Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 6740523, 15 pages
https://doi.org/10.1155/2019/6740523

Research Article

Incremental Multiple Hidden Layers Regularized Extreme Learning Machine Based on Forced Positive-Definite Cholesky Factorization

Jingyi Liu [1] **and Ba Tuan Le** [2,3]

[1] *College of Sciences, Northeastern University, 110819 Shenyang, China*
[2] *Information Science and Engineering School, Northeastern University, 110819 Shenyang, China*
[3] *NTT Hi-Tech Institute, Nguyen Tat Thanh University, Ho Chi Minh City 700000, Vietnam*

Correspondence should be addressed to Ba Tuan Le; lebatuan@qq.com

Received 17 October 2018; Accepted 4 April 2019; Published 24 April 2019

Academic Editor: Alberto Olivares

The theory and implementation of extreme learning machine (ELM) prove that it is a simple, efficient, and accurate machine learning method. Compared with other single hidden layer feedforward neural network algorithms, ELM is characterized by simpler parameter selection rules, faster convergence speed, and less human intervention. The multiple hidden layer regularized extreme learning machine (MRELM) inherits these advantages of ELM and has higher prediction accuracy. In the MRELM model, the number of hidden layers is randomly initiated and fixed, and there is no iterative tuning process. However, the optimal number of hidden layers is the key factor to determine the generalization ability of MRELM. Given this situation, it is obviously unreasonable to determine this number by trial and random initialization. In this paper, an incremental MRELM training algorithm (FC-IMRELM) based on forced positive-definite Cholesky factorization is put forward to solve the network structure design problem of MRELM. First, an MRELM-based prediction model with one hidden layer is constructed, and then a new hidden layer is added to the prediction model in each training step until the generalization performance of the prediction model reaches its peak value. Thus, the optimal network structure of the prediction model is determined. In the training procedure, forced positive-definite Cholesky factorization is used to calculate the output weights of MRELM, which avoids the calculation of the inverse matrix and Moore-Penrose generalized inverse of matrix involved in the training process of hidden layer parameters. Therefore, FC-IMRELM prediction model can effectively reduce the computational cost brought by the process of increasing the number of hidden layers. Experiments on classification and regression problems indicate that the algorithm can be effectively used to determine the optimal network structure of MRELM, and the prediction model training by the algorithm has excellent performance in prediction accuracy and computational cost.

1. Introduction

The neural network is a complex nonlinear system interconnected by a large number of neurons, and it is based on the research of human brain information processing ability by modern neurobiology and cognitive science. The neural network is also a mathematical simulation form of human brain physiological structure, with strong adaptability, self-learning ability, and nonlinear mapping, and it has been widely used by researchers in many scientific fields [1–3]. However, the above prediction models are all based on the traditional neural networks, and the network training process needs to modify the network weights repeatedly according to the training objectives and gradient information. The entire network training process usually takes hundreds or even thousands of iterations before it can be finally completed, which requires a large amount of calculation.

Extreme learning machine (ELM) is a novel single hidden layer feedforward neural network. It transforms the iterative adjustment process of traditional neural network parameter training into solving linear equations. According to Moore-Penrose generalized inverse matrix theory, the least

squares solution with the minimum norm is obtained analytically as the network weights. The whole training process can be completed in one time without iteration. Compared with the traditional neural network training algorithm, which requires several iterations to determine the network weights, the training speed of ELM is significantly improved [4, 5]. This advantage enables ELM to be successfully applied in pattern recognition [6, 7] and regression estimation [8–10]. In order to improve the generalization ability of ELM, the literature [11] draws on the principle of structural risk minimization in statistical learning theory and proposes a regularized extreme learning machine (RELM). RELM has a better generalization ability by introducing parameters λ to weigh structural risks and empirical risks [12–15]. For the single hidden layer RELM model with multiple input and single output, the literature [16] designed the Cholesky factorization method for regularized output weight matrix. In the learning and forgetting process of the sample sequence, the Cholesky factorization factor is calculated recursively by adding and deleting samples one by one, and then the output weights are adjusted, and the network structure is fixed. However, if dealing with input data with complex noise signals and high-dimensional information, or with more classification categories, RELM also shows its own shortcomings, and the accuracy of the established model is greatly reduced.

In order to improve the embarrassing situation of RELM, the literature [17, 18] starts from improving its network structure. On the basis of the traditional RELM three-layer structure, the number of hidden layers is increased to form a neural network with one input layer, multiple hidden layers, and one output layer, that is, the multiple hidden layers RELM network model (MRELM), in which the neuron nodes of each hidden layer are fully connected. MRELM inherits the idea that RELM randomly initializes the weights matrix between the input layer and the hidden layer as well as the bias vector of the hidden layer. By forcing the actual output of the hidden layer to be as close as possible to the expected output, the weights matrix and the bias vector of the added hidden layers are calculated; thereby a neural network model with multiple hidden layers is established. The parameter training process needs to calculate the inverse matrix and the MP generalized inverse matrix, in which the first hidden layer parameters are randomly initialized, and the remaining hidden layers parameters are obtained by minimizing the error between the actual output and the expected output of the corresponding hidden layer. Compared with the traditional RELM model, MRELM can effectively improve the prediction accuracy through the layer-by-layer optimization of network parameters between different hidden layers. Moreover, it has the advantages of strong generalization ability and fast computing speed and is not easy to fall into local optimum [19–22]. However, since the initial parameter values of MRELM are randomly initialized, although this avoids the situation that the algorithm falls into local optimum and overfitting, it also leads to the failure of some hidden layers or the reduction of the effect on the neural network during the modeling process. As a result, there are some redundant hidden layers in the MRELM network, which often require more reasonable selection methods and theories for the number of hidden layers. Meanwhile, the network structure of MRELM is determined by the users based on their own practical experience, but this empirical choice is not reasonable, and it is difficult to guarantee the optimality. In practical applications, users often need to carry out repeated experiments for many times and choose the network structure with the least time consuming and the highest accuracy from the complex results comparison as the optimal network model for the training and prediction of the actual data.

In order to realize the effective design of MRELM network structure, select the number of hidden layers reasonably, and achieve the desired accuracy requirements, an incremental MRELM training algorithm based on forced positive definite Cholesky factorization (FC-IMRELM) [23, 24] is put forward in this paper. The algorithm can adjust the number of hidden layers in the network adaptively according to the predicted data, so as to determine the optimal network structure of FC-IMRELM. At the same time, a novel method is adopted to calculate the parameters of the newly added hidden layers, that is, the connection weight matrix and the bias vector of the hidden layers. Based on previous research, MRELM typically requires fewer hidden neurons than ELM to achieve a desirable performance level. This is a basic requirement for considering the multiple-hidden-layers structure presented. The foundational ideas for the FC-IMRELM algorithm are far simpler to produce and more stable by comparing and contrasting its characteristics with other ELM variants. Experimental results for classification and regression problems show that the proposed FC-IMRELM algorithm has more advantages in terms of average accuracy compared to the traditional RELM model and other improved models of MRELM.

The rest of this paper is organized as follows: Section 2 presents a brief review of the basic concepts and related work of multiple hidden layers RELM, Section 3 describes the proposed incremental FC-IMRELM technique, Section 4 reports and analyzes the experimental results, and, finally, Section 5 summarizes key conclusions of the present study.

2. Brief Review of Multiple Hidden Layers Regularized Extreme Learning Machine

The MRELM algorithm tries to find a mapping relationship that makes the output predicted by the ELM neural network with multiple hidden layers infinitely close to the actual given result. This mapping relationship will be embodied in the solution process of the weight and bias parameters of the hidden layers. The number of hidden layers in the MRELM neural network needs to be selected according to the change of the predicted data. Therefore, in the training process of network parameters, in order to ensure that the final hidden layer output is closer to the expected hidden layer output, in addition to the random initialization of the parameters of the first hidden layer, the parameter training process starts from the second hidden layer to optimize the network parameters until all the network parameters are completed. Furthermore, during the establishment of the neural network, the weight

matrix and bias vector of each hidden layer are acquired and recorded, so as to obtain the final predicted output result of the MRELM neural network. The solving process of the network parameters will be explained in detail in the following algorithm flow.

Suppose that a set of training sample dataset given in MRELM neural network is $\{X, T\} = \{x_i, t_i\}$ $(i = 1, 2, \cdots, N)$, where $X = [x_1, x_2, \cdots, x_N]^T$ is the input samples, $x_i = [x_{i1}, x_{i2}, \cdots, x_{in}]^T \in R^n$ is the $n \times 1$ input vector, $T = [t_1, t_2, \cdots, t_N]^T$ is the corresponding labeled samples, $t_i = [t_{i1}, t_{i2}, \cdots, t_{im}]^T \in R^m$ is the $m \times 1$ observation vector, and N is the total number of training samples. Meanwhile, it is assumed that all hidden layers in the MRELM model contain the same number of hidden nodes L, and each hidden node chooses the same activation function $g(x)$. In the modeling process of MRELM algorithm, multiple hidden layers in the neural network are first treated as a single hidden layer, and then the hidden layer parameters in the MRELM network containing only a single hidden layer are randomly initialized, namely, the input weights matrix $W_1 = [(W_1)_1, (W_1)_2, \cdots, (W_1)_L]^T \in R^{L \times n}$ connecting the input layer and the first hidden layer, and the bias vector $B_1 = [b_{11}, b_{12}, \cdots, b_{1L}]^T \in R^L$ of the first hidden nodes. Thus, the output matrix of the first hidden layer H_1 can be calculated as follows:

$$H_1 = g\left(W_1 X + B_1\right) \tag{1}$$

whose scalar entries $(h_1)_{ij} = g(W_{1j} x_i + b_{1j})$ $(i = 1, 2, \cdots, N,\ j = 1, 2, \cdots, L)$ are interpreted as the output of the $j - th$ hidden node in the first hidden layer with respect to x_i and $(W_1)_j = [(W_1)_{j1}, (W_1)_{j2}, \cdots, (W_1)_{jn}]^T$ is the vector of connection weights between n input nodes and the $j - th$ hidden node in the first hidden layer. To better balance the empirical risk and structural risk, the MRELM adjusts the proportion of the two risks by introducing parameter λ, which can be expressed as the following constrained optimization problem.

$$\begin{aligned} \min \quad & \frac{1}{2}\left\|\beta_1\right\|^2 + \frac{\lambda}{2}\|\varepsilon\|^2 \\ \text{s.t.} \quad & T = H_1\beta_1 - \varepsilon \end{aligned} \tag{2}$$

where $\beta_1 = [(\beta_1)_1, (\beta_1)_2, \cdots, (\beta_1)_L]^T \in R^{L \times m}$ is the connection weights matrix between the first hidden layer and the output layer, with vector components $(\beta_1)_j = [(\beta_1)_{j1}, (\beta_1)_{j2}, \cdots, (\beta_1)_{jm}]^T$ $(j = 1, 2, \cdots, L)$ that denote the connection weights between the $j - th$ hidden node in the first hidden layer and m output nodes, $\varepsilon = [\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_N]^T$ denotes the training error, and $\lambda > 0$ is the regularization parameter.

