Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 7, No. 1, 2024

Extreme Learning Machine Classification Method based on Twin Strategy Salp Swarm Algorithm

Meiling Shang *, Rongguo Qu, Deqing Ji, Zhenxing Yu, Qinwei Fan

Xi'an Polytechnic University, Xi'an Shaanxi, 710600, China

*** Corresponding author:** Meiling Shang

Abstract: Extreme Learning Machines (ELM) are a type of Single Hidden Layer Feedforward Neural Network (SLFN). In recent years, they have attracted attention for their powerful approximation ability and fast learning speed. Compared with traditional neural network algorithms, ELM have the advantages of simple structure, fast learning speed, and good generalization performance. However, since the input weights and biases of ELM are randomly generated, there may be some suboptimal or unnecessary input weights and biases. In addition, ELM may require more hidden nodes, which may slow down its response to unknown test data. To address these issues, a twin strategy Salp Swarm Algorithm (TSSA) is proposed to increase the population diversity to improve the convergence speed of the algorithm through chaotic initialization, while shock inertia weights and learning paradigms are introduced into the follower position updating to enhance the stochasticity of the particles. Classification tests are conducted on six datasets using the proposed TSSA, and the experimental results show that the proposed TSSA-ELM has higher accuracy and better generalization performance than some existing ELM variants.

Keywords: Extreme Learning Machine; Salp Swarm Algorithm; Multi-Strategy; Inertia Weights; Classification Problems.

1. Introduction

Huang et al. proposed a non-iterative learning algorithm based on feedforward neural networks called Extreme Learning Machine (ELM) in 2004 [1]. Compared to other traditional learning algorithms such as Feedforward Neural Networks (FNN) and Backpropagation (BP), ELM has advantages in terms of good global generalization, high real-time performance, and minimal manual parameter tuning [2-3]. It has been widely applied in various fields, including cloud computing [4], data visualization [5], and random projection [6]. However, due to the random generation of input layer weights and thresholds in ELM, it may result in unnecessary or suboptimal weights and thresholds. Furthermore, when using ELM for prediction, it may require more hidden nodes, leading to slow response to unknown test data. The accuracy of prediction results depends directly on the input weights and thresholds of the hidden layer neurons, and using appropriate input weights and thresholds can effectively improve the accuracy of prediction. Therefore, many researchers have employed various methods to adjust the input layer weights and hidden layer thresholds of ELM.

Zhu et al. [7] proposed a fusion of the Differential Evolution algorithm to optimize the input layer weights of ELM. Ling et al. [8] used the Particle Swarm Optimization algorithm to optimize the weights of the input layer and biases of the hidden layer in ELM. Liu et al. [9] combined the Flower Pollination Algorithm (FA) and Firefly Algorithm (FPA) to optimize the input layer parameters of ELM. They conducted numerical experiments to validate the network's excellent generalization performance even with fewer parameters and a simpler network structure. Derya et al. [10] applied wavelet kernel functions to ELM and proposed a Genetic Algorithm-Wavelet Kernel-Extreme Learning Machine (GA-WK-ELM) method, which was used for Parkinson's disease diagnosis. The aforementioned papers all utilized swarm intelligence algorithms to optimize the input weights and thresholds of ELM, thereby improving the network's generalization ability. However, this also increased the algorithm's complexity and training time.

In 2016, Seyedali Mirjalili proposed the Salp Swarm Algorithm (SSA) [11], inspired by the natural foraging and migration behavior of salp organisms. SSA simulates the predation behavior of salps, considering the ocean as the solution space, and randomly generates a population of salps to search for the optimal solution. Compared to other swarm intelligence algorithms, SSA has advantages such as fast convergence, strong adaptability, and robustness, making it suitable for optimizing the ELM model. Tu et al. [12] incorporated the Harris Hawk Algorithm into SSA to obtain an excellent population and modified the follower's position update formula using the Multiverse Algorithm, allowing individual salps to perform global and local searches in different ranges. The improved algorithm was applied to parameter optimization of ELM weights and thresholds. Although the aforementioned improved algorithms have achieved some degree of enhancement in the search performance of the salp chain, they still fail to fully balance the algorithm's exploration and exploitation capabilities to achieve optimal performance.

