Original Research Article

High-order neural network based on a hybrid firefly flower pollination algorithm

Rongguo Qu, Yunlong Liu, Qingmei Dong, Jing Zhao, Manyuan Li, Qinwei Fan[Corresponding author].*

College of Science, Xi'an Polytechnic University, Xi'an, Shaanxi, 710600,China

Abstracts: Pi Sigma neural network is a kind of high-order feedforward neural network, which is characterized by fast convergence speed and strong nonlinear mapping ability. However, for the growing large dataset, the traditional Pi Sigma neural network suffers from the problems of complex network structure, difficulty in determining weights, and low learning efficiency. Therefore, this paper proposes a hybrid heuristic algorithm that combines the flower pollination algorithm with the firefly algorithm to optimize the weights and biases of the Pi Sigma neural network. The experimental results show that the optimized neural network has good performance in many aspects.

Keywords: High order neural network, Firefly algorithm, Flower pollination algorithm.

1. Introduction

In recent years, more and more researchers are using artificial intelligence to solve real-world problems due to the swift advancement of artificial intelligence. Neural networks, as an important part of AI, are also widely used by researchers[1-3]. So far, various neural network models have been proposed, among which the feedforward neural network is one of the most widely used. Early feedforward neural networks contain only summation neurons, which are less efficient in dealing with complex nonlinear problems. To improve the learning efficiency and nonlinear mapping ability of the network, the summation neurons were introduced into the

feedforward neural network, which resulted in the formation of high-order feedforward neural networks (HONNs) [4,5]. However, simply constructing the product neuron by the simple product of the input node values leads to an exponential increase in the number of weights, which is known as the "dimensionality catastrophe". To efficiently avoid the exponential growth of weight vectors and processing units, Shin Y. designed a higher-order neural network called the Pi Sigma neural network[6]. It is widely used in solving different problems such as classification, function approximation, and prediction[7-9].

For the performance of the neural network to be fully demonstrated, it is crucial to choose an appropriate training method. Traditional network training methods usually use local search methods, such as gradient descent, but this method has some problems. First, gradient descent is prone to fall into local optimal solutions, i.e., the network may converge to the local optimal point during the training process without being able to find the global optimal solution. This is because gradient descent can only update the current weights according to their gradient direction, which may result in missing better weight combinations. Second, gradient vanishing is another common problem, especially in deep networks. As the gradient decreases to near zero during backpropagation, the update of the weights also becomes very small, causing network learning to become slow or stagnant. In contrast, intelligent optimization algorithms have a global search capability and can search in the entire search space, thus making it more likely to find the global optimal solution. Such algorithms not only rely on the gradient information of the current weights but are also able to explore the entire weight space to find better weight combinations through different search strategies and optimization operators, thus improving the training effect and generalization performance of neural networks[10,11]. The intelligent optimization algorithm, also known as a modern heuristic algorithm, is a stochastic search and optimization algorithm that developed rapidly in the late 20th century. It breaks the dependence of traditional optimization algorithms on the mathematical characteristics of the optimal solution and can directly solve the approximate solution to the optimization problem, which is why it has been widely used.

Two famous population-based intelligence optimization algorithms, the flower pollination algorithm[12] and the firefly algorithm[13], were introduced by Yang. Due to their uncomplicated structure, independence from gradient information, and minimal required parameters, these two algorithms have garnered more attention among researchers. They have been extensively adopted in various fields such as structural optimization of neural networks, economic allocation, and production scheduling[14-16]. Although the flower pollination algorithm has many advantages, it has disadvantages like other intelligent optimization algorithms. For example, it tends to fall into local optima and converges slowly at later stages. To reduce these drawbacks, many researchers have tried to improve the algorithm.