According on the KKT theorem, the constrained optimization of (2) can be transformed into the following dual optimization problem:

$$L(\alpha, \beta_1, \varepsilon) = \frac{1}{2}\left\|\beta_1\right\|^2 + \frac{\lambda}{2}\|\varepsilon\|^2 - \alpha\left(H_1\beta_1 - T - \varepsilon\right) \tag{3}$$

where $\alpha = [\alpha_1, \alpha_2, \cdots, \alpha_N]$ is the Lagrange multipliers vector. Utilizing KKT optimality conditions, the following equations can be obtained:

$$\frac{\partial L}{\partial \beta_1} = 0 \Longrightarrow (\beta_1)^T = \alpha H_1 \tag{4a}$$

$$\frac{\partial L}{\partial \varepsilon} = 0 \Longrightarrow \lambda \varepsilon^T + \alpha = 0 \tag{4b}$$

$$\frac{\partial L}{\partial \alpha} = 0 \Longrightarrow H_1\beta_1 - T - \varepsilon = 0 \tag{4c}$$

Finally, β_1 can be gotten as follows:

$$\beta_1 = \left(\lambda^{-1} I + (H_1)^T H_1\right)^{-1} (H_1)^T T \tag{5a}$$

or

$$\beta_1 = (H_1)^T \left(\lambda^{-1} I + H_1 (H_1)^T\right)^{-1} T \tag{5b}$$

In order to reduce the computational costs, if $N > L$, one may prefer to apply the solution (5a), and if $N < L$, one may prefer to apply the solution (5b).

Now the second hidden layer is added to the MRELM neural network, the network structure with two hidden layers is restored, and the two hidden layers are fully connected, so the prediction output of the second hidden layer H_2 can be obtained as follows:

$$H_2 = g\left(W_2 H_1 + B_2\right) \tag{6}$$

where W_2 denotes the weights matrix between the first hidden layer and the second hidden layer. We suppose that the first and second hidden layers have the same number of nodes, and thus W_2 is a square matrix. The matrix B_2 represents the bias of the second hidden layer. The expected output of the second hidden layer H_{2*} can be calculated as

$$H_{2*} = T(\beta_1)^+ \tag{7}$$

where $(\beta_1)^+$ is the MP generalized inverse of the matrix β_1, which can be calculated using the orthogonal projection method. Namely, if $(\beta_1)^T\beta_1$ is nonsingular, then $(\beta_1)^+ = ((\beta_1)^T\beta_1)^{-1}(\beta_1)^T$; otherwise $(\beta_1)^+ = (\beta_1)^T(\beta_1(\beta_1)^T)^{-1}$ if $\beta_1(\beta_1)^T$ is nonsingular. To make the predicted output of the hidden layer in the MRELM neural network infinitely close to the expected output, we may set $H_2 = H_{2*}$. Subsequently, we define the augmented matrix $(W_2)_{HE} = [B_2 \ \ W_2]$, and it can be gotten as

$$(W_2)_{HE} = g^{-1}(H_{2*})(H_{2E})^+ \tag{8}$$

where $(H_{2E})^+$ is the MP generalized inverse of the matrix $H_{2E} = [1 \ \ H_1]^T$, and 1 represents a one-column vector of size N whose elements are the scalar unit 1. The solving method of $(H_{2E})^+$ is the same as previously discussed for $(\beta_1)^+$. The notation $g^{-1}(x)$ indicates the inverse of the activation function $g(x)$. For classification and regression problems, we all invoke the widely used logistic sigmoid function $g(x) =$

$1/(1+e^{-x})$. The predicted output of the second hidden layer H_2 is obtained as

$$H_2 = g\left(W_2 H_1 + B_2\right) = g\left(\left(W_2\right)_{HE} H_{2E}\right) \tag{9}$$

Therefore, the connection weights matrix β_2 between the second hidden layer and the output layer is calculated as

$$\beta_2 = \left(\lambda^{-1} I + \left(H_2\right)^T H_2\right)^{-1} \left(H_2\right)^T T \tag{10a}$$

or

$$\beta_2 = \left(H_2\right)^T \left(\lambda^{-1} I + H_2 \left(H_2\right)^T\right)^{-1} T \tag{10b}$$

The solving method of β_2 is chosen according to what is previously discussed for β_1.

According to the MRELM algorithm flow, the third hidden layer is added to the MRELM network, and restore the network structure with three hidden layers. Since the nodes between each hidden layer are all connected together, the prediction output of the third hidden layer can be obtained as

$$H_3 = g\left(W_3 H_2 + B_3\right) \tag{11}$$

where W_3 represents the weights matrix between the second hidden layer and the third hidden layer, and the vector B_3 denotes the bias of the third hidden layer. Thus, the expected output of the third hidden layer can be gotten as

$$H_{3*} = T\left(\beta_2\right)^+ \tag{12}$$

where $(\beta_2)^+$ is the MP generalized inverse of the weights matrix β_2, obtained using the approach described before. To meet the requirement that the predicted output of the third hidden layer is infinitely close to the expected output, let $H_3 = H_{3*}$. Accordingly, the augmented matrix can be defined as $(W_3)_{HE} = [B_3 \;\; W_3]$, and we can solve it as follows.

$$\left(W_3\right)_{HE} = g^{-1}\left(H_{3*}\right)\left(H_{3E}\right)^+ \tag{13}$$

where $(H_{3E})^+$ is the MP generalized inverse of the matrix $H_{3E} = [1 \;\; H_2]^T$, the specific meaning of the symbol 1 is described above, and the calculation of $(H_{3E})^+$ also proceeds in the manner discussed before. Therefore, we can update the predicted output of the third hidden layer as

$$H_3 = g\left(W_3 H_2 + B_3\right) = g\left(\left(W_3\right)_{HE} H_{3E}\right) \tag{14}$$

Finally, the connection weight matrix β_3 between the third hidden layer and the output layer can be calculated as

$$\beta_3 = \left(\lambda^{-1} I + \left(H_3\right)^T H_3\right)^{-1} \left(H_3\right)^T T \tag{15a}$$

or

$$\beta_3 = \left(H_3\right)^T \left(\lambda^{-1} I + H_3 \left(H_3\right)^T\right)^{-1} T \tag{15b}$$

The calculation approach of β_3 is still selected according to the principle of β_1 discussed previously. The final output of the MRELM network with three hidden layers after training can be expressed as

$$f(x) = H_3 \beta_3 \tag{16}$$

If the number of hidden layers l in the MRELM network is more than 3, an iterative format can be adopted to realize the calculation process. In other words, the iterative calculation of formula (6) to formulas (15a) and (15b) is performed for $l-3$ times until all hidden layer parameters are solved. Emphasized finally, this algorithm does not add all hidden layers to the network at one time, nor does it calculate all hidden layer parameters at one time, but one hidden layer after another is added to the network. Every time a new hidden layer is added, the weights matrix and the bias vector of the hidden layers are calculated immediately to prepare for the parameter calculation of the hidden layer to be added next time.

3. Solutions of IMRELM by the Forced Positive-Definite Cholesky Factorization

For the single hidden layer feed-forward neural network, the literature [15] puts forward the regularized extreme learning machine algorithm based on Cholesky factorization (CF-FORELM), introduces the Cholesky factorization of positive definite matrix into the solving process of RELM, and designs a recursive solution method for the calculation of the regularized output matrix Cholesky factorization factor. The advantages of CF-FORELM algorithm prompted us to introduce the forced positive-definite Cholesky factorization method into the framework of MRELM algorithm with multiple hidden layers, and we then proposed an MRELM neural network training algorithm based on forced positive definite Cholesky factorization (FC-IMRELM). Compared with the inverse matrix calculation of invertible matrix $\lambda^{-1} I + (H_l)^T H_l$ in traditional RELM algorithm and the calculation of MP generalized inverse of matrix $(H_l)^+$ in MELM algorithm (l denotes the number of hidden layers), the algorithm effectively reduces the computational cost and complexity brought by the matrix inverse process. Meanwhile, the numerical stability of the forced positive definite Cholesky factorization method also greatly weakens the randomness effect of the ELM algorithm on the prediction results.

3.1. Forced Positive-Definite Cholesky Factorization (FC). The main difficulty of the MRELM algorithm is the calculation of the inverse matrix and the MP generalized inverse matrix involved in the training process, including the inverse calculation of symmetric positive semidefinite matrix. In this case, the improved MRELM mode based on the traditional Cholesky factorization could not be realized, because the Cholesky factorization of the symmetric positive semidefinite matrix might not exist. Even if such a factorization exists, the calculation process is generally numerically unstable for the elements of the matrix factorization factor may be unbounded. In order to overcome these difficulties, we put forward a modified approach based on the forced positive definite Cholesky factorization for the MRELM algorithm

with multiple hidden layers, which is a numerical stability approach.

When the forced positive definite strategy is adopted to improve the MRELM algorithm, the key problem is how to form the positive definite matrix from the modified Cholesky decomposition of the undetermined matrix. If the matrix G is not a positive definite matrix, the Cholesky factorization method, which forces the matrix to have positive definite property, is to find a unit lower triangular matrix L and a positive definite diagonal matrix D for the general symmetric matrix G, so that the matrix $\overline{G} = LDL^T$ is positively definite, and it is only one diagonal matrix E away from the matrix G.

$$\overline{G} = LDL^T = G + E \tag{17}$$

In fact, the Cholesky factorization of symmetric positive definite matrix can be described as follows:

$$d_{jj} = g_{jj} - \sum_{s=1}^{j-1} d_{ss} l_{js}^2 \tag{18a}$$

$$l_{ij} = \frac{1}{d_{jj}} \left(g_{ij} - \sum_{s=1}^{j-1} d_{ss} l_{js} l_{is} \right), \quad i \geq j-1 \tag{18b}$$

where g_{ij} represents the element of matrix G and d_{jj} denotes the main diagonal element of matrix D. Here, the Cholesky factorization factors L and D are required to satisfy two requirements: one is that all elements of D are strictly positive, and the other is that the elements of the factorization factor L are uniformly bounded. That is, for $k = 1, \cdots, n$ and a positive number ρ, the formula (19) is required:

$$\begin{gathered} d_{kk} > \delta, \\ |r_{ik}| \leq \rho, \\ (i > k) \end{gathered} \tag{19}$$

where the auxiliary quantity $r_{ik} = l_{ik}\sqrt{d_{kk}}$, δ is a given small positive number. The matrix satisfying the above conditions is said to be sufficiently positive definite, where E is a zero matrix.

