Hindawi
Computational Intelligence and Neuroscience
Volume 2018, Article ID 3635845, 10 pages
https://doi.org/10.1155/2018/3635845

Research Article

An Extreme Learning Machine Based on Artificial Immune System

Hui-yuan Tian, Shi-jian Li, Tian-qi Wu, and Min Yao

School of Computer Science and Technology, Zhejiang University, Hangzhou, China

Correspondence should be addressed to Shi-jian Li; shijianli@zju.edu.cn

Received 5 March 2018; Accepted 27 May 2018; Published 25 June 2018

Academic Editor: Rodolfo Zunino

Extreme learning machine algorithm proposed in recent years has been widely used in many fields due to its fast training speed and good generalization performance. Unlike the traditional neural network, the ELM algorithm greatly improves the training speed by randomly generating the relevant parameters of the input layer and the hidden layer. However, due to the randomly generated parameters, some generated "bad" parameters may be introduced to bring negative effect on the final generalization ability. To overcome such drawback, this paper combines the artificial immune system (AIS) with ELM, namely, AIS-ELM. With the help of AIS's global search and good convergence, the randomly generated parameters of ELM are optimized effectively and efficiently to achieve a better generalization performance. To evaluate the performance of AIS-ELM, this paper compares it with relevant algorithms on several benchmark datasets. The experimental results reveal that our proposed algorithm can always achieve superior performance.

1. Introduction

In recent years, many computational intelligence techniques, such as neural networks and support vector machines (SVMs) [1], have been widely used in many real-world applications. However, those algorithms face some defects such as slow learning speed, trivial human intervention, and poor computational scalability.

Recently, to solve the drawbacks mentioned above, Huang et al. [2–5] proposed a new method named extreme learning machine (ELM) which has attracted ever-growing research attention. In contrast to the traditional neural networks such as BP [6], ELM is a tuning-free algorithm with fast learning speed by randomly generating input weights and hidden biases. With the help of least square method and Moore-Penrose generalized inverse, the ELM is transferred as a linear learning system. In addition, ELM is theoretically proved to have a good generalization performance with least human intervention. Therefore, ELM is widely used in many fields [5]. For example, Chaturvedi et al. [7] extended the extreme learning machine (ELM) paradigm to a novel framework that exploits the features of both Bayesian networks and fuzzy recurrent neural networks to perform subjectivity detection. Gastaldo et al. [8] addressed the specific role played by feature mapping in ELM. Cambria et al. [9] explored how the high generalization performance, low computational complexity, and fast learning speed of extreme learning machines can be exploited to perform analogical reasoning in a vector space model of affective common-sense knowledge. Recently Ragusa et al. [10] tackled the implementation of single hidden layer feedforward neural networks (SLFNs), based on hard-limit activation functions, on reconfigurable devices.

It is known that an appropriate selection of initial weight sets is very vital for training a neural network model [11]. There is a strong correlation between the final solution and the initial weight. However, due to the random determination of some learning parameters, some nonoptimal parameter may be introduced to the model [5], which may put negative impact on the final performance. To solve such a drawback, many relative works have been proposed in the past ten years. A straightforward way is to combine evolutionary methods with ELM [12]. For instance, Zhu et al. [13] utilized differential evolutionary algorithm (DE) to optimize ELM's generated parameters to achieve better performance. In [14], Xue et al. combined genetic algorithm (GA), ELM, and ensemble

learning to get a better and stable result. Rather than using GA or DE method, Sarasw athi et al. presented a PSO driven ELM [15], combining with Integer Coded Genetic Algorithm (ICGA) to solve gene selection and cancer classification. In [16], Cao et al. proposed an improved learning algorithm named self-adaptive evolutionary extreme learning machine (SaE-ELM). Similarly, Wu et al. presented a novel algorithm named dolphin swarm algorithm extreme learning machine (DSA-ELM) [17] to solve optimization problems.

However, all the above evolutionary algorithms have different search efficiency to optimize the problem. There is still much space to improve. For example, it is one of the biggest challenges in ELM that some nonoptimal parameters may be introduced to ELM algorithm due to the random generation of parameters. To overcome that challenge, in this paper we propose a new extreme method named artificial immune system extreme learning machine (AIS-ELM). Because artificial immune system (AIS) [18–20] has global search ability [21] and good convergence [22], it can solve some difficulties like slow convergence, getting stuck in local minima, etc. Therefore, we use AIS to optimize ELM to get a better initial weight sets capable of avoiding the training process falling into the local optimum. The original version and preliminary results of this paper's method were proposed by us in ELM2017 [23]. In this paper we have revised the original formulas, compared the AIS-ELM with more algorithms and added new expressions, regression validation and more datasets.

The rest of the paper is arranged as follows. Sections 2 and 3 briefly describe the traditional ELM and AIS methods. Section 4 proposes the detailed description of AIS-ELM. Section 5 carries out corresponding experiment: AIS-ELM algorithm is compared with traditional ELM, PSO-ELM, SaE-ELM, and DSA-ELM on five regression problems and eight classification benchmark problems obtained from the UCI Machine Learning Repository [24]; the training times between AIS-ELM and BP and SVM and traditional ELM are compared on three benchmark classification problems. The last section gives a conclusion of this paper.

2. Extreme Learning Machine

This section will introduce the extreme learning machine [2–5] proposed by Professor Huang. ELM is developed from a single hidden layer feedforward network and is extended to a generalized single hidden layer feedforward network. Compared to other conventional learning algorithms, the extreme learning algorithm's advantage is that the nodes of the single hidden layer feedforward network need not be adjusted.