Guo Z.[17] proposed a pollination algorithm (LNFPA) based on logical chaotic mapping and natural mutation, which utilizes logical chaotic mapping to generate the initial population, then uses a crossover operation to perform a local search and performs a natural mutation operation after each iteration to improve the search effect. Compared with FPA, LNFPA has a more stable performance, faster iteration speed, and higher accuracy. Nevertheless, the improvement of this algorithm ignores the balance between the local and global search of the algorithm. Cai C.[18] proposed a novel Improved Flower Pollen Algorithm (IFPA), which is based on the basic pollen algorithm with the addition of Cauchy mutation, Elite's strategy, and the dynamic transition probability, and used the improved algorithm for the synthesis of antenna arrays, but neglected the algorithm's population initial permutation of the antenna array. Although several scholars have optimized the basic pollen pollination algorithm, the problem of slow convergence of the pollen pollination algorithm in the later iterations still needs to be further solved.

To address the above problems, this paper proposes a hybrid firefly flower pollination algorithm (FA-FPA) Subsequently, we used the algorithm for the training of Pi Sigma neural networks to make the Pi Sigma networks have better generalization performance, which can make them better used in various fields. In summary, the main contributions of this paper are as follows:

This paper proposes an intelligent optimization algorithm known as a hybrid firefly flower pollination algorithm. The main idea of the algorithm is to use SPM chaotic mapping to initialize the population, so that the pollen algorithm converges to the global optimal solution faster based on obtaining a better initialized population, and then uses the dynamic conversion probability to control the balance between exploitation and exploration to avoid the algorithm from converging to the local optimal solution prematurely, moreover, to accelerate the algorithm's searching efficiency, it is incorporated into the searching method of the firefly algorithm in the global searching stage.

To demonstrate the effectiveness of the improved algorithm for Pi Sigma neural network training, this paper demonstrates the effectiveness of the proposed hybrid firefly flower pollination Pi Sigma neural network in dealing with classification through numerical experiments.

The remainder of this manuscript is structured in the following manner: we give a relevant description and introduction of the Pi Sigma neural network in the second section. In the third section, we elaborate and explain the hybrid firefly flower pollination algorithm. Section 4 describes how to train the Pi Sigma neural network using the hybrid firefly pollination algorithm. Section 5 conducts experiments on classification problems using Pi Sigma neural networks based on the hybrid firefly pollination algorithm. The sixth section is the conclusion and outlook.

2. Preparation

2.1. Pi Sigma Neural Network

The structure of the Pi Sigma neural network is bifurcated into three layers: an input layer, a hidden layer, and an output layer. The input layer employs a conventional summation layout, with its output utilized as the input for the summation unit of the hidden layer. The hidden layer's output is then propagated to the output layer. As the weights linking the hidden and output layers remain constant at 1, the output of the output layer is determined by multiplying the output units of all hidden layers. Consequently, the Pi Sigma neural network surpasses other networks in terms of efficiency and accuracy, as it employs fewer weight units to process a larger number of nodes. Moreover, the Pi Sigma neural network is adept at tackling intricate tasks that traditional feedforward neural networks struggle with, including zeroing polynomials and polynomial decomposition. **Figure 1** illustrates the architecture of a Pi Sigma neural network with n inputs and one output.

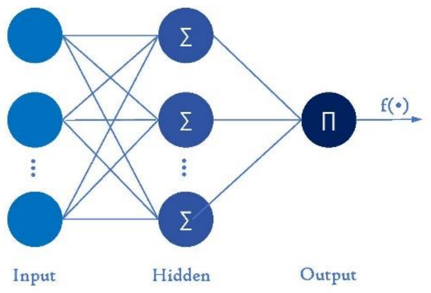


Figure 1. The architecture of the Pi Sigma neural network

It is supposed that the input $X = (X_1, X_2, \cdots, X_i, \cdots, X_n)^T$ is an n-dimensional input vector, where X_i denotes the ith component of X. The $k \times n$-dimensional power vector makes $W_{ji} = \left(W_{ji_1}, W_{ji_2}, \dots, W_{ji_n}\right)^T$, $j = 1,2,\cdots,k$ attributes to a layer of k summation units, where k is the corresponding order of the network and B_j is the bias unit. The output at the hidden layer h in **Figure 1** can be calculated by the following equation.

$$h_j = B_j + \sum_{i=1}^{n} W_{ji} X_i \tag{1}$$

where W_{ji} denotes the weight from the input to the summation unit, since the weights from the implicit layer to the output layer are fixed at 1, the output O can be calculated by the following equation.

$$O = f\left(\prod_{j=1}^{k} h_j\right) \tag{2}$$

where $f(\cdot)$ is a suitable activation function.