Next, we describe the $j-th$ step of this factorization. Suppose the $j-1$ column of the forced positive-definite Cholesky factorization has been calculated. For $k = 1, \cdots, j-1$, equation (19) holds. First calculate

$$\gamma_j = \left| \xi_j - \sum_{s=1}^{j-1} d_{ss} l_{js}^2 \right|, \tag{20}$$

where ξ_j is taken as g_{jj}, and the test value $\overline{d}$ is defined as

$$\overline{d} = \max\left\{\gamma_j, \delta\right\} \tag{21}$$

where δ is a small positive number. In order to determine whether $\overline{d}$ can accept as the $j-th$ element of D, we check whether $r_{ij} = l_{ij}\sqrt{\overline{d}}$ satisfies the formula (19). If so, let $d_{jj} = \overline{d}$, and get the $j-th$ column of L from $l_{ij} = r_{ij}/\sqrt{d_{jj}}$. Otherwise,

$$d_{jj} = \left| \xi_j - \sum_{s=1}^{j-1} d_{ss} l_{js}^2 \right| \tag{22}$$

let $\xi_j = g_{jj} + e_{jj}$, select positive number e_{jj} to make $\max|r_{ij}| = \rho$, and produce the $j-th$ column of L.

If the above process is completed, we obtain the Cholesky factorization formula (17) of the positive definite matrix $\overline{G}$, where E is a nonnegative diagonal matrix and the diagonal element is e_{jj}. For the given matrix G, this nonnegative diagonal matrix E depends on ρ. If $n > 1$,

$$\left\|E(\rho)\right\|_\infty \leq \left(\frac{\xi}{\rho} + (n-1)\rho\right)^2 + 2\left(\gamma + (n-1)\rho^2\right)\delta \tag{23}$$

where ξ is the maximum norm of the nondiagonal elements of G and γ is the maximum norm of the diagonal elements of G. If $\rho^2 = \xi/\sqrt{n^2-1}$, the upper bound is minimized. So, let ρ satisfy formula (24):

$$\rho^2 = \max\left\{\gamma, \frac{\xi}{\sqrt{n^2-1}}, \varepsilon_M\right\} \tag{24}$$

where ε_M represents the machine precision. We increase ε_M to prevent $\|G\|$ from being small.

Finally, we present the forced positive-definite Cholesky factorization algorithm, where the auxiliary quantity $c_{is} = l_{is}d_{ss}$, $s = 1, \cdots, j$, $i = j, \cdots, n$. These values need not be stored separately; they can be stored in the matrix G.

Algorithm 1. Forced Positive-Definite Cholesky Factorization (FC)

Step 1. Calculate the bounds of the elements of the factorization factor. Let $\rho^2 = \max\{\gamma, \xi/\nu, \varepsilon_M\}$, where $\nu = \max\{1, \sqrt{n^2-1}\}$ and γ and ξ are the maximum norm of the diagonal and nondiagonal elements of G, respectively.

Step 2. Initialization. Let $j = 1$, $c_{ii} = g_{ii}$, $i = 1, \cdots, n$.

Step 3. Determine the minimum index q, so that $|c_{qq}| = \max_{j \leq i \leq n}|c_{ii}|$, and exchange the information of q rows and j rows, q columns and j columns of G.

Step 4. Calculate the $j-th$ row of L, and solve the maximum norm of $l_{ij}d_{jj}$. Let $l_{js} = c_{js}/d_{ss}$, $s = 1, \cdots, j-1$, calculate $c_{ij} = g_{ij} - \sum_{s=1}^{j-1} l_{js}c_{is}$, $i = j+1, \cdots, n$, and let $\theta_j = \max_{j+1 \leq i \leq n}|c_{ij}|$. If $j = n$, let $\theta_j = 0$.

Step 5. Calculate the $j-th$ diagonal element $d_{jj} = \max\{\delta, |c_{jj}|, \theta_j^2/\rho^2\}$ of D. The diagonal element of E is modified to $E_j = d_{jj} - c_{jj}$. If $j = n$, stop.

Step 6. Correct the diagonal elements and column index, and let $c_{ii} = c_{ii} - c_{ij}^2/d_{jj}$, $i = j+1, \cdots, n$ and $j = j+1$; jump to Step 3.

3.2. Process of Matrices Decomposition for FC-IMRELM. According to the MRELM training process shown in equations (1) to (16), its essence is to solve the connection weight matrix β_l between the hidden layer and the output layer. However, it can be seen from equations (5a), (5b), (10a), (10b), (15a), and (15b) that the solution method given in literature

[18] involves matrix inversion, and the solution process of each hidden layer's parameters $(W_l)_{HE}$ (l is the number of hidden layers) involves the calculation of the MP generalized inverse of matrix. The problem of large amount of calculation reduces the modeling efficiency of the MRELM prediction model. In order to solve the above problems effectively, we propose a solution method of the weights matrix β_l and hidden layer parameters $(W_l)_{HE}$ based on the forced positive definite Cholesky factorization.

First, on the basis of equations (4a), (4b), and (4c), $\beta_1 = (H_1)^T\alpha^T$ can be obtained from equation (4a), and $\varepsilon = -\alpha^T/\lambda$ can be obtained from equation (4b), and then substituting β_1 and ε into equation (4c), we can obtain

$$A\left(\beta_1\right)^{(k)} = \left(b_A\right)^{(k)}, \quad k = 1, 2, \cdots, m. \tag{25}$$

where $A = \lambda^{-1}I + (H_1)^T H_1$, $((b_A)^{(1)}, (b_A)^{(2)}, \cdots, (b_A)^{(m)}) = (H_1)^T T$, and $\beta_1 = ((\beta_1)^{(1)}, (\beta_1)^{(2)}, \cdots, (\beta_1)^{(m)})$, and

$$B\left(U_B\right)^{(k)} = T^{(k)}, \quad k = 1, 2, \cdots, m \tag{26}$$

where $B = \lambda^{-1}I + H_1(H_1)^T$, $T = (T^{(1)}, T^{(2)}, \cdots, T^{(m)})$, and $\beta_1 = (H_1)^T((U_B)^{(1)}, (U_B)^{(2)}, \cdots, (U_B)^{(m)})$.

Therefore, the process of solving β_1 according to equation (5a) can be transformed into solving m linear equations in the form of equation (25), and the process of solving β_1 according to equation (5b) can be transformed into solving m linear equations in the form of equation (26); m is the dimension of observation vector.

If the number of training samples is greater than the number of hidden nodes, that is, $N > L$, the solution process of β_1 based on the Cholesky factorization is as follows. First, calculate the Cholesky factorization of matrix A:

$$A = S_A\left(S_A\right)^T \tag{27}$$

where S_A is a lower triangular matrix with positive diagonal elements. The nonzero element $(s_A)_{ij}$ in S_A can be calculated by the element a_{ij} of A according to equation (28):

$$\left(s_A\right)_{ij} = \begin{cases} \sqrt{a_{ii} - \sum_{n=1}^{i-1}\left(s_A\right)_{in}^2} & i = j, \\ \dfrac{\left(a_{ij} - \sum_{n=1}^{j-1}\left(s_A\right)_{in}\left(s_A\right)_{jn}\right)}{\left(s_A\right)_{jj}} & i > j \end{cases} \tag{28}$$

where $i = 1, 2, \cdots, L$, $j = 1, 2, \cdots, L$. Substitute equation (27) into equation (25) and multiply both sides of the equation by $(S_A)^{-1}$:

$$\left(S_A\right)^T\left(\beta_1\right)^{(k)} = \left(V_A\right)^{(k)}, \quad k = 1, 2, \cdots, m, \tag{29}$$

where $(V_A)^{(k)} = (S_A)^{-1}(b_A)^{(k)}$. Solving $\beta_1 = ((\beta_1)^{(1)}, (\beta_1)^{(2)}, \cdots, (\beta_1)^{(m)})$ is equivalent to solving equation (29). Because $(V_A)^{(k)} = (S_A)^{-1}(b_A)^{(k)}$ is equivalent to $S_A(V_A)^{(k)} = (b_A)^{(k)}$, the calculation formula for the elements $(v_A)_i^{(k)}$ of $(V_A)^{(k)}$ can be obtained by comparing the elements on both sides of the equation

$$\left(v_A\right)_i^{(k)} = \begin{cases} \dfrac{\left(b_A\right)_i^{(k)}}{\left(s_A\right)_{ii}} & i = 1, \\ \dfrac{\left(\left(b_A\right)_i^{(k)} - \sum_{n=1}^{i-1}\left(s_A\right)_{ni}\left(v_A\right)_n^{(k)}\right)}{\left(s_A\right)_{ii}} & i > 1 \end{cases} \tag{30}$$

where $i = 1, 2, \cdots, L$, $(b_A)_i^{(k)}$ is the element of $(b_A)^{(k)}$. Finally, on the basis of obtaining S_A and $(V_A)^{(k)}$, the element $(\beta_1)_i^{(k)}$ of $(\beta_1)^{(k)}$ can be calculated by using the elements of S_A and $(V_A)^{(k)}$:

$$\left(\beta_1\right)_i^{(k)} = \begin{cases} \dfrac{\left(v_A\right)_i^{(k)}}{\left(s_A\right)_{ii}} & i = L, \\ \dfrac{\left(\left(v_A\right)_i^{(k)} - \sum_{n=1}^{L-i}\left(s_A\right)_{i+n,i}\left(\beta_1\right)_{i+n}^{(k)}\right)}{\left(s_A\right)_{ii}} & i < L \end{cases} \tag{31}$$

where $i = 1, 2, \cdots, L$, $k = 1, 2, \cdots, m$, $\beta_1 = ((\beta_1)^{(1)}, (\beta_1)^{(2)}, \cdots, (\beta_1)^{(m)})$.