Compared with the traditional learning algorithm, the extreme learning machine not only has the smaller error but can reach the smallest norm of weights [5]; because the hidden layer need not to be adjusted in the limit learning machine algorithm, the output weight matrix can be solved by the least squares method.

For N arbitrary training samples $\{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=1}^{N}$, where $\boldsymbol{x_i} = [x_{i1}, x_{i2}, \ldots, x_{in}]^T \in \boldsymbol{R}^{\mathrm{n}}$ and $\boldsymbol{t_i} = [t_{i1}, t_{i2}, \ldots, t_{im}]^T \in \boldsymbol{R}^m$, and given activation function $g(x)$, the standard mathematical model of SLFNs with $\widetilde{N}$ hidden nodes is modeled as follows:

$$\sum_{i=1}^{\widetilde{N}} \beta_i g\left(\mathbf{w}_i \cdot \mathrm{x}_j + b_i\right) = \mathbf{o}_j, \quad j = 1, \ldots, N \tag{1}$$

where $\mathbf{w}_i = [w_{i1}, w_{i2}, \ldots w_{in}]^T$ is the weight vector connecting the input neurons and ith hidden neuron, $\beta_i = [\beta_{i1}, \beta_{i2}, \ldots \beta_{im}]^T$ is the weight vector connecting the ith hidden neuron and the output neurons, and b_i is the threshold of the ith hidden neuron.

That standard SLFNs with $\widetilde{N}$ hidden neurons given activation function $g(x)$ can approximate these N samples with zero error which means that

$$\sum_{j=1}^{N} \left\| \mathbf{o}_j - \mathbf{t}_j \right\| = 0 \tag{2}$$

There exist β_i, $\mathbf{w}_i$, and b_i such that

$$\sum_{i=1}^{\widetilde{N}} \beta_i g\left(\mathbf{w}_i \cdot \mathrm{x}_j + b_i\right) = \mathbf{t}_j, \quad j = 1, \ldots, N \tag{3}$$

The above N equations can be written compactly as

$$H\beta = T \tag{4}$$

where

$$\begin{aligned} \mathbf{H} &= \begin{bmatrix} \mathbf{h}\left(\mathbf{x}_1\right) \\ \vdots \\ \mathbf{h}\left(\mathbf{x}_N\right) \end{bmatrix} \\ &= \begin{bmatrix} g\left(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1\right) & \cdots & g\left(\mathbf{w}_{\widetilde{N}} \cdot \mathbf{x}_1 + b_{\widetilde{N}}\right) \\ \vdots & \cdots & \vdots \\ g\left(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1\right) & \cdots & g\left(\mathbf{w}_{\widetilde{N}} \cdot \mathbf{x}_N + b_{\widetilde{N}}\right) \end{bmatrix}_{N \times \widetilde{N}} \end{aligned} \tag{5}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_{\widetilde{N}}^T \end{bmatrix}_{\widetilde{N} \times m}, \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}_{N \times m} \tag{6}$$

Here, $\mathbf{H}$ is called the hidden layer output matrix [3].The column of $\mathbf{H}$ is the ith hidden node's output vector with respect to inputs $\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N$ and the jth row of H is the output vector of the hidden layer with respect to $\mathbf{x}_j$. Then the vector $\boldsymbol{\beta}$ (connecting the hidden layer with the output layer) is estimated using the Moore-Penrose generalized inverse of the matrix $\boldsymbol{H}$:

$$\widehat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{T} \tag{7}$$

ELM algorithm can be summarized as shown in Algorithm 1.

Step 1 Randomly generate the input weights $\mathbf{w}_i$ and hidden biases b_i
Step 2 Calculate the hidden-layer output matrix $\boldsymbol{H}$
Step 3 Compute the output weights matrix as $\widehat{\boldsymbol{\beta}} = \boldsymbol{H}^{\dagger}\boldsymbol{T}$

Algorithm 1: Standard ELM.

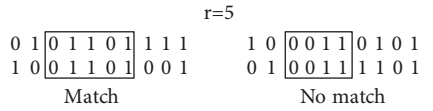

Figure 1: Matching under the rule of *r-contiguous bits*. In this example, $r = 5$, so the left is matching while the right is not.

3. Artificial Immune System

A relatively new area of bioinspired computing is Artificial Immune Systems (AIS). It is inspired by biological models of the natural immune system which has the properties of diversity, distributed computation, dynamic learning, error tolerance, adaptation, and self-monitoring. AIS can be applied to many domains [19] in principle for the reason that it is a general framework for a distributed adaptive system. This section will introduce the AIS in three aspects. Firstly, the clonal selection algorithm is briefly described in Section 3.1. Secondly, Section 3.2 introduces the mathematical model of BCA. Finally, the mathematical model describing the interaction of Antigen-Antibody is represented in Section 3.3.

3.1. Mathematical Model of BCA. Each B cell is modeled as binary strings of fixed length l for simplicity of calculation. One of the most important design choices in developing an Artificial Immune Systems algorithm is similarity measure or matching rule [25], and it is closely coupled to the encoding scheme. Hamming distance and edit distance are the obvious approximate matching rules.

However, there is a more immunologically plausible rule, called *r-contiguous bits* [26]: two strings will match if they have r-contiguous bits in common (see Figure 1). The value r is a threshold which serves as indication of the size of the subset of strings that a single string can match. For example, if $r = l$, the matching is completely special; i.e., the string will match only a single string (itself), but if $r = 0$, the matching is absolutely general; that is, the string will match every single string of length l.