Therefore, to better balance the exploration and exploitation capabilities of SSA, this paper proposes the Twin Strategy Salp Swarm Algorithm (TSSA). Firstly, the excellent traversal capability of the Cubic chaotic mapping is utilized by introducing it into the population initialization process, allowing the population to obtain better initial positions. Secondly, a dual-strategy mechanism is employed to update the follower's position information, incorporating oscillating inertia weight and learning patterns to balance local and global search, improve the ability to escape from local optima, and enhance the optimization performance of the algorithm.

The remaining sections of this paper are organized as follows. Section 2 provides an introduction to ELM and SSA. Section 3 elaborates on the process of improving the Salp Swarm Algorithm. Section 4 outlines the model construction process. Section 5 analyzes the experimental results,

demonstrating the effectiveness of the algorithm.

2. Related Work

(1) Extreme Learning Machine

Extreme Learning Machine is a type of single-hidden-layer feedforward neural network. In ELM, the input weights and thresholds are assigned randomly. Unlike traditional gradient descent methods, ELM adopts the framework of least squares to calculate the optimal output weights by solving the corresponding Moore-Penrose pseudo-inverse matrix. As a result, ELM exhibits advantages such as fast convergence and a reduced likelihood of getting trapped in local optima. This article believes that the concept of fintech can be discussed from two aspects: on the one hand, fintech has emerged in product innovation and traditional financial services, ultimately still being finance. On the other hand, financial technology is a technological means derived from the development of modern technology that specifically serves the financial industry. Financial technology has a very powerful technological gene that can empower the financial industry to improve quality and efficiency.

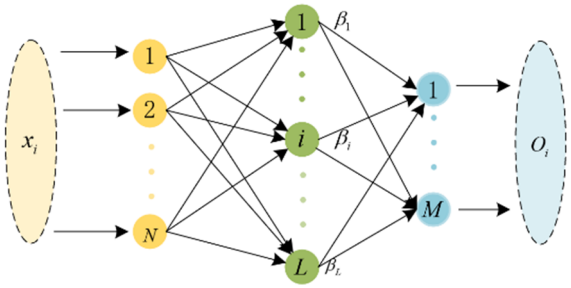


Figure 1. Extreme Learning Machine

The topological structure of the Extreme Learning Machine with N input nodes, L hidden nodes, and M output nodes is illustrated in Figure 1. Let's assume that the activation function for the hidden layer is denoted as G, and the threshold values are represented by B. For a set of P different samples $(\mathbf{x}_i, \mathbf{o}_i) \in \mathbf{R}^n \times \mathbf{R}^m$, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}] \in \mathbf{R}^n$ denotes the n-dimensional input samples and $\mathbf{o}_i = [o_{i1}, o_{i2}, \ldots, o_{im}] \in \mathbf{R}^m$ represents the desired output of the model.

The mathematical model of the ELM can be expressed as:

$$\sum_{i=1}^{n} \boldsymbol{\beta}_i G(\mathbf{w}_i \cdot \mathbf{x}_i + b_i) = \mathbf{t}_j, j = 1, 2, \ldots, m \quad (1)$$

where $\mathbf{w}_i = [w_{1i}, w_{2i}, \ldots, w_{mi}]^T$ denotes the input weights pointing to the first hidden node, $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{im}]^T$ denotes the output weights between the first hidden layer node and the output layer, and $\mathbf{t}_i$ is the actual output of the network.

During the training and learning process, when the learning error drops to 0, which is $\sum_{j=1}^{m} \|\mathbf{t}_j - \mathbf{o}_j\| = 0$, it indicates that the learning ability of the extreme learning machine is optimal at this point, which means that there exists a set of $(\mathbf{w}_i, \boldsymbol{\beta}_i, b_i)$ such that:

$$\mathbf{t}_j = \sum_{i=1}^{n} \boldsymbol{\beta}_i G(\mathbf{w}_i \cdot \mathbf{x}_i + b_i) = \mathbf{o}_j \quad (2)$$

The above P equations can be written in the following matrix form:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{O} \quad (3)$$

