserving Projection (LPP) [4] as the linear approximations to LLE and LE are proposed to handle the out-of-sample problem, respectively. A complete review and systematic comparison of current methods of dimensionality reduction are presented in [17].

Our generalized autoencoder (GAE) can produce a series of nonlinear dimensionality reduction methods which share the similar ideas to the above methods in Section 3.3. The most related work to our GAE is multilayer autoencoder [5]

which is indeed a special case of deep GAE by defining the input itself as the only reconstructed instance and the weights of all the reconstruction errors as 1.

# 3. Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction

In this section, we will explain the generalized autoencoder as a general neural network framework for dimensionality reduction in detail.

## 3.1. Generalized Autoencoder

The generalized autoencoder consists of two parts, an encoder and a decoder. As shown in Fig. 1 (b), the encoder maps an input $x_i \in \mathcal{R}^{d_x}$ to a reduced hidden representation $y_i \in \mathcal{R}^{d_y}$ with a function $g(x)$, which is as follows

$$y_i = g(Wx_i) \tag{1}$$

where $g(x)$ is either the identity for linear projection or a sigmoid function $\frac{1}{1+e^{-x}}$ for nonlinear projection , and the parameter $W$ is a $d_y \times d_x$ weight matrix. In this paper, we ignore the biases of the neural network for simple notations.

The decoder maps hidden representation $y_i$ to a reconstruction $x_i'$

$$x_i' = f(W'y_i) \tag{2}$$

The parameter $W'$ is another $d_x \times d_y$ weight matrix which can be either $W^T$ or not. $f(x)$ is either the identity for linear reconstruction or a sigmoid function for binary reconstruction.

To model the relationship between instances, the decoder reconstructs a set of instances indexed by $\Omega_i = \{j, k...\}$ with specific weights $S_i = \{s_{ij}, s_{ik}...\}$ for the input $x_i$. The weighted reconstruction error corresponding to $x_i$ is computed as follows

$$e_i(W, W') = \sum_{j \in \Omega_i} s_{ij} L(x_j, x_i') \tag{3}$$

where $L$ is the reconstruction error. Generally, the squared error $L(x_j, x_i') = ||x_j - x_i'||^2$ is used for linear reconstruction and the cross-entroy loss $L(x_j, x_i') = -\sum_{q=1}^{d_x} x_j^{(q)} log(x_i'^{(q)}) + (1 - x_j^{(q)}) log(1 - x_i'^{(q)})$ for binary reconstruction [5].

For $n$ training samples, the total reconstruction error $E$ is the summation of each $e_i$

$$E(W, W') = \sum_{i=1}^{n} e_i(W, W') = \sum_{i=1}^{n} \sum_{j \in \Omega_i} s_{ij} L(x_j, x_i') \tag{4}$$

Generalized autoencoder needs to learn the parameters $\{W, W'\}$ through minimizing $E$.

## 3.2. Connection to Graph Embedding

As we know, graph embedding [20] is a general framework for dimensionality reduction, which represents each vertex of a graph as a low-dimensional vector that preserves similarities between the vertex pairs. The similarities characterize certain statistical or geometric properties. The formulation of graph embedding is as follows:

$$y^* = arg \min_{y^T B y = c} \sum_{i,j} s_{ij} ||y_i - y_j||^2 \tag{5}$$

where $y$ is the low-dimensional representation, $s_{ij}$ is the similarity between the vertex $i$ and $j$ which often satisfies $s_{ij} = s_{ji}$, $c$ is a constant and $B$ is the constraint matrix to avoid a trivial solution[1].