Besides, in the algorithm a contiguous region hypermutation operator [27] is used and its form is

$$f_T = \frac{1}{L^2}\left[\sum_{n=1}^{a}\sum_{m=b}^{L-1}(1-r)^{m+1-n-k}\,r^k + \sum_{n=1}^{a} n\,(1-r)^{L+1-n-k}\,r^k\right] \tag{8}$$

where f_T is the probability of transition from zero to some number T ($0 \le T \le 2^L - 1$); L is the length of the binary string; a is the bit position of the first "flip" bit starting from the most significant bit; b is the bit position of the last "flip" bit starting from the most significant bit; k is the number of bits that must be flipped to mutate from 0 to T. r is the mutation probability of a bit given a contiguous region. $\{L, a, b, k, T\} \in \mathbb{Z}^+; 0 \le T \le 2^L - 1; a \le b \le L; \{f_T, r\} \in \mathbb{R}; 0 \le \mathrm{r} \le 1$

3.2. Clonal Selection Theory. The clonal selection theory (CST) [28] is used to explain how the adaptive immune system responds to an antigenic stimulus basically. It establishes the theory that only cells that are capable of recognizing an antigen will proliferate, while those that are incapable of doing so will be eliminated.

Both T cells and B cells can operate clonal selection. In the case of B cells, when the antigen receptors bind with an antigen, B cells begin to clone themselves and undergo somatic hypermutation to introduce diversity into the B cell population. After that B cells become activated and differentiate into plasma or memory cells. Plasma cells produce numerous antigen-specific antibodies leading to the removal of the antigen in a successful immune response. Memory cells remain within the host and promote a rapid secondary response when encountering the same (or similar) antigen. This is the operation of acquired immunity [19].

The B Cell Algorithm (BCA) as a simple clonal selection method was introduced in [22]. An outline of BCA is shown in Algorithm 2.

3.3. Shape Space. An abstract model describing the interaction of Antigen-Antibody is introduced by Perelson & Oster [29]. In this model, it is assumed that the characteristics of the antibody receptor (combined region) associated with antigen binding can be described by specifying a total of L shape parameters. It is also assumed that the same L parameters can be used to describe the antigen. These L parameters are incorporated into the vector, and the antibody receptor and antigenic determinant are described as Ab and Ag points, respectively, in an L–Euclidean shape space. Each molecule can be considered as a point in the L-dimensional real space mathematically and the affinity of Ag-Ab is related to the reciprocal of the Euclidean distance between them.

It is assumed that the antibody is capable of binding to any antigenic complement in the distance ε (stimulus region). Each N dimensional ball of radius ε takes up a volume $c_N\varepsilon^N$, where c_N is a constant which depends upon the

Step 1 **Initialization**: create an initial random population of individuals P
Step 2 **Main loop**: $\forall v \in P$:
(a) Affinity Evaluation: evaluate g(v);
(b) Clonal Selection and Expansion:
(i) Clone each B-cell: clone v and place in clonal pool C;
(ii) Select a random member of $v' \in C$ and apply the contiguous region hypermutation operator
(iii) Evaluate $g(v')$; if $g(\mathbf{v}') > g(\mathbf{v})$ then replace $\mathbf{v}$ by clone $\mathbf{v}'$
Step 3 **Cycle**: repeat step (2) until a certain stopping criterion is met.

Algorithm 2: B cell algorithm.

dimensionality of N (for arbitrary N, $c_N = 2\pi^{N/2}/N\Gamma(N/2)$, where $\Gamma(\cdot)$ is the Gamma function). If there are a total of N_{Ab} antibodies, its total coverage volume would not be greater than $N_{Ab}c_N\varepsilon^N$ since balls would overlap. Let us assume V is an n-dimensional cube with edge length R. The total volume of V is then R^N.

The goal is to maximize the coverage of antibody which can make the immune approach more reliable. Then the following equation must come into existence:

$$N_{Ab}c_N\varepsilon^N \geq R^N \tag{9}$$

Therefore, the range of N_{Ab} is as follows:

$$N_{Ab} \geq \frac{1}{c_N} \cdot \left(\frac{R}{\varepsilon}\right)^N \tag{10}$$

where $0 < \varepsilon < 1$.

4. Proposed Extreme Learning Machine Based on Artificial Immune System

This section proposes an Extreme learning machine based on artificial immune system, namely, AIS-ELM. Traditional ELM algorithm randomly generates input weights and hidden biases, and among them there may be some sets of nonoptimal input weights and hidden biases. It is necessary to optimize these nonoptimal input weights and hidden biases. Two methods can be used to solve this problem. One is to increase hidden neurons which is time-consuming and may not get a good result. The other is to optimize the input weights and hidden biases.

This paper combines AIS with ELM to optimize the input weights and hidden biases. AIS-ELM has three main phases: clone phase, mutation phase, and substitution phase. After the three phases, an optimal antibody will be produced. And the performance of ELM will be improved if the optimal antibody is used as the input weights and hidden biases.

One set of input weights and hidden biases are modeled by an antibody; the ith antibody is represented by

$$\begin{aligned}\mathbf{Ab_i} = [&a_{11}, a_{12}, \ldots, a_{1n}, a_{21}, a_{22}, \ldots, a_{2n}, a_{\widetilde{N}1}, a_{\widetilde{N}2}, \ldots, a_{\widetilde{N}n},\\ &b_1, b_2, \ldots, b_{\widetilde{N}}]\end{aligned} \tag{11}$$

where $i = 1, 2, \ldots N$, N is the number of training data and the number of population members. $\widetilde{N}$ is the number of hidden nodes and n is the dimension of input samples. $a_{ij}(j = 1, 2, \ldots, n)$ are the input weights. b_i are the hidden biases. The initial values of a_{ij} and b_i are randomly generated within the range of [-1, 1]. Then we calculate each antibody's fitness $Fit_{\mathbf{Ab}_i}(i = 1, 2, \ldots, N)$ according to the following equation with the validation data $\{(\mathbf{x}_j, \mathbf{t}_j)\}_{j=1}^V$.

$$\mathbf{Fit}\left(\mathbf{Ab_i}\right) = \sqrt{\frac{\sum_{j=1}^{V} \left\| \sum_{i=1}^{\widetilde{N}} \left\| \boldsymbol{\beta}_i g\left(\mathbf{a_i} \cdot \mathbf{x_j} + b_i\right) - \mathbf{t_j} \right\|_2^2}{V}} \tag{12}$$

where $\boldsymbol{T} = [\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_V]^T$ is the validation data. The reason for using validation data instead of training data is to alleviate possible overfitting. The corresponding output weights $\boldsymbol{\beta}$ are computed by using the MP generalized inverse by (7).

The clone section creates a clone pool having N-1 clonal antibodies $\mathbf{Ac_{ij}}(j = 1, 2, \ldots, N\text{-}1)$ for every antibody $\mathbf{b}_i$, and each clonal antibody is identical to the original antibody, i.e., $\mathbf{Ac_{ij}} = \mathbf{Ab_i}$ $(j = 1, 2, \ldots, N\text{-}1)$.

In the mutation procedure, each clonal antibody $\mathbf{Ac_{ij}}$ in the clone pool is mutated by the following formula:

$$\mathbf{Ac_{ij}} = \mathbf{Ab_i}\left(1 + P_{mutation}\right) \tag{13}$$

where $j = 1, 2, \ldots, N\text{-}1$ and $P_{mutation}$ is the mutation probability of the clonal antibody.

$$P_{mutation} = (-1)^j \cdot j \cdot f_T' \cdot \mathbf{Fit}_{\mathbf{Ab}_i} \cdot \varepsilon \tag{14}$$

Where the following holds.

(1) $(-1)^j \cdot j$ avoids the situation in which the directions of mutation are the same and the result falls into local optimal.

(2) f_T' is as follows:

$$\begin{aligned}f_T' = \frac{1}{N'^2}\Bigg[&\sum_{y=1}^{a}\sum_{x=b}^{N'-1}(1-r)^{x+1-y-k}\, r^k\\ &+ \sum_{y=1}^{a} y\,(1-r)^{N'+1-y-k}\, r^k\Bigg]\end{aligned} \tag{15}$$

The above equation is an application of (8), where N' is the total elements of the antibody and $N' = \widetilde{N}(n+1)$; f_T' is the probability of transition from zero to some number T $(0 \leq T \leq 2^{N'} - 1)$; a is the bit position of the first "on" bit starting from the most significant bit; b is the bit position of the last "on" bit starting from the most significant bit; k is the number

Table 1: Detailed description of the eight benchmark classification datasets

Dataset	Data			Attributes	Classes
	Training	Validation	Testing		
Ecoli	180	78	78	7	8
Diabetes	384	22	192	8	2
Epileptic Seizure	6000	2750	2750	179	5
Heart Disease	150	76	76	75	5
Iris	70	40	40	4	3
Glass	100	57	57	9	7
Image	1200	555	555	19	7
Satellite	3435	1500	1500	36	7

of bits that must be flipped to mutate from 0 to T; r is the mutation probability of a bit given a contiguous region.

$$\begin{aligned} &\left\{N', a, b, k, T\right\} \in \mathbb{Z}^{+}; \quad 0 \le T \le 2^{N'} - 1; \ a \le b \le N'; \\ &\left\{f_T', r\right\} \in \mathbb{R}; \quad 0 \le \mathrm{r} \le 1. \end{aligned} \tag{16}$$

(3) $\mathbf{Fit}_{\mathbf{Ab}_i}$ is used to adjust the range of mutation. The smaller the value of fitness is, the smaller the error is, so the requirement of mutation changes is tinier. On the other hand, the greater the value of fitness is, the bigger the need for mutation changes will be.

(4) ε $(0 < \varepsilon < 1)$ is stimulus region in which the antibody is capable of binding to any antigenic complement [29].

The substitution phase is to calculate each clonal antibody's fitness $\mathbf{Fit}_{\mathbf{Ac}_{ij}}$ $(j = 1, 2, \ldots, N\text{-}1)$ in the clone pool, and to compare $\boldsymbol{Fit}_{\boldsymbol{Ac}_{ij}}$ with the cloned antibody's fitness $\mathbf{Fit}_{\mathbf{Ab}_i}$. If $\mathbf{Fit}_{\mathbf{Ac}_{ij}}$ is smaller than $\mathbf{Fit}_{\mathbf{Ab}_i}$, corresponding fitness and antibody will be replaced. For instance, if $i = 1$, $j = 1$, and $Fit_{\boldsymbol{Ab}_1} < Fit_{\boldsymbol{Ac}_{11}}$, it is necessary to replace $\mathbf{Ab_1}$ and $Fit_{\boldsymbol{Ab}_1}$ with $\mathbf{Ac}_{11}$ and $Fit_{\boldsymbol{Ac}_{11}}$. After this iterative process, the antibody population evolves forward global optimization. Then an antibody with minimal fitness which indicates smallest error is the optimal antibody.

In the above process, our algorithm uses the clonal selection principle to ensure diversity which has been proved by De Castro et al. [30]. In addition, the ELM is optimized by BCA to get a better convergence which has been proved by Clark et al. [22] through an exact Markov chain model. Besides, using $\mathbf{Fit}_{\mathbf{Ab}_i}$ to adjust the mutation matches up the theory of immune network. Last but not least, the requirement of shape space is also satisfied.

In the specific experiment process, a number of other algorithms have to be compared, so the input data $\mathbf{z}$ should be normalized to ensure fairness.

$$z^* = \frac{z - z_{\min}}{z_{\max} - z_{\min}} \tag{17}$$

Then the stop criterion is as follows:

$$SC = 1 - \sqrt{\overline{z}^T \overline{z}} \tag{18}$$

where $\overline{z}$ is the mean of the group of z.

$$\overline{z} = \frac{1}{N} \sum_{i=1}^{n} {z_i}^* \tag{19}$$

All in all, the AIS-ELM have three parts. The first part is initialization including input data, normalization, and set parameters. The second part applies AIS to ELM. After cloning, mutation, and substitution phase, an optimal antibody meeting the requirements is acquired. Then the antibody can be used as the input weights and hidden biases in ELM. The AIS-ELM is presented in Algorithm 3.

5. Performance Verification

In this section, AIS-ELM is compared with DS-ELM, PSO-ELM, SaE-ELM, traditional ELM, SVM, and BP. The experiments are divided into two parts. In the first part, the first five algorithms are tested on eight benchmark classification problems; next we compare AIS-ELM with SVM, BP, and traditional ELM on training time on three benchmark classification problems. In the second part, five benchmark regression problems are carried out. The experimental environment is MATLAB R2014b running on a windows pc with Intel 2.7 GHz CPU and 8GB RAM.

All the inputs have been normalized into the range [-1, 1] for fairness. The number of hidden neurons depends on different problems and it will be listed in specific experiment. Besides, the parameters for AIS-ELM are set as follows: $a = 10, b = 50, \varepsilon = 0.1, k = 5, r = 0.2$.

5.1. Classification. In this subsection, five algorithms' performances on eight benchmark classification problems are evaluated. The eight datasets are Ecoli, Pima Indians Diabetes (Diabetes), Epileptic Seizure, Iris, Heart Disease, Glass Identification (Glass), Image Segmentation (Image), and Statlog (Satellite), respectively. The detailed description of the eight datasets is listed in Table 1.

Attributes of all the dataset have been normalized to [-1, 1], and the output is the training time, testing accuracy's mean and variance. A 20-fold cross validation method is taken to get the average of 20 repeated experiments to minimize the error. The whole dataset is divided into training set, validation set, and testing set without overlap. And the three sets are kept coincident for each trial of the five

Step 1 Initialization
Randomly generate the initial antibody population $\mathbf{Ab}_i$ $(i = 1, 2, \ldots, \mathrm{N})$
where $\mathbf{Ab}_i = [a_{11}, a_{12}, \ldots, a_{1n}, a_{21}, a_{22}, \ldots, a_{2n}, a_{\widetilde{N}1}, a_{\widetilde{N}2}, \ldots, a_{\widetilde{N}n}, b_1, b_2, \ldots, b_{\widetilde{N}}]$.
Then calculate the fitness of $\mathbf{Ab}_i$ by Eq.(12),
and get $\mathbf{Fit}_{\mathbf{Ab}_i} = \{Fit_{\boldsymbol{Ab}_1}, Fit_{\boldsymbol{Ab}_2}, \ldots, Fit_{\boldsymbol{Ab}_N}\}$.
Step 2 Clone Selection
while the stop criterion is not met with **do**
Step 2.1 Clone Phase
For each antibody $\mathbf{Ab_i}$ clone **N-1** antibody, the clone pool named $\mathbf{Ac_i} = \{\mathbf{Ac_{i1}}, \mathbf{Ac_{i2}}, \ldots, \mathbf{Ac_{iN\text{-}1}}\}$ where $\mathbf{Ac_{i1}} = \mathbf{Ac_{i2}} = \mathbf{Ac_{iN\text{-}1}} = \mathbf{Ab_i}$
Step 2.2 Mutation Phase
For each clone antibody $\mathbf{Ac_{ij}}$, where $i = 1, 2, \ldots, N$ and $\mathrm{j} = 1, 2, \ldots, N - 1$
$\mathbf{Ac_{ij}} = \mathbf{Ab_i}(1 + P_{mutation})$
$P_{mutation} = (-1)^j \cdot j \cdot f_T' \cdot \boldsymbol{Fit}_{\boldsymbol{Ab}_i} \cdot \varepsilon$

$$f_T' = \frac{1}{N'^2}\left[\sum_{y=1}^{a}\sum_{x=b}^{N'-1}(1-r)^{x+1-y-k}\, r^k + \sum_{y=1}^{a} y\,(1-r)^{N'+1-y-k}\, r^k\right]$$

