Generalized Autoencoder: A Neural Network Framework for Dimensionality Reduction

1Wei Wang  1Yan Huang  2Yizhou Wang  1Liang Wang

1Center for Research on Intelligent Perception and Computing, CRIPAC
Nat'l Lab of Pattern Recognition, CASIA
2Nat'l Engineering Lab for Video Technology
Key Lab. of Machine Perception (MoE), PKU, Beijing, China
Outline

- Motivation
- Related work
- Generalized autoencoder
- Experimental results
- Discussion and Conclusion
The autoencoder algorithm and its regularized variants are widely used in dimensionality reduction and manifold learning.

The autoencoder algorithm and its regularized variants are widely used in dimensionality reduction and manifold learning.
Motivation

- The autoencoder algorithm and its regularized variants are widely used in dimensionality reduction and manifold learning.

(b) P. Vincent et al. Extracting and composing robust features with denoising autoencoders. ICML2008
Motivation

- Modeling data relation is missing, which is very important in dimensionality reduction and manifold learning

(a) Isomap  
(b) Locally linear embedding  
(c) Laplacian eigenmaps
Motivation

- Modeling data relation is missing, which is very important in dimensionality reduction and manifold learning.

(a) Isomap  (b) Locally linear embedding  (c) Laplacian eigenmaps

How to model data relation in a neural network?
Related Work

- Modeling data relation with a siamese network

(a) Siamese network
(b) G.W. Taylor et al. Learning invariance through imitation. CVPR2011
(c) Y. Huang et al. A general nonlinear embedding framework based on deep neural network. ICPR2014
Related Work

- Modeling data relation with a siamese network

How to model data relation in an autoencoder from a viewpoint of reconstruction?
Generalized Autoencoder (GAE)

Three key ingredients

\[ \| x_i - x'_i \|^2 \]

\[ y_i \]

\[ W \]

\[ W' \]
Generalized Autoencoder (GAE)

Three key ingredients

- Each instance $x_i$ is used to reconstruct a set of instances $\{x_j\}$ rather than itself.
Generalized Autoencoder (GAE)

Three key ingredients

- Each instance $x_i$ is used to reconstruct a set of instances $\{x_j\}$ rather than itself.
- Each reconstruction error is weighted by a relational function of $x_i$ and $x_j$. 

![Diagram of generalized autoencoder]
Generalized Autoencoder (GAE)

Three key ingredients

- Each instance $x_i$ is used to reconstruct a set of instances $\{x_j\}$ rather than itself.
- Each reconstruction error is weighted by a relational function of $x_i$ and $x_j$.
- Considering that fixed data relation defined on the original high-dimensional space may not be valid on the manifold, the data relation is iteratively updated.
Two View Angles to GAE

- Preserve the relationship between reconstruction $x_i'$ and other raw input $x_j$, $x_k$, ...
Two View Angles to GAE

- Preserve the relationship between reconstruction $x_i'$ and other raw input $x_j, x_k, \ldots$

- Preserve the relationship between raw input $x_i$ and other reconstructions $x_j', x_k', \ldots$
The Formulation of GAE

- **Objective function**

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

where \( s_{ij} \) is reconstruction weight, \( \Omega_i \) is the reconstruction set, \( L(x_j, x'_i) \) is the reconstruction error

for binary reconstruction

\[
L(x_j, x'_i) = -\sum_{q=1}^{d_x} x_j^{(q)} \log(x_i^{(q)}) + (1 - x_j^{(q)}) \log(1 - x_i^{(q)})
\]

for linear reconstruction

\[
L(x_j, x'_i) = \|x_j - x'_i\|^2
\]
Iterative learning Procedure for GAE

**Input:** training set $\{x_i\}_1^n$

Parameters: $\Theta = (W, W')$

Notation: $\Omega_i$: reconstruction set for $x_i$

$S_i$: the set of reconstruction weight for $x_i$

$\{y_i\}_1^n$: hidden representation
Iterative learning Procedure for GAE

**Input:** training set \( \{x_i\}_1^n \)

Parameters: \( \Theta = (W, W') \)