The linearization extension of graph embedding considers that the low-dimensional representation can be obtained from a linear projection $y = X^T w$, where $w$ is the projection vector and $X = [x_1, x_2, ..., x_n]$. The objective function of the linearization extension becomes

$$w^* = arg \min_{\substack{w^T X B X^T w = c \\ \text{or } w^T w = c}} \sum_{i,j} s_{ij} ||w^T x_i - w^T x_j||^2 \tag{6}$$

Similarly, the generalized autoencoder represents the hidden representation of a neural network as a low-dimensional representation when assuming $d_y < d_x$. The hidden representation also preserves the similarities between pairs of instances through one reconstructing the other. We expand the total reconstruction error $E$ in Eqn. 4 for the linear reconstruction case as follows:

$$W^*, W'^* = arg\, min \sum_{i,j} s_{ij} ||x_j - f(W' g(W x_i))||^2 \tag{7}$$

When both $f$ and $g$ are the identity, we obtain a linearization extension of the generalized autoencoder. Tying the weights $W' = W^T$ like [12], and assuming only one hidden node in the network, e.g. $d_y = 1$, $w$ as the column of $W$, the objective function in Eqn. 7 becomes

$$w^* = arg\, min \sum_{i,j} s_{ij} ||x_j - w w^T x_i||^2 \tag{8}$$

Considering $w^T w = c$ and $y_i = w^T x_i$, it is not difficult to rewrite Eqn. 8 as

$$w^* = arg \min_{w^T w = c} \sum_{i,j} s_{ij} (||w^T x_i - w^T x_j||^2 + (\frac{c}{2} - 1) y_i^2) \tag{9}$$

where $c$ determines the scale of $w$ as that in Eqn. 6. We can see that the generalized autoencoder has an additional term which shows different tuning behaviors over hidden

[1]Original constraint $i \neq j$ is not used here because $i = j$ does not influence the objective function

Table 1. Six generalized autoencoders inspired by PCA [11], LDA [2], ISOMAP [16], LLE [13], LE [1] and MFA [20]

| Method | Reconstruction Set | Reconstruction Weight |
|---|---|---|
| GAE-PCA | $j = i$ | $s_{ij} = 1$ |
| GAE-LDA | $j \in \Omega_{c_i}$ | $s_{ij} = \frac{1}{n_{c_i}}$ |
| GAE-ISOMAP | $j : x_j \in X$ | $s_{ij} \in S = -H\Lambda H/2$ |
| GAE-LLE | $j \in N_k(i)$, | $s_{ij} = (M + M^T - M^T M)_{ij}$ if $i \neq j$; |
| | $j \in (N_k(m) \cup m), j \neq i$ if $\forall m, i \in N_k(m)$ | 0 otherwise |
| GAE-LE | $j \in N_k(i)$ | $s_{ij} = exp\{-\|\|x_i - x_j\|\|^2/t\}$ |
| GAE-MFA | $j \in \Omega_{k_1}(c_i)$, | $s_{ij} = 1$ |
| | $j \in \Omega_{k_2}(\bar{c}_i)$ | $s_{ij} = -1$ |

representation $y$ along with varying $c$. If $c = 2$, the generalized autoencoder has the similar objective function to graph embedding, without obvious tuning on $y$. If $c > 2$, this term penalizes $y$ not too large even with a large scale $w$. If $c < 2$, this term encourages $y$ to be large enough in terms of a small scale $w$.

For the case of $d_y > 1$, the generalized autoecoder solves Eqn.7 with stochastic gradient descent to obtain $d_y$-dimensional projection vectors. Graph embedding generally transforms original objective function to an eigendecomposition problem, and the eigenvectors to the smallest $d_y$ eigenvalues span the $d_y$-dimensional projection space.

The linearization extension above just illustrates the relationship between generalized autoencoder and graph embedding. In real applications, generalized autoencoder with $g(x)$ as a nonlinear projection is more preferred due to its powerful representability.

### 3.3. General Framework for Dimensionality Reduction

As can be seen from the above analysis, the generalized autoencoder can also be a general platform for dimensionality reduction through defining different reconstructed instances and different weights for reconstruction error. In the following, we propose six generalized autoencoders inspired by previous dimensionality reduction ideas, namely PCA [11], LDA [2], ISOMAP [16], LLE [13], LE [1] and MFA [20].

