Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Representation learning via a semi-supervised stacked distance autoencoder for image classification*

Liang HOU, Xiao-yi LUO, Zi-yang WANG, Jun LIANG†‡

College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China

†E-mail: jliang@zju.edu.cn

Received Feb. 28, 2019; Revision accepted Sept. 16, 2019; Crosschecked June 10, 2020

Abstract: Image classification is an important application of deep learning. In a typical classification task, the classification accuracy is strongly related to the features that are extracted via deep learning methods. An autoencoder is a special type of neural network, often used for dimensionality reduction and feature extraction. The proposed method is based on the traditional autoencoder, incorporating the "distance" information between samples from different categories. The model is called a semi-supervised distance autoencoder. Each layer is first pre-trained in an unsupervised manner. In the subsequent supervised training, the optimized parameters are set as the initial values. To obtain more suitable features, we use a stacked model to replace the basic autoencoder structure with a single hidden layer. A series of experiments are carried out to test the performance of different models on several datasets, including the MNIST dataset, street view house numbers (SVHN) dataset, German traffic sign recognition benchmark (GTSRB), and CIFAR-10 dataset. The proposed semi-supervised distance autoencoder method is compared with the traditional autoencoder, sparse autoencoder, and supervised autoencoder. Experimental results verify the effectiveness of the proposed model.

Key words: Autoencoder; Image classification; Semi-supervised learning; Neural network
https://doi.org/10.1631/FITEE.1900116 **CLC number:** TP391.9

1 Introduction

Image classification has become a hotspot in machine learning over the last few years. In a classification task, the key issues are dimensionality reduction, extraction of high-level features, and removal of redundant information (Haralick et al., 1973; Sun YN et al., 2017). He et al. (2018) and Peng et al. (2018) separately proposed different supervised approaches for fast fine-grained image classification. Such methods aim at addressing time- and labor-consuming problems. He et al. (2019) proposed a multi-scale and multi-granularity deep reinforcement learning approach to discriminate similar subcategories belonging to the same superclass.

As the data dimension increases (Bianco et al., 2018; Feng and Duarte, 2018; Rahmani et al., 2018), such as with video and image data, the dimensionality reduction of the data becomes a necessity. When applying machine learning models, high-dimensional data may lead to many problems. The most serious one is overfitting, i.e., high accuracy in the training set but a low score in the test set. In addition, the extracted low-dimensional data can effectively reduce the computational complexity and help better understand the data (Bengio et al., 2013; Meng LH et al., 2018).

The autoencoder (AE) (Kingma and Welling, 2016; Santana et al., 2016) is a typical neural network structure, learning features from the input data, in a process called encoding. Then, the original input data

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (Nos. U1664264 and U1509203)
ORCID: Liang HOU, https://orcid.org/0000-0003-0887-627X; Jun LIANG, https://orcid.org/0000-0003-1115-0824
© Zhejiang University and Springer-Verlag GmbH Germany, part of Springer Nature 2020

can be reconstructed based on the learned features, in a process called decoding. Through the processes of encoding and decoding, the output vector is reconstructed by the least but most representative information (Bengio, 2009; Tolstikhin et al., 2017). The basic AE is an unsupervised learning method (Hinton and Salakhutdinov, 2006; Hinton, 2007; Taherkhani et al., 2018). Usually, the mean squared error (MSE) between the output and input is used as a term of the objective function. The reconstruction error decreases as the training loss decreases (Vincent et al., 2010). Based on the AE proposed by Hinton, many improved models of AEs have been proposed. Deng et al. (2013) proposed a sparse AE (SAE) to create transfer learning based on the features. Zhang et al. (2016) proposed a denoising AE to extract and combine robust features against variable noise. Rifai et al. (2011) added a penalty into the traditional AE and proposed the contractive AE (CAE) for feature extraction. Xu et al. (2016) applied a semi-supervised variational AE (VAE) into the field of text classification and added a conditional long short-term memory model, making the text classification more effective. Wang et al. (2014) proposed a generalized AE (GAE), taking the relevance of the data into consideration by adding a weighted relational function to modify the AE.

If there are a sufficient number of data samples with labels, carrying out supervised learning is the first choice. If there are a limited number of data samples with labels, we have to resort to unsupervised learning. Some AEs, such as the SAE, are unsupervised learning models that extract features from unlabeled data. Some unsupervised methods (Gong et al., 2013; Tang et al., 2015) aim at generating hash codes by capturing a feature when the similarity in the feature space is preserved. Thus, these unsupervised methods retrieve the neighbors according to the Euclidean distance in the feature space, which does not always achieve a good performance. However, collected data samples without labels are universal in the real world, so it is important to make full use of the unlabeled data for supervised learning models.

Metric learning, also called distance metric learning (Du et al., 2018), is a typical classifier in unsupervised learning. DeepID2 is one of the metric learning ideas in the field of face recognition based on the deep learning proposed in Sun Y et al. (2014). A "feature," also called the DeepID2 vector, increases the difference between different classes, extracts features from different faces, and reduces the difference between intra-classes. DeepID2 uses face verification information and face identification information on the same network, both of which are supervised signals. Contrastive loss was introduced into the verification loss in the feature layer, making the distance for the same person's face in the feature space small and the distance for other people's faces large. If we do not consider the label information and introduce it into the loss function, we may lose the intrinsic information between different classes. Based on the above introduction, the contributions of this study are as follows:

1. To leverage the advantages of unsupervised training methods and class information, a semi-supervised learning method is proposed.

2. To obtain deep information, the stacked AE is proposed.

3. Distance information is introduced into the objective function for improving the classification performance.

2 Related work

In this section, we briefly introduce the models used in this study, including the AE, SAE, some improved AEs, and the softmax function for classification.

2.1 Autoencoder

The AE is an unsupervised learning algorithm whose goal is to keep the difference between the input and output as small as possible. The traditional basic AE is a three-layer neural network: an input layer, a representation layer, and an output layer. The diagram of its network structure is shown in Fig. 1. $\boldsymbol{X}=(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)\in\mathbb{R}^{D\times n}$ is an n-dimensional dataset and $\boldsymbol{x}_i$ (i=1, 2, …, n) is the ith D-dimensional data sample. $\boldsymbol{Y}=(\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n)\in\mathbb{R}^{D\times n}$ contains the reconstructed input samples. The entire network consists of two networks called the encoder and decoder, respectively. The representation layer, as the core part of the whole network, expresses the compressed features of the high-dimensional data.