Here, $\mathbf{H}$ is called the hidden layer output matrix, and we aim to obtain a set of parameters $(\mathbf{w}^*, b^*, \boldsymbol{\beta}^*)$ such that

$$\left\|\mathbf{H}(\mathbf{w}^*, b^*)\boldsymbol{\beta}^* - \mathbf{O}\right\| = \min_{\mathbf{w}, b, \boldsymbol{\beta}} \left\|\mathbf{H}(\mathbf{w}, b)\boldsymbol{\beta} - \mathbf{O}\right\| \quad (4)$$

When the number of nodes in the hidden layer is equal to the number of samples in the training set, $\mathbf{H}$ is an invertible square matrix, and the network can approximate the training samples with zero error, but in practical applications, the number of nodes in the hidden layer is usually smaller than the number of training samples. Therefore, when the activation function is infinitely differentiable, the output weights can be obtained by solving the least squares solution of the linear system $\mathbf{H}\boldsymbol{\beta} = \mathbf{O}$ with the explicit solution:

$$\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{O} \quad (5)$$

where is the generalized inverse of Moore-Penrose.

(2) The classic salp swarm algorithm

The Salp Swarm Algorithm (SSA) is designed based on the biological characteristics of salp swarms. In SSA, the first half of the salps are leaders, while the rest are followers. Unlike other swarm intelligence algorithms, leaders do not influence the movement of the entire swarm, and the positions of the followers are updated based on the position of the preceding individual. The leaders actively search for food sources (i.e., optimal solutions) in an n-dimensional space. The movement trajectory is determined by the following equation:

$$x_j^1 = \begin{cases} F_j + c_1\left(c_2\left(ub_j - lb_j\right) + lb_j\right), c_3 \geq 0.5 \\ F_j - c_1\left(c_2\left(ub_j - lb_j\right) + lb_j\right), c_3 < 0.5 \end{cases} \quad (6)$$

where x_j^1 denotes the position of the leader in the j th dimensional space and F_j denotes the position of the food source in the same space. In addition, ub_j and lb_j denote the upper and lower bounds of the dimensional space, respectively. The variables c_2 and c_3 are random numbers uniformly distributed in the range of $[0,1]$. The parameter c_1 in the SSA algorithm is used as a coefficient to balance the exploration and exploitation, which is given by:

$$c_1 = 2e^{-\left(\frac{4l}{L}\right)^m} \quad (7)$$

where l denotes the current number of iterations and L is the maximum number of iterations of the algorithm. The position of the follower is updated according to Newton's law of motion, which can be expressed as:

$$x_j^i = \frac{1}{2}\left(x_j^i + x_j^{i-1}\right) \quad (8)$$

3. Extreme Learning Machine Model with Twin Strategy Salp Swarm Algorithm

(1) Twin strategy salp swarm algorithm

In the search space of an evolutionary algorithm, the initial population setting plays a crucial role in determining the starting point and search direction. Different initialization methods may lead to different search results for the algorithm and may even affect its ability to discover globally optimal solutions. Inappropriate population initialization may cause the algorithm to converge prematurely, hindering its ability to explore better solutions. Conversely, overly random initialization can lead to a large number of low-quality individuals in the population. This will inevitably reduce the search efficiency of the algorithm and require more time to reach the optimal solution. In order to improve the training effect and generalization ability of the model, this paper uses a chaotic approach to initialize the population.

Chaotic random numbers are highly sensitive and nonlinear, so much so that slight perturbations in the initial conditions or parameters may lead to significantly different trajectories of the system, exhibiting robust statistical randomness properties. Specifically, we use Cubic chaotic mapping instead of random initialization, which is defined by the following formula.

$$ch_{n+1} = \rho ch_n \left(1 - ch_n^2\right) \tag{9}$$

$$x_{i,j} = x_{\min,j} + ch_{i,j}\left(x_{\max,j} - x_{\min,j}\right) \tag{10}$$

where the Cubic chaotic mapping has good chaotic traversal properties when $ch_0 = 0.3$ and $\rho = 2.595$.

