Scientific Research Publishing

Journal of Computer and Communications, 2022, 10, 138-153
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Double-Weighted Deterministic Extreme Learning Machine Based on Sparse Denoising Autoencoder and Its Applications

Liang Luo[1], Bolin Liao[1], Cheng Hua[1], Rongbo Lu[2*]

[1]College of Computer Science and Engineering, Jishou University, Jishou, China
[2]College of Computer and Artificial Intelligence, Huaihua University, Huaihua, China
Email: luoliang991105@163.com, bolinliao@jsu.edu.cn, hc@stu.jsu.edu.cn, *lurongbo8563@163.com

How to cite this paper: Luo, L., Liao, B.L., Hua, C. and Lu, R.B. (2022) A Double-Weighted Deterministic Extreme Learning Machine Based on Sparse Denoising Autoencoder and Its Applications. *Journal of Computer and Communications*, **10**, 138-153.
https://doi.org/10.4236/jcc.2022.1011010

Received: October 25, 2022
Accepted: November 27, 2022
Published: November 30, 2022

Abstract

Extreme learning machine (ELM) is a feedforward neural network-based machine learning method that has the benefits of short training times, strong generalization capabilities, and will not fall into local minima. However, due to the traditional ELM shallow architecture, it requires a large number of hidden nodes when dealing with high-dimensional data sets to ensure its classification performance. The other aspect, it is easy to degrade the classification performance in the face of noise interference from noisy data. To improve the above problem, this paper proposes a double pseudo-inverse extreme learning machine (DPELM) based on Sparse Denoising AutoEncoder (SDAE) namely, SDAE-DPELM. The algorithm can directly determine the input weight and output weight of the network by using the pseudo-inverse method. As a result, the algorithm only requires a few hidden layer nodes to produce superior classification results when classifying data. And its combination with SDAE can effectively improve the classification performance and noise resistance. Extensive numerical experiments show that the algorithm has high classification accuracy and good robustness when dealing with high-dimensional noisy data and high-dimensional noiseless data. Furthermore, applying such an algorithm to Miao character recognition substantiates its excellent performance, which further illustrates the practicability of the algorithm.

Keywords

Extreme Learning Machine, Sparse Denoising Autoencoder, Pseudo-Inverse Method, Miao Character Recognition

1. Introduction

Extreme Learning Machine (ELM), a brand-new single hidden layer feedforward neural network learning algorithm, was first put forth by Professor Huang of Nanyang Technological University in Singapore in 2004 [1]. Since ELM has the advantages of short training times and excellent generalization performance, it has received extensive attention and investigation from many scholars in the fields of pattern classification and data prediction, etc., unlike other conventional neural network algorithms (such as BP neural network algorithm [2]). The fundamental idea of ELM is to randomly create the input weights and hidden layer offsets of the network structure throughout the training phase and preserve them unmodified. Then the output weight of ELM is obtained by pseudo inverse operation [3]. The whole training process does not need iteration, only the hidden layer's number of neurons and activation function need to be set [4]. At present, ELM is being utilized extensively in the detection of diseases [5], the understanding of intelligent decision-making [6], and the recognition of traffic signs [7].

However, it is worth pointing out that the input weights, hidden layer bias, and the numbers of hidden layer neurons of the ELM algorithm are uncertain, conducting to the disadvantages of high complexity and low stability of the algorithm [8]. In view of this problem, many scholars have carried out a lot of research and experiments. In [9], combining a Memetic Algorithm (MA) and ELM to embed the local search strategy into the global optimization framework to seek the optimal parameters of the network structure. Moreover, Rong *et al.* proposed the pruned-ELM algorithm [10], which uses statistical methods to measure the correlation of hidden nodes and optimizes hidden layer nodes via pruning. In [11], a generalized interval-2 type fuzzy neural network establishes hidden nodes on the basis of interval-2 type multivariate Gaussian function, realizes the optimization of ELM hidden layer nodes and achieves good results. Although the aforementioned algorithms can optimize the structure of neurons in the hidden layer, the defects that cause a large number of hyperparameters are nonnegligible. Hyperparameters are obtained by iterative optimization in the process of neural network training, which increases the computational complexity and reduces the real-time stability of the algorithm. To surmount this problem, a double pseudo-inverse extreme learning machine (DPELM) algorithm is proposed in to directly determine the input and output weights of the neural network [12], which improves computational efficiency and stability.

On the other hand, aiming at high-dimensional or noisy data, single hidden layer ELM cannot identify data features well, thus affecting the final data classification results. Many researchers try to solve this problem by combining the ELM algorithm with autoencoder, which can not only reduce the computational complexity of deep networks, but also reduce the training time. In [13], Cambria *et al.* propose ELM-AE by combining the advantages of ELM and autoencoder, which has good feature representation ability and classification performance. On this basis, Sun *et al.* proposed the discriminant Graph regularized extreme learning machine autoencoder (GELM-AE) to improve ELM-AE [14], which can extract more abstract high-level features and improve the overall performance of the network model. Inspired by the above research, this paper combines SDAE and DPELM, and further proposes a double pseudo-inverse extreme learning machine based on sparse denoising autoencoder (SDAE-DPELM) [15]. SDAE is an improved autoencoder deep learning algorithm, it adds sparse constraints to the autoencoder to optimize the network structure, then better extract the deep characteristics of the sample data, and enhance the robustness and regulation ability of the algorithm. The algorithm uses the sparse denoising automatic encoder to extract features from the input data and takes the extracted features as the input data of DPELM, and then carries out network training. The network training speed is guaranteed while also improving the feature extraction capability and noise immunity of the network.

2. Relate Work

In this paper, the proposed algorithm is an extension of ELM algorithm. For comparison and connection, this section will briefly review the relevant concepts of ELM and the necessary theory of SDAE. It includes the combination of original ELM and SDAE.

2.1. Traditional Extreme Learning Machine

Given $\aleph$ randomly distinct sample sets $(\boldsymbol{x}_n, \boldsymbol{y}_n)$, where $\boldsymbol{x}_n$ is the input vector (sample feature) and $\boldsymbol{y}_n$ is the matching sample label vector, with $\boldsymbol{x}_n = (x_{n1}, x_{n2}, \cdots, x_{nN})^{\mathrm{T}} \in \mathbb{R}^N$, $\boldsymbol{y}_n = (y_{n1}, y_{n2}, \cdots, y_{nM})^{\mathrm{T}} \in \mathbb{R}^M$. The ELM network has the following parameters: N input neurons, K_1 hidden layer neurons, M output neurons, and $G(\cdot)$ denotes the activation function in the hidden layer neuron. The ELM's output can be expressed as

$$\boldsymbol{y}_n = \sum_{k_1=1}^{K_1} \boldsymbol{\lambda}_{k_1} G(\boldsymbol{\zeta}_{k_1}, \phi_{k_1}, \boldsymbol{x}_n), \tag{1}$$