If the number of hidden nodes is greater than the number of training samples, that is, $N < L$, the solution process of β_1 based on the Cholesky factorization is as follows. First, calculate the Cholesky factorization of B:

$$B = S_B\left(S_B\right)^T \tag{32}$$

where S_B is a lower triangular matrix with positive diagonal elements. The nonzero element $(s_B)_{ij}$ in S_B can be calculated by the element b_{ij} of B according to equation (33):

$$\left(s_B\right)_{ij} = \begin{cases} \sqrt{b_{ii} - \sum_{n=1}^{i-1}\left(s_B\right)_{in}^2} & i = j, \\ \dfrac{\left(b_{ij} - \sum_{n=1}^{j-1}\left(s_B\right)_{in}\left(s_B\right)_{jn}\right)}{\left(s_B\right)_{jj}} & i > j \end{cases} \tag{33}$$

where $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, N$. Substitute equation (32) into equation (26) and multiply both sides of the equation by $(S_B)^{-1}$:

$$\left(S_B\right)^T\left(U_B\right)^{(k)} = \left(V_B\right)^{(k)}, \quad k = 1, 2, \cdots, m, \tag{34}$$

where $(V_B)^{(k)} = (S_B)^{-1}T^{(k)}$. Solving $\beta_1 = (H_1)^T((U_B)^{(1)}, (U_B)^{(2)}, \cdots, (U_B)^{(m)})$ is equivalent to solving equation (34). Because $(V_B)^{(k)} = (S_B)^{-1}T^{(k)}$ is equivalent to $S_B(V_B)^{(k)} = T^{(k)}$, the calculation formula for the element $(v_B)_i^{(k)}$ of $(V_B)^{(k)}$ can be obtained by comparing the elements on both sides of the equation

$$\left(v_B\right)_i^{(k)} = \begin{cases} \dfrac{T_i^{(k)}}{\left(s_B\right)_{ii}} & i = 1, \\ \dfrac{\left(T_i^{(k)} - \sum_{n=1}^{i-1}\left(s_B\right)_{ni}\left(v_B\right)_n^{(k)}\right)}{\left(s_B\right)_{ii}} & i > 1 \end{cases} \tag{35}$$

where $i = 1, 2, \cdots, N$, $T_i^{(k)}$ is the element of $T^{(k)}$. Finally, on the basis of obtaining S_B and $(V_B)^{(k)}$, the element $(u_B)_i^{(k)}$ of $(U_B)^{(k)}$ can be calculated by using the elements of S_B and $(V_B)^{(k)}$.

$$(u_B)_i^{(k)} = \begin{cases} \dfrac{(v_B)_i^{(k)}}{(s_B)_{ii}} & i = L, \\ \dfrac{\left((v_B)_i^{(k)} - \sum_{n=1}^{L-i} (s_B)_{i+n,i} (u_B)_{i+n}^{(k)}\right)}{(s_B)_{ii}} & i < L \end{cases} \tag{36}$$

where $i = 1, 2, \cdots, N$, $k = 1, 2, \cdots, m$, $\beta_1 = (H_1)^T((U_B)^{(1)}, (U_B)^{(2)}, \cdots, (U_B)^{(m)})$.

At this point, we get the connection weight matrix β_1 between the first hidden layer and the output layer. The connection weight matrix β_l between the other hidden layer and output layer can also be calculated by the above method. Compared with the solution method of the connecting weight matrix β_l as shown in equations (5a), (5b), (10a), (10b), (15a), and (15b), the solution of β_l based on Cholesky factorization does not involve the inverse operation of the matrix, and it can be achieved by using simple algebraic operations.

The MRELM model contains multiple hidden layers, and the solving process of each hidden layer parameter $(W_l)_{HE}$ needs to calculate the MP generalized inverse of the corresponding matrices. However, it can be concluded from equations (7), (8), (11), and (13) that the solution method of $(\beta_l)^+$ and $(H_{lE})^+$ using orthogonal projection method [5] involves the inverse calculation of symmetric semipositive definite matrixes $(\beta_l)^T\beta_l$, $\beta_l(\beta_l)^T$, $(H_{lE})^T H_{lE}$, and $H_{lE}(H_{lE})^T$, which has the problem of large computational cost and numerical instability, and if the condition number of the above matrices is too large, the calculation results of MP generalized inverse of matrices $(\beta_l)^+$ and $(H_{lE})^+$ are usually unable to be obtained. This not only affects the modeling efficiency and prediction effect of MRELM model, but also may make the modeling process impossible to complete. However, the traditional Cholesky factorization method can only be used to solve the calculation of symmetric positive definite matrix. To effectively overcome the above difficulties, we use the forced positive definite Cholesky factorization to solve the MP generalized inverse of matrices $(\beta_l)^+$ and $(H_{lE})^+$, and we then get the hidden layer parameters $(W_l)_{HE}$.

If $(\beta_l)^T\beta_l$ is nonsingular, then take $(\beta_l)^+ = ((\beta_l)^T\beta_l)^{-1}(\beta_l)^T$ and substitute it into (7) and (11):

$$C\left((\beta_l)^+\right)^{(k)} = \left((\beta_l)^T\right)^{(k)}, \quad k = 1, 2, \cdots, L, \tag{37}$$

where $C = (\beta_l)^T\beta_l$, $H_{(l+1)*} = T(((\beta_l)^+)^{(1)}, ((\beta_l)^+)^{(2)}, \cdots, ((\beta_l)^+)^{(L)})$.

If $\beta_l(\beta_l)^T$ is nonsingular, then take $(\beta_l)^+ = (\beta_l)^T(\beta_l(\beta_l)^T)^{-1}$ and substitute it into (7):

$$D^T\left(\left((\beta_l)^+\right)^T\right)^{(k)} = (\beta_l)^{(k)}, \quad k = 1, 2, \cdots, m, \tag{38}$$

where $D = \beta_l(\beta_l)^T$, $H_{(l+1)*} = T((((\beta_l)^+)^T)^{(1)}, (((\beta_l)^+)^T)^{(2)}, \cdots, (((\beta_l)^+)^T)^{(m)})^T$.

Therefore, the process of solving $H_{(l+1)*}$ can be transformed into solving L linear equations in the form of equation (37), or solving m linear equations in the form of equation (38), L is the number of hidden nodes, and m is the dimension of the observation vector.

If $(\beta_l)^T\beta_l$ is nonsingular, the solution process of $H_{(l+1)*}$ based on forced positive definite Cholesky factorization is as follows. First, calculate the modified Cholesky factorization result of matrix C:

$$C = P_C M_C (P_C)^T \tag{39a}$$

where P_C is the unit lower triangular matrix and M_C is the positive definite diagonal matrix. $(p_C)_{ij}$ is the nonzero element below the main diagonal in P_C, and $(m_C)_{ii}$ is the main diagonal element in M_C; it can be calculated according to Algorithm 1 by using element c_{ij} of C, where $i = 1, 2, \cdots, m$, $j = 1, 2, \cdots, m$. If we set $M_C = (M_C)^{1/2}(M_C)^{1/2}$, then (39a) can be written as

$$C = Q_C (Q_C)^T \tag{39b}$$

where $Q_C = P_C(M_C)^{1/2}$. Substitute equation (39b) into equation (37) and multiply both sides of the equation by $(Q_C)^{-1}$.

$$(Q_C)^T\left((\beta_l)^+\right)^{(k)} = (V_C)^{(k)}, \quad k = 1, 2, \cdots, L, \tag{40}$$

where $(V_C)^{(k)} = (Q_C)^{-1}((\beta_l)^T)^{(k)}$. Solving $H^{(l+1)*} = T(((\beta_l)^+)^{(1)}, ((\beta_l)^+)^{(2)}, \cdots, ((\beta_l)^+)^{(L)})$ is equivalent to solving equation (40). Since $(V_C)^{(k)} = (Q_C)^{-1}((\beta_l)^T)^{(k)}$ is equivalent to $Q_C(V_C)^{(k)} = ((\beta_l)^T)^{(k)}$, by comparing the elements on both sides of the equation, the calculation formula for the element $(v_C)_i^{(k)}$ of $(V_C)^{(k)}$ can be obtained:

$$(v_C)_i^{(k)} = \begin{cases} \dfrac{\left((\beta_l)^T\right)_i^{(k)}}{(q_C)_{ii}} & i = 1, \\ \dfrac{\left(\left((\beta_l)^T\right)_i^{(k)} - \sum_{n=1}^{i-1} (q_C)_{ni} (v_C)_n^{(k)}\right)}{(q_C)_{ii}} & i > 1 \end{cases} \tag{41}$$

where $((\beta_l)^T)_i^{(k)}$ is the element of $((\beta_l)^T)^{(k)}$, and $(q_C)_{ij}$ is the element of Q_C, $i = 1, 2, \cdots, m$, $j = 1, 2, \cdots, m$. Finally, on the basis of obtaining Q_C and $(V_C)^{(k)}$, the element $((\beta_l)^+)_i^{(k)}$ of $((\beta_l)^+)^{(k)}$ can be calculated by using the elements of Q_C and $(V_C)^{(k)}$.

$$\left((\beta_l)^+\right)_i^{(k)} = \begin{cases} \dfrac{(v_C)_i^{(k)}}{(q_C)_{ii}} & i = L, \\ \dfrac{\left((v_C)_i^{(k)} - \sum_{n=1}^{L-i} (q_C)_{i+n,i} \left((\beta_l)^+\right)_{i+n}^{(k)}\right)}{(q_C)_{ii}} & i < L \end{cases} \tag{42}$$

where $i = 1, 2, \cdots, m$, $k = 1, 2, \cdots, L$, $(\beta_l)^+ = (((\beta_l)^+)^{(1)}, ((\beta_l)^+)^{(2)}, \cdots, ((\beta_l)^+)^{(L)})$; consequently, we can get $H_{(l+1)*} = T(((\beta_l)^+)^{(1)}, ((\beta_l)^+)^{(2)}, \cdots, ((\beta_l)^+)^{(L)})$.

If $\beta_l(\beta_l)^T$ is nonsingular, the solution process of $H_{(l+1)*}$ based on forced positive definite Cholesky factorization is as follows. First, calculate the modified Cholesky factorization results of matrix D:

$$D = P_D M_D (P_D)^T \tag{43a}$$

where P_D is the unit lower triangular matrix and M_D is the positive definite diagonal matrix. $(p_D)_{ij}$ is the nonzero element below the main diagonal in P_D, and $(m_D)_{ii}$ is the main diagonal element in M_D; it can be calculated according to Algorithm 1 by using the element d_{ij} of D, $i = 1, 2, \cdots, L$, $j = 1, 2, \cdots, L$. If we set $M_D = (M_D)^{1/2}(M_D)^{1/2}$, then (43a) can be written as

$$D = Q_D (Q_D)^T \tag{43b}$$

where $Q_D = P_D(M_D)^{1/2}$. Substitute equation (43b) into equation (38) and multiply both sides of the equation by $(Q_D)^{-1}$.

$$(Q_D)^T \left(\left((\beta_l)^+\right)^T\right)^{(k)} = (V_D)^{(k)}, \quad k = 1, 2, \cdots, m, \tag{44}$$

where $(V_D)^{(k)} = (Q_D)^{-1}(\beta_l)^{(k)}$. Solving $H_{(l+1)*} = T((((\beta_l)^+)^T)^{(1)}, (((\beta_l)^+)^T)^{(2)}, \cdots, (((\beta_l)^+)^T)^{(m)})^T$ is equivalent to solving equation (44). Since $(V_D)^{(k)} = (Q_D)^{-1}(\beta_l)^{(k)}$ is equivalent to $Q_D(V_D)^{(k)} = (\beta_l)^{(k)}$, by comparing the elements on both sides of the equation, the calculation formula for the element $(v_D)_i^{(k)}$ of $(V_D)^{(k)}$ can be gotten as

$$(v_D)_i^{(k)} = \begin{cases} \dfrac{(\beta_l)_i^{(k)}}{(q_D)_{ii}} & i = 1, \\ \dfrac{\left((\beta_l)_i^{(k)} - \sum_{n=1}^{i-1} (q_D)_{ni} (v_D)_n^{(k)}\right)}{(q_D)_{ii}} & i > 1 \end{cases} \tag{45}$$

where $(\beta_l)_i^{(k)}$ is the element of $(\beta_l)^{(k)}$ and $(q_D)_{ij}$ is the element of Q_D, $i = 1, 2, \cdots, L$, $j = 1, 2, \cdots, L$. Finally, on the basis of obtaining Q_D and $(V_D)^{(k)}$, the element $(((\beta_l)^+)^T)_i^{(k)}$ of $(((\beta_l)^+)^T)^{(k)}$ can be obtained by using elements of Q_D and $(V_D)^{(k)}$.

$$\left(\left((\beta_l)^+\right)^T\right)_i^{(k)} = \begin{cases} \dfrac{(v_D)_i^{(k)}}{(q_D)_{ii}} & i = L, \\ \dfrac{\left((v_D)_i^{(k)} - \sum_{n=1}^{L-i} (q_D)_{i+n,i} \left(\left((\beta_l)^+\right)^T\right)_{i+n}^{(k)}\right)}{(q_D)_{ii}} & i < L \end{cases} \tag{46}$$

where $i = 1, 2, \cdots, L$, $k = 1, 2, \cdots, m$, $(\beta_l)^+ = ((((\beta_l)^+)^T)^{(1)}, (((\beta_l)^+)^T)^{(2)}, \cdots, (((\beta_l)^+)^T)^{(m)})^T$; therefore, we can get $H_{(l+1)*} = T((((\beta_l)^+)^T)^{(1)}, (((\beta_l)^+)^T)^{(2)}, \cdots, (((\beta_l)^+)^T)^{(m)})^T$.