The position update of a follower in classical SSA is related to the position information of its previous individual, but it is a purely blind follower behavior, which updates in a single way and limits the search scope. In order to quickly traverse the search space for global search this paper adopts a twin strategy follower update mechanism.

Strategy 1: Introduce the oscillatory inertia weight w to update the follower position information, the oscillatory inertia weight performs global exploration in the pre-evolutionary stage, and the movement decreases in the late evolutionary stage, focusing on the local exploitation to be able to excavate the optimal solution more accurately, so as to be able to balance the exploitation and exploration line ability during the global search. At this time, the follower's position update formula is as follows:

$$w = \left(rand - 0.5\right)\left(2 - \tan(\frac{l}{L})\right) \tag{11}$$

$$x_j^i = \frac{1}{2}\left(x_j^i + w x_j^{i-1}\right) \tag{12}$$

where $rand$ is a uniformly distributed random number between $(0,1)$.

Strategy 2: According to the fitness information of the population in the solution space, sort all the individuals of the population, and get the sequence information $rank(i)$ of each individual, according to the sorting result, we select the individual with better fitness information as the learned example, in order to avoid arbitrary selection of the learned example, we use probabilistic way to select. When $rank(i) < rand \bullet P$ is satisfied, $rand$ is a random number between $(0,1)$, which is put into the set Q of learned examples. This means that when the individual's fitness information is better, the individual's ranking is more advanced, and then there is a greater possibility to satisfy the probabilistic selection formula we set. The follower's position update formula is as follows:

$$x_j^i = \frac{1}{2}\left(x_j^i + q_j^i\right) \tag{13}$$

where q_j^i is the j th dimension of the i th randomly selected individual in the set Q of learning examples.

In order to ensure that the population evolves in a better direction, we adopt a pairwise comparison mechanism to select better adapted individuals as the offspring of individuals in the current population.

Algorithm optimization step

(2) Algorithm optimization step

The Twin Strategy Salp Swarm Algorithm is illustrated in Figure 2.

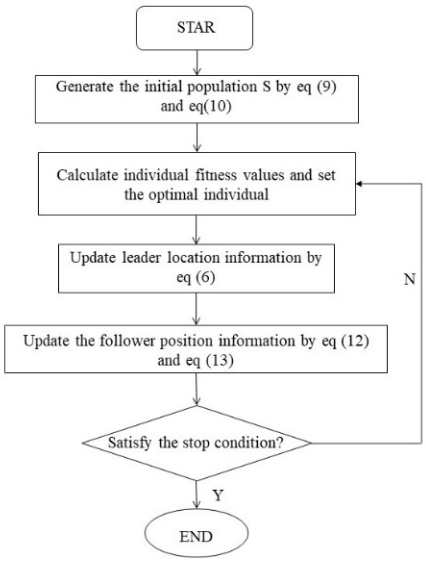


Figure 2. The flowchart of the Twin Strategy Salp Swarm Algorithm.

4. The Extreme Learning Machine Model based on the Twin Strategy Salp Swarm Algorithm

Since ELM randomly generates the weight matrices connecting the input and hidden layers as well as the thresholds for the hidden layers, inappropriate connection weights and thresholds may lead to invalid nodes, node redundancy, and insufficient generalization ability in some hidden layers. Therefore, in this paper, the TSSA algorithm is used to optimize the connection weights and biases of ELM. The specific steps are as follows:

Step 1: Construct the ELM model, define the TSSA algorithm and ELM model related parameters, and set the number of neurons in the hidden layer of the ELM. Initialize P initial solutions, the generated initial solution dimensions are $N*L+L$. The former $N*L$ dimensions denote the input layer weight matrix, and the remaining L dimensions denote the hidden layer threshold.

Step 2: Input the population P and the training set into the

ELM model to obtain the weights β and of the output layer and use the network test accuracy as the individual fitness value, and set the optimal individual as the best food source location for the current population.

Step 3: Update the leader location information according to equation (6) and update the follower location information according to equation (12) and equation (13) and the dual policy mechanism.

Step 4: Calculate the current population fitness value, check the feasibility of the new position, and obtain the current iteration optimal solution.

Step 5: Determine whether the algorithm reaches the maximum number of iterations, if it meets, use the obtained individual optimal solution as the input weights and thresholds of the ELM model to train the ELM model; otherwise, return to Step 2.

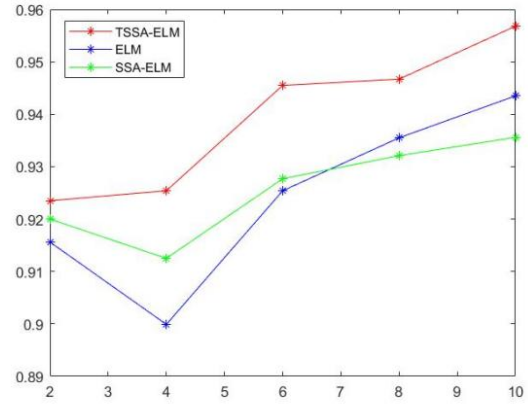
5. Numerical Experiment

In this section, we use the TSSA algorithm to optimize the input layer weights of ELM for classification purposes. We perform classification experiments of the proposed new algorithm on four real datasets and compare it with ELM, SSA-ELM, algorithms. Using the cross-validation method, 70% of each dataset is selected as the training set and the rest as the test set each time to obtain the average results of each algorithm under 10 simulation experiments, and then their performance is analyzed and compared in detail. The same network structure is used for testing in each experiment, the number of hidden layer neurons is fixed to 3, the population size P is set to 50, and the maximum number of iterations is 100.The datasets for the classification experiments are all from the UCI machine learning repository, and the corresponding experimental results are shown in Table 1.

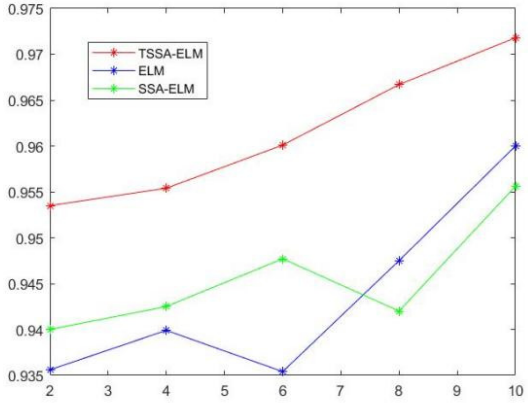
Table 1. Classification test results

Data Sat	Accuracy Rate	TSSA-ELM	SSA-ELM	ELM
Seeds	Training	0.9419	**0.9516**	0.9410
	Testing	**0.9330**	0.9232	0.9200
Tae	Training	**1.0000**	0.9823	0.9647
	Testing	**0.9562**	0.9485	0.9300
Iris	Training	**1.0000**	**1.0000**	**1.0000**
	Testing	**0.9824**	0.9710	0.9520
Cleveland	Training	**0.9272**	0.9141	0.8992
	Testing	**0.9088**	0.8605	0.8048

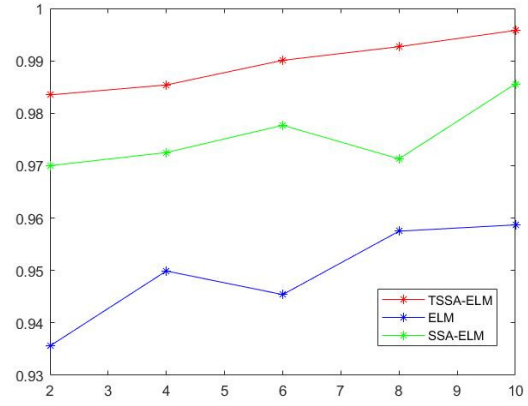
As can be seen from the experimental results in Table 1, the TSSA algorithm proposed in this paper consistently gives the best test results for the four high or low feature classification datasets. It is worth noting that TSSA-ELM improves the training test accuracy by 2.9725% compared with ELM and 1.2288% compared with SSA-ELM. The above results can indicate that TSSA has better generalization performance and stability, and can effectively train ELM to deal with classification problems.



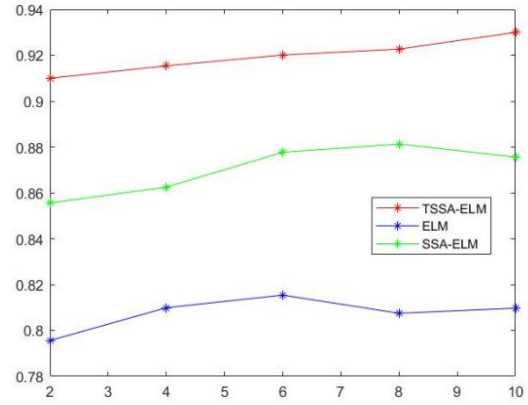
(a) Seeds



(b) Tae



(c) Iris



(d) Cleveland

Figure 3. Classification accuracy of different datasets with different hidden layer neuron nodes

Figure 3 gives the classification accuracy of different datasets under different numbers of hidden layer neurons, and the experimental results show that with the increase of the number of hidden layer neuron nodes the classification accuracy is also on the rise, but the rise reaches a certain accuracy and then will remain stable. Compared with its comparison algorithm, the change of the hidden layer nodes has less influence on SSA-ELM, and its experimental results remain stable, which indicates that SSA-ELM has better stability.

6. Conclusion

In order to mitigate the impact of the randomness in the input weights and thresholds of the Extreme Learning Machine (ELM) model, this paper proposes a Twin Strategy Salp Swarm Algorithm (TSSA) to optimize the input weights and thresholds of ELM. Additionally, a chaotic initialization method is introduced to generate more diverse initial solutions in the early stages of the algorithm. Through experimental validation, the proposed algorithm effectively balances the exploration and exploitation capabilities, achieving optimization of the weights and thresholds in ELM. Furthermore, classification tests on UCI machine learning repository its good generalization performance and stability.

References

[1] Huang G, Zhu Q, Siew C. Extreme learning machine: a new learning scheme of feedforward neural networks[J]. Proceedings International Joint Conference Neural Networks, 2004, 2: 985-990.

[2] Zhao Y, Huerta R. Improvements on parsimonious extreme learning machine using recursive orthogonal least squares[J]. Neurocomputing, 2016, 191: 82-94.

[3] Han M, Yang X, Jiang E. An extreme learning machine based on Celluar Automata of edge detection for remote sensing images [J]. Neurocomputing, 2016, 198: 27-34.

[4] Jeddi S, Sharifian S. A hybrid wavelet decomposer and GMDH-ELM ensemble model for Network function virtualization workload forecasting in cloud computing[J]. Applied Soft Computing, 2019, 105940.

[5] Altay O, Ulas M, Alyamac K. DCS-ELM: a novel method for extreme learning machine for regression problems and a new approach for the SFRSCC[J]. PeerJ Comput Science, 2021, 7.

[6] Fan J, Sun H, Su Y, Huang. Muspel-Fi: Multipath subspace projection and Elm-based fingerprint localization[J]. IEEE Signal Processing Letters, 2022, 29: 329-333.

[7] Zhu Q, Qin A, Suganthan P. Evolutionary extreme learning machine[J]. Pattern Recognit,2005, 38, 1759-1763.

[8] Ling Q, Han F. Han, Yao H. An improved evolutionary extreme learning machine based on particle swarm optimization [J]. Neurocomputing, 2013, 116: 87-93.

[9] Liu T, Fan Q, Kang Q, Niu L. Extreme learning machine based on Firefly adaptive flower pollination algorithm optimization [J]. Processes, 2020, 8(12): 1583.

[10] Derya A, Akif D. An Expert Diagnosis System for Parkinson Disease Based on Genetic Algorithm-Wavelet Kernel-Extreme Learning Machine[J]. Parkinson's Disease, 2016, 1-9.

[11] Mirjalili S, Gandomi A H, Mirjalili S Z, et al. Salp swarm algorithm: a bio-inspired optimizer for engineering design problems[J]. Advances in Engineering Software, 2017, 114: 163-191.

[12] Tu Q, Liu X, Xie Y, Han G. Range-Free Localization Using Extreme Learning Machine and Ring-Shaped Salp Swarm Algorithm in Anisotropic Networks[J]. IEEE Internet of Things Journal, 2022,10(9): 8228.