$n = 1, 2, \cdots, \aleph$, the k_1th neuron of the hidden layer with the weight vector in the input layer is represented by $\boldsymbol{\zeta}_{k_1} = (\zeta_{k_1 1}, \zeta_{k_1 2}, \cdots, \zeta_{k_1 N})$, ϕ_{k_1} is a representation of the k_1th neuron's bias. $\boldsymbol{\lambda}_{k_1} = (\lambda_{k_1 1}, \lambda_{k_1 2}, \cdots, \lambda_{k_1 M})$ represents the vector of weights of the k_1th hidden layer neuron with the output layer. The above linear equations can be written in the matrix from

$$\boldsymbol{H}\boldsymbol{\Lambda} = \boldsymbol{Y}, \tag{2}$$

where Y expresses the training sample's desired output matrix, and

$$\boldsymbol{H} = \begin{bmatrix} G(\boldsymbol{\zeta}_1, \phi_1, \boldsymbol{x}_1) & \cdots & G(\boldsymbol{\zeta}_{K_1}, \phi_{K_1}, \boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ G(\boldsymbol{\zeta}_1, \phi_1, \boldsymbol{x}_\aleph) & \cdots & G(\boldsymbol{\zeta}_{K_1}, \phi_{K_1}, \boldsymbol{x}_\aleph) \end{bmatrix}$$

is the randomized matrix mapping. Randomly generated the hidden layer neuron parameters (ζ_n, ϕ_n) and their values will remain constant throughout the training procedure. Given sample data and a known matrix $\boldsymbol{H}$, obtain the least squares solution $\tilde{\Lambda}$ of Equation (2), $\tilde{\Lambda} = \boldsymbol{H}^{\dagger}\boldsymbol{Y}$, $\dagger$ represents the pseudo-inverse of the matrix.

2.2. Sparse Denoising Autoencoder

The traditional autoencoder (AE) has limited ability to reconstruct input data and poor ability to extract data features. Sparse autoencoder (SAE) is improved by traidional AE. Unlike AE, it adds sparsity constraint to the error function, so that most hidden layer nodes are set to 0, a few hidden layer nodes are not 0, and the network is more sparse. In this way, the SAE has good adjustment ability, which makes the learning process of the model more similar to that of the human brain, which is conducive to extracting more representative features and improving the classification accuracy of the algorithm. The SDAE is based on SAE to degrade the original sample data. Its goals are to reduce noise interference, enhance data reconstructions, and increase the algorithm's robustness. Figure 1 depicts the structure of the SDAE network.

Degradation, sparse coding, and decoding are the three stages of the SDAE training process. To construct the degraded data $\ddot{\boldsymbol{X}}$, the original input data data $\boldsymbol{X}$ is first set to 0 with the specified deterioration rate $\boldsymbol{v}$. Then the degraded data $\ddot{\boldsymbol{X}}$ is sparsely encoded to get the encoded data $\boldsymbol{h}$. Finally, the encoded data $\boldsymbol{h}$ is decoded by SDAE to obtain the reconstructed data $\boldsymbol{Y}$. Minimizing the loss function $L(X,Y)$ obtains the best feature representation of the original data.

The calculation formula of sparse coding and decoding process is shown in Equation (3):

$$\boldsymbol{h} = g\left(\ddot{\boldsymbol{X}}\right) = s_g\left(w\ddot{\boldsymbol{X}} + b\right), y = f\left(\boldsymbol{h}\right) = s_f\left(w'\boldsymbol{h} + b'\right), \tag{3}$$

where w and b are the sparse coding weight matrix and offset vector, respectively, and $s(\cdot)$ is the activation function; typically, a sigmoid function is used. w' and b' are the decoding weight matrix and offset vector, respectively. Given the training set $D = \left\{x^{(i)}, y^{(i)}\right\}_{i=1}^{N}$, the SDAE's overall loss function is:

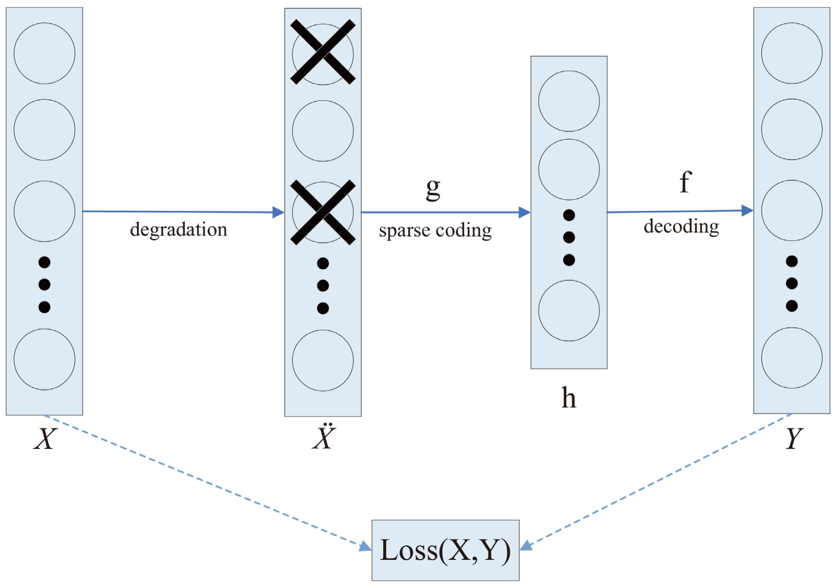

Figure 1. Network structure diagram of SDAE.

$$L(x,y)=\frac{1}{N}\sum_{D} J(x,y)+\eta\sum_{k_2=1}^{K_2} KL\left(\rho \,\|\, \hat{\rho}_{k_2}\right). \tag{4}$$

In Equation (4), the first part $J(x,y)=\frac{1}{2}\|x-y\|_2^2$ is the square difference error term. η is used to control the weight of the sparse penalty term and can take any value between 0 and 1, and K_2 represents the number of hidden layer nodes.

The Kullback-Leibler Divergence (KL) method is used for sparse penalty, as shown in Equation (5):

$$KL\left(\rho \,\|\, \hat{\rho}_{k_2}\right)=\rho \lg\left(\frac{\rho}{\hat{\rho}_{k_2}}\right)+(1-\rho)\lg\left(\frac{1-\rho}{1-\hat{\rho}_{k_2}}\right), \tag{5}$$

where $\hat{\rho}_{k_2}=\frac{1}{N}\sum_{i=1}^{N}\left(a_{k_2}(x_i)\right)$, represents the average activation value of all training samples on hidden layer neuron k_2, and a_{k_2} is the activation value on hidden layer neuron k_2. To achieve the effect that most of the neurons are inhibited, ρ is generally yielded to a value close to 0. If ρ takes the value 0.03, then by this constraint, the average activation value of each implicit layer neuron k_2 of the self-encoder will be close to 0.03.

3. SDAE-DPELM Algorithm Design

3.1. DPELM Learning Algorithm

This section focuses on the weight determination method of the improved ELM. The DPELM consists of N (input neurons), K_1 (hidden layer neurons) and M (output neurons), which is similar to classic ELM. DPELM algorithm is shown in Figure 2. By further analyzing the traditional ELM principle, we can rewrite Equation (1) as the following equation:

$$\boldsymbol{Y}=\Lambda G\left(\boldsymbol{Z}\boldsymbol{X}-\boldsymbol{\Phi}\right), \tag{6}$$

where $\boldsymbol{Y}=\left[\boldsymbol{y}_1,\boldsymbol{y}_2,\cdots,\boldsymbol{y}_{\aleph}\right]\in\mathbb{R}^{M\times\aleph}$, $\boldsymbol{X}=\left[\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_{\aleph}\right]\in\mathbb{R}^{N\times\aleph}$, $\boldsymbol{\Phi}=\left[\phi_1,\phi_2,\cdots,\phi_{\aleph}\right]\in\mathbb{R}^{K\times\aleph}$, $\boldsymbol{\Lambda}$ is the output weight matrix and $\boldsymbol{Z}$ is the input weight matrix,

$$\boldsymbol{\Lambda}=\begin{bmatrix}\lambda_{11} & \lambda_{12} & \cdots & \lambda_{1K_1}\\ \lambda_{21} & \lambda_{22} & \cdots & \lambda_{2K_1}\\ \vdots & \vdots & \ddots & \vdots\\ \lambda_{M1} & \lambda_{M2} & \cdots & \lambda_{MK_1}\end{bmatrix}\in\mathbb{R}^{M\times K_1},$$

$$\boldsymbol{Z}=\begin{bmatrix}\zeta_{11} & \zeta_{12} & \cdots & \zeta_{1N}\\ \zeta_{21} & \zeta_{22} & \cdots & \zeta_{2N}\\ \vdots & \vdots & \ddots & \vdots\\ \zeta_{K_11} & \zeta_{K_12} & \cdots & \zeta_{K_1N}\end{bmatrix}\in\mathbb{R}^{K_1\times N}.$$

Theorem 1. Assume that activation function $G(\cdot)$ is strictly monotonous. The bias Φ and output weights Λ are generated from interval randomly, the ideal input weights $\boldsymbol{Z}=\left(G^{-1}\left(\boldsymbol{\Lambda}^{\dagger}\boldsymbol{Y}\right)+\boldsymbol{\Phi}\right)\boldsymbol{X}^{\dagger}$.

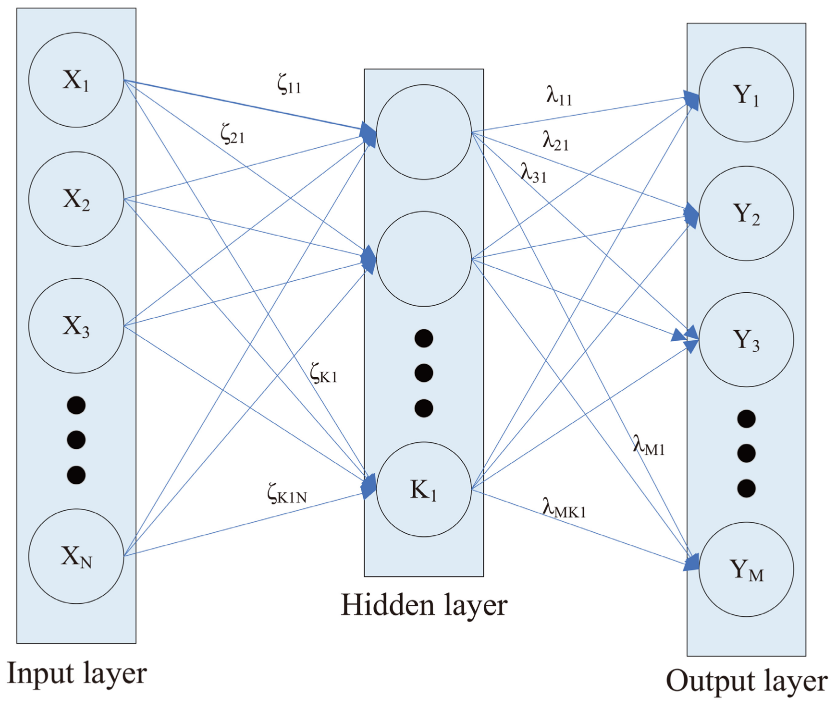

Figure 2. DPELM network structure diagram.

The bias Φ and output weights Λ are generated from interval randomly,
Prove: Simultaneously left multiply both sides of Equation (6) by $\Lambda^{\dagger}$ to get

$$\Lambda^{\dagger}\boldsymbol{Y} = \Lambda^{\dagger}\Lambda G\left(\boldsymbol{ZX} - \boldsymbol{\Phi}\right) = G\left(\boldsymbol{ZX} - \boldsymbol{\Phi}\right). \quad (7)$$

Then, solve the inverse function of the above equation, we have

$$G^{-1}\left(\Lambda^{\dagger}\boldsymbol{Y}\right) = \boldsymbol{ZX} - \boldsymbol{\Phi},$$

where $G^{-1}(\cdot)$ is the inverse function of $G(\cdot)$. The above equation can be rewritten as follows

$$\boldsymbol{ZX} = G^{-1}\left(\Lambda^{\dagger}\boldsymbol{Y}\right) + \boldsymbol{\Phi}. \quad (8)$$

Finally, we right multiply in both sides of the above equation by $\boldsymbol{X}^{\dagger}$ to obtain

$$\boldsymbol{ZXX}^{\dagger} = \left(G^{-1}\left(\Lambda^{\dagger}\boldsymbol{Y}\right) + \boldsymbol{\Phi}\right)\boldsymbol{X}^{\dagger},$$

that is,

$$\boldsymbol{Z} = \left(G^{-1}\left(\Lambda^{\dagger}\boldsymbol{Y}\right) + \boldsymbol{\Phi}\right)\boldsymbol{X}^{\dagger}.$$

The proof is finished.

The best output weight is determined using the pseudo-inverse approach once the optimal $\boldsymbol{Z}$ has been found. The formula $\tilde{\Lambda} = \boldsymbol{Y}\left(G\left(\boldsymbol{ZX} - \boldsymbol{\Phi}\right)\right)^{\dagger}$ can be used to determine the value of $\tilde{\Lambda}$.

3.2. Combination Method of SDAE and DPELM

The DPELM algorithm's network structure is straightforward, and it can quickly classify data. However, the classification performance of the data will be considerably impacted whenever the DPELM algorithm works with high-dimensional and noisy data. SDAE is capable of capturing characteristic representations of input data. Due to the increased sparsity constraint in SDAE, the network can better learn the structural features of the input data and thus better describe the input data. Therefore, we consider combining SDAE with DPELM. SDAE is responsible for extracting the input data required by DPELM, and then DPELM classifies the sample data. In addition to successfully reducing noise interference and enhancing algorithm robustness, it may also simplify the network topology and boost classification performance. Figure 3 depicts the SDAE-DPELM algorithm's network structure.

Below is a description of the SDAE-DPELM training process in full.

- Step 1: The sample data $\boldsymbol{X}$ is degraded into $\ddot{\boldsymbol{X}}$. Gradient descent training is carried out for SDAE to make the input data of the network equal to the output data. When the error function L is below the set threshold, SDAE training is finished.
- Step 2: After the SDAE training process, we obtain the hidden layer data $\boldsymbol{h}$ from the SDAE network. $\boldsymbol{h}$ is the result of a high level of abstraction of the original input layer sample data and network parameters. Due to the degradation of the original input and the addition of sparsity constraints to the network, the essential characteristics of the input data can be better reflected and the algorithm is more robust.
- Step 3: In this part, we take the extracted $\boldsymbol{h}$ as input layer sample data of DPELM classification algorithm, and use DPELM classification algorithm for classification processing. We conducted supervised training on these labeled sample data and obtained the fina classification results. At this point, the entire SDAE-DPELM algorithm training is finished.

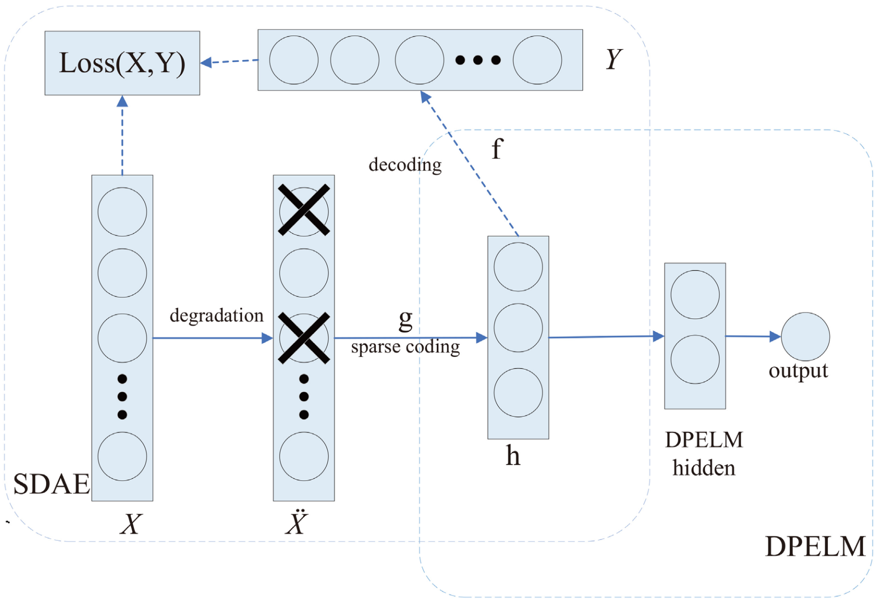

Figure 3. SDAE-DPELM network structure diagram.

4. Experimental Results and Analysis

4.1. Introduction of Experimental Environment and Experimental Dataset

In this section, aimed at examining the performance of our proposed algorithm, we will select the MINIST dataset on UCI for multiple experiments. The experimental hardware consists of an Intel(R) Core(TM) i9-10850K CPU 3.60 GHz with 16 GB of memory, 64-bit Windows 10, and Matlab 2016(a) as the experimental software.

4.2. Analysis and Determination of Network Structure

The number of input layer nodes N and output layer nodes M are fixed during process of neural network training, which corresponds to the number of training data dimensions and classes, respectively. Therefore, the key to neural network structure determination lies in the number of hidden layer nodes K_1. To determine K_1, scholars usually use the trial and error method to change the network structure according to certain standards for repeated experiments and then use the network structure with the best performance. This paper needs to conduct comparative experiments on performance impact. The different number of hidden layer nodes will affect the objectivity of model performance comparison. In SDAE-DPLEM neural network, it is divided into two stages: SDAE feature extraction and DPELM classification. The number of hidden nodes is different at different stages, so it has different effects on the performance of the network structure. Since it is challenging to count the hidden layer nodes K_1 and K_2 in both phases at once, the following experiments are used in this study to incrementally count the hidden layer nodes in each phase.

Firstly, this paper selected the MINIST dataset to train ELM and observe the relationship curve between ELM hidden layer node K_1 and classification performance in the training set and test set. The value range of K_1 is [100, 2000]. The performance trend is shown in Figure 4(a).

It can be seen from Figure 4(a) that the classification accuracy tends to increase incrementally as hidden layer node K_1 increases. Considering the compactness of the ELM algorithm network structure and the cost of training time

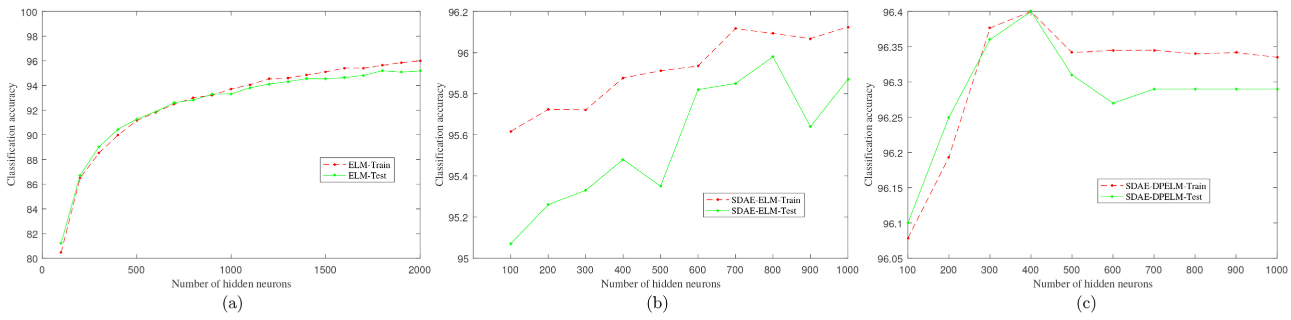

Figure 4. Influence of hidden layer node number on classification performance. (a) ELM Hidden layer node K_1; (b) SDAE hidden layer node K_2; (c) DPELM hidden layer node K_1.

burden, 1500 nodes are determined to be the ideal number of ELM hidden layer nodes K_1. Then, SDAE hidden layer nodes K_2 is determined by trial and error method again. The variation trend of the classification accuracy of the algorithm SDAE-ELM with SDAE hidden layer nodes K_2 is shown in **Figure 4(b)**.

In **Figure 4(b)**, after SDAE hidden layer nodes K_2 increases, the classification accuracy of SDAE-ELM gradually increases. When K_2 gets to 800, the classification accuracy is higher, and then there will be a small fluctuation in this range. While ensuring high classification accuracy, to reduce the complexity of time and space, the convergence of the network needs to be considered. Therefore, K_2 is determined to be 800.

As can be seen from the **Figure 4(c)**, when DPELM hidden layer nodes K_1 is less than 400, the classification accuracy of DPELM increases rapidly and reaches the highest point at 400. However, with the increasing DPELM hidden layer nodes K_1, the classification accuracy of DPELM begins to decline. In conclusion, in the following experimental analysis and comparison, the optimal DPELM hidden layer nodes K_1 is set to 400.

Finally, the optimal network structure 784-800-400-M of SDAE-DPELM is determined. Where 784 is the input layer nodes N of the data set and M is the number of categories, that is, the output vector.

4.3. SDAE-DPELM Numerical Comparison Experiment

SDAE-DPELM is compared to existing DPELM and SDAE-ELM to validate the algorithm's overall performance. Concretely, SDAE-DPELM is compared with DPELM to verify the impact of SDAE (sparse denoising autoencoder) on model accuracy, and the relevant results are shown in **Figure 5(a)**. Similarly, SDAE-ELM is used to verify the superiority of double pseudo inverse and the related results are depicted in **Figure 5(b)**. Noteworthy, the four algorithms have $N = 784$ nodes in the input layer and $M = 10$ nodes in the output layer, whose network structures are 784-K_1-10, 784-800-K_1-10,784-K_1-10, 784-800-K_1-10, respectively, where 800 is hidden layer nodes K_1.

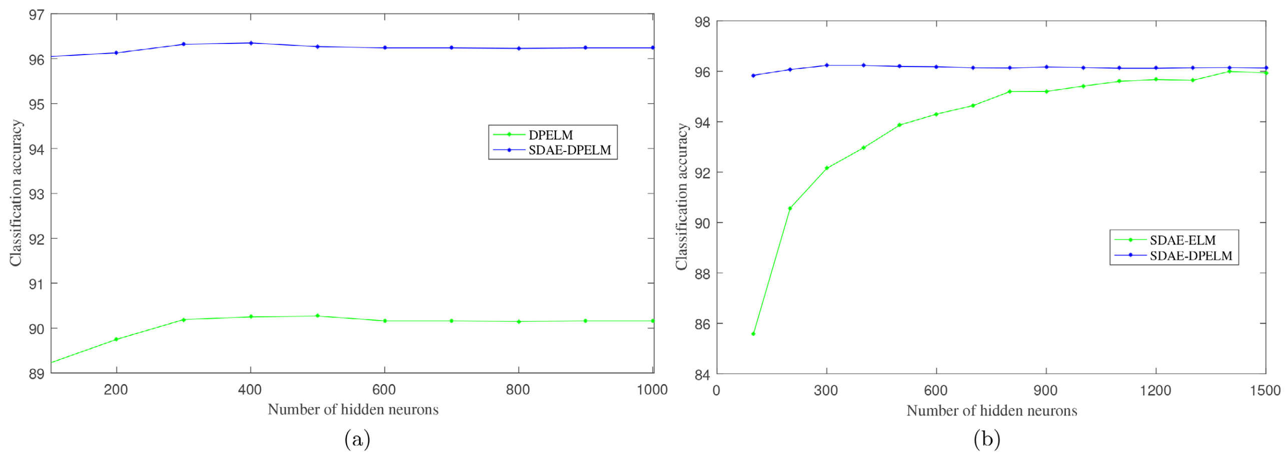

Figure 5. Curve of classification accuracy of DPELM, SDAE-ELM and SDAE-DPELM with hidden layer nodes K_1.

Obviously, SDAE-DPELM is more accurate than DPELM in any number of hidden nodes from Figure 5(a). As can been seen from Figure 5(b), when K_1 is large enough, the identification accuracy of SDAE-ELM in data set MINIST can be close to that of SDAE-DPELM, but too many hidden nodes will lead to more complex network structure and increase the burden of network training time. Therefore, this shows that SDAE-DPELM algorithm is effective by altering the conventional weight calculation approach.

As shown in Figure 6(b), when Gaussian noise (G.N.) is added to the MINIST dataset, the classification accuracy of the four models does not decrease significantly. The accuracy of data classification by SDAE-DPELM is still around 96%. At the same time, when the number of nodes is small, SDAE-DPELM can quickly reach a high value of classification accuracy. However, when salt and pepper noise (S&P.N.) is added to the data, the classification accuracy of the four models decreases to different degrees. SDAE-DPELM reduced classification accuracy by about 3%. Nevertheless, no matter what noise is added, the classification accuracy of SDA-DPELM is better than the other three models at any node between 0 and 1500. This indicates that SDAE-DPELM has better anti-noise than the other three models.

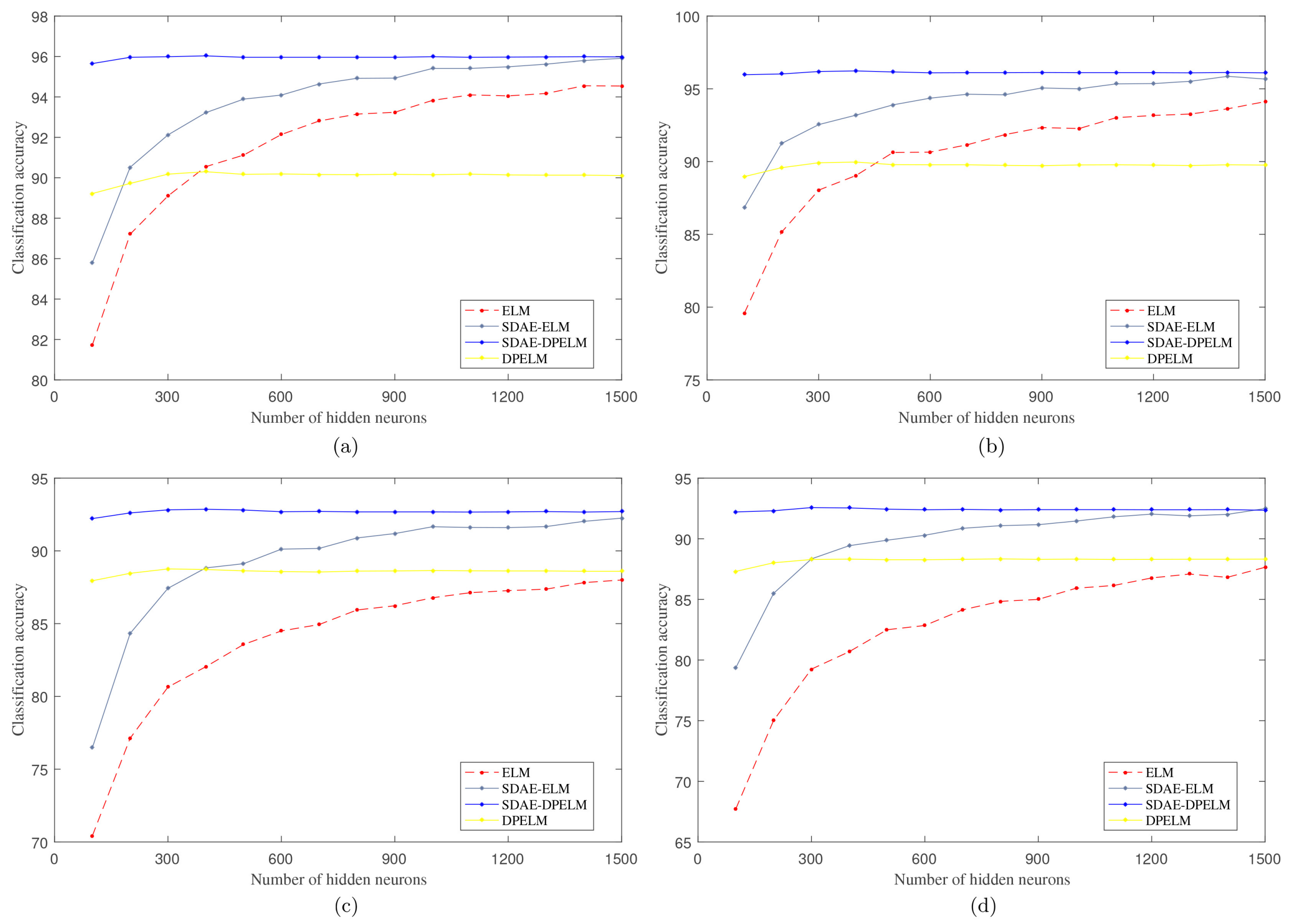

Figure 6. Influence of hidden layer nodes K_1 on classification accuracy of four algorithms under different noise conditions. (a) No noise. (b) G.N. (c) S&P.N. (d) G.N. and S&P.N.

4.4. Robustness Analysis of SDAE-DPELM Algorithm

SDAE-DPELM degenerates the original data samples, and its purpose is to eliminate noise interference and extract more essential features, thereby improving the robustness of the model. To verify the robustness of the SDAE-DPELM algorithm, different proportions of white Gaussian noise were added to the MNIST data set to compare the classification accuracy of the algorithm. The added Gaussian noise is subject to N(0, 0.01), N(0, 0.02), N(0, 0.03), N(0, 0.04), N(0, 0.05). In order to enhance the persuasiveness of the experiment, this experiment is carried out under the condition that the sparsity parameter ρ is 0.03, 0.05, and 0.07. The experimental results are shown in Figure 7.

It can be seen from Figure 7 that the classification accuracy of the algorithm slightly decreases when different proportions of white Gaussian noise are added to the dataset. However, when the sparsity parameter E takes different values, the variation range of classification accuracy is still less than 2%, which indicates that the data with certain noise will not significantly affect the classification accuracy of SDAE-DPELM.

4.5. Stability Analysis of SDAE-DPELM Algorithm

To verify the stability of the SDAE-DPELM algorithm, this section compares the mean, variance and range of SDAE-DPELM and ELM. Gradually increase the number of neurons in the hidden layer of the algorithm, and each time increase 50 neurons to observe the change of classification accuracy. Because the results of each classification of traditional ELM will have certain differences due to the different random initialization weights. In order to ensure the objectivity of algorithm comparison results and not be interfered by random values, during the experiment, every 50 hidden layer neurons are added, the ELM algorithm and

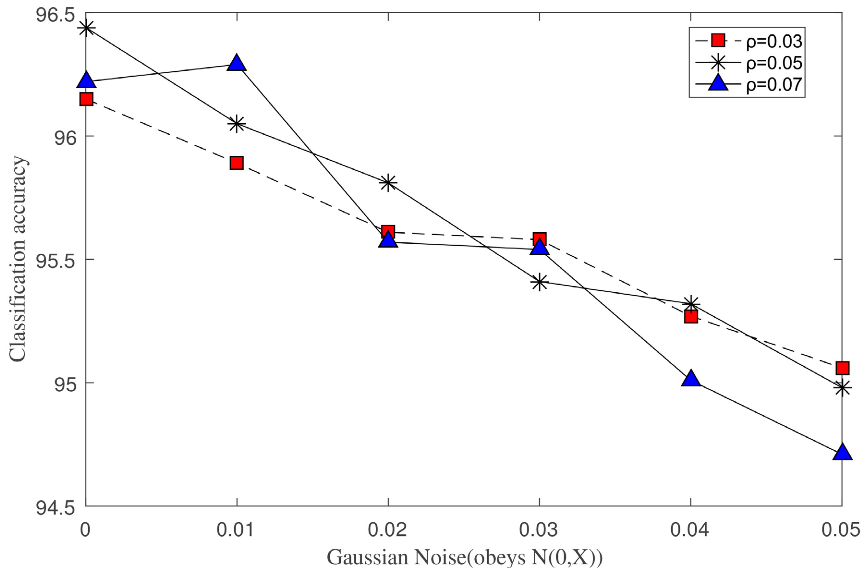

Figure 7. Comparison of SDAE-DPELM accuracy in the MINIST dataset of white Gaussian noise with different proportions.

SDAE-DPELM algorithm are required to run 10, 20 and 50 times respectively. The average accuracy of the test set is taken as the final classification result of each algorithm. The experimental results are shown in **Table 1**.

It can be seen from the results in **Table 1** that the classification performance of SDAE-DPELM algorithm is higher than that of the original ELM no matter whether it has been tested 10 times, 20 times or 50 times. In order to further analyze the stability of SDAE-DPELM algorithm, the range and variance comparison experiments between SDAE-DPELM and ELM are conducted again according to the above methods.

The variance value and the range value reflect the discrete degree of the output results of the two algorithms, which can be used to judge the stability of the algorithm output. The greater the dispersion, the more unstable the algorithm is. On the contrary, the smaller it is, the better the stability of the algorithm is. As can be seen from **Figure 8**, the variance and range of the proposed algorithm in MINIST dataset are smaller than those of the traditional ELM algorithm, indicating that the stability of the proposed algorithm is better than that of the traditional ELM algorithm.

Table 1. Comparison of ELM and SDAE-DPELM classification average accuracy (%).

