Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 5588426, 11 pages
https://doi.org/10.1155/2021/5588426

Research Article

Pruning Multilayered ELM Using Cholesky Factorization and Givens Rotation Transformation

Jingyi Liu,[1] Xinxin Liu,[2] Chongmin Liu,[2] Ba Tuan Le,[3] and Dong Xiao[2]

[1]*College of Sciences, Northeastern University, Shenyang 110819, China*
[2]*Information Science and Engineering School, Northeastern University, Shenyang 110819, China*
[3]*Control, Automation in Production and Improvement of Technology Institute (CAPITI), Hanoi 100000, Vietnam*

Correspondence should be addressed to Dong Xiao; xiaodong@ise.neu.edu.cn

Received 20 January 2021; Revised 27 March 2021; Accepted 5 April 2021; Published 27 April 2021

Academic Editor: Sotiris B. Kotsiantis

Extreme learning machine is originally proposed for the learning of the single hidden layer feedforward neural network to overcome the challenges faced by the backpropagation (BP) learning algorithm and its variants. Recent studies show that ELM can be extended to the multilayered feedforward neural network in which the hidden node could be a subnetwork of nodes or a combination of other hidden nodes. Although the ELM algorithm with multiple hidden layers shows stronger nonlinear expression ability and stability in both theoretical and experimental results than the ELM algorithm with the single hidden layer, with the deepening of the network structure, the problem of parameter optimization is also highlighted, which usually requires more time for model selection and increases the computational complexity. This paper uses Cholesky factorization strategy and Givens rotation transformation to choose the hidden nodes of MELM and obtains the number of nodes more suitable for the network. First, the initial network has a large number of hidden nodes and then uses the idea of ridge regression to prune the nodes. Finally, a complete neural network can be obtained. Therefore, the ELM algorithm eliminates the need to manually set nodes and achieves complete automation. By using information from the previous generation's connection weight matrix, it can be evitable to re-calculate the weight matrix in the network simplification process. As in the matrix factorization methods, the Cholesky factorization factor is calculated by Givens rotation transform to achieve the fast decreasing update of the current connection weight matrix, thus ensuring the numerical stability and high efficiency of the pruning process. Empirical studies on several commonly used classification benchmark problems and the real datasets collected from coal industry show that compared with the traditional ELM algorithm, the pruning multilayered ELM algorithm proposed in this paper can find the optimal number of hidden nodes automatically and has better generalization performance.

1. Introduction

Extreme learning machine (ELM) was first proposed by Huang and has attracted extensive attentions for its extremely fast learning speed, least human intervention, and easy implementation [1–3]. In the ELM algorithm, the input weights and hidden biases are randomly generated from any continuous probability distribution, and then, the output weights can be solved using the generalized Moore–Penrose inverse. Compared with the BP neural network, this algorithm has a good performance network in regression [4–6], classification [7–9], feature learning [10–12], and cluster tasks [13–15]. Different from conventional gradient-based neural network learning algorithms, which are sensitive to the combination of parameters and easy to trap in local optimum, ELM not only has a faster training speed but also has a smaller training error. The learning theories of ELM show that it can maintain the universal approximation and classification capability of SLFNs even if it works with randomly generated hidden nodes. However, because the essence of the extreme learning machine is the empirical risk minimization model and the randomness of the input weights and hidden biases, the extreme learning machine requires more hidden nodes than the BP neural network. Inspired by the structural risk minimization (SRM) principle of statistical learning theory and the weighted least square

method, Deng et al. [16] proposed a regularized extreme learning machine (RELM). On the basis of regularization, Zhou et al. [17] used Cholesky factorization to solve the regularized extreme learning machine and recursive least squares method for online learning. Chou's method implements online training by adding new training samples one by one and introduces the forgetting factor to reduce the influence of old training samples, so as to highlight the role of new training samples.

Although RELM has better robustness, it does not solve how to obtain the most suitable network structure like ELM. When there are too few hidden nodes in the neural network, the phenomenon of under fitting will appear, and when there are too many hidden nodes, there will be over fitting. Therefore, suitable hidden nodes are particularly important for neural networks. Rong et al. [18] proposed a pruning extreme learning machine. This method first defines a large amount of hidden nodes for the network, then prunes the label according to the correlation degree of each node, and finally obtains a suitable network. Martinez-Martinez et al. [19] used regularization to prune the hidden nodes. The initial stage of this method is the same as the pruning extreme learning machine, which is to first define a large number of hidden nodes for the network. The difference is that the method proposed by Martinez uses ridge regression to prune the network structure, thereby pruning redundant nodes.

When the input of the extreme learning machine contains a lot of noise, the fitting effect is often not ideal. Therefore, Miche et al. [20] proposed an optimal pruning extreme learning machine (OP-ELM) based on the correlation of input variables. This method uses the leave-one-out criterion to determine whether the node is left or not, which can effectively reduce the sensitivity to noise and obtain a more concise network model. Miche et al. [21] proposed a new and improved ELM (TROP-ELM) on the basis of the optimal pruning extreme learning machine. TROP-ELM adds L1 and L2 penalty terms to solve the OP-ELM instability in the process of solving the residual sum of squares. Indeed, the first step in the improvement process performs a L1 penalty on the output layer to rank the hidden nodes by implementing a LARS and then followed sequentially by a L2 penalty applied on the L1 penalized production from the LARS, thus avoiding numerical instabilities and efficient pruning of the network accordingly.

In recent years, due to the successful application of deep learning in many practical problems, academic research on ELM has shifted from single-layer networks to multilayer networks. The multilayered extreme learning machine (MELM) [22] is a new framework that attempts to integrate the characteristics of deep neural networks with a multilayer structure into ELM, but it also brings huge hidden layers of MELM that need to be predefined manually. Therefore, the exploration of the neural network structure is still an important issue. Moreover, the Moore–Penrose generalized inverse matrix is involved in the process of using the least squares method to solve the output matrix. If the matrix is not full rank, it will often cause numerical calculation errors.

Aiming at the above open problem of MELM, this paper proposes a pruning multilayer ELM algorithm framework (P-MELM). The algorithm adds the Tikhonov regularization factor in the process of calculating the output weights to achieve efficient pruning of nodes and then uses fast matrix decomposition strategy to automatically learn the optimal number of nodes. Since a lot of operations are involved in the pruning process of hidden nodes, this paper proposes a more stable numerical operation method, which uses Cholesky factorization of the covariance matrix and Givens rotation transformation to calculate pseudo inverse. This method can avoid large-scale matrix inversion, so compared with the direct calculation of the least square solution, this method needs less computing time and memory.

The chapters of this article are arranged as follows. Section 1 reviews the improvements made by previous generations to ELM. Section 2 mainly introduces MELM. Section 3 proposes and introduces our algorithm P-MELM. Section 4 establishes a model and compares it with other algorithms. Section 5 summarizes the full text.

2. Multilayered ELM (MELM)

Suppose there are N arbitrarily different samples $\{X, T\} = \{x_i, t_i\}$ $(i = 1, 2, \ldots, N)$, where $X = [x_1, x_2, \ldots, x_N]^T \in R^{N\times n}$ is the input sample, $T = [t_1, t_2, \ldots, t_N]^T \in R^{N\times m}$ is the corresponding sign sample, and $G(x) = 1/(1 + e^{-x})$ is the selected activation function. First, using the principle of random sampling, a value is randomly selected within the range of [0-1] to become the input weight $W_{1,L} = [w_{1,1}, w_{1,2}, \ldots, w_{1,L}]^T \in R^{L\times n}$ of the first hidden layer and the threshold $B_{1,L} = [b_{11}, b_{12}, \ldots, b_{1L}]^T \in R^{L\times N}$ of the first hidden layer. The standard mathematical formula for MELM is as follows:

$$\sum_{j=1}^{L} \left(\beta_{1,L}\right)_j G\left(w_{1,j}x_i + b_{1j}\right) = t_i, \quad i = 1, 2, \ldots, N. \tag{1}$$

By using the ELM algorithm to write formula (1) in the matrix form, we can get

$$H_{1,L}\beta_{1,L} = T, \tag{2}$$

where

$$\begin{aligned} H_{1,L} &= G\left(W_{1,L}X + B_{1,L}\right), \\ &= \begin{bmatrix} G\left(w_{1,1}x_1 + b_{11}\right) & \cdots & G\left(w_{1,L}x_1 + b_{1L}\right) \\ \vdots & \vdots & \vdots \\ G\left(w_{1,1}x_N + b_{11}\right) & \cdots & G\left(w_{1,L}x_N + b_{1L}\right) \end{bmatrix}, \\ &= [\, h_{1,1} \ \cdots \ h_{1,L} \,], \end{aligned} \tag{3}$$

$$\beta_{1,L} = \left[\left(\beta_{1,L}\right)_1, \left(\beta_{1,L}\right)_2, \ldots, \left(\beta_{1,L}\right)_L\right]^T. \tag{4}$$

As described in RMELM [23], the output matrix of the first hidden layer in MELM is defined as $H_{1,L} \in R^{N\times L}$, whose elements $(h_{1,j})_i = G(w_{1,j}x_i + b_{1j})$ $(i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, L)$ may be explained as the output of the jth node in

the first hidden layer in reference to x_i, where $h_{1,j} = \left[g(w_{1,j}x_1 + b_{1j}) \ \cdots \ g(w_{1,j}x_N + b_{1j}) \right]^T$ $(j = 1, 2, \ldots, L)$.

Unlike the conventional learning algorithm for SLFNs, the MELM approach is thus to generate randomly the weights $w_{1,j}$ and the bias b_{1j} in the first hidden layer and determine analytically the connection weights $\beta_{1,L} \in R^{L\times m}$ between the first hidden layer and the output layer using Moore–Penrose generalized inverse, which is identical to finding the minimum norm least square solution of the linear system, and therefore, representation (2) can be written as

$$\beta_{1,L} = H_{1,L}^{+}T, \tag{5}$$

where $H_{1,L}^{+}$ is the Moore–Penrose (MP) pseudoinverse of the matrix $H_{1,L}$. If $(H_{1,L})^T H_{1,L}$ is nonsingular, then $(H_{1,L})^{+} = (H_{1,L}^T H_{1,L})^{-1} H_{1,L}^T$, or alternatively $H_{1,L}^{+} = H_{1,L}^T (H_{1,L} H_{1,L}^T)^{-1}$ if $H_{1,L}H_{1,L}^T$ is nonsingular.

According to the solving process of the MELM algorithm, we can get

$$H_{2,L} = g\left(W_{2,L}H_{1,L} + B_{2,L}\right), \tag{6}$$

where $W_{2,L}$ represents the weights' matrix that links the first hidden layer to the second hidden layer. The matrices $B_{2,L}$ and $H_{2,L}$, respectively, denote the bias and the expected output of the second hidden layer. Moreover, the output matrix $H_{2,L}$ is made on the basis of the multilayer theory and can be rewritten as

$$H_{2,L} = T\beta_{1,L}^{+}, \tag{7}$$

where $(\beta_{1,L})^{+}$ is the MP pseudoinverse of the matrix $\beta_{1,L}$, obtained using the approach discussed before, namely, $(\beta_{1,L})^{+} = (\beta_{1,L}^T \beta_{1,L})^{-1} \beta_{1,L}$ if $\beta_{1,L}^T \beta_{1,L}$ is nonsingular; otherwise, $(\beta_{1,L})^{+} = \beta_{1,L}^T (\beta_{1,L}\beta_{1,L}^T)^{-1}$ if $\beta_{1,L}\beta_{1,L}^T$ is nonsingular.

The augmented matrix is then defined as $W_{2HE,L} = \left[B_{2,L} \ \ W_{2,L} \right]$ and computed as follows:

$$W_{2HE,L} = g^{-1}\left(H_{2*,L}\right) H_{2E,L}^{+}, \tag{8}$$

where $H_{2E,L}^{+}$ represents the pseudoinverse of the matrix $H_{2E,L} = \left[1 \ H_{1,L} \right]^T$, $1 = [1, 1, \ldots, 1]^T \in R^N$, and $g^{-1}(x)$ represents the inverse function of $g(x)$. The calculation of $H_{2E,L}^{+}$ is the same as the previous description of $H_{1,L}^{+}$. Subsequently, we can calculate the actual output matrix of the second hidden layer as

$$H_{2,L} = g\left(W_{2,L}H_{1,L} + B_{2,L}\right) = g\left(W_{2HE,L}H_{2E,L}\right). \tag{9}$$

Therefore, $\beta_{2,L}$ can be updated as follows:

$$\beta_{2,L} = H_{2,L}^{+}T. \tag{10}$$

The calculating method of $H_{2,L}$ proceeds in the form discussed before. If MELM has only two hidden layers, the final output of MELM can be expressed as

$$t(x) = H_{2,L}\beta_{2,L}. \tag{11}$$

When the number of hidden layers of MELM is greater than 2, as indicated in (5)–(10), an iterative solution scheme is adopted to implement the calculation process. Then, all the related parameters of these hidden layers can be obtained after the (M-2)-iteration operation. To make the final actual hidden layer output close to the expected hidden layer output, in the training stage, the MELM designs a novel parameter-setting step for the newly added hidden layer, as described in (8), which is the key step to guarantee the stability and feasibility of the MELM model.

3. Pruning Multilayered ELM (P-MELM)

This newly modified approach is in the form of a L2 regularization penalty applied within the MELM. First, the regularization method can constrain the norm of the output weights to avoid the possible numerical instabilities and prevent the model from overfitting. Second, the regularization term makes the coefficient matrix of linear equations positive definite and thus uses a fast matrix factorization strategy to recursively update the output weights in the calculation of the pruning criterion, which provides more accurate and reliable computation results. The pruning multilayered ELM algorithm based on the Cholesky factorization and Givens rotation transformation are deduced in this section, which avoids the complexities of matrix inversion.

3.1. The Solutions of P-MELM by Cholesky Factorization and Givens Rotation Transformation. Previous research has shown that due to the randomly generated parameters of the hidden nodes, MELM typically requires more hidden nodes than conventional gradient descent methods to provide desired performance levels. This essence of MELM clearly brings faster learning speed, least human intervention, and easy implementation. However, it suffers from performance problems connected with too many hidden nodes chosen at random, that is, the lack of sparsity and adjustment ability in the hidden layers. Furthermore, another common risk is that the network will produce some redundant nodes that add little or nothing to the existing efforts to respond to the model training. As a result of all these trends, the initial output matrix may not be a full column rank or even ill-conditioned; then, the exact analytical solution of the output weights can hardly be found, which directly affects the generalization capability and stability of the network. Aiming at this problem, it is proposed here to modify the pruning strategy for the selection of the optimal number of nodes by taking heuristic techniques to tentatively prune out the redundant nodes of the model that will decrease the computational complexity of providing an accurate ranking of the hidden layer nodes and raise the efficiency of the network learning and the adaptability to real problem resolving. In the following, the most well-known algorithms used to perform regularization are reviewed, using a L_2 penalty in the MELM, to regularize the network.