Notation: \( \Omega_i \): reconstruction set for \( x_i \)

\( S_i \): the set of reconstruction weight for \( x_i \)

\( \{y_i\}_1^n \): hidden representation

1. Compute the reconstruction weights \( S_i \) from \( \{x_i\}_1^n \) and determine the reconstruction set \( \Omega_i \), e.g. by \( k \)-nearest neighbor
Iterative learning Procedure for GAE

**Input**: training set \( \{x_i\}_1^n \)

Parameters: \( \Theta = (W, W') \)

Notation: \( \Omega_i \): reconstruction set for \( x_i \)

\( S_i \): the set of reconstruction weight for \( x_i \)

\( \{y_i\}_1^n \): hidden representation

1. Compute the reconstruction weights \( S_i \) from \( \{x_i\}_1^n \) and determine the reconstruction set \( \Omega_i \), e.g. by \( k \)-nearest neighbor

2. Minimize \( E \) in Eqn. 4 using the stochastic gradient descent and update \( \Theta \) for \( t \) steps
Iterative learning Procedure for GAE

**Input:** training set \( \{x_i\}^n_1 \)

Parameters: \( \Theta = (W, W') \)

Notation: \( \Omega_i \): reconstruction set for \( x_i \)

\( S_i \): the set of reconstruction weight for \( x_i \)

\( \{y_i\}^n_1 \): hidden representation

1. Compute the reconstruction weights \( S_i \) from \( \{x_i\}^n_1 \) and determine the reconstruction set \( \Omega_i \), e.g. by \( k \)-nearest neighbor

2. Minimize \( E \) in Eqn. 4 using the stochastic gradient descent and update \( \Theta \) for \( t \) steps

3. Compute the hidden representation \( \{y_i\}^n_1 \), and update \( S_i \) and \( \Omega_i \) from \( \{y_i\}^n_1 \).
Iterative learning Procedure for GAE

**Input:** training set \( \{x_i\}_1^n \)
Parameters: \( \Theta = (W, W') \)
Notation: \( \Omega_i \): reconstruction set for \( x_i \)
\( S_i \): the set of reconstruction weight for \( x_i \)
\( \{y_i\}_1^n \): hidden representation

1. Compute the reconstruction weights \( S_i \) from \( \{x_i\}_1^n \) and determine the reconstruction set \( \Omega_i \), e.g. by \( k \)-nearest neighbor
2. Minimize \( E \) in Eqn. 4 using the stochastic gradient descent and update \( \Theta \) for \( t \) steps
3. Compute the hidden representation \( \{y_i\}_1^n \), and update \( S_i \) and \( \Omega_i \) from \( \{y_i\}_1^n \).
4. Repeat step 2 and 3 until convergence.
Connection to Graph Embedding

- The linearization extension of graph embedding is as follows

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

- The linearization extension of GAE is as follows

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

where \( y_i = w^T x_i \) and \( w^T w = c \)
Connection to Graph Embedding

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

- The additional term \( \left( \frac{c}{2} - 1 \right) y_i^2 \) controls different tuning behaviors over the hidden representation by varying \( c \)
  - When \( c = 2 \), GAE has the similar objective function to graph embedding
  - When \( c > 2 \), this term prevents the hidden representation from being too large, even if the norm of \( w \) could be large
  - When \( c < 2 \), this term encourages the hidden representation to be large enough when \( w \) is small
GAE Variants

- Six implementations of GAE inspired by PCA, LDA, ISOMAP, LLE, LE, MFA
  - define different reconstruction sets and weights
  - preserve various kinds of data relation

<table>
<thead>
<tr>
<th>Method</th>
<th>Reconstruction Set</th>
<th>Reconstruction Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAE-PCA</td>
<td>$j = i$</td>
<td>$s_{ij} = 1$</td>
</tr>
<tr>
<td>GAE-LDA</td>
<td>$j \in \Omega_{e_i}$</td>
<td>$s_{ij} = \frac{1}{n_{e_i}}$</td>
</tr>
<tr>
<td>GAE-ISOMAP</td>
<td>$j : x_j \in X$</td>
<td>$s_{ij} \in S = -HH^T/2$</td>
</tr>
<tr>
<td>GAE-LLE</td>
<td>$j \in N_k(i)$, $j \in (N_k(m) \cup m), j \neq i$ if $\forall m, i \in N_k(m)$</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></td>
</tr>
<tr>
<td>GAE-MFA</td>
<td>$j \in \Omega_{k_1}(c_i)$, $j \in \Omega_{k_2}(\bar{c_i})$</td>
<td>$s_{ij} = \exp{-</td>
</tr>
</tbody>
</table>

$s_{ij}$ is the reconstruction weight between samples $i$ and $j$. The weight is determined by the specific reconstruction set and weight definition for each method.
Deep Generalized Autoencoder

- Stack multi-layers to form a deep GAE to handle more complex data
Experimental Results

- A face dataset from a small video (F1)
  - 1965 grayscale face images
  - 20×28 pixels

- CMU PIE face dataset
  - 68 subjects in 41,368 face images
  - 32×32 pixels

- MNIST handwritten digits
  - Only 5000 training images and 5000 testing images used for computational cost consideration
  - 28×28 pixels
Manifold Learning

- (a) change of a face’s 18 nearest neighbors during the first 20 iterations of dGAE-LE on CMU PIE dataset
- (b) change of impurity during the iteration learning

(a) Change of nearest neighbors

(b) Change of purity
Manifold Learning

- 2D visualization of the face image manifold on the F1
  - Radial patterns: along the radial axes and angular dimension, the facial expression and the pose change smoothly
Manifold Learning

- 2D visualization of the learned digit image manifold on the MNIST

(a) LPP [4]  
(d) GAE-LE
Manifold Learning

- 2D visualization of the learned digit image manifold on the MNIST

(b) MFA [20]

(e) GAE-MFA
Manifold Learning

- 2D visualization of the learned digit image manifold on the MNIST

(c) LDA [2]  
(f) GAE-LDA
Manifold Learning

- 2D visualization of the learned digit image manifold on the MNIST

Compared with other methods, the 2D data points from our GAE variants are more distinctive.
Face Recognition

- Face recognition on the CMU PIE
  - 85 training images and 85 testing images for each individual
  - 157-d features after PCA preprocessing
  - A 157-200-100 encoder network is used

<table>
<thead>
<tr>
<th>Method</th>
<th>ER</th>
<th>Our Model</th>
<th>ER</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>20.6%</td>
<td>dGAE-PCA</td>
<td>3.5%</td>
</tr>
<tr>
<td>Kernel PCA</td>
<td>8.1%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDA</td>
<td>5.7%</td>
<td>dGAE-LDA</td>
<td>1.2%</td>
</tr>
<tr>
<td>Kernel LDA</td>
<td>1.6%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISOMAP</td>
<td>–</td>
<td>dGAE-ISOMAP</td>
<td>2.5%</td>
</tr>
<tr>
<td>LLE</td>
<td>–</td>
<td>dGAE-LLE</td>
<td>3.6%</td>
</tr>
<tr>
<td>LPP</td>
<td>4.6%</td>
<td>dGAE-LE</td>
<td>1.1%</td>
</tr>
<tr>
<td>Kernel LPP</td>
<td>1.7%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MFA</td>
<td>2.6%</td>
<td>dGAE-MFA</td>
<td>1.1%</td>
</tr>
<tr>
<td>Kernel MFA</td>
<td>2.1%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Digit Classification

- Digit classification on the MNIST
  - 500 training images and 500 testing images for each digit
  - No PCA preprocessing
  - A 784-500-200-30 encoder network is used

<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-LE</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>
Discussion and Conclusion

- The relationship between GAE and denoising autoencoder
  - Denoising autoencoder can be a special case of GAE by defining the reconstruction set as the corrupted version of the input and the reconstruction weight as 1

- Easy to devise new algorithms
  - It is more flexible to construct the reconstruction set by containing the instances with various relationships, such as the same class, knn

- Reduce the computational cost in the large-scale dataset
  - Adopt sampling strategy to construct the reconstruction set
Thanks! (Q&A)