Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction

Wei Wang¹, Yan Huang¹, Yizhou Wang², Liang Wang¹
¹Center for Research on Intelligent Perception and Computing, CRIPAC
Nat’l Lab of Pattern Recognition, Institute of Automation Chinese Academy of Sciences
²Nat’l Eng. Lab for Video Technology, Key Lab. of Machine Perception (MoE), Sch’l of EECS, Peking University, Beijing, China
{wangwei, yhuang, wangliang}@nlpr.ia.ac.cn, yizhou.wang@pku.edu.cn

Abstract

The autoencoder algorithm and its deep version as traditional dimensionality reduction methods have achieved great success via the powerful representability of neural networks. However, they just use each instance to reconstruct itself and ignore to explicitly model the data relation so as to discover the underlying effective manifold structure. In this paper, we propose a dimensionality reduction method by manifold learning, which iteratively explores data relation and use the relation to pursue the manifold structure. The method is realized by a so called “generalized autoencoder” (GAE), 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 \( x_j \) defined on the learned manifold. Hence, the GAE 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 promising performance.

1. Introduction

Real-world data, such as images of faces and digits, usually have a high dimension which leads to the well-known curse of dimensionality in statistical pattern recognition. However, high dimensional data usually lies in a lower dimensional manifold, so-called “intrinsic dimensionality s-space”. Various methods of dimensionality reduction have been proposed to discover the underlying manifold structure, which plays an important role in many tasks, such as pattern classification [9] and information visualization [16].

Principal Component Analysis (PCA) is one of the most popular linear dimensionality reduction techniques [10][17]. It projects the original data onto its principal directions with the maximal variance, and does not consider any data relation.

Linear Discriminant Analysis (LDA) is a supervised method to find a linear subspace, which is optimal for discriminating data from different classes [2]. Marginal Fisher Analysis (MFA) extends LDA by characterizing the intraclass compactness and interclass separability [19]. These two methods use class label information as a weak data relation to seek a low-dimensional separating subspace.

However, the low-dimensional manifold structure of re-
al data is usually very complicated. Generally, just using a simple parametric model, such as PCA, it is not easy to capture such structures. Exploiting data relations has been proved to be a promising means to discover the underlying structure. For example, ISOMAP [15] learns a low-dimensional manifold by retaining the geodesic distance between pairwise data in the original space. In Locally Linear Embedding (LLE) [12], each data point is a linear combination of its neighbors. The linear relation is preserved in the projected low-dimensional space. Laplacian Eigenmaps (LE) [1] purses a low-dimensional manifold by minimizing the pairwise distance in the projected space weighted by the corresponding distance in the original space. They have a common weakness of suffering from the out-of-sample problem. Neighborhood Preserving Embedding (NPE) [3] and Locality Preserving Projection (LPP) [4] are linear approximations to LLE and LE to handle the out-of-sample problem, respectively.

Although these methods exploit local data relation to learn manifold, the relation is fixed and defined on the original high-dimensional space. Such relation may not be valid on the manifold, e.g. the geodesic nearest neighbor on a manifold may not be the Euclidian nearest neighbor in the original space.

The autoencoder algorithm [13] belongs to a special family of dimensionality reduction methods implemented using artificial neural networks. It aims to learn a compressed representation for an input through minimizing its reconstruction error. Fig. 1(a) shows the architecture of an autoencoder. Recently, the autoencoder algorithm and its extensions [8][18][11] demonstrate a promising ability to learn meaningful features from data, which could induce the “intrinsic data structure”. However, these methods just consider self-reconstruction and ignore to explicitly model the data relation.