If $(H_{lE})^T H_{lE}$ is nonsingular, then let $(H_{lE})^+ = ((H_{lE})^T H_{lE})^{-1}(H_{lE})^T$ and substitute it into (8) and (13):

$$E(U_E)^{(k)} = \left(\left(g^{-1}(H_{l*})\right)^T\right)^{(k)}, \quad k = 1, 2, \cdots, L, \tag{47}$$

where $E = (H_{lE})^T H_{lE}$, $(W_l)_{HE} = ((U_E)^{(1)}, (U_E)^{(2)}, \cdots, (U_E)^{(L)})^T (H_{lE})^T$.

If $H_{lE}(H_{lE})^T$ is nonsingular, then let $(H_{lE})^+ = (H_{lE})^T (H_{lE}(H_{lE})^T)^{-1}$ and substitute it into (8) and (13):

$$F\left(((W_l)_{HE})^T\right)^{(k)} = (b_F)^{(k)}, \quad k = 1, 2, \cdots, L, \tag{48}$$

where $(W_l)_{HE} = ((((W_l)_{HE})^T)^{(1)}, (((W_l)_{HE})^T)^{(2)}, \cdots, (((W_l)_{HE})^T)^{(L)})^T$, $((b_F)^{(1)}, (b_F)^{(2)}, \cdots (b_F)^{(L)})^T = g^{-1}(H^{l*})(H_{lE})^T$, $F = H_{lE}(H_{lE})^T$.

Therefore, the process of solving $(W_l)_{HE}$ can be transformed into solving L linear equations in the form of equation (47), or solving L linear equations in the form of equation (48), where L is the number of hidden nodes.

If $(H_{lE})^T H_{lE}$ is nonsingular, the solution process of $(W_l)_{HE}$ based on the forced positive definite Cholesky factorization is as follows. First, calculate modified Cholesky factorization results of matrix E:

$$E = P_E M_E (P_E)^T \tag{49a}$$

where P_E is the unit lower triangular matrix and M_E is the positive definite diagonal matrix. $(p_E)_{ij}$ is the nonzero element below the main diagonal in P_E, and $(m_E)_{ii}$ is the main diagonal element in M_E; it can be calculated according to Algorithm 1 by using element E_{ij} of E, $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, N$. If we set $M_E = (M_E)^{1/2}(M_E)^{1/2}$, then (49a) can be written as

$$E = Q_E (Q_E)^T \tag{49b}$$

where $Q_E = P_E(M_E)^{1/2}$. Substitute equation (49b) into equation (47) and multiply both sides of the equation by $(Q_E)^{-1}$.

$$(Q_E)^T (U_E)^{(k)} = (V_E)^{(k)}, \quad k = 1, 2, \cdots, L, \tag{50}$$

where $(V_E)^{(k)} = (Q_E)^{-1}((g^{-1}(H_{l*}))^T)^{(k)}$. Solving $(W_l)_{HE} = ((U_E)^{(1)}, (U_E)^{(2)}, \cdots, (U_E)^{(L)})^T (H_{lE})^T$ is equivalent to solving equation (46). For $(V_E)^{(k)} = (Q_E)^{-1}((g^{-1}(H_{l*}))^T)^{(k)}$ is equivalent to $Q_E(V_E)^{(k)} = ((g^{-1}(H_{l*}))^T)^{(k)}$, by comparing the elements on both sides of the equation, the calculation formula for the element $(v_E)_i^{(k)}$ of $(V_E)^{(k)}$ can be calculated as

$$(v_E)_i^{(k)} = \begin{cases} \dfrac{\left(\left(g^{-1}(H_{l*})\right)^T\right)_i^{(k)}}{(q_E)_{ii}} & i = 1, \\ \dfrac{\left(\left(\left(g^{-1}(H_{l*})\right)^T\right)_i^{(k)} - \sum_{n=1}^{i-1} (q_E)_{ni} (v_E)_n^{(k)}\right)}{(q_E)_{ii}} & i > 1 \end{cases} \tag{51}$$

where $((g^{-1}(H_{l*}))^T)_i^{(k)}$ is the element of $((g^{-1}(H_{l*}))^T)^{(k)}$ and $(q_E)_{ij}$ is the element of Q_E, $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, N$. Finally, on the basis of obtaining Q_E and $(V_E)^{(k)}$, the element $(U_E)_i^{(k)}$ of $(U_E)^{(k)}$ can be calculated by using the elements of Q_E and $(V_E)^{(k)}$.

$$\left(U_E\right)_i^{(k)} = \begin{cases} \dfrac{\left(v_E\right)_i^{(k)}}{\left(q_E\right)_{ii}} & i = L, \\ \dfrac{\left(\left(v_E\right)_i^{(k)} - \sum_{n=1}^{L-i} \left(q_E\right)_{i+n,i} \left(U_E\right)_{i+n}^{(k)}\right)}{\left(q_E\right)_{ii}} & i < L \end{cases} \tag{52}$$

where $i = 1, 2, \cdots, N$, $k = 1, 2, \cdots, L$, $U_E = ((U_E)^{(1)}, (U_E)^{(2)}, \cdots, (U_E)^{(L)})$; accordingly, we can obtain the matrix $(W_l)_{HE} = ((U_E)^{(1)}, (U_E)^{(2)}, \cdots, (U_E)^{(L)})^T (H_{lE})^T$.

If $H_{lE}(H_{lE})^T$ is nonsingular, the solution process of $(W_l)_{HE}$ based on the forced positive definite Cholesky factorization is as follows. First, calculate modified Cholesky factorization results of matrix F:

$$F = P_F M_F \left(P_F\right)^T \tag{53a}$$

where P_F is the unit lower triangular matrix and M_F is the positive definite diagonal matrix. $(p_F)_{ij}$ is the nonzero element below the main diagonal in P_F, and $(m_F)_{ii}$ is the main diagonal element in M_F; it can be calculated according to Algorithm 1 by using element F_{ij} of F, $i = 1, 2, \cdots, L+1$, $j = 1, 2, \cdots, L+1$. If we set $M_F = (M_F)^{1/2}(M_F)^{1/2}$, then (53a) can be written as

$$F = Q_F \left(Q_F\right)^T \tag{53b}$$

where $Q_F = P_F(M_F)^{1/2}$. Substitute equation (53b) into equation (48) and multiply both sides of the equation by $(Q_F)^{-1}$.

$$\left(Q_F\right)^T \left(\left(\left(W_l\right)_{HE}\right)^T\right)^{(k)} = \left(V_F\right)^{(k)}, \quad k = 1, 2, \cdots, L, \tag{54}$$

where $(V_F)^{(k)} = (Q_F)^{-1}(b_F)^{(k)}$. Because solving $(W_l)_{HE} = ((((W_l)_{HE})^T)^{(1)}, (((W_l)_{HE})^T)^{(2)}, \cdots, (((W_l)_{HE})^T)^{(L)})^T$ is equivalent to solving equation (54) and $(V_F)^{(k)} = (Q_F)^{-1}(b_F)^{(k)}$ is equivalent to $Q_F(V_F)^{(k)} = (b_F)^{(k)}$, by comparing the elements on both sides of the equation, the calculation formula for the element $(v_F)_i^{(k)}$ of $(V_F)^{(k)}$ can be calculated as

$$\left(v_F\right)_i^{(k)} = \begin{cases} \dfrac{\left(b_F\right)_i^{(k)}}{\left(q_F\right)_{ii}} & i = 1, \\ \dfrac{\left(\left(b_F\right)_i^{(k)} - \sum_{n=1}^{i-1} \left(q_F\right)_{ni} \left(v_F\right)_n^{(k)}\right)}{\left(q_F\right)_{ii}} & i > 1 \end{cases} \tag{55}$$

where $(b_F)_i^{(k)}$ is the element of $(b_F)^{(k)}$ and $(q_F)_{ij}$ is the element of Q_F, $i = 1, 2, \cdots, L+1$, $j = 1, 2, \cdots, L+1$. Finally, on the basis of obtaining Q_F and $(V_F)^{(k)}$, the element $(((W_l)_{HE})^T)_i^{(k)}$ of $(((W_l)_{HE})^T)^{(k)}$ can be calculated by using the elements of Q_F and $(V_F)^{(k)}$.

$$\left(\left(\left(W_l\right)_{HE}\right)^T\right)_i^{(k)} = \begin{cases} \dfrac{\left(v_F\right)_i^{(k)}}{\left(q_F\right)_{ii}} & i = L, \\ \dfrac{\left(\left(v_F\right)_i^{(k)} - \sum_{n=1}^{L-i} \left(q_F\right)_{i+n,i} \left(\left(\left(W_l\right)_{HE}\right)^T\right)_{i+n}^{(k)}\right)}{\left(q_F\right)_{ii}} & i < L \end{cases} \tag{56}$$

where $i = 1, 2, \cdots, L+1$, $k = 1, 2, \cdots, L$. So we can get $(W_l)_{HE} = ((((W_l)_{HE})^T)^{(1)}, (((W_l)_{HE})^T)^{(2)}, \cdots, (((W_l)_{HE})^T)^{(L)})^T$.