2.1.1. Flower Pollination Algorithm

Self-pollination and cross-pollination represent the two main methods of pollination. Typically, self-pollination occurs when there is no reliable pollinator, allowing a plant's pollen to fertilize its own flowers. Heterogeneous pollination is when pollen from different plants can be pollinated. The behavior of pollinators during hetero flower pollination will follow a Levy distribution. The current investigation operates under the assumption that each plant possesses only one flower and one pollen gamete. Additionally, each pollen gamete maintains a one-to-one connection with a solution, with the pollination process adhering to a strict set of guidelines.

a) Heterogeneous pollination is the process of global pollination by $Lévy$ flight by transmitters carrying pollen gametes;

b) The act of abiotic self-pollination can be regarded as a type of local pollination, whereby a plant employs its own pollen to fertilize its own flowers without assistance from an external pollinator;

c) The degree of similarity between two flowers determines the likelihood of successful reproduction, as stipulated by the concept of flower constancy;

d) The degree to which cross-pollination is converted into self-pollination is determined by the conversion probability $p\epsilon[0,1]$. This is because various factors, including but not limited to distance, skew the pollination process towards self-pollination.

During the early stages of flower pollination algorithm, a population $P(t)=\{X_t^i\}$, $X_t^i=\left[x_{i,1}^t,x_{i,2}^t,\cdots,x_{i,j}^t,\cdots,x_{i,D}^t\right]$, $j=1,2,\cdots,D;\ i=1,2,\cdots,N$,containin N individuals is randomly generated, where D is the dimensionality of the optimization problem, t is the number of current iterations, and the population size N represents the diversity of solutions and the number of individuals iterating each time.

The process of self-pollination, i.e., local pollination, is as follows:

$$X_i^{t+1}=X_j^t+\varepsilon\left(X_j^t-X_i^t\right) \tag{3}$$

Here, the variables X_j^t and X_i^t denote pollen produced by two different flowers of the same plant class, and are represented by randomly chosen integers $i,j\in[1,N]$. Additionally, the variance factor ε is a random number selected uniformly from the interval [0,1].

In the context of this study, heterogeneous pollination is a global pollination process in which pollinators utilize $Lévy$ flights to facilitate pollination. The process can be summarized as follows:

$$X_i^{t+1}=X_i^t+\gamma L\left(X_i^t-g^*\right) \tag{4}$$

Here, X_i^t denotes the location of the i-th pollen particle at iteration t. The γ parameter governs the particle's step size, while g^* represents the current population's best solution. Additionally, the intensity of pollination is determined by the parameter L, which specifies the step size according to the $Lévy$ distribution.

2.1.2. Firefly Algorithm

The FA algorithm transforms the optimization problem into a search for the brightest firefly within the population and employs iterative position updates and mutual attraction between fireflies to achieve this goal. In order to simplify the algorithm, certain ideal assumptions are usually made, such as:

a) All fireflies do not distinguish between male and female;

b) The attraction between fireflies is only related to the luminous brightness and distance c) of the fireflies;

The brightness of the fireflies is determined by the objective function.

The attraction between fireflies in the FA algorithm is based on two factors: brightness and mutual attraction. The brighter the firefly is, the better its position within the solution space and the more attractive it becomes to other fireflies. The brightest firefly, which represents the global optima of the objective function, has the strongest attraction force and tends to attract other weaker fireflies toward it. If two or more fireflies have the same level of brightness, they will move randomly.