In this paper, we propose a method of dimension reduction by manifold learning, which extends the traditional autoencoder to iteratively explore data relation and use the relation to pursue the manifold structure. The proposed method is realized by a so called “generalized autoencoder” (GAE). As shown in Fig.1, it differs from the traditional autoencoder in two aspects, (i) the GAE learns a compressed representation $y_i$ for an instance $x_i$, and builds the relation between $x_i$ and other data $\{x_j, x_k\ldots\}$ by using $y_i$ to reconstruct each element in the set, not just $x_i$ itself. (ii) The GAE imposes a relational weight $s_{ij}$ on the reconstruction error $||x_j - x_i'||^2$. Hence, the GAE captures the structure of the data space through minimizing the weighted reconstruction error. When replacing the sigmoid function in the GAE with a linear transformation, we show that the linear GAE and the linear graph embedding [19] have similar objective functions. This indicates that the GAE can also be a general framework for dimensionality reduction by defining different reconstruction sets and weights. Inspired by existing dimensionality reduction ideas, we derive six implementations of GAE including both supervised and unsupervised models. Furthermore, we propose a deep GAE (dGAE), which extends GAE to multiple layers.

The rest of the paper is organized as follows. In Section 2, we first introduce the formulation of the generalized autoencoder and its connection to graph embedding, then propose the generalized autoencoder as a general framework of dimensionality reduction and derive its six implementations. In Section 2.3, we illustrate deep generalized autoencoder as a multilayer network of GAE. The experimental results on three datasets are presented in Section 3. Finally, we discuss the proposed method and conclude the paper in Section 4.

2. Dimensionality Reduction by Manifold Learning

2.1. The method

Algorithm 1 shows the proposed method. (1) We initialize the data relation by computing the data pairwise similarity/distance in the original high dimension space, then determine a “relation/reconstruction” set $\Omega_i$ for each data point $x_i$. For example, the set can be composed of $k$-nearest neighbors. (2) Local manifold structure around each data is learned from its reconstruction set using the proposed “generalized autoencoder” (GAE) introduced below. (3) On the learned manifold, data relation is updated according to the pairwise similarity/distance defined on the hidden representation derived from the GAE. Step (2) and (3) are iterated until convergence or the maximum iteration number being reached.

2.2. 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 $y_i \in \mathcal{R}^{d_y}$ by a function $g()$,

$$y_i = g(W x_i)$$

(1)

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

The decoder reconstructs $x_i' \in \mathcal{R}^{d_x}$ from the hidden representation $y_i$

$$x_i' = f(W' y_i)$$

(2)

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

To model the relation between data, the decoder reconstructs a set of instances indexed by \( \Omega_i = \{ j, k, \ldots \} \) with specific weights \( S_i = \{ s_{ij}, s_{ik}, \ldots \} \) for \( x_i \). The weighted reconstruction error is

\[
e_i(W, W') = \sum_{j \in \Omega_i} s_{ij} L(x_j, x'_i) \tag{3}
\]

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-entropy loss \( L(x_j, x'_i) = -\sum_{q=1}^{d_y} x_j^{(q)} \log(x_i^{(q)}) + (1 - x_j^{(q)}) \log(1 - x_i^{(q)}) \) for binary reconstruction [5].

The total reconstruction error \( E \) of \( n \) samples is

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

The generalized autoencoder learns the parameters \( (W, W') \) via minimizing \( E \).

### 2.2.1 Connection to Graph Embedding

Graph embedding [19] is known to be a general framework for dimensionality reduction, of which each data is represented as a graph node in a low-dimensional vector, the edges preserve similarities between the data pairs. The similarities characterize certain statistical or structural properties of data distribution. The formulation of graph embedding is

\[
y^* = \arg \min_{y^TWy = c} \sum_{i,j} s_{ij} ||y_i - y_j||^2, \tag{5}
\]

where \( y \) is the low-dimensional representation. \( s_{ij} \) is the similarity between the vertex \( i \) and \( j \), usually, \( s_{ij} = s_{ji}. c \)
is a constant and \( B \) is a constraint matrix to avoid a trivial solution.