So far we have obtained the parameters $(W_l)_{HE}$ of each hidden layer. Compared with the solving method of hidden layer parameters $(W_l)_{HE}$ shown in equations (7), (8), (11), and (13), the calculation approach of $(W_l)_{HE}$ based on the forced positive definite Cholesky factorization does not involve the operation of inverse matrix, which can be realized only by simple algebraic calculations. Meanwhile, the method of forcing the matrix to be positive definite matrix can guarantee the numerical stability of MRELM neural network training process.

3.3. FC-IMRELM Training Algorithm. Studies have shown that the number of hidden layers determines the learning accuracy and generalization ability of the MRELM model [25], and it is also a key factor that must be determined in advance when designing the MRELM network structure. Due to the complexity of various training samples applied in MRELM prediction model, it is difficult to accurately determine the optimal number of hidden layers by human experience, so that the MRELM prediction model has enough hidden layers to ensure its learning accuracy, while, at the same time, it has as few hidden layers as possible to maintain its contracted network structure. To avoid the disadvantages and difficulties of artificially selecting the number of hidden layers, we propose an incremental MRELM training algorithm based on forced positive Cholesky factorization, which can automatically determine the optimal number of hidden layers in MRELM, and the training process is as follows.

Algorithm 2. Incremental MRELM based on Forced Positive-Definite Cholesky Factorization (FC-IMRELM)

Step 1. Suppose the training sample dataset is $\{X, T\} = \{x_i, t_i\}$ $(i = 1, 2, \cdots, N)$, where $X = [x_1, x_2, \cdots, x_N]^T$ is the input samples, $T = [t_1, t_2, \cdots, t_N]^T$ is the labeled samples, each hidden layer contains the number of hidden nodes L, and the activation function is $g(x)$.

Step 2. Let the number of hidden layers in MRELM $l = 1$, randomly initialize the input weights matrix W_1 between the input layer and the first hidden layer and the bias vector B_1

of the first hidden layer, and let $(W_1)_{HE} = [B_1 \ W_1]$, $H_{1E} = [1 \ X]^T$.

Step 3. Calculate the output matrix of the first hidden layer $H_1 = g((W_1)_{HE}H_{1E})$, and calculate the matrices A, B, and $(b_A)^{(k)}$ $(k = 1, 2, \cdots, m)$.

Step 4. Calculate the connection weights matrix between the first hidden layer and the output layer $\beta_1 = (\lambda^{-1}I + (H_1)^T H_1)^{-1}(H_1)^T T$ or $\beta_1 = (H_1)^T(\lambda^{-1}I + H_1(H_1)^T)^{-1}T$.

(1) If $N > L$, the Cholesky factorization factor S_A of matrix A is calculated according to the formula (28), and $(V_A)^{(k)}$ $(k = 1, 2, \cdots, m)$ is calculated according to the formula (30) using matrices S_A and $(b_A)^{(k)}$ $(k = 1, 2, \cdots, m)$. Then matrices S_A and $(V_A)^{(k)}$ $(k = 1, 2, \cdots, m)$ are used to calculate $(\beta_1)^{(k)}$ according to equation (31), and we can get $\beta_1 = ((\beta_1)^{(1)}, (\beta_1)^{(2)}, \cdots, (\beta_1)^{(m)})$.

(2) If $N < L$, the Cholesky factorization factor S_B of matrix B is calculated according to the formula (33), and $(V_B)^{(k)}$ $(k = 1, 2, \cdots, m)$ is calculated according to the formula (35) using S_B and $T^{(k)}$ $(k = 1, 2, \cdots, m)$. Then the matrices S_B and $(V_B)^{(k)}$ $(k = 1, 2, \cdots, m)$ are used to calculate $(U_B)^{(k)}$ $(k = 1, 2, \cdots, m)$ according to the equation (36); subsequently, we can obtain $\beta_1 = (H_1)^T((U_B)^{(1)}, (U_B)^{(2)}, \cdots, (U_B)^{(m)})$.

Step 5. Calculate the expected output $H_{(l+1)*} = T(\beta_l)^+$ of the $(l+1) - th$ hidden layer.

(1) If $(\beta_l)^T\beta_l$ is nonsingular, the forced positive-definite Cholesky factorization factor Q_C of matrix C is calculated according to equations (39a) and (39b), and $(V_C)^{(k)}(k = 1, 2, \cdots, L)$ is calculated according to equation (41) using Q_C and $((\beta_l)^T)^{(k)}$ $(k = 1, 2, \cdots, L)$. Then the matrices Q_C and $(V_C)^{(k)}$ $(k = 1, 2, \cdots, L)$ are used to calculate $((\beta_l)^+)^{(k)}$ $(k = 1, 2, \cdots, L)$ according to formula (42), so we can obtain $H_{(l+1)*} = T(((\beta_l)^+)^{(1)}, ((\beta_l)^+)^{(2)}, \cdots, ((\beta_l)^+)^{(L)})$.

(2) If $\beta_l(\beta_l)^T$ is nonsingular, the forced positive-definite Cholesky factorization factor Q_D of D is calculated according to equations (43a) and (43b), and $(V_D)^{(k)}$ $(k = 1, 2, \cdots, m)$ is calculated according to equation (45) using Q_D and $(\beta_l)^{(k)}$ $(k = 1, 2, \cdots, m)$. Then the matrices Q_D and $(V_D)^{(k)}$ $(k = 1, 2, \cdots, m)$ are used to calculate $(((\beta_l)^+)^T)^{(k)}$ $(k = 1, 2, \cdots, m)$ according to formula (46); thus we can get $H_{(l+1)*} = T((((\beta_l)^+)^T)^{(1)}, (((\beta_l)^+)^T)^{(2)}, \cdots, (((\beta_l)^+)^T)^{(m)})^T$.

Step 6. Let $l = l + 1$, calculate the parameter $(W_l)_{HE} = g^{-1}(H_{l*})(H_{lE})^+$ of the $l - th$ hidden layer, set $(W_l)_{HE} = [B_l \ W_l]$, W_l is the weights matrix between the $(l-1) - th$ hidden layer and the $l - th$ hidden layer, and B_l is the bias vector of the $l - th$ hidden layer.

(1) If $(H_{lE})^T H_{lE}$ is nonsingular, the forced positive-definite Cholesky factorization factor Q_E of matrix E is calculated according to equations (49a) and (49b), and $(V_E)^{(k)}$ $(k = 1, 2, \cdots, L)$ is calculated according to equation (51) using Q_E and $((g^{-1}(H_{l*}))^T)^{(k)}$ $(k = 1, 2, \cdots, L)$. Then the matrices Q_E and $(V_E)^{(k)}$ $(k = 1, 2, \cdots, L)$are used to calculate $(U_E)^{(k)}$ $(k = 1, 2, \cdots, L)$ according to formula (52); therefore we can obtain $(W_l)_{HE} = ((U_E)^{(1)}, (U_E)^{(2)}, \cdots, (U_E)^{(L)})^T(H_{lE})^T$.

(2) If $H_{lE}(H_{lE})^T$ is nonsingular, the forced positive-definite Cholesky factorization factor Q_F of matrix F is calculated according to equations (53a) and (53b), and $(V_F)^{(k)}$ $(k = 1, 2, \cdots, L)$ is calculated according to equation (55) using Q_F and $(b_F)^{(k)}$ $(k = 1, 2, \cdots, L)$. Then the matrices Q_F and $(V_F)^{(k)}$ $(k = 1, 2, \cdots, L)$ are used to calculate $(((W_l)_{HE})^T)^{(k)}$ according to formula (56); thereby we can get $(W_l)_{HE} = ((((W_l)_{HE})^T)^{(1)}, (((W_l)_{HE})^T)^{(2)}, \cdots, (((W_l)_{HE})^T)^{(L)})^T$.

Step 7. Calculate the predicted output matrix of the $l - th$ hidden layer according to the equation $H_l = g(W_l H_{l-1} + B_l) = g((W_l)_{HE}H_{lE})$.

Step 8. Recalculate the connection weight matrix $\beta_l = (\lambda^{-1}I + (H_l)^T H_l)^{-1}(H_l)^T T$ or $\beta_l = (H_l)^T(\lambda^{-1}I + H_l(H_l)^T)^{-1}T$ between the $l - th$ hidden layer and the output layer according to the principle shown in (28)-(36).

Step 9. On the basis of β_l, the MRELM prediction model with l hidden layers is established, and the final output result $f(x) = H_l\beta_l$ is calculated.

Step 10. Calculate the sum of the empirical risk and the structural risk of the MRELM prediction model.

$$R_l = \frac{1}{2}(\beta_l)^T \beta_l + \frac{\lambda}{2}\varepsilon^T\varepsilon. \tag{57}$$

Then jump to Step 5. Judging the formula (58) from $l = 5$,

$$\left|\frac{(R_{l-i} - R_{l-i-1})}{R_{max}}\right| \le \xi, \tag{58}$$

where ξ is the learning precision and R_{max} is the maximum value of $R_1, \cdots R_l$, $i = 0, \cdots, 3$. If the formula (58) is satisfied, the training process terminates, determine l as the optimal number of hidden layers, and establish the corresponding MRELM prediction model; otherwise continue to increase l until the condition is met.

The number of hidden layers in MRELM increases successively from the initial value, and the expansion stops when R_l is no longer significantly reduced. At this time, even if the hidden layer is continued to be added, the R_l representing the learning accuracy and generalization ability of MRELM model will not be significantly improved but will lead to a large number of redundant hidden layers in MRELM. Therefore, the MRELM model at this time has the optimal number of hidden layers.

3.4. Proof of Positive Definiteness for FC-IMRELM. In the implementation of FC-IMRELM algorithm, the process of solving the connection weights matrix β_l and the learning parameters $(W_l)_{HE}$ in the $l - th$ hidden layer can be transformed into solving linear equations in the form of equations

(25), (26), (37), (38), (47), and (48). The premise of applying standard Cholesky factorization to solving linear equations is that its coefficient matrix must be a symmetric positive definite matrix, and thus we need to prove that the matrices $A = \lambda^{-1}I + (H_1)^T H_1$ and $B = \lambda^{-1}I + H_1(H_1)^T$ are symmetric positive definite matrices. The precondition of applying the forced positive definite Cholesky factorization to solving linear equations is that the coefficient matrix must be a symmetric matrix; therefore it is necessary to prove that the matrices $C = (\beta_l)^T\beta_l$, $D = \beta_l(\beta_l)^T$, $E = (H_{lE})^T H_{lE}$, and $F = H_{lE}(H_{lE})^T$ are symmetric semipositive definite matrices. The following theorem shows that the matrices $(H_l)^T H_l$ and $H_l(H_l)^T$ are symmetric semipositive definite matrices, and $\lambda^{-1}I$ is symmetric positive definite matrix; hence the matrices $\lambda^{-1}I + (H_l)^T H_l$ and $\lambda^{-1}I + H_l(H_l)^T$ are symmetric positive definite matrices, too. Consequently, Eq. (25) and Eq. (26) can be solved by the standard Cholesky factorization, and the connection weights matrix β_l in the $l-th$ hidden layer can be gotten. In addition, the matrices $(\beta_l)^T\beta_l$, $\beta_l(\beta_l)^T$, $(H_{lE})^T H_{lE}$, and $H_{lE}(H_{lE})^T$ are symmetric semipositive definite matrices, so Eq. (37), Eq. (38), Eq. (47), and Eq. (48) can be calculated by force positive-definite Cholesky factorization, and the learning parameters $(W_l)_{HE}$ in the $l-th$ hidden layer can be obtained.