When computing the brightness at an individual j in the FA algorithm, its proximity to other individuals, such as individual i located at a distance of r_{ij}, is considered.

$$I_{ij} = I_0 \times e^{-\lambda r_{ij}} \tag{5}$$

where I_0 denotes the maximum fluorescence brightness of the firefly, i.e., the fluorescence brightness of the individual i itself; λ is defined as the coefficient that describes the rate at which the light intensity is absorbed, whose data has a crucial role in the speed of convergence and performance of the FA algorithm itself, theoretically $\lambda \epsilon [0, \infty)$, but in practice, due to the characteristic scale of the optimization problem, as long as $\lambda = 0$ is satisfied, so in most applications, the value of λ is usually taken between 0.1 and 10, in this paper's numerical experiments, $\lambda = 1.0$. r_{ij} refers to the Euclidean distance between fireflies i and j in the solution space. It can be expressed as follows:

$$r_{ij} = |x_j - x_i| = \sqrt{\sum_{k=1}^{d} \left(x_j^k - x_i^k\right)^2} \tag{6}$$

The attraction force between individual i and individual j is proportional to a function of their relative brightness and distance, determining the degree of their attraction.

$$\beta_{ij} = \beta_0 \times e^{-\lambda r_{ij}^2} \tag{7}$$

Here r_{ij} and λ are the same as before; β_0 denotes the maximum attraction of the firefly (i.e., the attraction at $r = 0$).

The updated equation for the position of firefly j that is attracted to another firefly i and moves is

$$X_j^{t+1} = X_j^t + \beta_0 e^{-\lambda r_{ij}^2}\left(X_i^t - X_j^t\right) + \alpha\mu_j \tag{8}$$

By using a step factor α and a random vector μ_j, whose components obey either Gaussian or uniform distributions, the FA algorithm continually adjusts the positions of fireflies based on their attractiveness and brightness, resulting in fireflies gravitating towards those with higher fitness values and ultimately achieving optimization.

3. Proposed methods

3.1. A Hybrid Firefly Flower Pollination Optimization Algorithm

3.1.1. Initialization of population based on SPM chaotic mapping

Population initialization is a crucial step in evolutionary algorithms and optimization algorithms, and its quality directly affects the convergence speed of the algorithm and the quality of the results. The traditional population initialization method may lead to high similarity and lack of diversity of individuals in the population, which limits the exploration ability and global search ability of the algorithm. In contrast, the use of SPM chaotic mapping[19] for population initialization has a more uniform distribution, which can increase the diversity of the population, improve the coverage of the search space, and enhance the exploration ability of the algorithm. This approach provides better starting points, accelerates the convergence process of the algorithm, and helps to discover high-quality solutions, which improves the performance and effectiveness of the algorithm. The specific formulation of SPM chaotic mapping is as follows:

$$x(t+1)=\begin{cases}\mathrm{mod}(\frac{x(t)}{\omega}+\mu\sin(\pi x(t))+r,1),\ 0\le x(t)<\omega\\ \mathrm{mod}(\frac{x(t)/\omega}{0.5-\omega}+\mu\sin(\pi x(t))+r,1),\ \omega\le x(t)<0.5\\ \mathrm{mod}(\frac{(1-x(t))/\omega}{0.5-\omega}+\mu\sin(\pi(1-x(t)))+r,1),\ 0.5\le x(t)<1-\omega\\ \mathrm{mod}(\frac{1-x(t)}{\omega}+\mu\sin(\pi(1-x(t)))+r,1),\ 1-\omega\le x(t)<1\end{cases} \quad (9)$$

where, $\omega \in (0,1), \mu \in (0,1)$ when the system is in a chaotic state and r is a random number between 0 and 1.

a) Adaptive transformation probability

Based on the above introduction of the flower pollination algorithm, it is easy to know that the transformation probability controls the search method of the flower pollination algorithm. However, the transformation probability of the original flower pollination algorithm is a fixed value, which may cause the search process of the algorithm to be too conservative or too random, affecting the performance and effectiveness of the algorithm. To further improve the flower pollination algorithm, an adaptive mechanism can be introduced to dynamically adjust the transformation probability. The specific formula is as follows:

$$p = 0.8 + 0.2 \times r_1 \times \frac{t}{T} \quad (10)$$

Where $r_1 \in (0,1)$, t is the current iteration, T is the total iteration.