The theory behind the AE is as follows: first, the weights of the encoder and decoder are initialized,

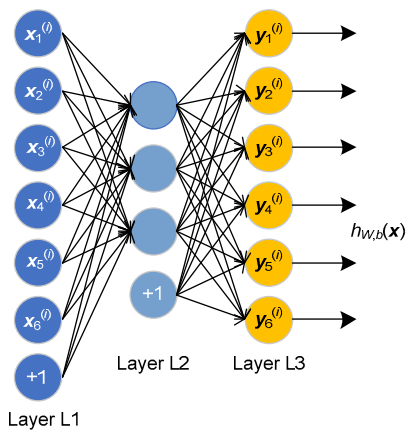


Fig. 1 Automated encoder network structure
"+1' denotes the bias of the layer. $h_{W,b}(\boldsymbol{x})$ is the activation function

and then the AE is trained according to an L_2 penalty and MSE between the original data and the reconstructed data.

The encoding process is as follows:

$$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}_h), \tag{1}$$

$$f(\boldsymbol{x}) = 1/[1+\exp(-\boldsymbol{x})], \tag{2}$$

where $f(\boldsymbol{x})$ is called the activation function, chosen to be the nonlinear sigmoid function, $\boldsymbol{W}$ is the weight matrix of the encoder, and $\boldsymbol{b}_h$ is the bias of the input.

The decoding process is as follows:

$$\boldsymbol{y} = f(\boldsymbol{W}'\boldsymbol{h} + \boldsymbol{b}_v), \tag{3}$$

where $\boldsymbol{W}'$ is the weight matrix of the decoder (in the following experiment, $\boldsymbol{W}'=\boldsymbol{W}^{\mathrm{T}}$, also called the tied weights), $\boldsymbol{b}_v$ is the bias in the output layer, and $\boldsymbol{y}$ obtained here is the reconstruction of the original data.

The reconstruction error between the reconstructed data $\boldsymbol{y}$ and input data $\boldsymbol{x}$ is formulated as

$$J_{\mathrm{MSE}}(\boldsymbol{\theta}) = \sum_{i=1}^{m} \left\| \boldsymbol{x}_i - \boldsymbol{y}_i \right\|^2, \tag{4}$$

where m is the number of samples, $\boldsymbol{x}_i$ is the input vector, $\boldsymbol{y}_i$ is the output vector, and $\boldsymbol{\theta}$ is the set of all parameters in the network.

To prevent overfitting during the training process, a weight regularization term is usually added to the objective function to put limits on the network parameters. The weight regularization term in the following experiments is formulated as

$$J_{\mathrm{w}}(\boldsymbol{\theta}) = \frac{\gamma}{2} \sum_{l=1}^{2} \sum_{i=1}^{s_l} \sum_{j=1}^{s_l+1} \left(w_{ji}^{(l)} \right)^2, \tag{5}$$

where γ determines the value of the weight regularization term in the objective function, l is the number of layers in the network, s_l is the dimensionality of the lth layer, and w_{ji} is the element in the ith row and j^{th} column.

$$J_{\mathrm{cost}}(\boldsymbol{\theta}) = J_{\mathrm{MSE}}(\boldsymbol{\theta}) + J_{\mathrm{w}}(\boldsymbol{\theta}). \tag{6}$$

2.2 Stacked autoencoder

As we know, the power of deep networks is that they can learn more comprehensive representations of raw data layer by layer. Each layer is based on the previous expression, which tends to be more abstract and more suitable for complex tasks such as classification.

A single AE can learn a representation through a three-layer (encoder-representation-decoder) network. In fact, when the training is over, the output layer is of no use. Now, we will simply focus on the expression of features.

During the whole training process of the stacked AE, we use a layer-by-layer pre-training strategy; i.e., the self-encoding transformation between every two layers is trained, and a layer's parameters are taken as the initial values of the next layer. The layer-by-layer approach can stabilize the convergence. Once a stack of encoders has been built, its output representation of the highest level can be used as input to a stand-alone supervised learning algorithm as shown in Fig. 2. The parameters of all layers can then be simultaneously fine-tuned using a gradient procedure such as stochastic gradient descent (Du et al., 2018).

2.3 Sparse autoencoder

As a neural network model based on feature extraction methods, a major advantage of the AE is its simple structure. The SAE is also an unsupervised learning method. In the SAE, the number of nodes in the hidden layer is greater than the number of input nodes, most nodes in the hidden layer are

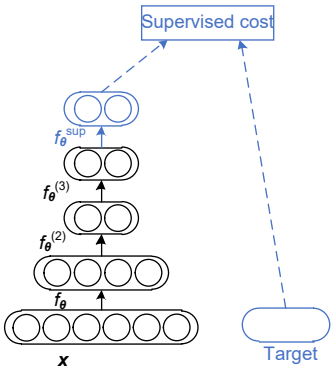


Fig. 2 Fine-tuning of a deep network for classification
After training a stack of encoders as explained in the figure, an output layer is added to the stack. The parameters of the whole system are fine-tuned to minimize the error in predicting the supervised target (e.g., class), by performing gradient descent on a supervised cost

"suppressed," and only a small portion are in an "active" state.

In an AE network structure, if $\boldsymbol{a}_j^2$ represents the activation value of the jth hidden node, then the average activation value of the jth hidden node in the training data $\boldsymbol{x}_i$ can be represented as follows:

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{a}_j^2 \boldsymbol{x}_i, \tag{7}$$

where the sparsity limitation $\hat{\rho}_j = \rho$ is added to make the neuron in the jth hidden layer close to ρ, and ρ indicates the sparsity parameter. To satisfy this limitation, in most cases the activation value of the neuron is close to zero, and there are only a small number of neurons that are not close to zero.

To achieve sparsity, we also need to add a penalty term in the objective function, which encourages the parameters to stay close to $\hat{\rho}_j$.

The penalty term for the SAE is as follows:

$$J_{\text{sparse}}(\boldsymbol{\theta}) = \sum_{j=1}^{s_2}\left[\rho \ln\left(\frac{\rho}{\hat{\rho}_j}\right) + (1-\rho)\ln\left(\frac{1-\rho}{1-\hat{\rho}_j}\right)\right], \tag{8}$$

where s_2 is the number of neurons in the hidden layer. To facilitate writing it, we have

$$\text{KL}(\rho \,\|\, \hat{\rho}_j) = \rho \ln\left(\frac{\rho}{\hat{\rho}_j}\right) + (1-\rho)\ln\left(\frac{1-\rho}{1-\hat{\rho}_j}\right), \tag{9}$$

where $\text{KL}(\rho \,\|\, \hat{\rho}_j)$ is a way to quantify the difference between two probability distributions ρ and $\hat{\rho}_j$, also known as relative entropy. The larger the relative entropy, the greater the difference between the two parts. When $\rho=\hat{\rho}_j$, $\text{KL}(\rho \,\|\, \hat{\rho}_j)=0$; it means that these two distributions are the same. With the increase of the difference between ρ and $\hat{\rho}_j$, $\text{KL}(\rho \,\|\, \hat{\rho}_j)$ increases. Therefore, minimizing this term can bring $\hat{\rho}_j$ closer to ρ.

Combining Eq. (6), the objective function of the SAE is as follows:

$$J_{\text{cost}}(\boldsymbol{\theta}) = J_{\text{MSE}}(\boldsymbol{\theta}) + J_{\text{w}}(\boldsymbol{\theta}) + J_{\text{sparse}}(\boldsymbol{\theta}). \tag{10}$$

2.4 Softmax regression model

The softmax regression model allows multiple classification results, not just two. For a softmax classifier, the probability output function is shown as follows:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) = \begin{bmatrix} p(y^{(i)}=1) \,|\, \boldsymbol{x}^{(i)};\boldsymbol{\theta} \\ p(y^{(i)}=2) \,|\, \boldsymbol{x}^{(i)};\boldsymbol{\theta} \\ \vdots \\ p(y^{(i)}=k) \,|\, \boldsymbol{x}^{(i)};\boldsymbol{\theta} \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} \mathrm{e}^{\boldsymbol{\theta}_j^{\mathrm{T}} \boldsymbol{x}^{(i)}}} \begin{bmatrix} \mathrm{e}^{\boldsymbol{\theta}_1^{\mathrm{T}} \boldsymbol{x}^{(i)}} \\ \mathrm{e}^{\boldsymbol{\theta}_2^{\mathrm{T}} \boldsymbol{x}^{(i)}} \\ \vdots \\ \mathrm{e}^{\boldsymbol{\theta}_k^{\mathrm{T}} \boldsymbol{x}^{(i)}} \end{bmatrix}. \tag{11}$$

These are parameters in the softmax model, including the bias. $\boldsymbol{\theta}_j$ represents the jth row of matrix $\boldsymbol{\theta}$. The objective function can be expressed as

$$J(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{k} 1\{y^{(i)}=j\} \ln \frac{\mathrm{e}^{\boldsymbol{\theta}_j^{\mathrm{T}} \boldsymbol{x}^{(i)}}}{\sum_{l=1}^{k} \mathrm{e}^{\boldsymbol{\theta}_l^{\mathrm{T}} \boldsymbol{x}^{(i)}}} + \frac{\boldsymbol{\lambda}_2}{2}\sum_{i=1}^{k}\sum_{j=1}^{D+1}\theta_{ij}^2, \tag{12}$$

where θ_{ij} is the element in the weight matrix, j is the jth term of the input layer, i is the ith term of the output layer, and $\boldsymbol{\lambda}_2$ is a weight attenuation matrix. The expression of the indicator function is given as follows:

$$1\{y^{(i)}=j\} = \begin{cases} 1, & y^{(i)}=j, \\ 0, & y^{(i)} \neq j. \end{cases} \tag{13}$$

3 Extracting features and the distance autoencoder

In this section, before introducing feature extraction and the distance AE (DAE), an introduction to the supervised AE (Sv-AE) is given (Du et al., 2018). We take this state-of-the-art method as our baseline.

Given training data $\boldsymbol{x}=(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_D)$ and label vector $\boldsymbol{y}=(y_1, y_2, \ldots, y_D)$, where y_i (i=1, 2, …, D) has only one non-zero element (y_{ic}=1 denotes that example $\boldsymbol{x}_i$ belongs to class c), the input is first preprocessed as described above. The obtained input is denoted as $z=(\tilde{x},\tilde{y})$, the encoding process of the Sv-AE can be expressed as follows:

$$\boldsymbol{h}=f(\boldsymbol{W}\tilde{\boldsymbol{x}}+\boldsymbol{U}\tilde{\boldsymbol{y}}+\boldsymbol{b}), \tag{14}$$

and the decoding process is added with a label reconstruction requirement:

$$\begin{cases}\boldsymbol{v}=f(\boldsymbol{W}'\boldsymbol{h}+\boldsymbol{b}'),\\ \boldsymbol{l}=g(\boldsymbol{U}'\boldsymbol{h}+\boldsymbol{c}),\end{cases} \tag{15}$$

where the activation function f and the network parameters ($\boldsymbol{W}$, $\boldsymbol{W}'$, $\boldsymbol{b}$, $\boldsymbol{b}'$) are similar to those described in the traditional AE. The objective function is as follows:

$$J_{\text{Sv-AE}}(\boldsymbol{\theta})=\frac{1}{n}\sum_{\boldsymbol{x}\in D_n}L_1(\boldsymbol{y},\boldsymbol{l})+\lambda\frac{1}{n}\sum_{\boldsymbol{x}\in D_n}L_2(\boldsymbol{x},\boldsymbol{v}), \tag{16}$$

where L_1 and L_2 denote the reconstruction errors of the label and data, respectively. λ is a hyperparameter that controls the trade-off between data reconstruction and label reconstruction. For details, please refer to Du et al. (2018).

3.1 Feature extraction

The traditional AE is an unsupervised learning algorithm, using a backpropagation algorithm to make the output value as close as possible to the input value. An AE consists of two parts, the encoding process and decoding process. For a trained AE, the vector containing the activation value of each layer composed of units $\boldsymbol{a}^{(2)}=(\boldsymbol{a}_1^{(2)}, \boldsymbol{a}_2^{(2)}, \ldots, \boldsymbol{a}_n^{(2)})$ can represent the input vector $\boldsymbol{x}=(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m)$; namely, $\boldsymbol{a}^{(2)}$ is a new feature value of $\boldsymbol{x}$ in the new feature space. Assuming that there are n units in the hidden layer, then there are n new features (namely $\boldsymbol{a}_i^{(2)}$ is n-dimensional). The compression of high-dimensional data into low-dimensional data to acquire features can be linear or nonlinear. A traditional AE is a nonlinear process mapping the original data to the hidden layer. Selecting different activation functions to extract features can satisfy different demands. The compressed data should reflect the characteristics of the original data as much as possible, so that the data features are more expressive, and the performance of the trained classifier will be better.

In the process of encoding, if no restrictions are imposed on the network, it is difficult to obtain features that are more expressive than the original input data. Therefore, we need to apply some constraints to effectively extract useful information and eliminate noise. For example, considering a given dataset $\boldsymbol{X}$ with n samples and a features, the original feature set is denoted as $\boldsymbol{Y}_\text{i}$ and the feature extraction function f generates a new feature set $\boldsymbol{Y}_\text{o}$, where $|\boldsymbol{Y}_\text{o}|<|\boldsymbol{Y}_\text{i}|$. Generally, the objective functions of the feature extraction methods minimize the difference between the original space $\boldsymbol{Y}_\text{i}$ and the new space $\boldsymbol{Y}_\text{o}$ (Meng QX et al., 2017). Moreover, the encoding process adopts the activation function as a nonlinear process; the weighting matrices of the encoding and decoding processes of the following experiments use the binding matrix to enhance the nonlinearity of the mapping process.

3.2 Distance autoencoder

The traditional AE simply relies on minimizing the error between the input and reconstructed signals to obtain the feature representations of the hidden layer from the input; however, this training strategy does not guarantee the most representative feature of the extracted data. The feature that the encoder has learned may cause the reconstruction information to be only a duplicate of the original input. In a multi-category task, the “distance” between different classes largely reflects whether the two categories can be easily classified correctly. If the “distance” between the centers of the two different classes is small (that is, the centers are close to each other), it is difficult to distinguish the categories to which they belong. If the “distance” between the two different classes is

relatively large, it is easier to determine the categories to which they belong. Suppose there are three different classes (Fig. 3), and "distance" is the distance between the center point of label 1 and the center point of label 2. Similarly, the distance between each two class labels is the distance between their respective centers. The SAE is an improved form of AE. It introduces the sparsity penalty into the loss function and minimizes the loss function to generate new features; the SAE uses the label information to train the classification model mentioned above. Inspired by this, we propose the DAE. By introducing distance into the loss function and minimizing the objective function, we can improve the classification accuracy of the classifier. Specifically, the loss function is defined as follows:

$$J_{\text{cost}}(\boldsymbol{\theta}) = J_{\text{MSE}}(\boldsymbol{\theta}) + J_{\text{w}}(\boldsymbol{\theta}) + J_{\text{sparse}}(\boldsymbol{\theta}) - \beta \sum_{i=1}^{m(m-1)/2} \text{distance}, \tag{17}$$

where $J_{\text{MSE}}(\boldsymbol{\theta})$, $J_{\text{w}}(\boldsymbol{\theta})$, and $J_{\text{sparse}}(\boldsymbol{\theta})$ have the same meanings as in Eq. (10), m is the number of label categories, $\sum_{i=1}^{m(m-1)/2}\text{distance}$ is the sum of the distances between labels, and β is the weight coefficient in the loss function.

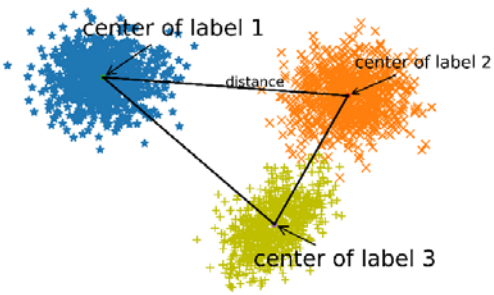


Fig. 3 Illustration of the distances between three labels
"distance" means the distance between each two class centers

There are multiple representations for "distance," such as the Euclidean distance, Manhattan distance, and Chebyshev distance. Here, the distance in Eq. (17) is chosen as the Euclidean distance.

However, in many cases, owing to a variety of reasons, the labeling of our data may not be complete. So, we use the unlabeled data to pre-train the network. In the experiments below, we manually delete some labels of the data to simulate cases where the data are contaminated. Based on the discussion above, we propose a semi-supervised DAE (Algorithm 1), and the flowchart is shown in Fig. 4.

Algorithm 1 Distance autoencoder

Input: data matrix $\boldsymbol{X}\in\mathbb{R}^{D\times n}$, parameters $\alpha>0$, $\beta>0$, $\gamma>0$

```
Initialize AE
model=DiscriminatedAutoencoder(θ, β)
a=model.encoder(X_unlabeled)
X_reconstruction=model.decoder(X_unlabeled)
Minimize J_cost(θ)=J_MSE(θ)+γJ_w(θ)
a_labeled=model.encoder(X_labeled)
X_reconstruction=model.encoder(a_labeled)
Calculate the center of each label (m labels)
distance=0
for (i=1; i<m; i++) do
  for (j=i; j<=m; j++) do
    distance=Σ||center_i−center_j||
  end for
end for
Minimize J_cost(θ)=J_MSE(θ)+γJ_w(θ)+αJ_sparse(θ)
                  −β∑_{i=1}^{m(m−1)/2} distance
prediction=model.fit
Classify the results
```

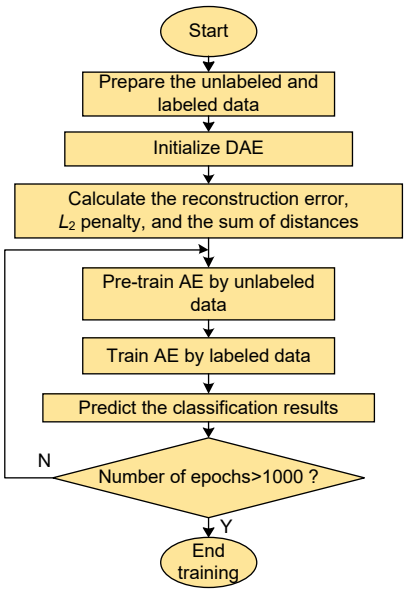


Fig. 4 The algorithm process for the semi-supervised autoencoder

4 Experimental results

4.1 Datasets

4.1.1 MNIST dataset

The MNIST dataset of handwritten digits is provided by the National Institute of Standards and Technique (NIST) and contains 60 000 training samples and 10 000 test samples. It has 10 labels from zero to nine (Fig. 5b). The size of each image is 28×28. Both the training set and test set are composed of handwritten numbers written by 250 different people. One hundred randomly selected images are shown in Fig. 5a.

For the MNIST dataset, the two-dimensional distribution of the data is shown in Fig. 6.

4.1.2 Street view house numbers

The street view house numbers (SVHN) dataset is a real-world image dataset for developing machine learning and object recognition algorithms with a minimal requirement for data preprocessing and formatting. The SVHN was obtained from the house numbers in Google street view images and it has 10 classes, 73 257 digits for training, and 26 032 digits for testing.

There are 10 images in each category, shown in Fig. 7.

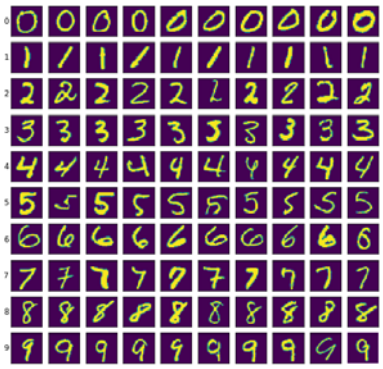
4.1.3 German traffic sign recognition benchmark

The German traffic sign recognition benchmark (GTSRB) is a multi-class, single-image classification dataset. We selected eight classes from it as our dataset. Fig. 8a shows the traffic signs randomly selected from the GTSRB. Since the images were collected in a real environment, the dataset includes a large number of images under various adverse conditions such as low resolution, different illumination intensities, partial occlusion, the tilt of view, and motion blur, which can comprehensively reflect the practical application potential of the classification algorithm.

4.1.4 CIFAR-10

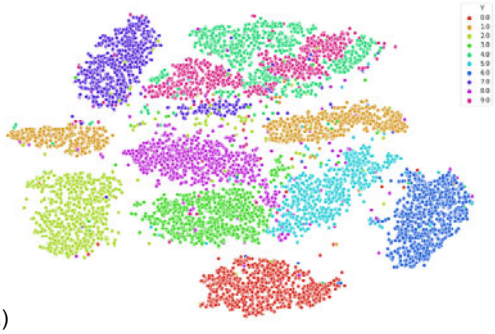
The CIFAR-10 dataset consists of 60 000 color images in 10 classes with 6000 images per class; it

(a)

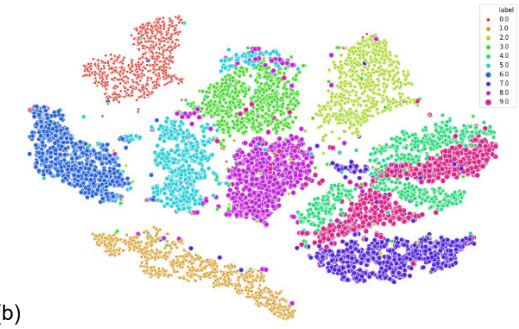


(b)

Fig. 5 The original (a) and preprocessed (b) MNIST data

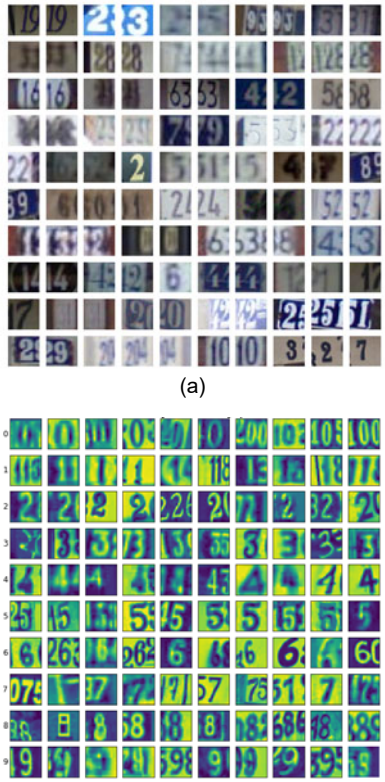


(a)



(b)

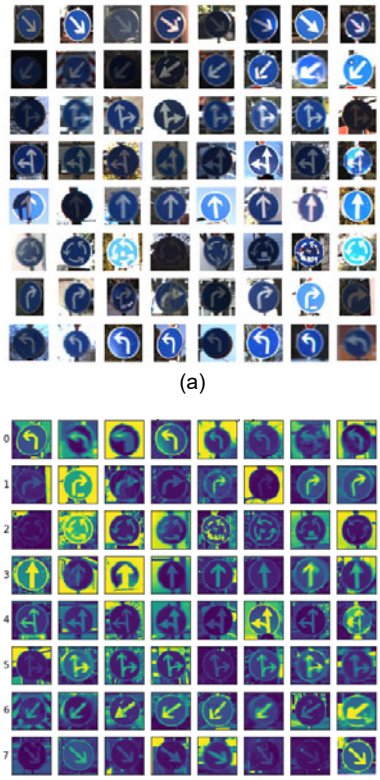
Fig. 6 Two-dimensional distribution of the MNIST data: (a) original data before using the DAE; (b) reconstructed data after using the DAE



(a)

(b)

Fig. 7 The original (a) and preprocessed (b) SVHN data



(a)

(b)

Fig. 8 The original (a) and preprocessed (b) GTSRB data

represents a labeled subset which includes 80 million tiny images. Fig. 9 shows 10 random images from each class. There are 50 000 training images and 10 000 test images.

4.2 Experiment

We carried out experiments using the datasets mentioned above. All models were carried out with the same configurations on the same dataset. Specifically, when building the network structure, we initialized the encoder weight matrix using the standard normal distribution. For the decoder weight matrix, we used $\boldsymbol{W}'=\boldsymbol{W}^{\mathrm{T}}$. The activation function of each layer of the encoder and decoder was selected as the sigmoid function. The objective function was trained with stochastic gradient descent (SGD) for 1000 epochs.

To verify the semi-supervised algorithm, we took some data from the original data as labeled and unlabeled data respectively. The setting values of this ratio were 10%, 30%, 50% (meaning that 10%, 30%, or 50% labeled/unlabeled data were extracted from the original data). Specifically, when we took 50% unlabeled data (artificially erasing the label) from the original data in unsupervised learning, the same unlabeled data were used in supervised learning, and the rest of the labeled data were used for fine-tuning in the supervised learning process. Both the unlabeled and labeled data would be used in semi-supervised learning.

At the beginning of the experiment, the classification performance of the DAE proposed in this study was compared with those of the traditional AE and the SAE (Wu et al., 2013). Then we compared the improved distance sparse autoencoders (distance sparse autoencoder, DSAE) with their original versions (SAE). We also incorporated the state-of-the-art Sv-AE mentioned in Section 3 as our baseline in the experiment.

Glorot and Bengio (2010) pointed out that the AEs in each of contrast experiment have their own loss functions, but for feature extraction, all of the AEs take the MSE of the reconstruction error. The classifier uses the softmax function of the fully connected layer.

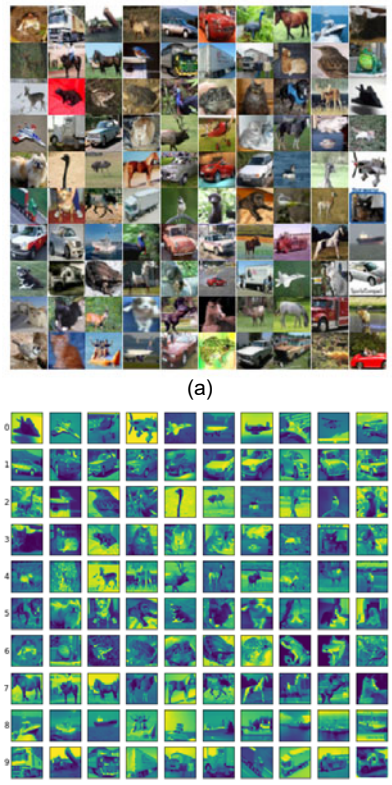


Fig. 9 The original (a) and preprocessed (b) CIFAR-10 data

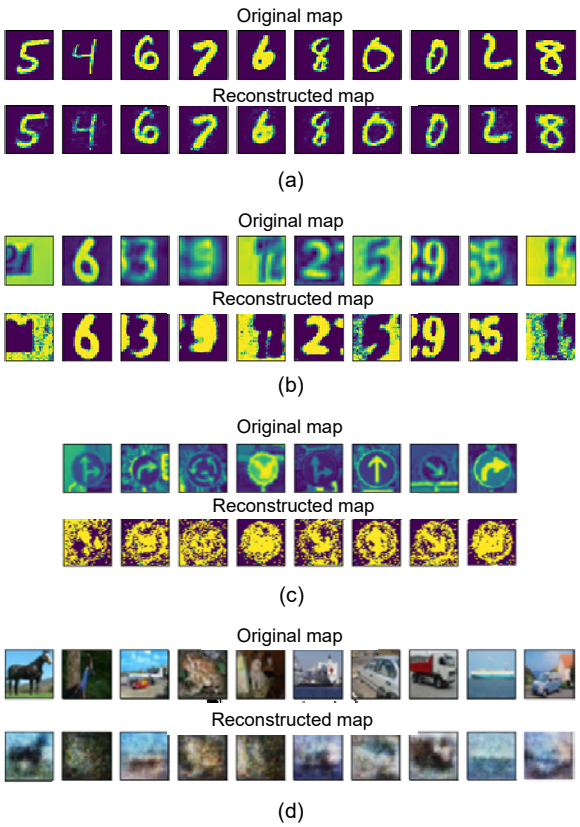


Fig. 10 Original and reconstructed maps of the traditional autoencoder: (a) MNIST dataset; (b) SVHN dataset; (c) GTSRB dataset; (d) CIFAR-10 dataset

4.2.1 Comparison with AE, SAE, and the state-of-the-art algorithm

We first compared the classification performance of our proposed DAE with those of the traditional AE and the SAE on the datasets described above. To objectively evaluate the performance of our method, we compared it with the state-of-the-art Sv-AE algorithm. In the experiment, we explored different parameters of the autoencoder so that we can know how they affect the performance of the autoencoder.

The original and reconstructed maps of the traditional AE, SAE, and DAE are shown in Figs. 10, 11, and 12, respectively.

As we know, the smaller the magnitude of the reconstruction error is, the more distinct the reconstruction map will be. From the results shown above, we can learn that the DAE has more effective classification than the traditional AE.

To verify the effectiveness of our algorithm in real scenes, we carried out a recognition and classification experiment using traffic lights. We made a dataset (DS) with a training set consisting of 1187 images and a test set consisting of 297 images. The traffic lights are shown in Fig. 13. The experimental results are shown in Tables 1–3.

4.2.2 Experiments on single autoencoders and stacked autoencoders

In the experiment, we explored different values of the scaling parameter β, which determines the weights of the reconstructed data and the original data. The value of β ranged from zero to two in the step of 0.02. Note that the AE and SAE do not have such a parameter; thus, their reconstruction loss does not change as β changes. The reconstruction loss was measured by the MSE, which confirmed that consideration of the distance information is distinctive in the processes of encoding and decoding. The proposed DAE model was influenced by the weight parameter β, and the value of β depended on the datasets.

We also evaluated the DAE as a pre-training

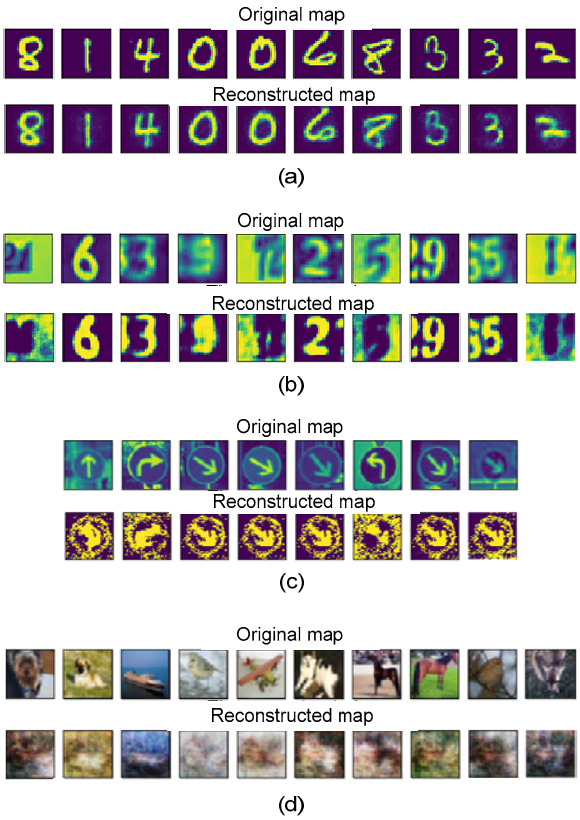


Fig. 11 Original and reconstructed maps of the sparse autoencoder: (a) MNIST dataset; (b) SVHN dataset; (c) GTSRB dataset; (d) CIFAR-10 dataset

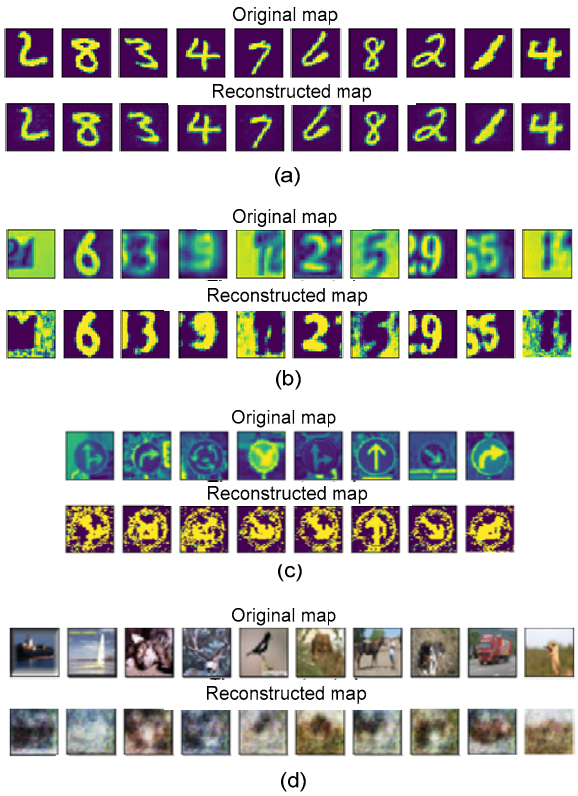


Fig. 12 Original and reconstructed maps of the distance autoencoder: (a) MNIST dataset; (b) SVHN dataset; (c) GTSRB dataset; (d) CIFAR-10 dataset

strategy for building deep networks, using the stacking procedure that we described in Section 2.2. We compared mainly the classification performance of networks pre-trained by stacking distance AEs (StackedDAE) with those of stacking traditional AEs and the corresponding improved versions.

Tables 1, 2, and 3 are the results of the experiments that 10%, 30%, and 50% labeled/unlabeled data were extracted from the original data, respectively.

We compared the performance of the extended variants of the autoencoders to those of their original versions, and the results are listed in Table 1–3. The more data we used, the better the results we achieved. For small-scale datasets, the methods performed well; for CIFAR-10, the best classification accuracy achieved was only 0.6591. The classification accuracy of our own dataset was up to 0.9868. Thus, the proposed method prefers small-scale datasets. The proposed method achieved better results than the algorithms compared (including the state-of-the-art supervised autoencoder algorithm). We observed that in most cases, considering the distance of the classes contributed to improving the classification performance, suggesting that autoencoders can generate more robust and useful features with less information and that these features are the key to achieving good classification results. In the real scenes, our proposed method also achieved satisfactory results. According to the results above, we conclude that under the condition of a lack of sufficient labeled data, unlabeled data can help improve the classification performance using our proposed method.

Fig. 13 Traffic lights in real scenes

Table 1 Experimental results of the MNIST, SVHN, GTSRB, DS, and CIFAR-10 datasets with 10% labeled/unlabeled data from the original data

Model	Classification accuracy (unsupervised)				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	0.5364	0.3202	0.4851	0.6287	0.2946
Sv-AE	–	–	–	–	–
DAE	0.5469	0.3316	0.5107	0.6540	0.3058
StackedDAE	–	–	–	–	–
SAE	0.5629	0.3573	0.5230	0.6630	0.3213
DSAE	–	–	–	–	–
StackedDSAE	–	–	–	–	–
Model	**Classification accuracy (supervised)**				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	–	–	–	–	–
Sv-AE	0.5729	0.3568	0.5379	0.6842	0.3147
DAE	0.5516	0.3486	0.5256	0.6760	0.3127
StackedDAE	0.5567	0.3541	0.5283	0.6789	0.3306
SAE	–	–	–	–	–
DSAE	0.5730	0.3564	0.5370	0.6831	0.3472
StackedDSAE	0.5792	0.3603	0.5401	0.6856	**0.3619**
Model	**Classification accuracy (semi-supervised)**				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	–	–	–	–	–
Sv-AE	–	–	–	–	–
DAE	0.5602	0.3517	0.5297	0.6778	0.3217
StackedDAE	0.5700	0.3554	0.5319	0.6821	0.3446
SAE	–	–	–	–	–
DSAE	0.5886	0.3614	0.5407	0.6843	0.3568
StackedDSAE	**0.6031**	**0.3661**	**0.5459**	**0.6889**	0.3608

The best results are in bold

5 Conclusions

In this paper, we have proposed a semi-supervised stacked DAE model that can extract the features of high-dimensional data based on the distance information for a semi-supervised image classification problem. As we know, the traditional AE is an unsupervised method, which does not use label information. We have extended this principle to other major autoencoder models including the SAE. From the experiments, we argued that the labels' distance information has an effect on the classification performance. The proposed semi-supervised distance autoencoder models have been evaluated on several datasets, and the experimental results showed that by considering the distance information, more robust features can be generated. The better features contribute to improved classification results.

Contributors

Liang HOU and Xiao-yi LUO designed the research and processed the data. Liang HOU drafted the manuscript. Zi-yang WANG helped organize the manuscript. Liang HOU and Jun LIANG revised and finalized the paper.

Compliance with ethics guidelines

Liang HOU, Xiao-yi LUO, Zi-yang WANG, and Jun LIANG declare that they have no conflict of interest.

References

Bengio Y, 2009. Learning deep architectures for AI. *Found Trends Mach Learn*, 2(1):1-127. https://doi.org/10.1561/2200000006

Table 2 Experimental results of the MNIST, SVHN, GTSRB, DS, and CIFAR-10 datasets with 30% labeled/unlabeled data from the original data

Model	Classification accuracy (unsupervised)				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	0.7260	0.5064	0.6051	0.7881	0.4519
Sv-AE	–	–	–	–	–
DAE	0.7504	0.5119	0.6107	0.7966	0.4592
StackedDAE	–	–	–	–	–
SAE	0.7603	0.5364	0.6232	0.8059	0.4629
DSAE	–	–	–	–	–
StackedDSAE	–	–	–	–	–
Model	Classification accuracy (supervised)				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	–	–	–	–	–
Sv-AE	0.7712	0.5508	0.6401	0.8171	0.4642
DAE	0.7516	0.5221	0.6282	0.8030	0.4613
StackedDAE	0.7547	0.5251	0.6314	0.8059	0.4737
SAE	–	–	–	–	–
DSAE	0.7692	0.5474	0.6361	0.8120	0.4879
StackedDSAE	0.7729	0.5532	0.6397	0.8158	0.4958
Model	Classification accuracy (semi-supervised)				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	–	–	–	–	–
Sv-AE	–	–	–	–	–
DAE	0.7597	0.5286	0.6328	0.8137	0.4685
StackedDAE	0.7631	0.5313	0.6357	0.8150	0.4732
SAE	–	–	–	–	–
DSAE	0.7739	0.5494	0.6409	0.8194	0.4916
StackedDSAE	**0.7764**	**0.5561**	**0.6461**	**0.8241**	**0.5007**

The best results are in bold

Bengio Y, Courville A, Vincent P, 2013. Representation learning: a review and new perspectives. *IEEE Trans Patt Anal Mach Intell*, 35(8):1798-1828. https://doi.org/10.1109/tpami.2013.50

Bianco S, Buzzelli M, Schettini R, 2018. Multiscale fully convolutional network for image saliency. *J Electron Imag*, 27(5):051221. https://doi.org/10.1117/1.jei.27.5.051221

Deng J, Zhang ZX, Marchi E, et al., 2013. Sparse autoencoder-based feature transfer learning for speech emotion recognition. Humaine Association Conf on Affective Computing and Intelligent Interaction, p.511-516. https://doi.org/10.1109/acii.2013.90

Du F, Zhang JS, Ji NN, et al., 2018. Discriminative representation learning with supervised auto-encoder. *Neur Process Lett*, 49(2):507-520. https://doi.org/10.1007/s11063-018-9828-2

Feng SW, Duarte MF, 2018. Graph autoencoder-based unsupervised feature selection with broad and local data structure preservation. *Neurocomputing*, 312:310-323. https://doi.org/10.1016/j.neucom.2018.05.117

Glorot X, Bengio Y, 2010. Understanding the difficulty of training deep feedforward neural networks. *J Mach Learn Res*, 9:249-256.

Gong YC, Lazebnik S, Gordo A, et al., 2013. Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans Patt Anal Mach Intell*, 35(12):2916-2929. https://doi.org/10.1109/tpami.2012.193

Haralick RM, Shanmugam K, Dinstein I, 1973. Textural features for image classification. *IEEE Trans Syst Man Cybern*, SMC-3(6):610-621. https://doi.org/10.1109/TSMC.1973.4309314

He XT, Peng YX, Zhao JJ, 2018. Fast fine-grained image classification via weakly supervised discriminative localization. *IEEE Trans Circ Syst Video Technol*, 29(5): 1394-1407. https://doi.org/10.1109/tcsvt.2018.2834480

He XT, Peng YX, Zhao JJ, 2019. Which and how many regions to gaze: focus discriminative regions for fine-grained visual categorization. *Int J Comput Vis*, 127(9):

Table 3 Experimental results of the MNIST, SVHN, GTSRB, DS, and CIFAR-10 datasets with 50% labeled/unlabeled data from the original data

Model	Classification accuracy (unsupervised)				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	0.9271	0.6564	0.7958	0.9293	0.6149
Sv-AE	–	–	–	–	–
DAE	0.9554	0.7289	0.8205	0.9630	0.6208
StackedDAE	–	–	–	–	–
SAE	0.9606	0.7364	0.8260	0.9667	0.6294
DSAE	–	–	–	–	–
StackedDSAE	–	–	–	–	–
Model	**Classification accuracy (supervised)**				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	–	–	–	–	–
Sv-AE	0.9636	0.7619	0.8355	0.9745	0.6315
DAE	0.9507	0.7523	0.8212	0.9730	0.6287
StackedDAE	0.9567	0.7541	0.8254	0.9759	0.6345
SAE	–	–	–	–	–
DSAE	0.9722	0.7614	0.8350	0.9831	0.6463
StackedDSAE	0.9759	0.7662	0.8361	0.9856	0.6549
Model	**Classification accuracy (semi-supervised)**				
	MNIST	SVHN	GTSRB	DS	CIFAR-10
AE	–	–	–	–	–
Sv-AE	–	–	–	–	–
DAE	0.9601	0.7486	0.8337	0.9798	0.6319
StackedDAE	0.9733	0.7584	0.8357	0.9825	0.6427
SAE	–	–	–	–	–
DSAE	0.9730	0.7694	0.8447	0.9835	0.6447
StackedDSAE	**0.9764**	**0.7723**	**0.8459**	**0.9868**	**0.6591**

The best results are in bold

1235-1255. https://doi.org/10.1007/s11263-019-01176-2

Hinton GE, 2007. Learning multiple layers of representation. *Trends Cogn Sci*, 11(10):428-434. https://doi.org/10.1016/j.tics.2007.09.004

Hinton GE, Salakhutdinov RR, 2006. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504-507. https://doi.org/10.1126/science.1127647

Kingma DP, Welling M, 2016. Auto-encoding variational Bayes. https://arxiv.org/abs/1312.6114

Meng LH, Ding SF, Zhang N, et al., 2018. Research of stacked denoising sparse autoencoder. *Neur Comput Appl*, 30(7): 2083-2100. https://doi.org/10.1007/s00521-016-2790-x

Meng QX, Catchpoole D, Skillicom D, et al., 2017. Relational autoencoder for feature extraction. Int Joint Conf on Neural Networks, p.364-371. https://doi.org/10.1109/ijcnn.2017.7965877

Peng YX, He XT, Zhao JJ, 2018. Object-part attention model for fine-grained image classification. *IEEE Trans Image Process*, 27(3):1487-1500. https://doi.org/10.1109/tip.2017.2774041

Rahmani MH, Almasganj F, Ali Seyyedsalehi S, 2018. Audio-visual feature fusion via deep neural networks for automatic speech recognition. *Dig Signal Process*, 82(5): 54-63. https://doi.org/10.1016/j.dsp.2018.06.004

Rifai S, Vincent P, Muller X, et al., 2011. Contractive auto-encoders: explicit invariance during feature extraction. Proc 28th Int Conf on Machine Learning, p.833-840.

Santana E, Emigh M, Principe JC, 2016. Information theoretic-learning auto-encoder. Int Joint Conf on Neural Networks. https://doi.org/10.1109/ijcnn.2016.7727620

Sun Y, Chen Y, Wang XG, et al., 2014. Deep learning face representation by joint identification-verification. Proc 27th Int Conf on Neural Information Processing, p.1988-1996.

Sun YN, Xue B, Zhang MJ, et al., 2017. A particle swarm optimization-based flexible convolutional autoencoder for image classification. *IEEE Trans Neur Netw Learn Syst*, 30(8):2295-2309. https://doi.org/10.1109/TNNLS.2018.2881143

Taherkhani A, Cosma G, Mcginnity TM, 2018. Deep-FS: a feature selection algorithm for deep Boltzmann machines. *Neurocomputing*, 322:22-37. https://doi.org/10.1016/j.neucom.2018.09.040

Tang JH, Li ZC, Wang M, et al., 2015. Neighborhood discriminant hashing for large-scale image retrieval. *IEEE Trans Image Process*, 24(9):2827-2840. https://doi.org/10.1109/tip.2015.2421443

Tolstikhin I, Bousquet O, Gelly S, et al., 2017. Wasserstein auto-encoders. https://arxiv.org/abs/1711.01558

Vincent P, Larochelle H, Lajoie I, et al., 2010. Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. *J Mach Learn Res*, 11(12):3371-3408.

Wang W, Huang Y, Wang YZ, et al., 2014. Generalized autoencoder: a neural network framework for dimensionality reduction. IEEE Conf on Computer Vision and Pattern Recognition. https://doi.org/10.1109/cvprw.2014.79

Wu J, Cai ZH, Zhu XQ, 2013. Self-adaptive probability estimation for Naive Bayes classification. Int Joint Conf on Neural Networks. https://doi.org/10.1109/ijcnn.2013.6707028

Xu WD, Sun HZ, Deng C, et al., 2016. Variational autoencoders for semi-supervised text classification. https://arxiv.org/abs/1603.02514

Zhang TS, Wang W, Ye H, et al., 2016. Fault detection for ironmaking process based on stacked denoising autoencoders. American Control Conf, p.3261-3267. https://doi.org/10.1109/acc.2016.7525420