Theorem 3. *Let H_l be a matrix. Then $\lambda^{-1}I + (H_l)^T H_l$ and $\lambda^{-1}I + H_l(H_l)^T$ are all symmetric positive definite matrices.*

Proof. Obviously, for the matrix $A = \lambda^{-1}I + (H_l)^T H_l$, then by properties of transpose we have

$$\begin{aligned} A^T &= \left(\lambda^{-1}I + (H_l)^T H_l\right)^T = \lambda^{-1}I + (H_l)^T \left((H_l)^T\right)^T \\ &= \lambda^{-1}I + (H_l)^T H_l = A. \end{aligned} \tag{59}$$

So, A is symmetric.

Similarly, for the matrix $B = \lambda^{-1}I + H_l(H_l)^T$, applying properties of transpose again, we get

$$\begin{aligned} B^T &= \left(\lambda^{-1}I + H_l(H_l)^T\right)^T = \lambda^{-1}I + \left((H_l)^T\right)^T H_l\Big)^T \\ &= \lambda^{-1}I + H_l(H_l)^T = B. \end{aligned} \tag{60}$$

Hence, B is symmetric.

On the other hand, let $\eta = (\eta_1, \eta_2, \cdots, \eta_L)^T$ be a nonzero vector in R^L. We can easily verify that

$$\eta^T\left(\lambda^{-1}I\right)\eta = \lambda^{-1}\sum_{i=1}^{L}\eta_i^2 > 0. \tag{61}$$

In addition,

$$\eta^T(H_l)^T H_l\eta = (H_l\eta)^T H_l\eta = \|H_l\eta\|^2 \geq 0. \tag{62}$$

Thus, the matrix $A = \lambda^{-1}I + (H_l)^T H_l$ is positive definite

Also, let $\mu = (\mu_1, \mu_2, \cdots, \mu_N)^T$ be a nonzero vector in R^N. Then we deduce that

$$\mu^T\left(\lambda^{-1}I\right)\mu = \lambda^{-1}\sum_{i=1}^{L}\mu_i^2 > 0. \tag{63}$$

Additionally,

$$\begin{aligned} \mu^T H_l(H_l)^T\mu &= \left((H_l)^T\mu\right)^T(H_l)^T\mu = \left\|(H_l)^T\eta\right\|^2 \\ &\geq 0. \end{aligned} \tag{64}$$

Consequently, the matrix $B = \lambda^{-1}I + H_l(H_l)^T$ is positive definite. □

Theorem 4. *Let β_l be a matrix. Then $(\beta_l)^T\beta_l$ and $\beta_l(\beta_l)^T$ are all symmetric positive semidefinite matrices.*

Proof. First, for the matrix $C = (\beta_l)^T\beta_l$, it is not hard to see from properties of transpose that

$$C^T = \left((\beta_l)^T\beta_l\right)^T = (\beta_l)^T\left((\beta_l)^T\right)^T = (\beta_l)^T\beta_l = C. \tag{65}$$

Thus, C is symmetric.

Second, by the same method we have for $D = \beta_l(\beta_l)^T$

$$D^T = \left(\beta_l(\beta_l)^T\right)^T = \left((\beta_l)^T\right)^T(\beta_l)^T = \beta_l(\beta_l)^T = D. \tag{66}$$

Therefore, D is symmetric.

Finally, it is clear that $(\beta_l)^T\beta_l$ is a square symmetric matrix. For any nonzero vector $\sigma = (\sigma_1, \sigma_2, \cdots \sigma_m)^T$ in R^m, we have

$$\sigma^T(\beta_l)^T\beta_l\sigma = (\beta_l\sigma)^T\beta_l\sigma = \|\beta_l\sigma\|^2 \geq 0. \tag{67}$$

Thus, the matrix $C = (\beta_l)^T\beta_l$ is positive semidefinite.

It is also clear that $\beta_l(\beta_l)^T$ is a square symmetric matrix. For any nonzero vector $\tau = (\tau_1, \tau_2, \cdots, \tau_L)^T$ in R^L, we get

$$\tau^T\beta_l(\beta_l)^T\tau = \left((\beta_l)^T\tau\right)^T(\beta_l)^T\tau = \left\|(\beta_l)^T\tau\right\|^2 \geq 0. \tag{68}$$

Consequently, the matrix $D = \beta_l(\beta_l)^T$ is positive semidefinite. □

Theorem 5. *Let H_{lE} be a matrix. Then $(H_{lE})^T H_{lE}$ and $H_{lE}(H_{lE})^T$ are all symmetric positive semidefinite matrices.*

Proof (apparent). We can prove Theorem 5 using the same idea as in Theorem 4. □

4. Results and Discussion

In this section, experiments of our proposed FC-IMRELM are conducted on benchmark data sets for classification and regression problems. In order to investigate the improvement of learning accuracy of our methods, original ELM [5], TELM [17], and MRELM [18] are also evaluated. All the

Table 1: Properties of the classification datasets.

Data set	Attributes	Classes	Training	Testing
Banknote	4	2	857	515
Blood	4	2	500	258
Diabetic	19	2	719	432
Image	19	7	1443	867
Wilt	5	2	4339	500
Coal spectral data	8	4	196	119
Iron spectral data	5	2	86	37

Table 2: Average testing classification correct percentage for the algorithms ELM, TELM, MRELM, and FC-IMRELM using classification datasets (%).

Data set	ELM	TELM	MRELM	FC-IMRELM
Banknote	99.82	99.88	99.93	99.95
Blood	78.95	80.78	80.71	81.12
Diabetic	72.90	73.79	73.92	74.05
Image	86.61	91.09	91.29	91.98
Wilt	62.39	66.35	66.32	67.26
Coal spectral data (Coal)	89.31	91.22	91.02	94.12
Iron spectral data (Iron)	96.26	96.15	97.56	97.83

Table 3: Specifications for the regression datasets.

Data set	Attributes	Training	Testing
Bodyfat	14	212	40
Pyrim	27	64	10
Triazines	60	161	25

performance assessments are conducted in the MATLAB 2014b computational environment running on Windows 10 operating system with Intel Core™ i5-7200U CPU @2.7 GHz, 8 GB RAM, NVIDIA M150 Graphics card, and 2 GB GDDR5 video memory. Furthermore, to comprehensively compare resulting performances, the activation function of each algorithm used in the experiments is uniformly assigned as sigmoid function: $g(x) = 1/(1 + \exp(-\lambda x))$, where λ is 1. The number of hidden neurons is set to 20. Moreover, each algorithm runs for 100 trials and the results obtained will be averaged as the final value.

4.1. Classification Problems

4.1.1. Characteristics of Classification Datasets. The classification datasets are obtained from the UCI website [26] and the literature [27, 28]. To evaluate the robustness of our FC-IMRELM algorithm, we conducted the tests using simple benchmark datasets and real datasets collected from coal and iron ores industries. The characteristics of the datasets are shown in Table 1.

4.1.2. Evaluation of Testing Accuracy on Classification Datasets. In order to make the performance evaluation more comprehensive, the real datasets that are related to complex industrial data were added to our performance evaluation.

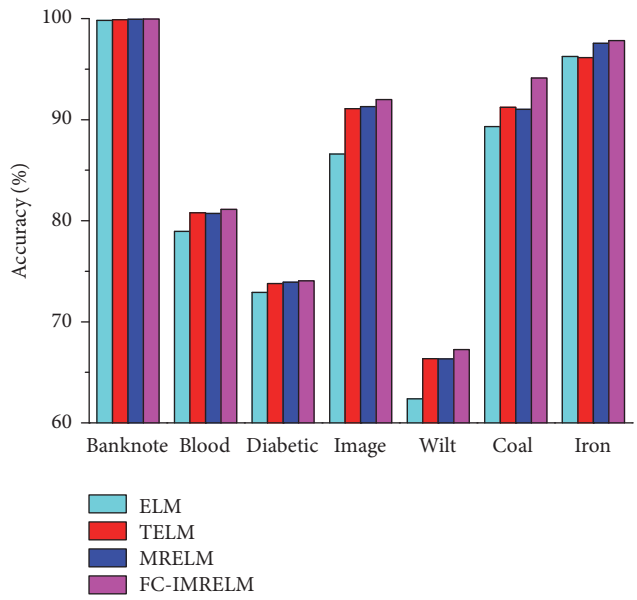

Figure 1: The average testing classification accuracy for the algorithms ELM, TELM, MRELM, and FC-IMRELM using classification datasets.

The original ELM, TELM, and MRELM algorithms are tested using the simple benchmark datasets and real datasets to validate the improvement of learning accuracy of our IMRELM algorithm. From Table 2 and Figure 1, we can see that each algorithm has a good classification accuracy for Banknote dataset. For Blood, Diabetic, Wilt, Coal spectral, and Iron spectral datasets, the algorithms TELM, MRELM, and FC-IMRELM all outperform the ELM algorithm, and the FC-IMRELM algorithm has the highest classification

Table 4: Performance comparison among the original ELM, TELM, MRELM, and FC-IMRELM on regression datasets.