b) Flower pollination algorithm with hybrid firefly algorithm

To improve the global search capability of the flower pollination algorithm, this paper introduces the firefly algorithm in the global search of flower pollination, which provides more search directions for the algorithm during the global search. The flowchart of the "Flower pollination algorithm with hybrid firefly algorithm" is shown below:

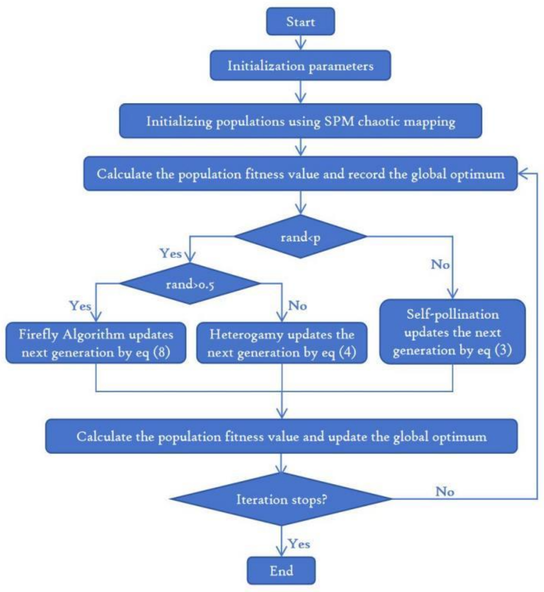

Start
Initialization parameters
Initializing populations using SPM chaotic mapping
Calculate the population fitness value and record the global optimum
rand<p
Yes
No
rand>0.5
Yes
No
Firefly Algorithm updates next generation by eq (8)
Heterogamy updates the next generation by eq (4)
Self-pollination updates the next generation by eq (3)
Calculate the population fitness value and update the global optimum
Iteration stops?
No
Yes
End

Figure 2. The flowchart of the Flower pollination algorithm with hybrid firefly algorithm.

3.1.2. Pi Sigma Neural Network based on Hybrid Firefly Flower Pollination Optimization Algorithm

In this chapter, we adopt the FA-FPA algorithm to calculate the optimal weights and deviations of PSNN, to improve the classification accuracy of PSNN. We start by applying FA-FPA to achieve the optimal combination of weights and deviations and use them directly as input weights and deviations for PSNN training. To reduce the impact of large differences between variables on the algorithm performance, we normalize the data using equation (11):

$$X = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{11}$$

Here, X_{max}, X_{min} represent the maximum and minimum samples in the training phase, respectively.

Naturally, in encoding the input variables of the PSNN, we indicate the input weights and deviations as individual pollens using real number encoding. Chapter 2 specifies that the input layer has n neurons, and the hidden layer has k neurons The length of the string for each pollen is therefore determined accordingly, as follows:

$$\rho = (n + 1) \times k \tag{12}$$

The specific stages of the FA-FPA-PSNN method are as follows:

a) Initialization parameters, setting the maximum number of iterations or search precision as the loop termination condition;

b) Normalize the input according to equation (11);

c) Initialize the population position using equation (9) and calculate the objective function value according to the FFT algorithm;

d) Randomly generate and calculate the conversion probability according to equation (10);

e) Randomly generate $rand \epsilon [0,1]$, if $rand \geq p$, then search by equation (3), else randomly generate $rand \epsilon [0,1]$, if $rand > 0.5$ then search by equation (8), else search by equation (4);

f) Calculate the fitness value of each pollen according to the FFT algorithm and find the current optimal solution;

g) Determine whether the loop termination condition is satisfied. If not, go to step d); if satisfactory, go to step h);

h) Output the result and the algorithm ends.

FFT Algorithm

A. Setting the loop termination condition;

B. Calculating the output of the hidden layer of the PSNN using an equation (1);

C. Calculating the output of the output layer of the PSNN using equation (2);

D. Calculating the fitness value based on classification accuracy;

E. Determine whether the loop termination condition is satisfied. If not, go to step B.; if satisfied, the algorithm ends.