The linear graph embedding (LGE) assumes that the low-dimensional representation can be obtained by a linear projection \( y = X^T w \), where \( w \) is the projection vector and \( X = [x_1, x_2, \ldots, x_n] \). The objective function of LGE is

\[
w^* = \min_{w^T X = c} \sum_{i,j} s_{ij} ||w^T x_i - w^T x_j||^2 \tag{6}
\]

Similarly, the hidden representation of the generalized autoencoder induces a low-dimensional manifold of data when \( d_y < d_x \), on which the relation/similarity/distance between data pairs is defined through one reconstructing the others. From Eqn.2,1,4, the total reconstruction error \( E \) is

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

In the linear reconstruction case, if \( W' = W^T \) (like [11]) and assuming only one hidden node in the network, i.e. \( d_y = 1 \), \( W \) degenerates to a column vector \( w \), the objective function Eqn. 7 becomes

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

Let \( w^T w = c \) and \( y_i = w^T x_i \), Eqn. 8 becomes

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

Compared with the linear graph embedding (LGE) Eqn. 6, we can see that the generalized autoencoder (GAE) has an additional term which controls different tuning behaviors over the hidden representation \( y \) by varying \( c \). If \( c = 2 \), the GAE has the similar objective function to LGE. If \( c > 2 \), this term prevents \( y \) being too large, even if the norm of \( w \) could be large. If \( c < 2 \), this term encourages \( y \) to be large enough when \( w \) is small.

The linear version of the above models just illustrates the connection between the generalized autoencoder and the graph embedding. In real applications, \( g() \) in the generalized autoencoder is usually preferred to be nonlinear due to its powerful representability.

### 2.2.2 Implementation of the GAE Variants

As can be seen from the above analysis, the generalized autoencoder can also be a general framework for dimensionality reduction through defining different reconstruction sets and weights. In the following, we derive six implementations of the GAE inspired by previous dimensionality reduction ideas, namely PCA [10], LDA [2], ISOMAP [15], LLE [12], LE [1] and MFA [19].

\[^1\) Original constraint \( i \neq j \) is not used here, because \( i = j \) does not violate the objective function}
Table 1. Six implementations of the generalized autoencoders inspired by PCA [10], LDA [2], ISOMAP [15], LLE [12], LE [1] and MFA [19]

<table>
<thead>
<tr>
<th>Method</th>
<th>Reconstruction Set</th>
<th>Reconstruction Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAE-PCA</td>
<td>( j \equiv i )</td>
<td>( s_{ij} = 1 )</td>
</tr>
<tr>
<td>GAE-LDA</td>
<td>( j \in \Omega_{c_i} )</td>
<td>( s_{ij} = \frac{1}{n_{c_i}} )</td>
</tr>
<tr>
<td>GAE-ISOMAP</td>
<td>( j : x_j \in X )</td>
<td>( s_{ij} = S = \frac{-HAH}{2} )</td>
</tr>
<tr>
<td>GAE-LLE</td>
<td>( j \in N_k(i), )</td>
<td>( s_{ij} = (M + M^T - M^T M)_{ij} ) if ( i \neq j ); ( 0 ) otherwise</td>
</tr>
<tr>
<td>GAE-LE</td>
<td>( j \in N_k(i) )</td>
<td>( s_{ij} = \exp{-</td>
</tr>
<tr>
<td>GAE-MFA</td>
<td>( j \in \Omega_{c_i}(c_i) )</td>
<td>( s_{ij} = 1 )</td>
</tr>
</tbody>
</table>

GAE-PCA: an input \( x_i \) is used to reconstruct itself with unit weight, \( i.e. \Omega_i = \{i\}, s_{ii} = 1 \). Given the constraint \( w^T w = 1 \), Eqn. 8 can be rewritten as

\[
w^* = \arg \max_{w_i = 1} \sum_i w_i^T x_i x_i^T w
\]

which is the formulation of traditional PCA with zero mean [2]. The neural network implementation of GAE-PCA is exactly the traditional autoencoder [3], which is a special case of the proposed GAE.

GAE-LDA: an input \( x_i \) is used to reconstruct all the data from the same class as \( x_i \), \( i.e. j \in \Omega_{c_i} \), where \( \Omega_{c_i} \) is the index set of Class \( c_i \), which \( x_i \) belongs to. The weight \( s_{ij} \) is inversely proportional to the sample size \( n_{c_i} \) of class \( c_i \), \( s_{ij} = \frac{1}{n_{c_i}} \). The one-dimensional linear GAE-LDA follows Eqn. 9. It should be noted that the GAE-LDA does not need to satisfy the following two constraints as the traditional LDA: (1) data in each class follow Gaussian distribution. (2) The projection dimension is lower than the number of classes.

GAE-ISOMAP: an input \( x_i \) is used to reconstruct all the data \( X \). Let \( D_G \) be the geodesic distances of all the data pairs [15]. The weight matrix \( S = -HAH/2 \), where \( H = I - \frac{1}{n} ee^T \), \( e \) is the \( n \)-dimensional 1-vector. \( A_{ij} = D_G^2(i,j) \) [19]. The one-dimensional linear GAE-ISOMAP follows Eqn. 9.

GAE-LLE: an input \( x_i \) is used to reconstruct its \( k \)-nearest neighbors and some other inputs’ \( k \)-nearest neighbors which \( x_i \) belongs to, \( i.e. j \in \{N_k(i), N_k(m), m\} \) 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 \( x_i \). Let \( M \) be the local reconstruction coefficient matrix, and \( \sum_{j \in N_k(i)} M_{ij} = 1, M_{ij} = 0 \) if \( j \notin N_k(i) \) [12]. The weight matrix \( S_{ij} = (M + M^T - M^T M)_{ij} \) if \( i \neq j \); \( 0 \) otherwise [19]. The one-dimensional linear GAE-LLE follows Eqn. 9.

GAE-LE: an input \( x_i \) is used to reconstruct its \( k \) nearest neighbors, \( i.e. j \in N_k(i) \). The reconstruction weight 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 linear GAE-LE follows Eqn. 9.

GAE-MFA: an input \( x_i \) is used to reconstruct its \( k_1 \)-nearest neighbors of the same class, and its \( k_2 \)-nearest neighbors of other classes, \( i.e. j \in \Omega_{c_i}^{k_1}, s_{ij} = 1 \) and if \( j \in \Omega_{c_i}^{k_2}, s_{ij} = -1 \), where \( \Omega_{c_i}^{k_1} \) is the index set of \( x_i \)’s \( k_1 \)-nearest neighbors in Class \( c_i \) and \( \Omega_{c_i}^{k_2}(c_i) \) is the index set of \( x_i \)’s \( k_2 \)-nearest neighbors in all the other classes. Similar to MFA, the positive reconstruction characterizes the “intraclass compactness” while the negative reconstruction characterizes the “interclass separability”.

We give a summary of the reconstruction sets and weights of the 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. For example, when introducing class labels to GAE-LE (or GAE-LLE) and choosing the \( k \)-nearest neighbors of the same class as \( x_i \), we can obtain a supervised GAE-LE (or supervised GAE-LLE).

2.3. Deep Generalized Autoencoder

To handle more complex datasets, we extend the generalized autoencoder to a multilayer network called deep generalized autoencoder (dGAE). It contains a multilayer encoder network to transform the high-dimensional input data to a low-dimensional hidden representation and a multilayer decoder network to reconstruct the data. The structures of the two networks are symmetric with respect to the low-dimensional hidden representation layer. For example, Fig. 2(a) shows a five-layer dGAE on a face dataset used in Section 3.3. 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 of 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. The first hidden layer and the input layer are modeled as a Gaussian restrict-
ed 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. The first and second hidden layers are combined to form another GRBM as shown in Fig. 2(b). These two GRBMs can be trained efficiently using Contrastive Divergence method [5].

After pretraining, we obtain the weights \( \{ W_i \}_{i=1,2} \) for both the encoder and the decoder. We 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 through reconstructing the other faces in the same class.