**GAE-PCA:** an input $x_i$ only needs to reconstruct itself with unit weight for the reconstruction error, namely $\Omega_i = \{i\}$, $s_{ii} = 1$. Given the constraint $w^T w = 1$, Eqn. 8 can be rewritten as

$$w^* = arg \max_{w^T w = 1} \sum_i w^T x_i x_i^T w \qquad (10)$$

which is the formulation of traditional PCA with $E(x) = 0$ [2].

The neural network implementation of GAE-PCA is exactly the *traditional autoencoder* [5] which is a special case of our generalized autoencoder.

**GAE-LDA:** an input $x_i$ needs to reconstruct all the instances belonging to same class as the input, namely $j \in \Omega_{c_i}$ where $\Omega_{c_i}$ is the index set of the class $c_i$ that $i$ belongs to. The weight $s_{ij}$ is inversely proportional to the sample size $n_{c_i}$ of class $c_i$, namely $s_{ij} = \frac{1}{n_{c_i}}$. The one-dimensional linearization extension of GAE-LDA follows Eqn. 9. It should be noted that GAE-LDA does not need to satisfy two constraints as traditional LDA: 1) the data of each class is of Gaussian distribution; 2) the number of projection dimensions is lower than the class number.

**GAE-ISOMAP:** an input $x_i$ needs to reconstruct all the instances $X$. Let $D_G$ be the geodesic distances of all the data pairs [16]. The weight matrix for reconstruction errors is $S = -H\Lambda H/2$, where $H = I - \frac{1}{n} ee^T$, $e$ is the $n$-dimensional one vector, $\Lambda_{ij} = D_G^2(i,j)$ [20]. The one-dimensional linearization extension of GAE-ISOMAP follows Eqn. 9.

**GAE-LLE:** an input $x_i$ needs to reconstruct its $k$ nearest neighbors and some others' $k$ nearest neighbors which the input $x_i$ belongs to, namely $j \in N_k(i)$, $j \in (N_k(m) \cup m)$ and $j \neq i$ if $\forall m, i \in N_k(m)$, where $N_k(i)$ is the index set of the $k$ nearest neighbors of the $i$-th instance. Let $M$ be the local reconstruction coefficient matrix, and $\sum_{j \in N_k(i)} M_{ij} = 1$, $M_{ij} = 0$ if $j \notin N_k(i)$ [13]. The weight matrix for reconstruction errors is $S_{ij} = (M + M^T - M^T M)_{ij}$ if $i \neq j$; 0 otherwise [20]. The one-dimensional linearization extension of GAE-LLE follows Eqn. 9.

**GAE-LE:** an input $x_i$ needs to reconstruct its $k$ nearest neighbors, namely $j \in N_k(i)$. The weight for reconstruction error is $s_{ij} = exp\{-||x_i - x_j||^2/t\}$, where $t$ is a tuning parameter [1]. Given that the objective function of LE is similar to Eqn. 5, the one-dimensional linearization extension of GAE-LE follows Eqn. 9.

**GAE-MFA:** an input $x_i$ needs to reconstruct its $k_1$ nearest neighbors of the same class with the weight 1, and its $k_2$ nearest neighbors of the different class with the weight $-1$, namely if $j \in \Omega_{k_1}(c_i)$, $s_{ij} = 1$ and if $j \in \Omega_{k_2}(\bar{c}_i)$, $s_{ij} = -1$, where $\Omega_{k_1}(c_i)$ is the index set of the input $x_i$'s $k_1$ nearest neighbors in the class $c_i$ and $\Omega_{k_2}(\bar{c}_i)$ is the index set of $x_i$'s $k_2$ nearest neighbors in all the other classes except $c_i$. In a similar way to MFA, the positive reconstructions

characterize the intraclass compactness while the negative reconstructions characterize the interclass separability.

We give a summary of the reconstruction sets and reconstruction weights of these six methods in Table 1. As can be seen from this table, we can easily devise new algorithms in this framework by combining the ideas of these methods, e.g., introduce class labels to GAE-LE (or GAE-LLE) and reconstruct the neighbors in the same class. All these extensions will be further explored in our future work.

## 4. Deep Generalized Autoencoders

To handle more complex dataset, we extend the generalized autoencoder to a multilayer network called deep generalized autoencoder (dGAE), which contains a multilayer encoder network to transform the high-dimensional input data to a low-dimensional representation and a multilayer decoder network to recover the data. The structures of these two networks are symmetric with respect to the layer of low-dimensional representation. For example, Fig. 2(a) shows a five-layer dGAE on the face dataset used in Section 5.3. It has a three-layer encoder network and a three-layer decoder network labeled with a box. They are symmetric with respect to the second hidden layer of 100 real-valued nodes. In order to obtain good initial weights for the network, we adopt the layer-wise pretraining procedure introduced in [5]. When pretraining the first hidden layer of 200 binary features, the real-valued face data in the input layer is modeled by a Gaussian distribution and these two layers are modeled as a Gaussian restricted Boltzmann machine (GRBM) [5]. When pretraining the second hidden layer of 100 real-valued features, the first hidden layer is now considered as the input to the second hidden layer and these two layers are combined to form another GRBM as shown in Fig. 2(b). These two GRBMs can be trained efficiently using Contrastive Divergence [5]. After pretraining, we obtain the initial weights $\{W_i\}_{i=1,2}$ for both the encoder network and the decoder network, and further fine-tune them by backpropagating the derivatives of the total reconstruction error in Eqn. 4. Fig. 2(c) illustrates the fine-tuning procedure of a deep GAE-LDA (dGAE-LDA) through reconstructing the other faces in the same class. The $\epsilon_i$ finally adjusts the initial weight.

## 5. Experimental Results

In this section, we will discuss two applications of the GAE algorithm, one for face recognition and the other for digital classification. We begin with a face manifold analysis and a digit data visualization to illustrate the effectiveness of GAE.

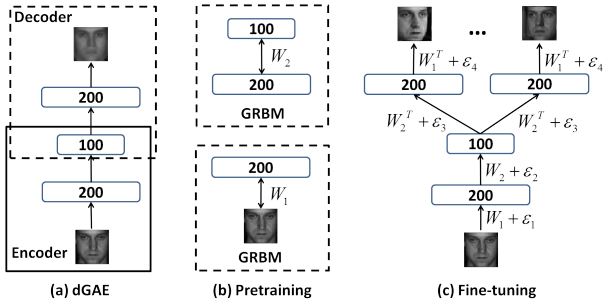



Figure 2. Training a deep generalized autoencoder (dGAE) on the face dataset used in Section 5.3. (a) The dGAE consists of a three-layer encoder network and a three-layer decoder network. A reconstructed face is presented in the output layer. (b) Pretraining the dGAE through learning a stack of Gaussian restricted Boltzmann machines (GRBM). (c) Fine-tuning a deep GAE-LDA.

### 5.1. Data Sets

Three datasets are employed in our experiments. (1) The first face dataset (F1) is the same as that used in [13], which contains 1,965 grayscale face images taken from sequential frames of a small video. The size of each image is 20× 28 pixels. The face images in F1 are believed to generate from a manifold with few degrees of freedom, which will be used in our manifold analysis experiments. (2) The second face dataset (F2) is from CMU PIE face database which contains 68 subjects with 41,368 face images [15]. The face images in F2 are captured under varying pose, illumination and expression, which will be used in our face recognition experiments. (3) The MNIST dataset[2] contains 60,000 grayscale images of handwritten digits, which will be used in our data visualization and digit classification experiments. The size of each digit image is 28 × 28 pixels. We randomly select 1,000 images of each digit for computational reasons, 500 for training and the other 500 for testing.

### 5.2. Manifold Analysis and Data Visualization

Generally, high-dimensional data is considered to be embedded in a low-dimensional space, such as face images and handwritten digit images mentioned above. Similar to previous methods [16][1][13], GAE provides an effective means to discover the nonlinear manifold structure.

**Manifold Analysis** We map F1 face images into a two-dimensional space using LPP [4], dGAE-PCA and dGAE-LE [3]. Both of our methods consist of an encoder with layers of size (20×28)-200-2 and a symmetric decoder. The 2-D

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

[3]Due to limited space, we only show the mapping results of dGAE-PCA and dGAE-LE in manifold analysis. dGAE-PCA is actually the deep autoencoder [5]. We randomly choose dGAE-LE to make a comparison with dGAE-PCA. In the following data visualization, we only show the mapping results of three representative methods, dGAE-LE as an unsupervised method and dGAE-MFA, dGAE-LDA as two supervised methods.

data points are shown in Fig. 3. The representative faces are shown next to the data points. As can be seen, the distribution of the data points of our two methods approximates a circular planar around a center point. Along the circular surface, the facial expression and the viewing point of faces change smoothly. Moreover, the mapping results of dGAE-PCA and dGAE-LE all present a continuous neighborhood structure in the circular planar, which differ from that of LPP very much. LPP divides the face images into two parts with mouth open or not, and change the facial expression and pose in an orthogonal way. It is because the weight vectors learned in our neural network framework are not orthogonal and the obtained components tend to have equal variances. Comparing with dGAE-PCA (deep autoencoder [5]), the data points of dGAE-LE distribute much more evenly because dGAE-LE preserves neighborhood structure explicitly.

**Data Visualization** We generate the 2-D data points of 0∼9 digit images in MNIST dataset using two unsupervised methods LPP [4], dGAE-LE and four supervised methods MFA [20], LDA [2], dGAE-MFA, dGAE-LDA. The six figures in Fig. 4 show the results in which the data points of 10 digits are labeled with different colors and marks. Our three methods consist of an encoder with layers of size (28×28)-1000-500-250-2 and a symmetric decoder. As can be seen, the data points of different digits overlap seriously in LPP, MFA and LDA's results while our three methods obtain more distinctive clusters than them. Moreover, by fully exploiting class label information, the clusters from dGAE-MFA, dGAE-LDA are more distinguishable than dGAE-LE.

### 5.3. Face Recognition

In this section, we evaluate the performance of our six GAEs on PIE dataset, and compare the results to the other four methods.

Similar to the experimental setting in [4], 170 face images of each individual are used in the experiments, 85 for training and the other 85 for testing, and all the images are first projected to a PCA subspace for noise reduction. We retain 98% of the energy in the PCA step and obtain a 157-dimensional feature for each image. In the experiments, we adopt a 157-200-100 encoder network for the deep generalized autoencoders. After learning the parameters of the deep generalized autoencoders, the low-dimensional representations are computed for all the face images. Then we apply a nearest neighborhood classifier to classify the testing data and compute the error rate as an evaluation measure.

Table 2 shows the recognition results of the 10 methods on this dataset, namely PCA [11], LDA [2], LPP [4], MFA [20] and our six dGAEs. Due to the out-of-sample problem, we can not give the results of ISOMAP [16], LLE [13] and LE [1]. Considering LPP as a popular linear approximation to LE, we present its result here. In the experiments, we select the reduced dimensions with the best performance for the other four methods, which is shown in parentheses in Table 2. Because the reduced dimensions of the dGAEs are not very sensitive in a large range, we select 100 as the reduced dimension for all our methods.

Table 2. Performance comparison on the PIE dataset (ER is short for error rate. The reduced dimensions of the other four methods are in parentheses while those of our methods are all 100.)

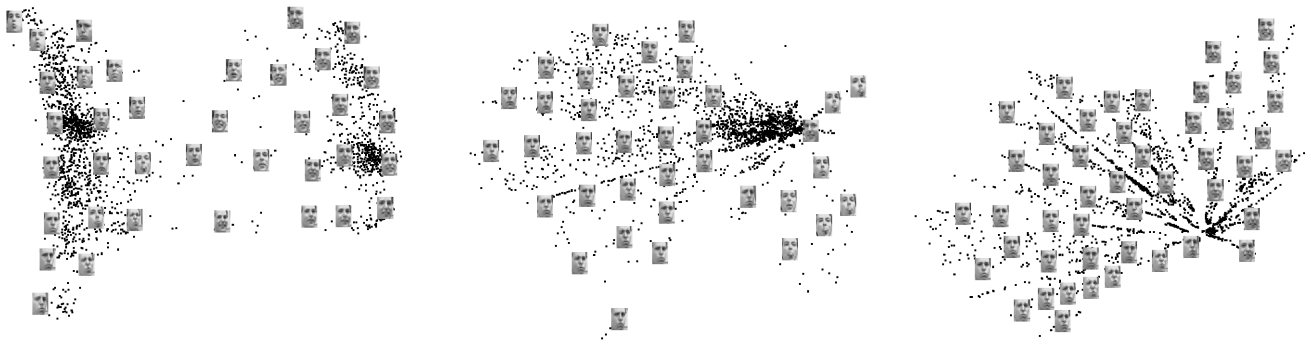
| Method | ER | Our Method | ER |
|---|---|---|---|
| PCA | 20.6% (150) | dGAE-PCA | 3.5% |
| LDA | 5.7% (67) | dGAE-LDA | 1.2% |
| ISOMAP | – | dGAE-ISOMAP | 2.5% |
| LLE | – | dGAE-LLE | 3.6% |
| LPP | 4.6%(110) | dGAE-LE | 3.0% |
| MFA | 2.6% (85) | dGAE-MFA | 1.1% |

As can be seen, our methods generally perform much better than their corresponding methods. Two supervised methods dGAE-LDA and dGAE-MFA with the error rate 1.2%, 1.1% achieve the best performances. Except dGAE-LLE, the other four dGAEs all have a lower error rate than dGAE-PCA (deep autoencoder [5]). It proves the importance of modelling the relationship between instances, which verifies the motivation of this paper. In particular, the global method dGAE-ISOMAP (2.5%) performs better than two local methods dGAE-LLE (3.6%) and dGAE-LE (3.0%), which demonstrates that the deep architecture of dGAE has the potential to improve nonlocal techniques. Moreover, the low error rate of dGAE-LDA indicates that it has broken through the limit of LDA that the data of each class should belong to a Gaussian distribution.

### 5.4. Digit Classification

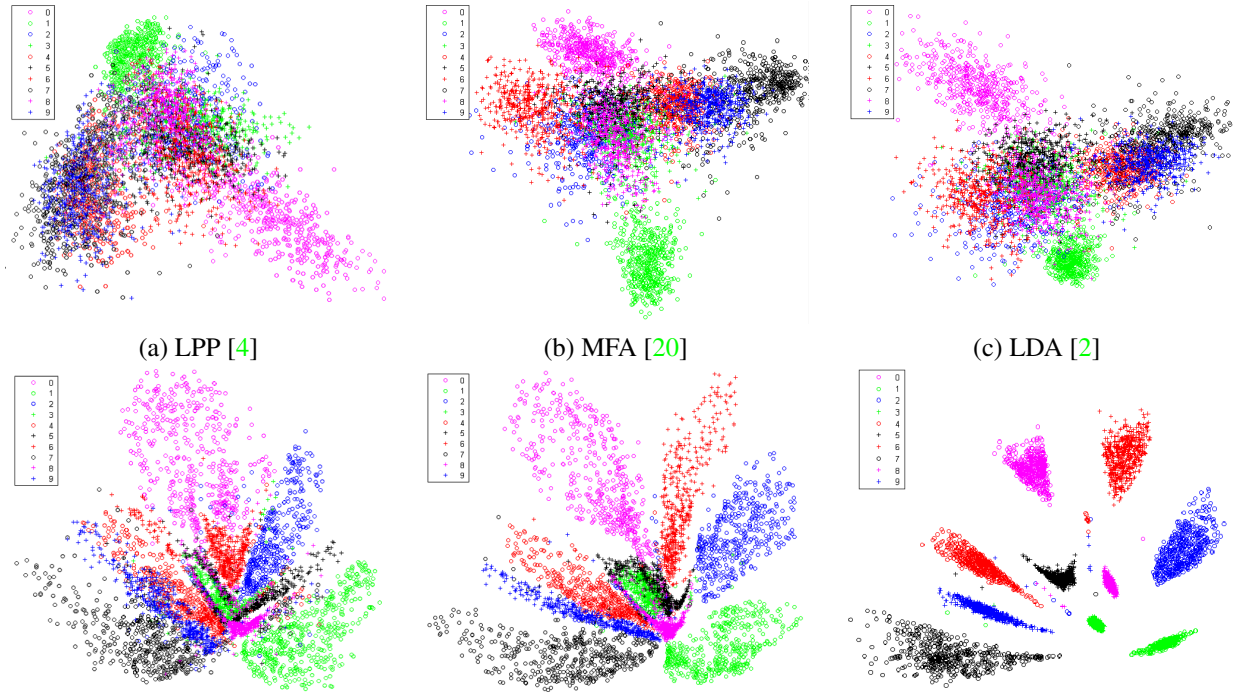
To further evaluate the performance of our GAEs, we perform digit classification on the MNIST dataset.

Different from face recognition experiments, we do not preprocess the digit images and use the 784-dimensional original data as the input. A 784-500-200-30 encoder network is applied to our deep generalized autoencoders in these experiments. We adopt the same testing procedure as face recognition experiments, and select 30 as the reduced dimension for all the dGAEs. Table 3 shows the classification results of the 10 methods on this dataset. As can be seen, our methods still perform much better than the corresponding methods. However, due to the large variations of digit images, the reconstruction weights computed from the original data space are hard to reflect the real relationship between instances. So the three unsupervised dGAEs, namely dGAE-ISOMAP, dGAE-LLE and dGAE-LE , perform comparably to or weaker than dGAE-PCA. The recon-



(a) LPP [4] (b) dGAE-PCA (deep autoencoder [5]) (c) dGAE-LE

Figure 3. 2-D mapping results of F1 face images with LPP [4], dGAE-PCA (deep autoencoder [5]) and dGAE-LE. (a) LPP divides face images into two parts with mouth open or not, and changes the facial expression and pose in an orthogonal way. (b) and (c) The data points of our two methods approximate to form a circular planar around a center point. Along the circular surface, the facial expression and viewing point change smoothly.



(a) LPP [4] (b) MFA [20] (c) LDA [2]

(d) dGAE-LE (e) dGAE-MFA (f) dGAE-LDA

Figure 4. Digit data visualization using 6 methods: two unsupervised methods (a) LPP [4] and (d) dGAE-LE, four supervised methods (b) MFA [20], (c) LDA [2], (e) dGAE-MFA and (f) dGAE-LDA. As can be seen, our three methods obtain much better clusters than their counterparts, respectively. Moreover, by exploiting class label information, the obtained clusters with dGAE-MFA and dGAE-LDA are distinguished better than dGAE-LE. (best viewed in color)

struction weights in dGAE-LDA and dGAE-MFA are only related to class labels, so they achieve the best performance as before.

## 6. Discussion

It is known that the denoising autoencoder [19] is trained to reconstruct an instance from one of its corrupted versions. In the generalized autoencoder, an instance is reconstructed by each instance in a specific set. From the view point of the denoising autoencoder, the GAE uses a set of specific "corrupted versions" of the input, such as its neighbors or the instances of the same class, instead of the versions with Gaussian noise or *masking noise* [19]. In the future, we

Table 3. Performance comparison on the MNIST dataset (ER is short for error rate. The reduced dimensions of the other four methods are in parentheses while those of our methods are all 30.)

| Method | ER | Our Method | ER |
|---|---|---|---|
| PCA | 6.2% (55) | dGAE-PCA | 5.3% |
| LDA | 16.1% (9) | dGAE-LDA | 4.4% |
| ISOMAP | – | dGAE-ISOMAP | 6.4% |
| LLE | – | dGAE-LLE | 5.7% |
| LPP | 7.9%(55) | dGAE-LE | 5.1% |
| MFA | 9.5% (45) | dGAE-MFA | 3.9% |

will compare the generalized autoencoder with the denoising autoencoder on more tasks, such as feature learning and classification, and also consider incorporating of the merits of both sides.

We argue that the generalized autoencoder can embed label information effectively in the supervised or semi-supervised cases. As we know, the traditional autoencoder is an unsupervised method, which cannot utilize class label. Classification restricted Boltzmann machine (ClassRBM) [8] may provide a solution to explicitly modelling class labels by fusing label vectors into the visible layer (1 for the current label and 0 for the others in a label vector). However, training a neural network with such a visible layer needs a large amount of labeled data. Our generalized autoencoder embeds class labels in an implicit way and can decorrelate the reconstructions of labeled data and unlabeled data, which is more flexible and can be used in all the cases.

## 7. Conclusion

This paper proposed a generalized autoencoder to model the relationship between instances, which provided a general neural network framework for dimensionality reduction. Two experiments on face recognition and digit classification demonstrated encouraging results. As can be seen, we did not give a detailed investigation on the optimal architecture of GAE, such as the number of the layers and the nodes of each layer. In the future, we will explore it deeply and design more powerful dimensionality reduction algorithms in this framework.

## References

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in Neural Information Processing Systems*, 2002. 2, 4, 5, 6

[2] R. Duda, P. Hart, and D. Stork. Pattern classification, 2nd edition. *Wiley-Interscience, Hoboken, NJ*, 2000. 2, 4, 6, 7

[3] X. He, D. Cai, S. Yan, and H. Zhang. Neighborhood preserving embedding. *International Conference on Computer Vision*, 2005. 2

[4] X. He and P. Niyogi. Locality preserving projections. *Advances in Neural Information Processing Systems*, 2004. 2, 5, 6, 7

[5] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006. 2, 3, 4, 5, 6, 7

[6] H. Hoffmann. Kernel pca for novelty detection. *Pattern Recognition*, 2007. 2

[7] A. Krizhevsky. Using very deep autoencoders for content-based image retrieval. *European Symposium on Artificial Neural Networks*, 2011. 1

[8] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio. Learning algorithms for the classification restricted boltzmann machine. *Journal of Machine Learning Research, Vol. 13*, 2012. 8

[9] H. Lee, C. Ekanadham, and A. Ng. Sparse deep belief net model for visual area v2. *Advances in Neural Information Processing Systems*, 2008. 1, 2

[10] K. Lee, J. Ho, M. Yang, and D. Kriegman. Video-based face recognition using probablistic appearance manifolds. *IEEE Conference on Computer Vision and Pattern Recognition*, 2003. 1

[11] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 1901. 2, 4, 6

[12] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. *International Conference on Machine Learning*, 2011. 1, 2, 3

[13] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 2000. 2, 4, 5, 6

[14] D. Rumelhart, G. Hinton, and R. Williams. Learning internal representations by error propagation. *Parallel Distributed Processing. Vol 1: Foundations. MIT Press, Cambridge, MA*, 1986. 1

[15] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression (pie) database. *Proc. IEEE Int'l Conf. Automatic Face and Gesture Recognition*, 2002. 5

[16] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000. 2, 4, 5, 6

[17] L. van der Maaten, E. Postma, and H. van den Herik. Dimensionality reduction: A comparative review. *Tilburg University Technical Report*, 2009. 2

[18] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *Journal of Machine Learning Research, Vol. 11*, 2010. 1

[19] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol. Extracting and composing robust features with denoising autoencoders. *International Conference on Machine Learning*, 2008. 1, 2, 7

[20] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Li. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2007. 1, 2, 3, 4, 6, 7