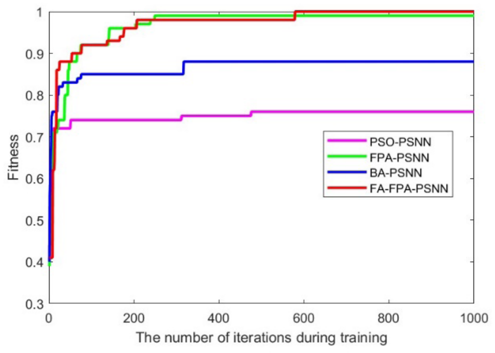
4. Numerical experiments

To validate the effectiveness of the FA-FPA-PSNN algorithm for classification, we have experimented with the algorithm using several sample datasets and compared the classification performance of FA-FPA-PSNN, PSO-PSNN, BA-PSNN, and FPA-PSNN on these samples. **Table 1** describes the sample attributes, the number of samples and the data types of the selected datasets.

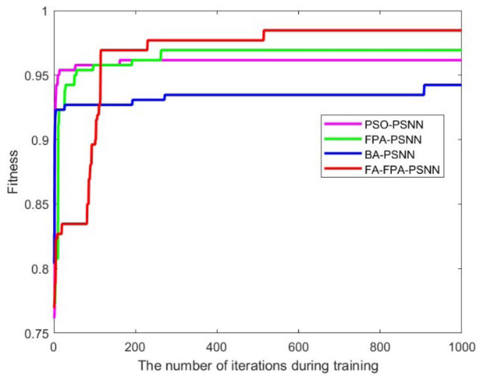
Table 1. Details of the classification experiment dataset.

Sample Name	Attributes	Sample Number	Category
Iris	4	150	3
Jain	2	372	2
Cleve	13	296	2
Cancer	9	683	2
Thyroid	5	215	3

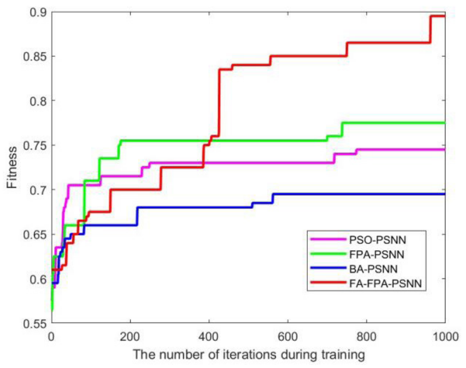
Table 2 shows the accuracy of the classification of FA-FPA-PSNN on different data sets. **Figure 3** shows the variation curves of the fitness values of FA-FPA-PSNN and other comparative algorithms on different data sets.



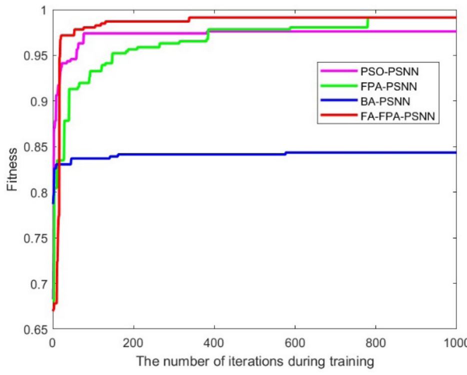
(a) Iris



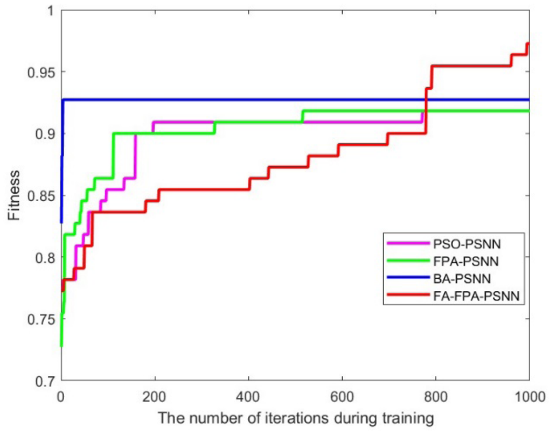
(b) Jain



(c) Cleve



(d) Cancer



(e) Thyroid

Figure 3. the variation curves of the fitness values of different algorithms on different data sets.

In this article, we set the initial population size to 20 and the number of population iterations to 1000 for initializing the proposed Pi Sigma neural network based on the FA-FPA algorithm. Also, we divided each dataset into a training set and a test set in a ratio of 2:1. In this experiment, the activation function chosen for the Pi Sigma neural network is the Sigmoid function. Our fitness function for classification is the correct rate of classification.

In addition, please note that the experiments in this paper were conducted using MATLAB version 2018a. According to the graphs of fitness values during training iterations of the datasets, we know that the FA-FPA-PSNN algorithm is the first to reach the maximum fitness value on the training sets of all datasets, which means that the FA-FPA-PSNN algorithm is more efficient in the merit-seeking ability compared to the FPA-PSNN, PSO-PSNN, and BA-PSNN algorithms are more powerful.

Alternatively, through **Table 2**, we can find that the FA-FPA-PSNN algorithm has higher test accuracy on all datasets than the FPA-PSNN, PSO-PSNN, and BA-PSNN algorithms, which shows the strong generalization ability and classification performance of FA-FPA-PSNN algorithm. In summary, after the classification experiments on the data, the experimental results demonstrate that the classification accuracy attained by the PSNN based on the firefly flower pollination algorithm surpasses that of the PSNN based on the BA, the PSNN based on the PSO, and the PSNN based on the FPA.

Table 2. The accuracy of the classification of different algorithm on different data sets.

Data Set	Algorithm	Training Accuracy	Testing Accuracy
Iris	BA-PSNN	0.7900	0.7600
	PSO-PSNN	0.6800	0.6400
	FPA-PSNN	0.9600	0.9600
	FA-FPA-PSNN	**0.9700**	**0.9800**
Jain	BA-PSNN	0.9078	0.8761
	PSO-PSNN	0.9192	0.9558
	FPA-PSNN	**0.9423**	0.9558
	FA-FPA-PSNN	**0.9423**	**0.9823**
Celve	BA-PSNN	0.6000	0.6563
	PSO-PSNN	0.6750	0.6979
	FPA-PSNN	0.7200	0.7292
	FA-FPA-PSNN	**0.8150**	**0.8333**
Cancer	BA-PSNN	0.8087	0.7982
	PSO-PSNN	0.9587	0.9596
	FPA-PSNN	0.9739	0.9686
	FA-FPA-PSNN	**0.9826**	**0.9741**
Thyroid	BA-PSNN	0.8182	0.8192
	PSO-PSNN	0.8182	0.8667
	FPA-PSNN	0.8364	0.8476
	FA-FPA-PSNN	**0.9091**	**0.9333**

5. Conclusion and potential future works

This study proposes a Pi Sigma neural network based on the hybrid flower pollination algorithm, i.e. the f flower pollination algorithm is used to train the weights and deviations of the Pi Sigma neural network from the input layer to the hidden layer, and the trained Pi Sigma neural network is used to solve the data classification problem.

In the classification experiments, compared with FPA-PSNN, PSO-PSNN, and BA-PSNN, the FA-FPA-PSNN had better classification performance on multiple data sets, was less prone to fall into local optima, and had stronger generalization performance. It also has stronger generalization performance.

In the future, we can improve the Pi Sigma neural network based on firefly flower pollination in several ways:

a) Develop more efficient optimization algorithms. In addition to the firefly algorithm, many other optimization algorithms can be used to train neural networks, such as genetic algorithms and ant colony algorithms. In the future, we can explore the use of these algorithms to optimize the Pi Sigma neural network.

b) Introduce more sophisticated data preprocessing techniques. Data preprocessing is a crucial step in neural network applications, which can greatly influence the performance and generalization ability of the model. We can introduce more sophisticated data preprocessing techniques to effectively extract hidden information from the data and enhance the performance of the PSNN.

About the Author

Name: Qu Rongguo[*Corresponding author] Gender: Female Ethnicity: Han Education：Master’s Degree Year of birth: 2000-11 Fields：Computational Mathematics Hometown：Weinan City, Shaanxi Province

References

[1] Sharma R., Pachori R.B., Sircar, P. Seizures classification based on higher-order statistics and deep neural network[J]. Biomedical Signal Processing and Control 59 (2020): 101921.