3. Experimental Results

In this section, we discuss two applications of the GAE, one for face recognition and the other for digit classification. First, we show the effectiveness of manifold learning by the GAE and its extensions.

3.1. Data Sets

Experiments are conducted on three datasets. (1) The face dataset F1 contains 1,965 grayscale face images from the frames of a small video. It is also used in [12]. The size of each image is 20 × 28 pixels. (2) The CMU PIE face database [14] contains 68 subjects in 41,368 face images. The size of each image is 32 × 32 pixels. These face images are captured under varying poses, illumination and expression. (3) The MNIST dataset contains 60,000 grayscale images of handwritten digits. The size of each digit image is 28 × 28 pixels. Considering that computing the reconstruction weights of some of the six GAE implementations, e.g. the geodesic distance of the GAE-ISOMAP, is time consuming when the dataset is large, we randomly select 10,000 images to use in our experiments. For each digit, 500 images are for training and the other 500 are used for testing.

3.2. Manifold Learning

Generally, high-dimensional real data is considered to lie in a low-dimensional space, such as face images and handwritten digit images. Similar to the previous methods [15][1][12], the GAE provides an effective means to discover the nonlinear manifold structure.

Firstly, we explore the ability of the GAE in local manifold structure discovery by checking the changes of the \( k \)-nearest neighbors of a data during the iterative learning process. Fig.3 shows the changes of a face’s 18 nearest neighbors during the first 20 iterations of our unsupervised dGAE-LE learning on the CMU PIE dataset. Each row shows the 18 nearest neighbors of the input data during in one iteration of learning. The faces in the colored boxes belong to other persons. The color indicates the identity of the persons. We can see that along the iterative learning, the nearest neighbors of a data converges to its own class.

In addition, we compare the manifold learning results of different models, i.e., LPP [4], dGAE-PCA and dGAE-LE, on the F1 dataset. For visualization, we map the face images into a two-dimensional manifold space learned from dGAE-PCA and dGAE-LE. dGAE-PCA is actually the deep autoencoder [5]. We choose the unsupervised form of dGAE-LE to make a compar-
These models. Both of our methods consist of an encoder with layers of size $(20 \times 28)-200-2$ and a symmetric decoder. The 2D manifolds are shown in Fig. 5. The representative faces are shown next to the data points. As can be seen, the distributions of the data points of our two methods appear radial patterns. Along the radial axes and angular dimension, the facial expression and the pose change smoothly.

We visualize the 2D data manifolds of $0 \sim 9$ digit images of the MNIST dataset learned from two unsupervised methods, (i.e. the LPP [4], dGAE-LE) and four supervised methods (i.e. MFA [19], LDA [2], dGAE-MFA, dGAE-LDA). Fig. 6 shows the results. The data points of the 10 digits are shown with different colors and symbols. Our three methods consist of an encoder with layers of size $(28 \times 28)-1000-500-250-2$ and a symmetric decoder. As can be seen, the data points of different digits overlap seriously derived from LPP, MFA and LDA. Whereas, data points form more distinctive clusters using the three proposed methods. Moreover, by employing class labels, the clusters from dGAE-MFA, dGAE-LDA are more distinguishable than dGAE-LE.

### 3.3. Application to Face Recognition

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

Similar to the experiment setting in [4], 170 face images of each individual are used in the experiments, 85 for training and the other 85 for testing. All the images are first projected to a PCA subspace for noise reduction. We retain 98% of the energy in the denoising step and use 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 GAEs, the low-dimensional representations are computed for all the face images. Then, we apply the nearest-neighbor classifier on the learned low-dimensional space to classify the testing data and compute the error rates.

Table 2 shows the recognition results of 14 models on this dataset, namely, (kernel) PCA [10], (kernel) LDA [2], (kernel) LPP [4], (kernel) MFA [19] and our six dGAEs. Due to the out-of-sample problem, we cannot give the results of ISOMAP, LLE and LE. However, considering the LPP as a popular linear approximation to the LE, we present the result of the supervised LPP as [4]. Correspondingly, we use the supervised version of dGAE-LE for comparison. For the kernel methods, we present the best performance from Gaussian kernel, polynomial kernel and polyplus kernel. For the other four non-kernel methods, we select the manifold dimensions with the best performance, which is shown in parentheses in Table 2. We select 100 as the manifold dimension for all our dGAEs.

As can be seen, our methods generally perform much better than their counterparts. Three supervised methods dGAE-LDA, dGAE-LE and dGAE-MFA with the error rate 1.2%, 1.1% and 1.1% achieve the best performance. Except for the LPP, MFA and LDA - the data of each class follow a Gaussian distribution - is not longer a necessity.
We use the 784-dimensional raw data as the input. In our deep GAEs, the encoder layers are of the size 784-500-200-30. We adopt the same testing procedure as the face recognition experiments, and the manifold dimension is set to 30 for all the dGAEs.

Table 3 shows the classification results of the 14 methods. The baseline uses the nearest-neighbor classifier on the raw data, and its error rate is 6.5%. As can be seen, on this dataset, (1) our methods still outperform the counterparts. (2) our methods all perform better than the baseline, but not all the other 8 methods. This demonstrates that the GAE may discover more reasonable manifold structure by iteratively exploring the data relation. (3) From the error rates of kernel PCA (8.5%) and PCA (6.2%), it can be seen that the kernel extensions may not always improve the original methods, and finding a suitable kernel function is often the key issue. However, due to the multilayer neural network architecture, the proposed deep GAE has the universal approximation property [6], which can capture more complicated data structure.

4. Discussion and Conclusion

It is known that the denoising autoencoder [18] is trained to reconstruct a data from one of its corrupted versions.
Table 3. Performance comparison on the MNIST dataset. ER is short for “error rate”. The reduced dimensions are in the parentheses. Our models use 30 dimensions. pp is short for “polyplus” kernel.

<table>
<thead>
<tr>
<th>Method</th>
<th>ER</th>
<th>Our Model</th>
<th>ER</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>6.2% (55)</td>
<td>dGAE-PCA</td>
<td>5.3%</td>
</tr>
<tr>
<td>Kernel PCA</td>
<td>8.5% (pp)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDA</td>
<td>16.1% (9)</td>
<td>dGAE-LDA</td>
<td>4.4%</td>
</tr>
<tr>
<td>Kernel LDA</td>
<td>4.6% (pp)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISOMAP</td>
<td>–</td>
<td>dGAE-ISOMAP</td>
<td>6.4%</td>
</tr>
<tr>
<td>LLE</td>
<td>–</td>
<td>dGAE-LLE</td>
<td>5.7%</td>
</tr>
<tr>
<td>LPP</td>
<td>7.9% (55)</td>
<td>dGAE-LLE</td>
<td>4.3%</td>
</tr>
<tr>
<td>Kernel LPP</td>
<td>4.9% (pp)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MFA</td>
<td>9.5% (45)</td>
<td>dGAE-MFA</td>
<td>3.9%</td>
</tr>
<tr>
<td>Kernel MFA</td>
<td>6.8% (pp)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In the generalized autoencoder, a data is reconstructed by those in a specific set. From the denoising autoencoder viewpoint, the GAE uses a set of “corrupted versions” for the reconstruction, such as its neighbors or the instances of the same class, instead of the versions with Gaussian noise or masking noise [18]. In the future, we 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.

As we know, the traditional autoencoder is an unsupervised method, which do not utilize class label. Classification restricted Boltzmann machine (ClassRBM) [7] may provide a solution to explicitly modeling class labels by fusing label vectors into the visible layer. However, training a neural network with such a visible layer needs a large amount of labeled data. We argue that the generalized autoencoder is able to exploit label information in learning more flexibly via using/selecting different data into the reconstruction sets. For example, we can build a reconstruction set according to the labels or just use nearest-neighbors by ignoring the labels, or even combining both strategies.

Acknowledgments

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

References