Nodes	$\text{ELM}_{\text{times}=10}$	$\text{SDAE-DPELM}_{\text{times}=10}$	$\text{ELM}_{\text{times}=20}$	$\text{SDAE-DPELM}_{\text{times}=20}$	$\text{ELM}_{\text{times}=50}$	$\text{SDAE-DPELM}_{\text{times}=50}$
50	72.745	95.974	72.357	96.063	72.529	95.888
100	81.096	95.932	81.499	95.982	81.431	95.946
150	84.849	95.986	85.021	96.131	84.817	96.018
200	86.494	96.045	86.712	96.247	86.777	96.146
250	88.003	96.124	88.066	96.320	88.071	96.248
300	88.968	96.190	88.982	96.359	88.980	96.288
350	89.699	96.234	89.730	96.358	89.677	96.296
400	90.430	96.243	90.323	96.361	90.294	96.290
450	90.738	96.199	90.732	96.353	90.794	96.283
500	91.229	96.167	91.259	96.340	91.156	96.263
550	91.608	96.119	91.495	96.316	91.510	96.246
600	91.828	96.133	91.833	96.315	91.848	96.241
650	92.210	96.132	92.287	96.320	92.142	96.237
700	92.430	96.128	92.485	96.317	92.397	96.237
750	92.703	96.124	92.716	96.316	92.653	96.236
800	92.793	96.134	92.911	96.316	92.893	96.236
850	93.096	96.132	93.122	96.321	93.070	96.228
900	93.253	96.133	93.267	96.315	93.295	96.231
950	93.421	96.130	93.404	96.315	93.437	96.229
1000	93.719	96.140	93.635	96.317	93.577	96.231

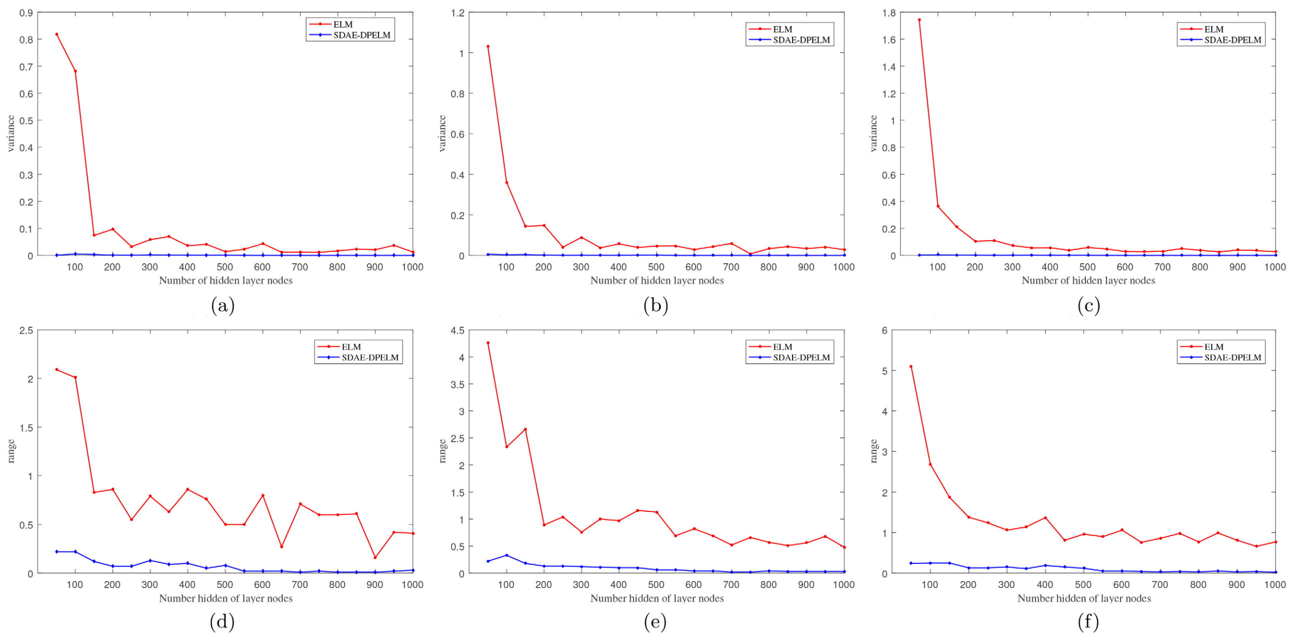

Figure 8. Variation of range and variance of ELM algorithm and DPLEM algorithm. (a) t = 10. (b) t = 20. (c) t = 50. (d) t = 10. (e) t = 20. (f) t = 50.

4.6. Recognition of Xiangxi Block Miao Characters Based on SDAE-DPELM

Xiangxi block Miao script is a written symbol created by Miao literati in the late Qing Dynasty to record, sort out and create Miao songs, which is still widely used in the Xiangxi Miao area. Xiangxi block Miao character recognition is to automatically recognize Miao characters written on a paper carrier by using computer and image processing technology, which is of great practical significance for the digital preservation of Miao characters. This section will use the new neural network SDAE-DPELM algorithm proposed in this paper to further explore the application of Xiangxi block Miao text recognition.

In this paper, we randomly select some characters from the Miao language library for the experiment. To make it easier for readers to read, we selected a portion from the original data set, as shown in Figure 9.

This experiment uses the chosen Miao character dataset to examine the highest correct recognition rate that both algorithms can obtain with the number of neurons in the hidden layer between 1 and 1000 to validate the performance of SDAE-DPELM in Miao language recognition.

The experimental results are shown in Table 2. The network structure of the SDAE-DPELM algorithm is more condensed than that of the original ELM algorithm, and it also achieves superior classification performance even with fewer hidden layer nodes. It is worth mentioning that the network structure of SDAE-DPELM algorithm given in Table 2, in which the input vector of the first layer is 2704, is the feature vector obtained by scaling the original Xiangxi square character picture and then extracting its features by LGS operator. The second layer of 2375 neurons is a deep feature vector obtained by further deep feature extraction

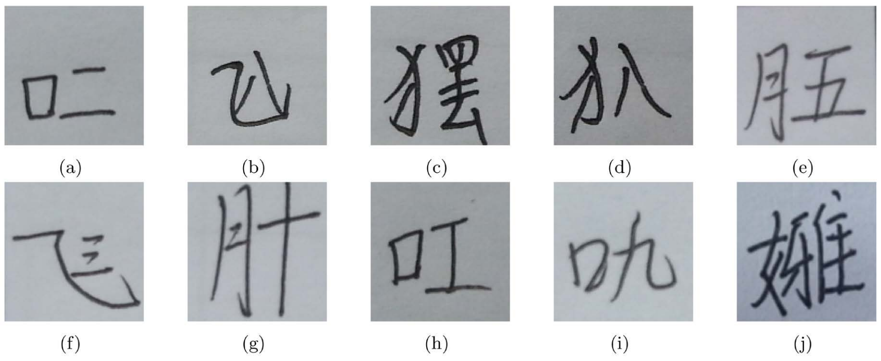

Figure 9. Some Miao characters in the original sample.

Table 2. Performance comparison of two algorithms in Miao character recognition.

Algorithm	Network structure	Accuracy of recognition
ELM	2704-1000-10	90%
SDAE-DPELM	2704-2375-30-10	80%

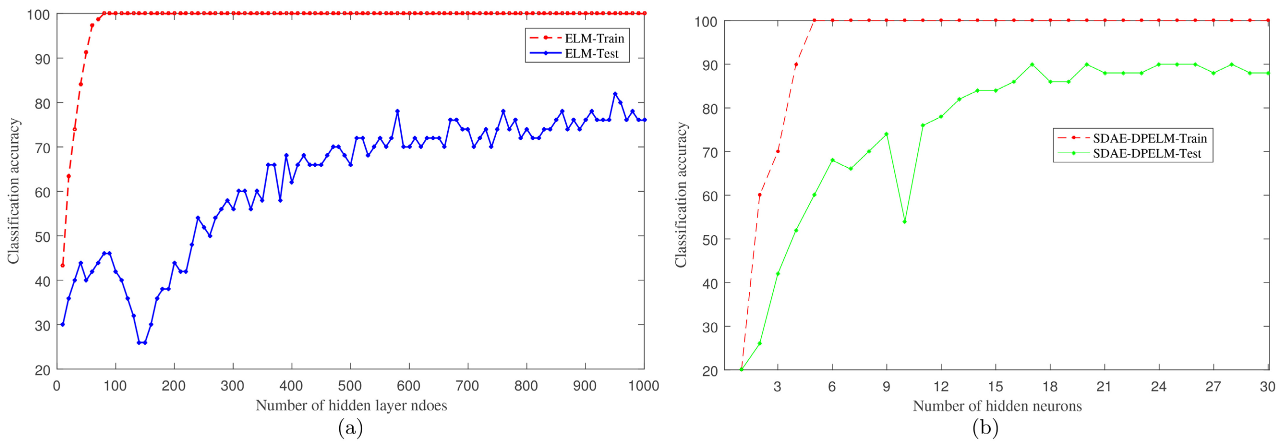

Figure 10. Comparison of recognition results of two algorithms. (a) ELM Algorithm. (b) SDAE-DPELM Algorithm.

by the SDAE. The following 30 (1000 in SDAE-ELM and ELM algorithm) is the number of neurons in the hidden layer required by the classifier. 10 represents the dimension of the output vector. Ten different output results can be represented.

As shown in **Figure 10**, this paper proposed algorithm can achieve higher accuracy than the original ELM algorithm with fewer hidden layer nodes. This provides an effective and practical tool for Xiangxi block Miao character recognition.

5. Conclusion

In this paper, a new deep learning algorithm SDAE-DPELM is proposed by combining SDAE with DPELM. SDAE is used to extract more representative deep abstract features of data, and the extracted features are used as input data of DPELM, and then network training is carried out. It not only overcomes the disadvantages of ELM's complex network structure and weak robustness caused by random assignment of hidden layer parameters, but also retains the advantages of ELM's fast operation speed. The experimental results show that the SDAE-DPELM algorithm has good robustness and stability. The practicality of SDAE-DPELM is also verified in the application of Miao character recognition. At present, the double-pseudo inverse weight determination method is only used for the traditional ELM. In future work, we will consider combining this weight determination method with ensemble learning to apply to the field of data prediction and image recognition.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant No. 62066015, the Natural Science Foundation of Hunan Province of China under Grant No. 2020JJ4511.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Huang, G.B., Zhu, Q.Y. and Siew, C.K. (2004) Extreme Learning Machine: A New Learning Scheme of Feedforward Neural Networks. 2004 *IEEE International Joint Conference on Neural Networks*, Vol. 2, 985-990. https://doi.org/10.1109/IJCNN.2004.1380068

[2] Jin, W., Li, Z.J., Wei, L.S. and Zhen, H. (2000) The Improvements of BP Neural Network Learning Algorithm. 5*th International Conference on Signal Processing Proceedings*. 16*th World Computer Congress* 2000, Vol. 3, 1647-1649.

[3] Xu, R., Liang, X., Qi, J.S., Li, Z.Y. and Zhang, S.S. (2019) Advances and Trends in Extreme Learning Machine. *Chinese Journal of Computers*, **42**, 1640-1670.

[4] Huang, G.B. and Wang, D.H. and Lan, Y. (2011) Extreme Learning Machines: A Survey. *International Journal of Machine Learning and Cybernetics*, **2**, 107-122. https://doi.org/10.1007/s13042-011-0019-y

[5] Chyzhyk, D., Savio, A. and Graa, M. (2015) Computer Aided Diagnosis of Schizophrenia on Resting State fMRI Data by Ensembles of ELM. *Neural Networks*, **68**, 23-33. https://doi.org/10.1016/j.neunet.2015.04.002

[6] Wang, G.H., Li, S.M., Zhu, D. and Yang, J. (2014) Application of Extreme Learning Machine in Objective Stereoscopic Image Quality Assessment. *Journal of Optoelectronics Laser*, **25**, 1837-1842.

[7] Huang, Z., Yu, Y. and Liu, H. (2016) An Efficient Method for Traffic Sign Recognition Based on Extreme Learning Machine. *IEEE Transactions on Cybernetics*, **47**, 920-933. https://doi.org/10.1109/TCYB.2016.2533424

[8] Zhang, W.B., Ji, H.B., Wang, L. and Zhu, M.Z. (2014) Multiple Hidden Layer Output Matrices Extreme Learning Machine. *Systems Engineering and Electronics*, **36**, 1656-1659.

[9] Zhang, Y., Wu, J., Cai, Z., Zhang, P. and Chen, L. (2013) Memetic Extreme Learning Machine. *Pattern Recognition*, **58**, 135-148. https://doi.org/10.1016/j.patcog.2016.04.003

[10] Rong, H.J., Ong, Y.S., Tan, A.H. and Zhu, Z. (2008) A Fast Pruned-Extreme Learning Machine for Classification Problem. *Neurocomputing*, **72**, 359-366. https://doi.org/10.1016/j.neucom.2008.01.005

[11] Pratama, M., Zhang, G., Er, M.J. and Anavatti, S. (2016) An Incremental Type-2 Meta-Cognitive Extreme Learning Machine. *IEEE Transactions on Cybernetics*, **47**, 339-353. https://doi.org/10.1109/TCYB.2016.2514537

[12] Chen, L., Huang, Z., Li, Y., Zeng, N., Liu, M., Peng, A. and Jin, L. (2019) Weight and Structure Determination Neural Network Aided with Double Pseudoinversion for Diagnosis of Flat Foot. *IEEE Access*, **7**, 33001-33008. https://doi.org/10.1109/ACCESS.2019.2903634

[13] Cambria, E. and Huang, G. (2013) Extreme Learning Machines-Representational Learning with ELMs for Big Data. *IEEE Intelligent Systems*, **28**, 30-59. https://doi.org/10.1109/MIS.2013.140

[14] Sun, K., Zhang, J., Zhang, C. and Hu, J. (2017) Generalized Extreme Learning Machine Autoencoder and a New Deep Neural Network. *Neurocomputing*, **230**, 374-381. https://doi.org/10.1016/j.neucom.2016.12.027

[15] Ng, A. (2011) Sparse Autoencoder. *CS*294*A Lecture Notes*, **72**, 1-19.