where ε is computed by Eq.(10).
Then compute the fitness of $\mathbf{Ac_{ij}}$ by Eq.(12).
And get $\mathbf{Fit}_{\mathbf{Ac}_i} = \{Fit_{\boldsymbol{Ac}_{i1}}, Fit_{\boldsymbol{Ac}_{i2}}, \ldots, Fit_{\boldsymbol{Ac}_{i1N-1}}\}$.
Step 2.3 Substitution Phase
For each antibody $\mathbf{Ab_i}$, compare $Fit_{\boldsymbol{Ab}_i}$ and $\mathbf{Fit}_{\mathbf{Ac}_{ij}}(j = 1, 2, \ldots, N\text{-}1)$
For $j = 1, 2, \ldots, N$-1

$$Fit_{Ab_i} = \begin{cases} Fit_{Ac_{ij}}, & \text{if } Fit_{Ac_{ij}} < Fit_{Ab_i} \\ Fit_{Ab_i}, & \text{otherwise} \end{cases}$$

$$\mathbf{Ab_i} = \begin{cases} \mathbf{Ac_{ij}}, & \text{if } Fit_{Ac_{ij}} < Fit_{Ab_i} \\ \mathbf{Ab_i}, & \text{otherwise} \end{cases}$$

end while
The final antibody population is $\mathbf{Ab}_{final} = \{\mathbf{Ab_1}, \mathbf{Ab_2}, \ldots, \mathbf{Ab_N}\}$ and $\mathbf{Ab_i}$ with the smallest fitness is the best antibody $\mathbf{Ab_{best}}$. Then $\mathbf{Ab_{best}}$ is used to opitimize the weight.
Step 3 ELM
Calculate the hidden-layer output matrix $\boldsymbol{H}$ with the set of input weights and hidden biases represented by the $\mathbf{Ab_{best}}$.
Compute the output weights matrix $\widehat{\boldsymbol{\beta}} = \boldsymbol{H}^{\dagger}\boldsymbol{T}$.

Algorithm 3: Artificial immune system extreme learning machine.

algorithms. The results are shown in Table 2, and the best results are emphasized in bold font.

Considering the training time, it is obvious that ELM is the fastest one because all the other four algorithms transfer ELM repeatedly. Besides, AIS-ELM's training time is slightly shorter than the other three methods because the times of ELM iteration are smaller than other three algorithms.

Then, focusing on the testing accuracy, it is easy to find that AIS-ELM has the highest mean testing accuracy in all the classification datasets. As for variance, AIS-ELM is the smallest in most instances and is slightly worse than the best one in a few cases. In addition, the good convergence property of clone selection algorithm shows that AIS-ELM outperforms the DSA-ELM, PSO-ELM, SaE-ELM, and ELM.

In addition, we have done the Wilcoxon's signed-rank test [31], and the W-value is 0, which is less than the critical value at p<=0.05. Therefore, the results show that AIS-ELM is significantly different from DS-ELM, PSO-ELM, SaE-ELM, and ELM, which indicates that AIS-ELM outperforms the other four approaches on eight classification datasets.

Besides, we compare the training time between AIS-ELM and BP, SVM, and traditional ELM on three benchmark classification problems. The results are shown in Table 3.

From Table 3, although AIS-ELM is slower than traditional ELM because of iterations, its training speed is significantly faster than that of BP and SVM.

5.2. Regression. In this subsection, the five algorithms are compared on the five regression benchmark problems. The five datasets are Breast Cancer, Parkinson, SinC, Servo, and Yacht Hydro (Yacht), respectively. Detailed description of the five datasets is shown in Table 4.

Attributes of all the datasets have been normalized to [-1, 1] and we focus on the training time and testing accuracy's means and variance. A 20-fold cross validation method is taken to get the average of 20 repeated experiments to minimize the error. The whole dataset is divided into training set, validation set, and testing set without overlap. The results are shown in Table 5.

Table 2: Results of the five algorithms on eight benchmark classification datasets.

Dataset	Algorithm	Training Time (s)	Testing Accuracy (%)		Hidden Nodes
			Means	StDev	
Ecoli	AIS-ELM	2.232	**87.087**	**1.32**	20
	DS-ELM	2.523	86.352	1.43	20
	PSO-ELM	2.257	86.623	1.67	20
	SaE-ELM	2.608	85.985	1.78	20
	ELM	0.003	84.678	2.12	30
Diabetes	AIS-ELM	2.865	**82.771**	0.67	20
	DS-ELM	3.192	80.667	0.65	20
	PSO-ELM	3.993	80.123	**0.53**	20
	SaE-ELM	4.216	81.673	0.76	20
	ELM	0.004	78.984	1.21	30
Epileptic	AIS-ELM	80.379	**83.412**	**0.78**	150
	DS-ELM	82.458	82.345	0.95	150
	PSO-ELM	84.912	81.627	1.12	150
	SaE-ELM	84.233	81.765	1.05	150
	ELM	3.976	80.026	1.21	180
Heart Disease	AIS-ELM	10.923	**80.234**	**1.53**	20
	DS-ELM	11.329	79.637	1.56	20
	PSO-ELM	10.993	78.942	1.73	20
	SaE-ELM	12.265	78.762	1.69	20
	ELM	0.013	76.149	1,96	30
Iris	AIS-ELM	1.0835	**96.921**	**0.35**	20
	DS-ELM	1.255	96.488	0.69	20
	PSO-ELM	1.1315	96.124	0.83	20
	SaE-ELM	1.362	95.642	0.74	20
	ELM	0.001	93.439	1.26	30
Glass	AIS-ELM	2.632	**68.453**	**1.67**	20
	DS-ELM	3.026	65.345	1.89	20
	PSO-ELM	3.067	65.438	1.91	20
	SaE-ELM	3.036	65.087	2.23	20
	ELM	0.003	60.267	2.12	30
Image	AIS-ELM	29.221	**94.55**	0.659	90
	DS-ELM	31.976	93.78	**0.528**	90
	PSO-ELM	32.103	93.23	1.014	90
	SaE-ELM	32.641	92.11	0.832	90
	ELM	0.0493	92.56	0.783	120
Satellite	AIS-ELM	37.424	**88.346**	**0.87**	100
	DS-ELM	39.856	87.265	0.97	100
	PSO-ELM	39.613	86.795	1.08	100
	SaE-ELM	40.238	86.715	0.96	100
	ELM	0.0624	85.028	0.99	150

Table 3: Results of the four algorithms on three benchmark classification datasets.

Algorithm	Satellite Training Times (s)	Image Training Times (s)	Epileptic Seizure Training Times (s)
AIS-ELM	37.424	29.221	80.379
SVM	129.235	103.496	339.648
BP	67.329	58.637	186.247
ELM	0.0624	0.0493	3.976

Table 4: Detailed description of the five benchmark regression datasets.

Dataset	Data			Attributes
	Training	Validation	Testing	
Breast Cancer	98	50	50	32
Parkinson	500	270	270	26
SinC	5000	2500	2500	1
Servo	384	192	192	4
Yacht Hydro	150	79	79	13

Table 5: Results of the five algorithms on the five benchmark regression datasets.

Dataset	Algorithm	Training	Testing Accuracy		Hidden
		Time (s)	Means	StDev	Nodes
Breast Cancer	AIS-ELM	14.898	**2.46E-01**	**1.32E-02**	30
	DS-ELM	15.367	2.53E-01	1.63E-02	30
	PSO-ELM	15.902	2.98E-01	2.35E-02	30
	SaE-ELM	16.342	2.65E-01	1.76E-02	30
	ELM	0.008	2.99E-01	2.07E-02	50
Parkinson	AIS-ELM	20.287	**1.68E-01**	**3.35E-02**	30
	DS-ELM	21.349	1.79E-01	3.67E-02	30
	PSO-ELM	22.547	1.87E-01	4.12E-02	30
	SaE-ELM	22.975	1.89E-01	3.95E-02	30
	ELM	0.011	2.12E-01	4.53E-02	50
Servo	AIS-ELM	14.942	**8.81E-02**	9.83E-03	20
	DS-ELM	15.256	9.41E-02	**9.63E-03**	20
	PSO-ELM	15.278	1.17E-01	1.35E-02	20
	SaE-ELM	15.456	1.07E-01	1.45E-02	20
	ELM	0.007	1.35E-01	1.95E-02	30
Yacht	AIS-ELM	13.478	**1.76E-01**	**4.32E-02**	20
	DS-ELM	13.755	1.87E-01	4.52E-02	20
	PSO-ELM	13.834	2.45E-01	5.63E-02	20
	SaE-ELM	14.292	2.23E-01	5.60E-02	20
	ELM	0.006	2.67E-01	8.43E-02	30
SinC	AIS-ELM	33.012	**6.12E-03**	**3.54E-04**	30
	DS-ELM	33.514	6.35E-03	4.32E-04	30
	PSO-ELM	33.095	7.46E-03	5.41E-04	30
	SaE-ELM	33.821	7.93E-03	5.76E-04	30
	ELM	0.013	8.01E-03	3.84E-04	50

Considering the training time, it is also obvious that traditional ELM is the fastest because of the same reason in 5.1. As for the testing accuracy, AIS-ELM, DS-ELM, PSO-ELM, and SaE-ELM obtain better results with less hidden nodes than ELM, which means that AIS-ELM, DS-ELM, PSO-ELM, and SaE-ELM can achieve better generalization performances with more compact networks. And the RMSE of AIS-ELM is smaller than other four algorithms. Therefore, it can be concluded that AIS-ELM can achieve better performance than other four algorithms on regression problems.

6. Conclusion

In this paper, first we introduce the standard ELM and artificial immune system; then we propose a new approach named artificial immune system extreme learning machine (AIS-ELM). In AIS-ELM, AIS is used to optimize the input weights through clone, mutation, and substitution process.

In the experiment part of this paper, AIS-ELM is compared with DS-ELM, PSO-ELM, SaE-ELM, and the traditional ELM on thirteen well-known benchmark datasets (eight classification datasets and five regression datasets) obtained from UCI Machine Learning Repository. Besides, the training times between AIS-ELM and BP, SVM, and traditional ELM are compared on three benchmark classification problems. Experimental results show that AIS-ELM can achieve better testing results (smaller RMSE on regression and higher accuracy on classification) than other DS-ELM, PSO-ELM, SaE-ELM, and the traditional ELM in most cases, and its training speed is significantly faster than that of BP

and SVM. According to the global search ability [21] and good convergence [22] of AIS, our artificial immune system extreme learning machine is superior to the other methods both on the eight classification datasets and five regression datasets in the experiments. In addition, there are six medical datasets among the thirteen datasets, which can prove that AIS-ELM can also play an excellent role in healthcare. Future research works will be concentered on how to apply the current immune system to some new directions, such as NLP and computer vision.

Data Availability

The classification datasets and regression datasets supporting the findings of this study are from previously reported studies and datasets, which have been cited. The processed data are available at UCI Machine Learning Repository [Online] (http://archive.ics.uci.edu/ml).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work is supported by National Key Research and Development Plan under Grant no. 2016YFB1001203.

References

[1] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, *Support vector machines*, vol. 13, IEEE Intelligent Systems and their applications, 1998.

[2] G. Huang, 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: 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.

[4] G.-B. Huang and C. K. Siew, "Extreme learning machine: RBF network case," in *Proceedings of the 8th International Conference on Control, Automation, Robotics and Vision*, pp. 1029–1036, 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] E. Fahlman S, "An empirical study of learning speed in back-propagation networks," 1988.

[7] I. Chaturvedi, E. Ragusa, P. Gastaldo, R. Zunino, and E. Cambria, "Bayesian network based extreme learning machine for subjectivity detection," *Journal of The Franklin Institute*, 2017.

[8] P. Gastaldo, R. Zunino, E. Cambria, and S. Decherchi, "Combining ELMs with random projections," *IEEE Intelligent Systems*, vol. 28, no. 6, pp. 46–48, 2013.

[9] E. Cambria, P. Gastaldo, F. Bisio, and R. Zunino, "An ELM-based model for affective analogical reasoning," *Neurocomputing*, pp. 443–455, 2015.

[10] E. Ragusa, C. Gianoglio, P. Gastaldo, and R. Zunino, *A Digital Implementation of Extreme Learning Machines for Resource-Constrained Devices*, IEEE Transactions on Circuits and Systems II: Express Briefs, 2018.

[11] M. Y. Rafiq, G. Bugmann, and D. J. Easterbrook, "Neural network design for engineering applications," *Computers & Structures*, vol. 79, no. 17, pp. 1541–1552, 2001.

[12] E. Cambria, G. B. Huang, L. L. C. Kasun et al., *Extreme learning machines [trends & controversies]*, vol. 28, IEEE Intelligent Systems, 2013.

[13] Q. Zhu, A. K. Qin, P. N. Suganthan, and G. Huang, "Evolutionary extreme learning machine," *Pattern Recognition*, vol. 38, no. 10, pp. 1759–1763, 2005.

[14] X. Xue, M. Yao, Z. Wu, and J. Yang, "Genetic ensemble of extreme learning machine," *Neurocomputing*, vol. 129, pp. 175–184, 2014.

[15] S. Saraswathi, S. Sundaram, N. Sundararajan, M. Zimmermann, and M. Nilsen-Hamilton, "ICGA-PSO-ELM approach for accurate multiclass cancer classification resulting in reduced gene sets in which genes encoding secreted proteins are highly represented," *IEEE Transactions on Computational Biology and Bioinformatics*, vol. 8, no. 2, pp. 452–463, 2011.

[16] J. Cao, Z. Lin, and G.-B. Huang, "Self-adaptive evolutionary extreme learning machine," *Neural Processing Letters*, vol. 36, no. 3, pp. 285–305, 2012.

[17] T. Wu, M. Yao, and J. Yang, "Dolphin swarm extreme learning machine," *Cognitive Computation*, vol. 9, no. 2, pp. 275–284, 2017.

[18] J. Timmis and M. Neal, "A resource limited artificial immune system for data analysis," *Knowledge-Based Systems*, vol. 14, no. 3-4, pp. 121–130, 2001.

[19] J. Timmis, A. Hone, T. Stibor, and E. Clark, "Theoretical advances in artificial immune systems," *Theoretical Computer Science*, vol. 403, no. 1, pp. 11–32, 2008.

[20] D. Dasgupta, S. Yu, and F. Nino, "Recent advances in artificial immune systems: models and applications," *Applied Soft Computing*, vol. 11, no. 2, pp. 1574–1587, 2011.

[21] L. N. De Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '02)*, pp. 699–704, May 2002.

[22] E. Clark, A. Hone, and J. Timmis, "A Markov Chain Model of the B-Cell Algorithm," in *Artificial Immune Systems*, vol. 3627 of *Lecture Notes in Computer Science*, pp. 318–330, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[23] H. Tian, S. T. Li, M. Wu, and Yao., "An extreme learning machine based on artificial immune system," in *Proceedings of the The 8th International Conference on Extreme Learning Machines (ELM, 2017)*, Yantai , China, 2017.

[24] D. Dua and E. Karra Taniskidou, *UCI Machine Learning Repository*, University of California, School of Information and Computer Science, 2017.

[25] S. A. Hofmeyr and S. Forrest, "Architecture for an artificial immune system.," *Evolutionary Computation*, vol. 8, no. 4, pp. 443–473, 2000.

[26] J. K. Percus, O. E. Percus, and A. S. Perelson, "Predicting the size of the T-cell receptor and antibody combining region from consideration of efficient self-nonself discrimination," *Proceedings of the National Acadamy of Sciences of the United States of America*, vol. 90, no. 5, pp. 1691–1695, 1993.

[27] J. Kelsey and J. Timmis, "Immune Inspired Somatic Contiguous Hypermutation for Function Optimisation," in *Genetic and Evolutionary Computation — GECCO 2003*, vol. 2723 of *Lecture Notes in Computer Science*, pp. 207–218, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[28] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *American Association for the Advancement of Science: Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[29] A. S. Perelson and G. F. Oster, "Theoretical studies of clonal selection: minimal antibody repertoire size and reliability of self–non-self discrimination," *Journal of Theoretical Biology*, vol. 81, no. 4, pp. 645–670, 1979.

[30] L. N. de Castro and F. J. von Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, 2002.

[31] http://www.socscistatistics.com/tests/signedranks/.