It is assumed that the initial value of hidden nodes is set as $L = L_{\max}$, the output matrix of the first hidden layer is

$H_{1,L}$, and the connection weights' matrix between the first hidden layer and the output layer is $\beta_{1,L}$. Through studying the training dataset, the optimal number of hidden nodes and hidden layers are determined, which use the P-MELM prediction model for a framework and employ the Cholesky factorization method and Givens rotation transformation to recursively calculate the factorization factor U_L. As the number of hidden nodes decreases, the new connection weights' matrix $\beta_{1,L+1}$ can be obtained rapidly according to the parameter information of the existing network. More specifically, if we reduce the total number of hidden nodes from L to $L-1$, the first hidden layer output matrix of P-MELM will change from $H_{1,L} \in R^{N\times L}$ to $H_{1,L-1} \in R^{N\times(L-1)}$, which can be intuitively represented as the following:

$$H_{1,L} = \left[\, h_{1,1} \;\vdots\; h_{1,2} \;\cdots\; h_{1,L} \,\right] = \left[\, h_{1,1} \;\vdots\; H_{1,L-1} \,\right], \tag{12}$$

where $h_{1,j} = \left[\, G(w_{1,j}x_1 + b_{1j}) \;\cdots\; G(w_{1,j}x_N + b_{1j}) \,\right]^T$ $(j = 1, 2, \ldots, L)$. According to (12), we can get

$$\begin{aligned} A_L &= CI_L + H_{1,L}^T H_{1,L}, \\ &= C\begin{bmatrix} 1 & 0 \\ 0 & I_{L-1} \end{bmatrix} + \begin{bmatrix} h_{1,1}^T \\ H_{1,L-1}^T \end{bmatrix}\left[\, h_{1,1} \;\; H_{1,L-1} \,\right], \\ &= \begin{bmatrix} C + h_{1,1}^T h_{1,1} & h_{1,1}^T H_{1,L-1} \\ H_{1,L-1}^T h_{1,1} & CI_{L-1} + H_{1,L-1}^T H_{1,L-1} \end{bmatrix}, \\ &= \begin{bmatrix} P_{L-1} & Q_{L-1} \\ Q_{L-1}^T & A_{L-1} \end{bmatrix}, \end{aligned} \tag{13}$$

where $I_L \in R^{L\times L}$ and $I_{L-1} \in R^{(L-1)\times(L-1)}$ denote the identity matrices of order L and (L-1), respectively, $Q_{L-1} = \left[\, h_{1,1}^T h_{1,2} \;\cdots\; h_{1,1}^T h_{1,L} \,\right]$, and $P_{L-1} = C + h_{1,1}^T h_{1,1}$. Due to the tight coupling that exists between A_{L-1} and A_L, as shown in (13) above, the Cholesky factorization U_{L-1} of matrix A_{L-1} can be analytically and easily determined based on the Cholesky factorization theorem [24]. The recursive calculation of matrix factorizations is implemented as follows. First, the matrices A_L and U_L^T are partitioned into four submatrices, and their corresponding block matrices have the same size. For convenience, the matrix U_L^T can thus be rewritten as

$$U_L^T = \begin{bmatrix} v & V^T \\ 0^T & W^T \end{bmatrix}, \tag{14}$$

where V^T is an $(L-1)$-dimensional row vector, W^T is an $(L-1)\times(L-1)$ square matrix, and v is a scalar. Substituting formula (14) to (13), A_{L-1} can be solved as

$$A_{L-1} = VV^T + WW^T = U_{L-1}U_{L-1}^T. \tag{15}$$

Next, the rank one update algorithm [25] is used to calculate the Cholesky factorization factor, so U_{L-1}^T can be obtained from the equation of the form:

$$J(L-1)J(L-2)\cdots J(k)\cdots J(1)\left[\, V \;\; W \,\right]^T = \left[\, 0 \;\; U_{L-1} \,\right]^T_{L\times(L-1)}. \tag{16}$$

Here, $J(k)$ is the Givens rotation transformation matrix, which can be expressed in the matrix form as

$$J(k) = I + (c_k - 1)\left(\delta_0\delta_0^T + \delta_k\delta_k^T\right) + s_k\left(\delta_0\delta_k^T - \delta_k\delta_0^T\right), \tag{17}$$

where $c_k = \cos\theta_k = W_{kk}^T/\sqrt{(W_{kk}^T)^2 + (V_k^T)^2}$, $s_k = \sin\theta_k = V_k^T/\sqrt{(W_{kk}^T)^2 + (V_k^T)^2}$, and $k = 1, 2, \ldots, L-1$.

It is worth noting that, in (16), the matrix V and W change with the rank one update algorithm proceeding, so the calculation process of (17) does not need to explicitly record the matrix $J(k)$. Meanwhile, as shown in (12), B_L can be represented as

$$\begin{aligned} B_L &= H_{1,L}^T T = \left[\, h_{1,1} \;\vdots\; H_{1,L-1} \,\right]^T T, \\ &= \begin{bmatrix} h_{1,1}^T T \\ H_{1,L-1}^T T \end{bmatrix} = \begin{bmatrix} h_{1,1}^T T \\ B_{L-1} \end{bmatrix}. \end{aligned} \tag{18}$$

If the number of hidden nodes decreases, the connection weights' matrix $\beta_{1,L-1}$ between the first hidden layer and the output layer can be directly and analytically determined by solving the following equations:

$$A_{L-1}\beta_{1,L-1} = B_{L-1}. \tag{19}$$

Then, substitute formula (15) into (19) and multiply U_{L-1}^{-1} of both sides, and the problem seeking for $\beta_{1,L-1}$ can thus be formulated as the following:

$$U_{L-1}^T\beta_{1,L-1} = F_{L-1}, \tag{20}$$

where $F_{L-1} = U_{L-1}^{-1}B_{L-1}$. For convenience, $F_{L-1} = U_{L-1}^{-1}B_{L-1}$ is intuitively transformed as

$$U_{L-1}F_{L-1} = B_{L-1}, \tag{21}$$

and it can be shown that F_{L-1} is associated with U_{L-1} and B_{L-1}, which are indeed obtained by solving the linear equation system. Accordingly, the optimal output weight matrix $\beta_{1,L-1}$ can be calculated simultaneously using the corresponding entries of matrices U_{L-1} and F_{L-1}.

3.2. Decremental Learning Procedure of P-MELM. The process builds an optimum network structure, known as decremental learning of P-MELM, through the calculation of the corresponding Cholesky factorization factor U_L. It can be seen from formulas (14)–(20) that the P-MELM algorithm adopts a fast calculation format and can achieve a decreasing update of the connection weight matrix. The solution method of $\beta_{1,L-1}$ based on Cholesky factorization and Givens rotation transformation makes full use of the information stored in the calculation of $\beta_{1,L}$, where U_{L-1} can be obtained through Givens rotation transformation on the basis of U_L, and B_{L-1} can be obtained directly on the basis of B_L. Therefore, if the number of hidden nodes decreases successively, $\beta_{1,L-1}$ can be rapidly calculated by simple matrix arithmetic operation on the basis of computing $\beta_{1,L}$. In this case, if $\beta_{1,L-1}$ is calculated by using the method shown in (5), it must be recalculated in the way of finding the

inverse of the higher-order matrix, and it cannot be directly solved on the basis of computing $\beta_{1,L}$. Therefore, the P-MELM algorithm can guarantee the learning accuracy and further improve the training speed. We have summarized the steps of the P-MELM algorithm as follows:

Step 1: first, we limit the number of nodes. The maximum number of nodes is $L_{\max}$, the minimum number of nodes is $L_{\min}$, and the iteration stopping criterion is ξ_L. Then, we set the initial number of hidden nodes as $L = L_{\max}$ and calculate A_L and B_L.

Step 2: calculate the Cholesky factorization U_L of A_L according to (13), and calculate F_L using the matrices U_L and B_L according to (21).

Step 3: calculate $\beta_{1,L}$ with the matrices U_L and F_L according to (20). Then, build a P-MELM model according to $\beta_{1,L}$. The network structure of the model has M hidden layers, and each hidden layer has L hidden layer nodes.

Step 4: the empirical risk and structural risk of the P-MELM algorithm are solved, and the sum of these two parts is carried out:

$$\begin{aligned} R_{1,L} &= \left\| H_{1,L}\beta_{1,L} - T \right\|^2 + C\left\| \beta_{1,L} \right\|^2, \\ &= \left(H_{1,L}\beta_{1,L} - T \right)^T \left(H_{1,L}\beta_{1,L} - T \right) + C\beta_{1,L}^T \beta_{1,L}. \end{aligned} \tag{22}$$

Step 5: let $L = L - 1$, calculate U_L and B_L on the basis of U_{L+1} and B_{L+1} according to the principle shown in (14)–(18), and then, go to Step 3. Start from $L = L_{\max} - 1$ to determine whether (23) is satisfied:

$$\left| \frac{R_{1,L+1} - R_{1,L}}{R_{1,\max}} \right| > \xi_L, \tag{23}$$

where $R_{1,\max}$ is the resulting maximum value of $R_{1,L}, \cdots, R_{1,L_{\max}}$. If the conditions of (23) are met, then modeling iteration is completed, the optimal number value of hidden nodes is confirmed as L, and the corresponding prediction model of P-MELM is built. Otherwise, if $L > L_{\min}$, continue to decrease L until the condition $L = L_{\min}$ is satisfied.

In the P-MELM algorithm flow, first, the neural network has a larger number of hidden nodes $L_{\max}$, and then, the number of nodes is gradually reduced until $R_{1,L}$ changes significantly. If the number of hidden nodes continues to decrease at this time, the generalization ability and fitting effect of P- P-MELM ELM will decrease along with the progress, and a large number of effective nodes will also be lost in the network. Therefore, when $R_{1,L}$ changes significantly, P-P-MELM ELM has the optimal number of hidden nodes.

3.3. Proposed P-MELM. P-MELM shows an obvious superiority in faster computations and numerical stability compared with the conventional MELM by introducing Cholesky factorization and Givens rotation transformation. This technique uses the basic arithmetic operation on matrix elements to construct weights matrix and avoids the complicated matrix inverse. In addition, when the hidden node is reduced one by one, the dynamically changing network structure needs to retrain the model by the existing training data, and this will lead to significantly longer model training times. Faced with this situation, a better alternative is to fully utilize information gained by the process of P-MELM training and implement the decremental updating of network parameters recursively.

First, train P-MELM with the dataset $\{X, T\}$ and then, use trained P-MELM to provide the prediction of the verifying data Z. The chosen neural network architecture consisted of M hidden layers, one input layer, and output layer; meanwhile, each of the hidden layers contains L hidden nodes. Then, the output of the standard MELM classifier with respect to the input x can be modeled as

$$t_{M,L}(x) = \sum_{j=1}^{L} \left(\beta_{M,L} \right)_j G\left(w_{M,j}, b_{Mj}, x \right), \quad x \in R^n. \tag{24}$$

If the number of training samples is $N >> L$, we set the initial value of hidden nodes as $L_{\max}$, the minimum number of hidden nodes as $L_{\min}$, the initial number of hidden layers as 1, and the maximum number of hidden layers as $M_{\max}$. ξ_L is expressed as the best accuracy of neural network learning.

Phase 1: initialization of the P-MELM network.

(1) Let $L = L_{\max}$ and $M = 1$. Randomly select the input weight $w_{1,j}$ and bias b_{1j} of the first hidden layer, $j = 1, \cdots, L$.

(2) Solve the output matrix $H_{1,L}$ of the first hidden layer according to (3), $H_{1,L} = G(w_{1,1}, \ldots, w_{1,L}, b_{11}, \ldots, b_{1L}, x_1, \ldots, x_N)$.

(3) Calculate the matrix A_L and B_L according to (13) and (18).

(4) Cholesky factorization of A_L according to (14) to get U_L.

(5) According to (21), calculate F_L by using the matrices U_L and B_L.

(6) According to (20), calculate $\beta_{1,L}$ by using the matrices U_L and F_L. Then, build a P-MELM model according to $\beta_{1,L}$. The network structure of the model has M hidden layers, and each hidden layer has L hidden layer nodes.

(7) According to (22), the empirical risk and structural risk of the P-MELM algorithm are obtained, and the two parts are summed.

Phase 2: Update P-MELM neural network sturctural parameters (the number of hidden nodes and hidden layers). While $L \geq L_{\min}$.

(1) Let $L = L - 1$. Calculate U_L and B_L on the basis of U_{L+1} and B_{L+1} according to the principle shown in (14)–(18), and then, go to Step 1 (5). If (23) is satisfied, then L is the optimal number of hidden nodes at this time. If (23) is not satisfied, then $L = L_{\max} - 1$ continues to solve.

(2) Let $M = M + 1$. Calculate the expected output matrix $H_{M,L}$ of the Mth layer according to (7), $H_{M,L} = T\beta^{+}_{M-1,L}$.

(3) Calculate the learning parameter $W_{MHE,L}$ in the Mth hidden layer according to (8), $W_{MHE,L} = g^{-1}(H_{M,L})H^{+}_{ME,L}$.

(4) Update the output matrix of the Mth hidden layer, $H_{M,L} = g(W_{MHE,L}H_{ME,L})$.

(5) According to (19)–(21), the connection weights between the Mth hidden layer and the output layer $\beta_{M,L}$ are updated.

(6) If $M = M_{\max}$, then stop training; otherwise, return to 2 (1), and continue to increase the number of hidden layer M until the condition is met.

(7) Update the output weight matrix $\beta_{M,L}$ and establish a P-MELM model. And, calculate the final output $t_{M,L}(x)$ of the P-MELM algorithm according to (24).Therefore, the predicted output of test instance Z can be represented as $t_{M,L}(Z)$.

If we set $M_{\max} = 1$ in the P-MELM approach, P-MELM becomes the pruning extreme learning machine (P-ELM). The flow chart of the P-MELM algorithm is shown in Figure 1. It can be seen from Figure 1 that when the number of hidden layer nodes in the P-MELM algorithm gradually decreases, P-MELM does not need to retrain the network and recalculate the weights, but updates the weights on the basis of the original network. MELM needs to recalculate the output weight every time once the number of hidden nodes is reduced. Therefore, when the number of hidden layer nodes decreases gradually, P-MELM is simpler and takes less time than MELM.

4. Algorithm Verification

In order to verify the stability of the P-MELM algorithm and P-ELM algorithm proposed in this paper, we have carried out experiments in MATLAB 2016a environment and carried out 100 experiments on each algorithm. Our experiment is divided into two parts. The first part is a simple benchmark classification [26], and the second part is to classify coal by using the spectral information of coal [27]. In addition, in order to comprehensively evaluate the performance of the P-ELM algorithm and P-MELM algorithm, the initial value of the number of nodes in a single hidden layer is set as 100 in the experiment, and the step size is set as 1 to delete redundant nodes. The number of layers of the multilayer extreme learning machine is set from 2 to 10.

4.1. Data Acquisition and Processing. We download two commonly used datasets of Diabetic Retinopathy Debrecen and image segmentation from the UCI Machine Learning Library to verify the reliability of the algorithm proposed in simple benchmark classification. In addition, when faced with extremely time-consuming and complex classification problems, researchers now tend to use machine learning methods to solve complex problems. Therefore, this paper also uses the spectral information of coal to classify coal to verify the reliability of P-ELM and P-MELM. The collected coal mine information is shown in Table 1.

We use the SVC-HR-1024 spectrometer produced by American Spectra Vista Company as the experimental instrument in the spectrum experiment. The spectral bandwidth of this instrument is 1024, and the spectral range is 350–2500 nm. In the experiment, firstly, the sample is cleaned. The spectrometer probe is 380 mm away from the surface of the ore sample, and as far as possible, it is set perpendicular to the sample surface. Secondly, the experimenters should reduce the walking process and wear dark clothes during the experiment to reduce the interference of environmental factors on the spectrum experiment. Then, the spectrometer was used to conduct 6 spectroscopic experiments on each sample of coal. Finally, summarize all the experimental data, and calculate the average of the six spectral experimental data of the same coal mine as the final spectral data. Figure 2 shows the process of data collection and processing, during which whiteboard measurement and calibration are carried out every 10 minutes. Because the original spectral data included 1024 bands, some of which are overlapped bands, it is necessary to remove the overlapped bands from the data, and finally, 973 useful bands are obtained.

4.2. Evaluation of Testing Accuracy. We use P-ELM and P-MELM to model the classification problem. Table 2 shows the parameters of P-ELM and P-MELM. Figures 3 and 4, respectively, show the accuracy of data classification by P-ELM and P-MELM under different hidden nodes. From Figures 3 and 4, we can see that P-ELM and P-MELM can obtain good learning accuracy and have a relatively accurate classification effect. If the redundant hidden nodes in the neural network are pruned, the testing accuracy of P-ELM and P-MELM enhance or remain unchanged, and this phenomenon is especially evident in Diabetic Retinopathy Debrecen, image segmentation, and coal spectral datasets. More importantly, the classification accuracy surfaces of P-MELM keep relatively smooth, but the curves of P-ELM have some oscillations in Diabetic Retinopathy Debrecen and image segmentation cases when the redundant hidden nodes are pruned. By comparing the algorithm process of P-ELM and P-MELM, it can be seen that P-MELM changes from a single hidden layer to multiple hidden layers on the basis of P-ELM, and the calculation method of the parameters in the P-MELM algorithm is different. So, P-MELM can eliminate the fluctuations caused by P-ELM to some extent. As a consequence, the P-MELM algorithm is more accurate in classification than the P-ELM algorithm, and the fitting effect is better.

4.3. Evaluation of Average Training Time. In Section 4.3, we compare the training time of MELM and P-MELM under the same parameters. The time required for MELM and P-MELM algorithms is shown in Figure 5. From Figure 5, we can see that whether it is coal mine classification, diabetic retinopathy classification, or image segmentation classification, and the training time required for P-MELM is much less than the

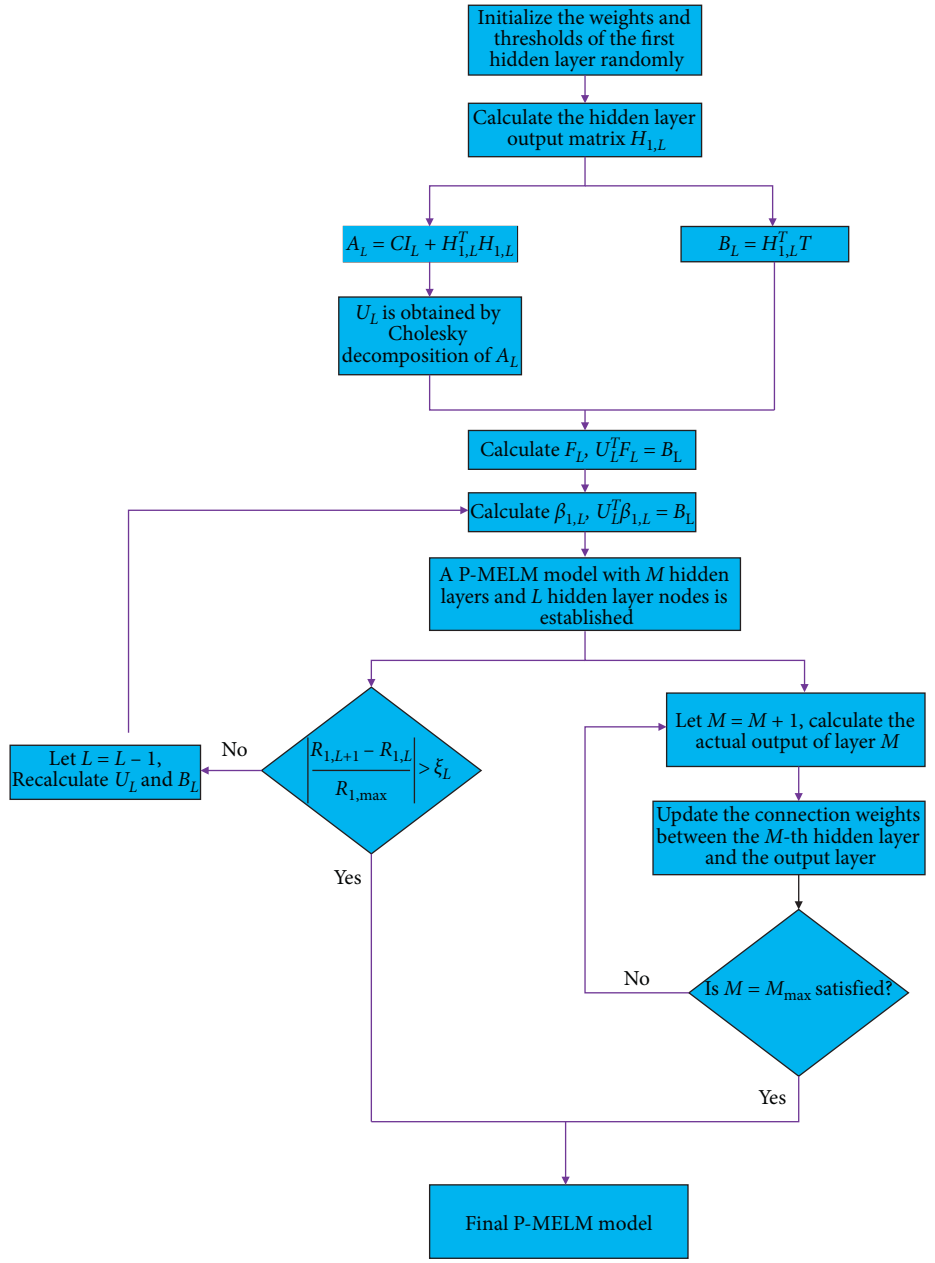


Figure 1: Flow chart of the P-MELM algorithm.

Table 1: Type and quantity of coal collected.

Mineral species	Attribute	Training data	Testing data
Coal	Anthracite	44	27
	Bituminous coal	50	30
	Lignite	36	22
	Coal gangue	66	40
	Total	196	119

training time required for MELM. This is because when the number of hidden nodes changes, MELM needs to retrain the network, and P-MELM does not need to recalculate the network parameters, but only updates the weights in the model. Therefore, under the same network structure, P-MELM has a faster convergence speed than MELM. Furthermore, the training time curves for several cases shown in Figure 4 also indicate that the spent learning time is still monotonically decreasing with the learning steps, which is consistent with MELM's conclusions. Compared with P-MELM, MELM needs more time to retrain the network and recalculate the output weights. However, with the increase of hidden layer nodes, the time difference of training time will gradually decrease.

4.4. Comparison of Testing Accuracy by Different Methods. The classification accuracy of the testing data is selected as the performance evaluation criterion for the algorithms investigated, namely, ELM, MELM, P-ELM, P-MELM, ELM-AE, and SELM. The best learning accuracy that six algorithms can achieve can be observed from Table 3. On the basis of these

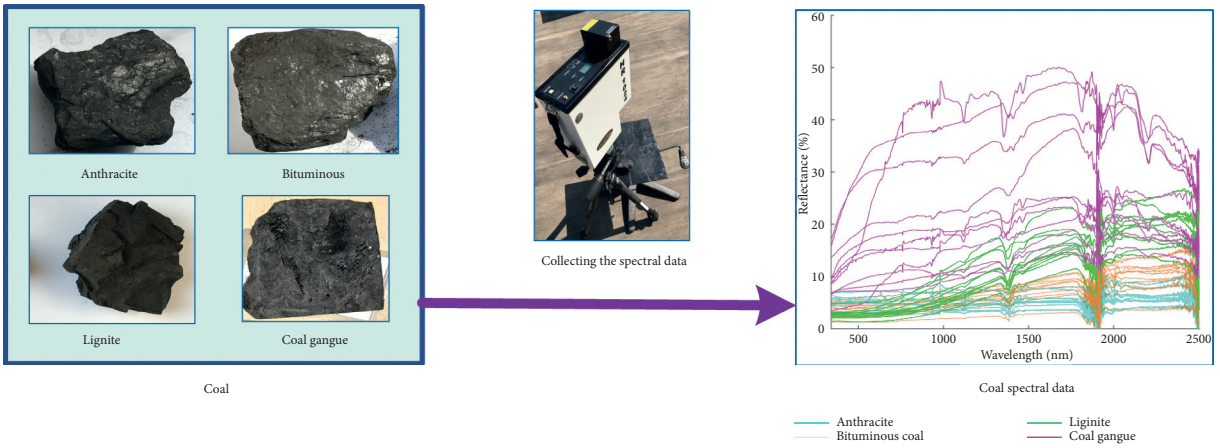


Figure 2: Spectrum experiment and spectrum curve of coal.

Table 2: Parameters of P-ELM and P-MELM under different classification problems.

Datasets	Algorithm	Initial nodes	End nodes	Hidden layers
Diabetic Retinopathy Debrecen	P-ELM	100	50	1
	P-MELM	100	80	2–10
Image segmentation	P-ELM	100	80	1
	P-MELM	80	55	2–10
Coal spectral data	P-ELM	100	75	1
	P-MELM	80	55	2–10

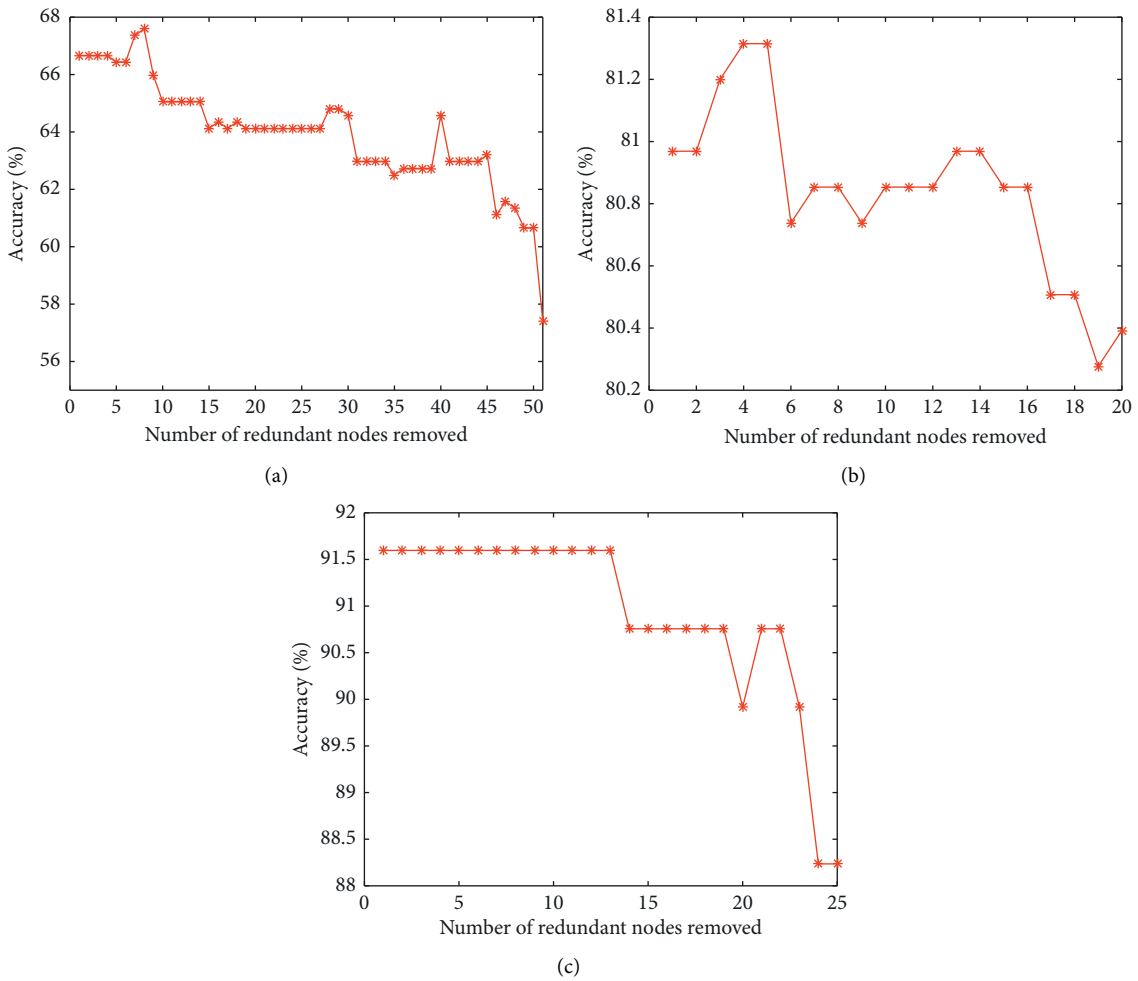


Figure 3: The average testing accuracy for the algorithm P-ELM using (a) Diabetic Retinopathy Debrecen, (b) image segmentation, and (c) coal spectral data, while the hidden node is decreased one by one.

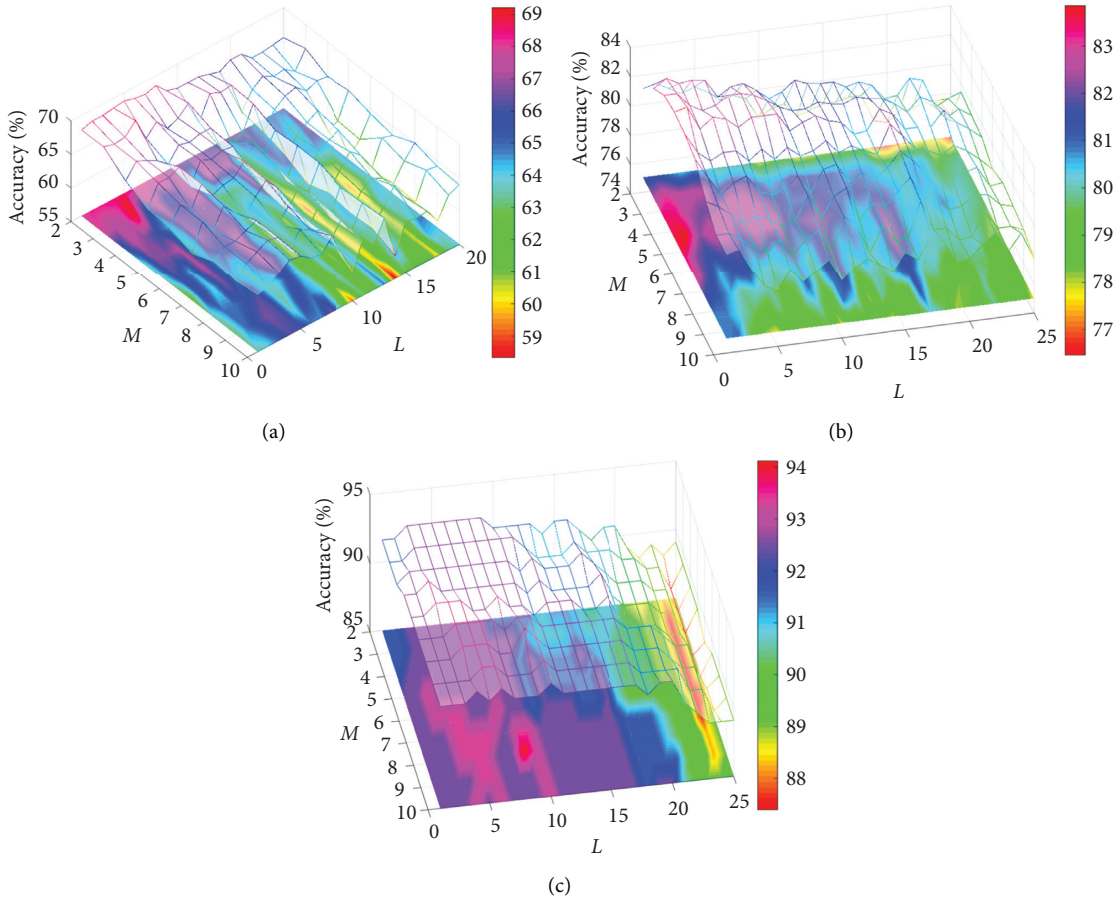


Figure 4: The average testing accuracy for the algorithm P-MELM with different hidden layers using (a) Diabetic Retinopathy Debrecen, (b) image segmentation, and (c) coal spectral data, while the hidden node is decreased one by one. Here, M and L are the number of hidden layers and the number of redundant nodes removed, respectively.

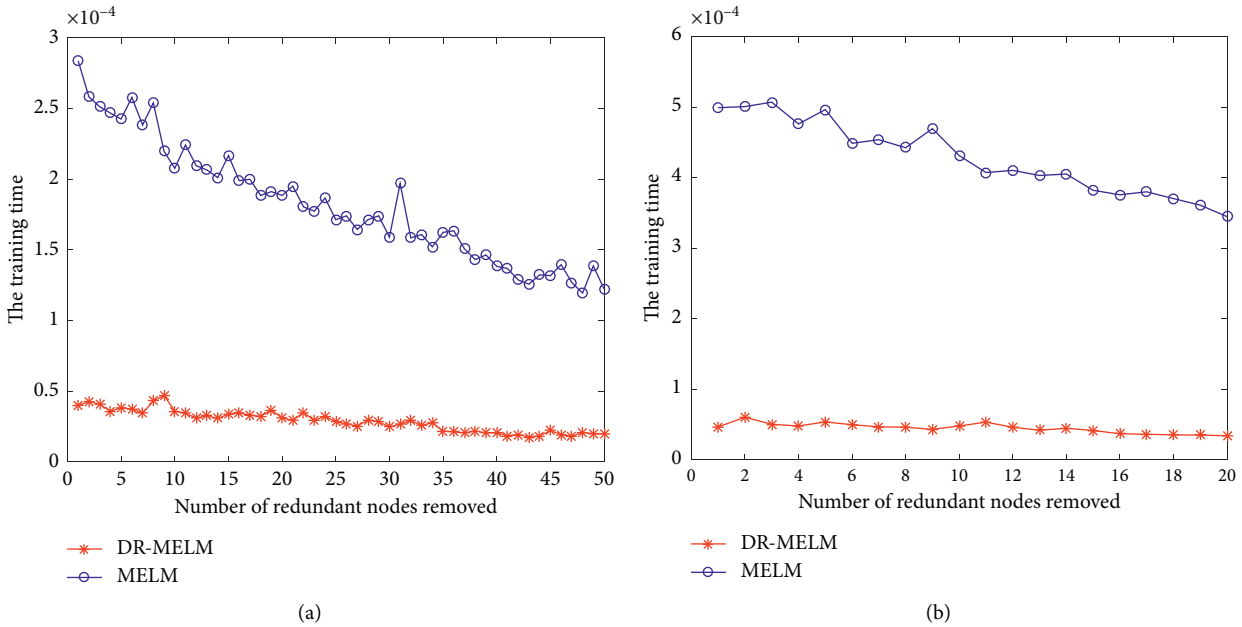
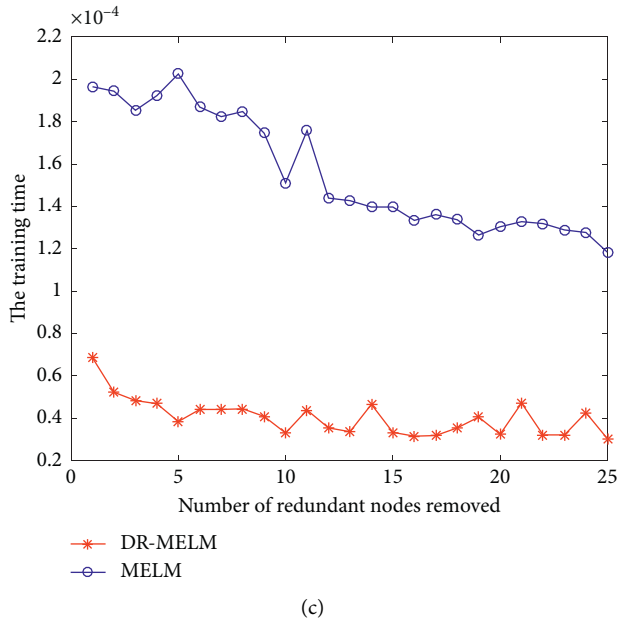


Figure 5: Continued.



(c)

Figure 5: Comparison of training time between the algorithms MELM and P-MELM using (a) Diabetic Retinopathy Debrecen, (b) image segmentation, and (c) coal spectral data, while the hidden node is decreased one by one.

Table 3: The best testing accuracy that the algorithms ELM, MELM, P-ELM, P-MELM, ELM-AE, and SELM can achieve.

Datasets	Best testing accuracy (%)					
	ELM	MELM	P-ELM	P-MELM	ELM-AE	SELM
Diabetic Retinopathy Debrecen	63.66	65.97	67.59	68.75	68.06	66.20
Image segmentation	75.08	75.89	81.31	83.04	82.4	93.0
Coal spectral data	89.91	89.91	91.60	92.24	91.60	67.23

experimental results, we can draw the following conclusions. If the initial network structure is fixed, P-ELM and P-MELM always get higher testing accuracy than ELM and MELM, respectively. This means that the P-ELM and P-MELM techniques can get more stable networks in most cases. What is more, a higher classification accuracy indicates a better classification algorithm performance. As observed from Table 3, it is obvious that the proposed P-MELM technique achieves the highest classification accuracy among all classification datasets, relative to the original ELM, MELM, and P-ELM algorithms. Finally, these results further prove that the P-ELM and P-MELM methods have a better classification effect than the ELM method and the training speed can meet the requirements.

5. Conclusions and Discussion

Based on MELM, this paper introduces the L2 penalty factor between the hidden layer and the output layer, uses the regularization parameter C to measure the structural risk and empirical risk of the MELM model, and proposes the P-MELM algorithm. P-MELM uses the Cholesky factorization method to effectively reduce the amount of calculation for a single solution of the output weights' matrix, and when pruning hidden nodes, P-MELM can also update the original connection weights without the need for network recalculation, so P-MELM greatly reduces the amount of calculation and improves the calculation efficiency. P-MELM introduces the L2 regularization term into MELM to reduce the sensitivity of nodes. Starting from the initial large number of hidden nodes, by minimizing its prediction error and regularization control item, redundant nodes are rationally pruned to obtain a more concise network. Finally, data simulation indicates that P-MELM features are fast, precise, and reliable.

Data Availability

The datasets are obtained from references [26, 27]. All data is available. The datasets are obtained from the University of California, Irvine, Machine Learning Repository, Coal Classification Method Based on Visible-Infrared Spectroscopy, and An Improved Multilayer Extreme Learning Machine. All data is available.

Conflicts of Interest

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

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant

2017YFB0304100, National Natural Science Foundation of China under Grant 71672032, Fundamental Research Funds for Central University under Grant nos. N2104026 and N2005011, and Scientific Research Funds of Educational Department of Liaoning Province under Grant LT2020008.

References

[1] G. Huang, G.-B. Huang, S. Song, and K. You, "Trends in extreme learning machines: a review," *Neural Networks*, vol. 61, pp. 32–48, 2015.

[2] G.-B. You, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.

[3] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[4] B. L. S. da Silva, F. K. Inaba, E. O. T. Salles, and P. M. Ciarelli, "Fast deep stacked networks based on extreme learning machine applied to regression problems," *Neural Networks*, vol. 131, pp. 14–28, 2020.

[5] K. Wang, H. Pei, J. Cao, and P. Zhong, "Robust regularized extreme learning machine for regression with non-convex loss function via DC Program," *Journal of the Franklin Institute*, vol. 357, no. 11, pp. 7069–7091, 2020.

[6] Y. Shi, P. Li, H. Yuan, J. Miao, and L. Niu, "Fast kernel extreme learning machine for ordinal regression," *Knowledge-Based Systems*, vol. 177, pp. 44–54, 2019.

[7] Y. Qing, Y. Zeng, Y. Li, and G.-B. Huang, "Deep and wide feature based extreme learning machine for image classification," *Neurocomputing*, vol. 412, pp. 426–436, 2020.

[8] C. Wu, Y. Li, Z. Zhao, and B. Liu, "Extreme learning machine with autoencoding receptive fields for image classification," *Neural Computing and Applications*, vol. 32, no. 12, pp. 8157–8173, 2020.

[9] S. Jing, Y. Wang, and L. Yang, "Selective ensemble of uncertain extreme learning machine for pattern classification with missing features," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5881–5905, 2020.

[10] G. Zhang, D. Cui, S. Mao, and G.-B. Huang, "Unsupervised feature learning with sparse Bayesian auto-encoding based extreme learning machine," *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 7, pp. 1557–1569, 2020.

[11] X. Fang, Y. Cai, Z. Cai, X. Jiang, and Z. Chen, "Sparse feature learning of hyperspectral imagery via multiobjective-based extreme learning machine," *Sensors*, vol. 20, no. 5, p. 1262, 2020.

[12] F. Cao, Z. Yang, J. Ren, W. Chen, G. Han, and Y. Shen, "Local block multilayer sparse extreme learning machine for effective feature extraction and classification of hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 8, pp. 5580–5594, 2019.

[13] C. Chen, C.-M. Vong, P.-K. Wong, and K.-I. Tai, "Approximate empirical kernel map-based iterative extreme learning machine for clustering," *Neural Computing and Applications*, vol. 32, no. 12, pp. 8031–8046, 2020.

[14] J. Chen, Y. Zeng, Y. Li, and G.-B. Huang, "Unsupervised feature selection based extreme learning machine for clustering," *Neurocomputing*, vol. 386, pp. 198–207, 2020.

[15] Y. S. Zhang, J. Wu, C. Zhou, Z. H. Cai, J. Yang, and P. S. Yu, "Multi-view fusion with extreme learning machine for clustering," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 5, p. 53, 2019.

[16] W. Y. Deng, Q. H. Zheng, and L. Chen, "Regularized extreme learning machine," in *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining*, Nashville, TN, USA, April 2009.

[17] X.-R. Zhou and C.-S. Wang, "Cholesky factorization based online regularized and kernelized extreme learning machines with forgetting mechanism," *Neurocomputing*, vol. 174, pp. 1147–1155, 2016.

[18] H.-J. Rong, Y.-S. Ong, A.-H. Tan, and Z. Zhu, "A fast pruned-extreme learning machine for classification problem," *Neurocomputing*, vol. 72, no. 1–3, pp. 359–366, 2008.

[19] J. M. Martínez-Martínez, P. Escandell-Montero, E. Soria-Olivas, J. D. Martín-Guerrero, R. Magdalena-Benedito, and J. Gómez-Sanchis, "Regularized extreme learning machine for regression problems," *Neurocomputing*, vol. 74, no. 17, pp. 3716–3721, 2011.

[20] Y. Miche, A. Sorjamaa, P. Bas et al., "Optimally pruned extreme learning machine," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 158–162, 2010.

[21] Y. Miche, M. Van Heeswijk, P. Bas, O. Simula, and A. Lendasse, "TROP-ELM: a double-regularized ELM using LARS and Tikhonov regularization," *Neurocomputing*, vol. 74, no. 16, pp. 2413–2421, 2011.

[22] D. Xiao, B. Li, and S. Zhang, "An online sequential multiple hidden layers extreme learning machine method with forgetting mechanism," *Chemometrics and Intelligent Laboratory Systems*, vol. 176, pp. 126–133, 2018.

[23] J. Liu, X. Liu, C. Liu, B. T. Le, and D. Xiao, "Random search enhancement of incremental regularized multiple hidden layers ELM," *IEEE Access*, vol. 7, pp. 36866–36878, 2019.

[24] M. Seeger, "Low rank updates for the Cholesky decomposition," Technical Report, 2007.

[25] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders, "Methods for modifying matrix factorizations," *Mathematics of Computation*, vol. 28, no. 126, p. 505, 1974.

[26] University of California, *Machine Learning Repository*, University of California, Irvine, CA, USAhttp://archive.ics.uci.edu/ml/datasets.html.

[27] Y. Mao, B. T. Le, D. Xiao et al., "Coal classification method based on visible-infrared spectroscopy and an improved multilayer extreme learning machine," *Optics & Laser Technology*, vol. 114, pp. 10–15, 2019.