[2] Nayak S.C., Misra B.B., Dehuri S., Hybridization of the higher order neural networks with the evolutionary optimization algorithms - An application to financial time series forecasting. Advances in Machine Learning for Big Data Analysis[J]. Singapore: Springer Nature Singapore, 2022. 119-144.

[3] Kang Q., Fan Q. W. Zurada J. M. Huang. T.W. A Pruning Algorithm with Relaxed Conditions for High-Order Neural Networks based on Smoothing Group L1/2 Regularization and Adaptive Momentum[J]. Knowledge Based Systems 257 (2022) 109858.

[4] Ivakhnenko A.G. Polynomial theory of complex systems[J]. IEEE transactions on Systems, Man, and Cybernetics 4 (1971): 364-378.

[5] Giles C.L., Maxwell T. Learning, invariance, and generalization in highorder neural networks[J]. Applied optics 26.23 (1987): 4972-4978.

[6] Shin Y., Ghosh J. The Pi Sigma network: An efficient higher-order neural network for pattern classification and function approximation[J]. IJCNN-91-Seattle international joint conference on neural networks. Vol.1. IEEE, 1991.

[7] Egrioglu E., Bas E. Modified pi sigma artificial neural networks for forecasting[J]. Granular Computing 8.1 (2023): 131-135.

[8] Bas E., Egrioglu E., Tunc T. Multivariate Picture Fuzzy Time Series: New Definitions and a New Forecasting Method Based on Pi Sigma Artificial Neural Network[J]. Computational Economics 61.1

(2023): 139- 164.

[9] Dash R., Rautray R., Dash R. Utility of a Shuffled Differential Evolution algorithm in designing of a Pi Sigma Neural Network based predictor model[J]. Applied Computing and Informatics 19.1/2 (2023): 22-40.

[10] Balasubramanian K., Ananthamoorthy N.P., Ramya K. An approach to classify white blood cells using convolutional neural network optimized by particle swarm optimization algorithm[J]. Neural Computing and Applications 34.18 (2022): 16089-16101.

[11] Gupta V., Gupta M. Improved PSO Algorithm-Based Convolutional Neural Network Approach for Ship Detection and Classifications[J]. SN Computer Science 3.4 (2022): 318.

[12] Yang X.S.. Flower pollination algorithm for global optimization[J]. International conference on unconventional computing and natural computation. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[13] Fister I., Fister Jr I., Yang X.S., Brest J. A comprehensive review of firefly algorithms[J]. Swarm and evolutionary computation 13 (2013): 34- 46.

[14] Abdalkareem Z.A., Al-Betar M.A., Amir A., Ehkan P., Hammouri A.I., Salman O.H. Discrete flower pollination algorithm for patient admission scheduling problem[J]. Computers in biology and medicine 141 (2022): 105007.

[15] Gupta S.K., Dalal A. Optimisation of hourly plants water discharges in hydrothermal scheduling using the flower pollination algorithm[J]. International Journal of Ambient Energy 44.1 (2023): 686-692.9

[16] Kaya C.B., Ebubekir K.A. Y.A. A novel approach based to neural network and flower pollination algorithm to predict number of COVID-19 cases[J]. Balkan Journal of Electrical and Computer Engineering 9.4 (2021): 327-336.

[17] Guo Z., Suo R., Zhang X. An improved flower pollination algorithm based on logistic chaotic mapping and natural mutation[J]. Proceedings of the 7th International Conference on Big Data and Computing. 2022.

[18] Cai C., Nie Y., Wu H., Fu M., Meng X. Pattern Synthesis of Antenna Array Based on Improved Flower Pollination Algorithm[J]. Proceedings of the 5th International Conference on Computer Science and Software Engineering. 2022.

[19] Peng X., Yan R., Zhao B.,et al. Fast Low-rank Representation based Spatial Pyramid Matching for Image Classification[J]. Knowledge-Based Systems, 2015, 90(C):14-22.