Data set	ELM		TELM		MRELM		FC-IMRELM	
	RMSE	R2	RMSE	R2	RMSE	R2	RMSE	R2
Bodyfat	0.0084	0.80	0.002	0.986	0.0021	0.984	0.002	0.99
Pyrim	0.12	0.34	0.08	0.47	0.09	0.61	0.06	0.67
Triazines	0.12	0.18	0.12	0.21	0.13	0.18	0.091	0.38

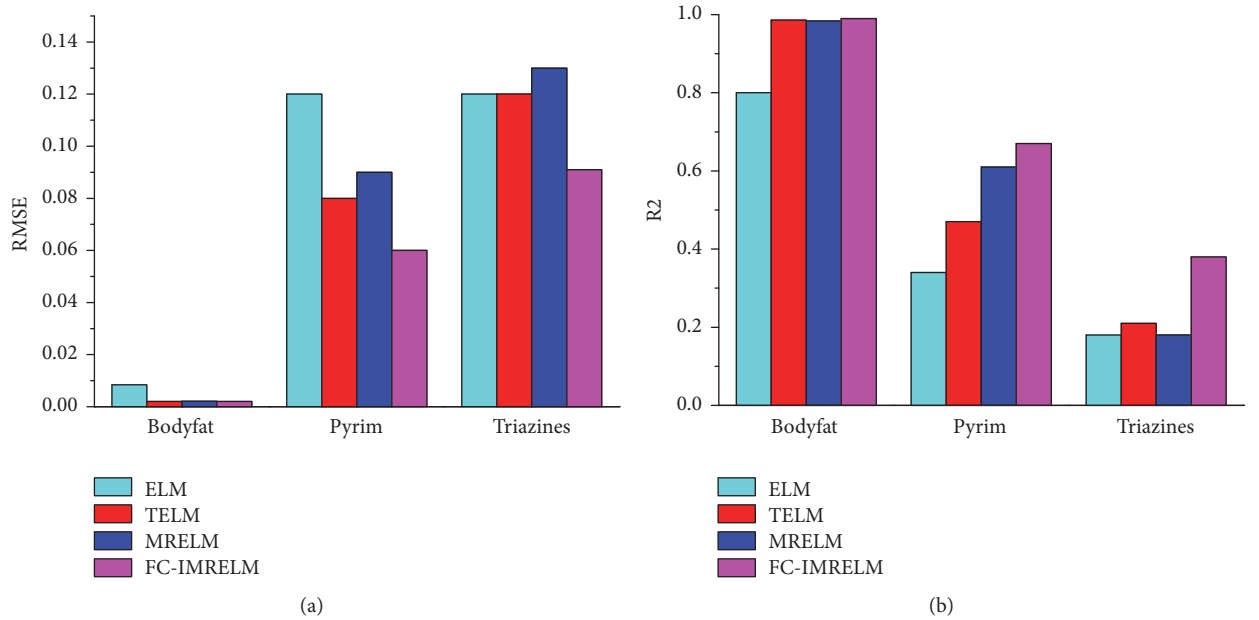

Figure 2: The average RMSE (a) and R2 (b) for the algorithms ELM, TELM, MRELM, and FC-IMRELM using regression datasets.

accuracy. For Image dataset, the classification accuracy of the algorithms TELM, MRELM, and FC-IMRELM is much higher than that of the ELM algorithm, and the classification accuracy of the FC-IMRELM algorithm is still the highest, reaching 91.98%. The experimental results show that the average classification accuracy of our FC-IMRELM algorithm is significantly higher than that of the original ELM, TELM, and MRELM algorithms, and the computational experiments using Coal spectral and Iron spectral datasets also demonstrate that our FC-IMRELM algorithm can be easily extended to practical application.

4.2. Regression Problems. For the regression model, the root-mean-square error (RMSE) and the coefficient of determination (R2) [29] are used as the model performance evaluation indexes in this study to verify the effectiveness of the proposed FC-IMRELM algorithm. R2 and RMSE are expressed as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N_{test}} (y_i - \widehat{y}_i)^2}{N_{test}}} \tag{69}$$

$$R2 = 1 - \frac{\sum_{i=1}^{N_{test}} (y_i - \widehat{y}_i)^2}{\sum_{i=1}^{N_{test}} (y_i - \overline{y})^2} \tag{70}$$

where N_{test} is the number of samples in the prediction set; y_i is the actual value of the sample; $\overline{y}$ is the average of the actual values; $\widehat{y}_i$ is the predicted value calculated by the model. The value range of R2 is between (0, 1), the closer the value of R2 to 1 and the smaller the value of RMSE, the better the performance of the model.

4.2.1. Characteristics of Regression Datasets. The regression datasets are obtained from the LIBSVM website [30]. The characteristics of the datasets are shown in Table 3.

4.2.2. Evaluation of Estimation Accuracy on Regression Datasets. As shown in Table 4 and Figure 2, we can conclude that the algorithms TELM, MRELM, and FC-IMRELM all have good prediction results for the Bodyfat dataset, where RMSE of these algorithms are small, and R2 are above 0.98, while the prediction ability of the ELM algorithm is slightly worse, and R2 of ELM is equal to 0.8. For Pyrim and Triazines datasets, the prediction results utilizing the FC-IMRELM algorithm are better than those utilizing the algorithms ELM, TELM, and MRELM, among which RMSE of the FC-IMRELM algorithm is the smallest, and R2 is the highest. The above analysis of experimental results indicates that the advantage of the FC-IMRELM algorithm is that it can better extract data characteristics for the multiattribute data, so it has better predictive ability.

5. Conclusions

(1) First of all, compared with MRELM, the FC-IMRELM algorithm proposed in this paper uses the idea of forced positive definite Cholesky factorization to determine the hidden layer parameters. The training process is more simplified, with low calculation amount and high numerical stability. In addition, the MRELM algorithm needs to set the network structure in advance, and the number of hidden layers remains unchanged in the training process, while the FC-IMRELM algorithm can automatically select the optimal number of hidden layers through the principle of structural risk and empirical risk minimization and adjust the network structure adaptively according to the training samples.

(2) Secondly, compared with CF-FORELM, the FC-IMRELM algorithm proposed in this paper is designed for the semipositive definite matrices appearing in the parameter solving process of MRELM model. The condition number of the matrix is improved while forcing the matrix positive definite, thereby accelerating the convergence speed of the MRELM model and ensuring the numerical stability of the modeling process.

(3) Finally, by introducing parameter λ to weigh the structural risk and empirical risk of ELM model, the FC-IMRELM algorithm has significantly improved its generalization ability compared with the traditional neural network. In addition, the forced positive definite Cholesky factorization is used to calculate its output weights, effectively reducing the computational cost brought by the increasing process of hidden layers. The prediction example shows that the FC-IMRELM algorithm can effectively avoid the numerical instability of MRELM model and has the advantages of high prediction accuracy and fast calculation speed, which can provide a novel and efficient solution to the prediction problem.

Data Availability

The datasets are obtained from [26–28, 30]. All data is available.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0304100, in part by the National Natural Science Foundation of China under Grant 71672032, and in part by the Fundamental Research Funds for Central University under Grant N180404012 and Grant N182608003.

References

[1] C.-M. Chang, T.-K. Lin, and C.-W. Chang, "Applications of neural network models for structural health monitoring based on derived modal properties," *Measurement*, vol. 129, pp. 457–470, 2018.

[2] H. Kim, C. Sui, K. Cai, B. Sen, and J. Fan, "An efficient high-speed channel modeling method based on optimized design-of-experiment (DoE) for artificial neural network training," *IEEE Transactions on Electromagnetic Compatibility*, vol. 60, no. 6, pp. 1648–1654, 2018.

[3] Z.-Y. Wang, C. Lu, and B. Zhou, "Fault diagnosis for rotary machinery with selective ensemble neural networks," *Mechanical Systems and Signal Processing*, vol. 113, pp. 112–130, 2018.

[4] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol. 2, pp. 985–990, July 2004.

[5] 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.

[6] M. Sahani and P. K. Dash, "Variational mode decomposition and weighted online sequential extreme learning machine for power quality event patterns recognition," *Neurocomputing*, vol. 310, pp. 10–27, 2018.

[7] L. Yang and S. Zhang, "A sparse extreme learning machine framework by continuous optimization algorithms and its application in pattern recognition," *Engineering Applications of Artificial Intelligence*, vol. 53, pp. 176–189, 2016.

[8] W. Zheng, X. Peng, D. Lu et al., "Composite quantile regression extreme learning machine with feature selection for short-term wind speed forecasting: a new approach," *Energy Conversion and Management*, vol. 151, pp. 737–752, 2017.

[9] Y. Chen and W. Wu, "Mapping mineral prospectivity using an extreme learning machine regression," *Ore Geology Reviews*, vol. 80, pp. 200–213, 2017.

[10] S. Yuong Wong, K. Siah Yap, and H. Jen Yap, "A Constrained Optimization based Extreme Learning Machine for noisy data regression," *Neurocomputing*, vol. 171, pp. 1431–1443, 2016.

[11] 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 (CIDM '09)*, pp. 389–395, April 2009.

[12] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[13] S. Ding, G. Ma, and Z. Shi, "A rough RBF neural network based on weighted regularized extreme learning machine," *Neural Processing Letters*, vol. 40, no. 3, pp. 245–260, 2014.

[14] 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 egression problems," *Neurocomputing*, vol. 74, no. 17, pp. 3716–3721, 2011.

[15] W. Zheng, Y. Qian, and H. Lu, "Text categorization based on regularization extreme learning machine," *Neural Computing and Applications*, vol. 22, no. 3-4, pp. 447–456, 2013.

[16] 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.

[17] B. Qu, B. Lang, J. Liang, A. Qin, and O. Crisalle, "Two-hidden-layer extreme learning machine for regression and classification," *Neurocomputing*, vol. 175, pp. 826–834, 2016.

[18] D. Xiao, B. Li, and Y. C. Mao, "A multiple hidden layers extreme learning machine method and its application," *Mathematical Problems in Engineering*, vol. 2017, Article ID 4670187, 10 pages, 2017.

[19] X. Wen, H. Liu, G. Yan, and F. Sun, "Weakly paired multimodal fusion using multilayer extreme learning machine," *Soft Computing*, vol. 22, no. 11, pp. 3533–3544, 2018.

[20] X. Su, S. Zhang, Y. Yin, Y. Liu, and W. Xiao, "Data-driven prediction model for adjusting burden distribution matrix of blast furnace based on improved multilayer extreme learning machine," *Soft Computing*, vol. 22, no. 11, pp. 3575–3589, 2018.

[21] X. Li, W. Mao, and W. Jiang, "Multiple-kernel-learning-based extreme learning machine for classification design," *Neural Computing and Applications*, vol. 27, no. 1, pp. 175–184, 2016.

[22] Y. Yang, Q. M. J. Wu, Y. Wang, K. M. Zeeshan, X. Lin, and X. Yuan, "Data partition learning with multiple extreme learning machines," *IEEE Transactions on Cybernetics*, vol. 45, no. 8, pp. 1463–1475, 2015.

[23] P. E. Gill and W. Murray, "Newton-type methods for unconstrained and linearly constrained optimization," *Mathematical Programming*, vol. 7, pp. 311–350, 1974.

[24] P. E. Gill and W. Murray, "Quasi-Newton methods for unconstrained optimization," *Journal of the Institute of Mathematics and Its Applications*, vol. 9, pp. 91–108, 1972.

[25] C. M. Wong, C. M. Vong, P. K. Wong, and J. Cao, "Kernel-based multilayer extreme learning machines for representation learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 3, pp. 757–762, 2018.

[26] http://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 machin," *Optics and Laser Technology*, vol. 114, pp. 10–15, 2019.

[28] D. Xiao, C. Liu, and B. T. Le, "Detection method of TFe content of iron ore based on visible-infrared spectroscopy and IPSO-TELM neural network," *Infrared Physics and Technology*, vol. 97, pp. 341–348, 2019.

[29] B. T. Le, D. Xiao, Y. Mao, and D. He, "Coal analysis based on visible-infrared spectroscopy and a deep neural network," *Infrared Physics & Technology*, vol. 93, pp. 34–40, 2018.

[30] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets.