IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 14, 2019, accepted December 10, 2019, date of publication December 19, 2019, date of current version December 27, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2960687

A Novel Heuristic Artificial Neural Network Model for Urban Computing

QI NA[1,2,3], GUISHENG YIN[2], AND ANG LIU[4]
[1]Centre for Big Data and Business Intelligence, Harbin Engineering University, Harbin 150001, China
[2]College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China
[3]School of Economics and Management, Harbin Engineering University, Harbin 150001, China
[4]School of Management, Heilongjiang University of Science and Technology, Harbin 150001, China

Corresponding author: Qi Na (naqi@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 71602041 and Grant 71874040.

ABSTRACT Urban computing brings powerful computational techniques to bear on such urban challenges as pollution, energy consumption, and traffic congestion. After decades of rapid development, artificial neural networks (ANN) have been successfully applied in many disciplines and have enabled many remarkable research achievements. However, no quantitative method has yet been found that can identify every parameter to optimize a neural network. The BP neural network is most frequently used but suffers from the following defects with respect to complex and multidimensional training data or setting of different parameters, i.e., overfitting, slow convergence speed, trapping in local optima and poor prediction effect, and these obstacles have greatly restricted its practical applications. Therefore, this paper proposes a method that uses ant colony optimization (ACO) to train the parameters and structure of the neural network, optimizes its weight and threshold to solve its defects, and applies the model in the optimization scheme of urban operation and management to verify its effect. The experimental simulation proves that the method in this paper is effective and that it makes certain improvements in the local and global search ability, speed, and accuracy of the neural network.

INDEX TERMS Urban computing, artificial neural network model, structure and parameter, ant colony optimization, speed and accuracy.

I. INTRODUCTION

Urban computing is a solution to the challenges facing cities based on continuous acquisition, integration and analysis of the vast amounts of big data in the city environment. Urban computing combines efficient data management and analysis algorithms with novel visualization techniques to aid in better understanding of the nature of various urban phenomena and prediction of the future of the city. The neural network is an important component of machine learning and deep learning. With learning and memory ability, the neural network learns the sample features of every class and compares the input vector with the feature information it memorizes when it encounters the samples to be recognized to determine which class the samples belong to [1]. This algorithm obtains the decision information through the learning mechanism, and it does not require a priori knowledge. The BP neural network is the most popular neural network model and is a multilayer feed-forward neural network that includes an input layer, hidden layer and output layer. In practical applications, such flaws as slow convergence speed, trapping in local optimum, poor network stability and overfitting severely affect its generalization performance and hinder its applications [2], [3]. In recent years, the emergence of swarm intelligence optimization algorithms has suggested a new idea for modeling research with neural networks. Ant colony optimization (ACO) is a swarm intelligence optimization method that simulates the biological behavior in which ants move along the shortest foraging path in the natural world [4]. This algorithm takes the pheromone as the indirect communication tool, gains success in a series of combinatorial optimization problems through an internal search mechanism and positive feedback characteristic, and produces good results in a wide range of applications in many fields such as function optimization, network routing, robot path planning and data mining [5], [6]. This paper leverages ACO to optimize the

The associate editor coordinating the review of this manuscript and approving it for publication was Rongbo Zhu.

structure, parameters and stability of the neural network and verifies its effectiveness in optimization problems of urban operation and management.

BP neural network is a frequently used training method and it uses error gradient descent information to confirm the training direction. It has a slow convergence speed and the speed is even slower especially when it approximates local or global extremums. Besides, it frequently gets trapped in local extremum. In the recent decade, many improved BP algorithms have shown up, including parameter and weight mixed learning method, conjugate gradient method, and quasi-Newton method and they have accelerated the convergence speed to a certain extent. However, these methods are based on gradient descent method in essence; so their convergence speed and accuracy are still limited and they are still easy to get trapped in local optimal solution. In the past ten years, ACO has become a research priority in the field of intelligent computation and it has made excellent effect. ACO can be used in such optimization problems as multi-objective optimization, task allocation, pattern classification and high-dimensional and complete data processing. But as ACO has a short development history, its theoretical foundation and application promotion shall be further studied. Under this context, this paper adopts ACO in the training of parameters and structure of BP neural network and applies this new method into urban computing because urban computing is more complex and difficult compared with other problems. So the research work in this paper has important practical significance and value.

The special contributions of this paper include the following:

- This work studies the basic idea and implementation of the artificial neural network (ANN), analyzes the relevant parameters and typical meaning of the BP neural network, and gives a brief description of its applications. Based on in-depth understanding, this approach examines and identifies the weaknesses of BP neural network based on an in-depth understanding of the algorithm and applies related theoretical analysis to improve its structure, parameters and stability.
- The basic concept and theories of ACO are subjected to in-depth research, supplying clear explanation on its implementation, conducting simulation experiments via ACO to solve complex function optimization problems, and verifying that ACO can better solve the relevant optimization problems.
- Because the BP neural network has certain limitations, this paper proposes optimization of the structure and parameters of the BP neural network via ACO to avoid trapping in local minimums and failing to obtain the global optimal solution, and it also performs stability analysis of the model.
- Using the optimization problems in operation and management as an example, the test experiment verifies the optimization method based on the BP neural network. The analysis and comparison results show that the neural network model proposed in this paper is far more effective than the conventional methods.

The remainder of this paper is arranged as follows. Section 2 introduces related work, and Section 3 presents theoretical analysis of the BP neural network, including its nonlinear transformation and linear separability. Section 4 discusses and tests parameter optimization and analysis of ACO, Section 5 elaborates on performance analysis and evaluation of the BP neural network optimized by ACO and discusses the results of the test experiment. Section 6 closes with conclusions and future research directions.

II. RELATED WORK

The neural network is a highly nonlinear dynamic system that has parallel distributed processing ability, strong robustness and fault-tolerant capability, as well as distributed storage and learning ability. The neural network can fully approximate complex nonlinear relationships. The BP neural network, which is short for error backpropagation neural network, consists of one input layer, one or several hidden layer(s), and one output layer, and every layer is composed of a certain number of neurons. These neurons are inter-related, similar to human nerve cells. The basic BP network includes two aspects: forward propagation of signals and backpropagation of errors, i.e., it proceeds from the input to the output when calculating the actual output and from the output to the input when modifying the weight and threshold [7], [8]. The Kolmogorove theorem has proven that the BP neural network has a strong nonlinear mapping capability and generalization ability and that it can realize any continuous or mapping function with a three-layer network [9]. Historically, the backpropagation algorithm (BP neural network) has been frequently used to train the feed-forward neural network, but it suffers from the shortcomings of easy trapping in local minimum, oversensitivity to the initial conditions and slowness in convergence speed. However, if the BP network has the proper initial conditions and network structure, it can fully leverage its advantages and perfectly fulfill various tasks [10]. Random initial conditions and an estimated network structure will result in a slow and complicated training process. To solve these problems, it is necessary to optimize the structure of the neural network by cutting unnecessary connections and thresholds and optimizing the weight and threshold. Therefore, a good method must be found to solve such optimization problems with respect to the neural network structure and parameters [11].

The ANN represents network problem-solving knowledge via the distributed memory of connection weights between many connected neurons. In essence, neural network learning is the process of optimization of the network structure and weight. Optimization of the neural network performance has long been a research priority for scholars. The BP algorithm is the most frequently used training method. In recent decades, although many scholars have made numerous improvements to the BP algorithm, the convergence speed and accuracy are still limited, and it is still easily trapped in a locally optimal solution. Swarm intelligence is an artificial intelligence technology that simulates bionic behaviors [12]. Different from conventional optimization methods, swarm intelligence has an intelligent and parallel nature and is not subject to selected continuous and differentiable conditions of the target function of the object. These strengths have made it suitable for application in selected polymorphous, discrete and noisy complex optimization problems and has offered a feasible approach for solving the defects of the neural network itself. The ACO was first introduced by Macro Dorigo and others in 1991, and it is a bionic algorithm that simulates the similarity between ant foraging behavior and the travelling salesman problem. The ACO exchanges information mainly through a substance known as the pheromone, performs selection and updating according to the pheromone, and generates a globally optimal solution through numerous iterations. The ACO is a positive feedback mechanism. Therefore, based on this background, this paper uses ACO to train the parameters and structure of the feed-forward neural network and offers certain application value [13], [14].

III. PROPOSED MODEL

The BP neural network is a supervised learning algorithm, and its main strategy is inputting learning samples and using the backpropagation algorithm to repeatedly adjust and train the network weights and deviations to make the output vector as similar to the expected vector as possible. When the square sum of error in the output layer is smaller than the specified error, the training is completed, and the algorithm saves the network weight and deviation. Backpropagation the adjusts network parameters by calculating the error between the output layer and the expected value, thus diminishing the error. The powerful force of ANN is almost entirely built upon the backpropagation algorithm [15].

A. BACKPROPAGATION PROCESS OF ERROR

In backpropagation of error, we calculate the output error of neurons in a layer-by-layer manner starting from the output layer and adjust the weights and thresholds in every layer according to the error gradient descent method to make the final output of the modified network approximate the expected value [16].

In the following text, X_j represents the input of the jth node in the input layer, and $j = 1, \ldots, M$; w_{ki} is the weight from the ith node in the hidden layer to the jth node in the input layer; θ_i is the threshold of the ith node in the hidden layer; $\Phi(x)$ is the activation function in the hidden layer; W_{ki} is the weight from the kth node in the output layer to the ith node in the hidden layer, and $i = 1, \ldots, q$; a_k is the threshold of the kth node in the output layer, and $k = 1, \ldots, L$; ψ_k is the activation function in output layer; and O_k is the output of the kth node in the output layer.

For every sample p, its quadratic error criterion function is E_p:

$$E_p = \frac{1}{2}\sum_{k=1}^{L}(T_k - o_k)^2 \tag{1}$$

The total error criterion function of P training samples is:

$$E = \frac{1}{2}\sum_{p=1}^{P}\sum_{k=1}^{L}(T_k^p - o_k^p)^2 \tag{2}$$

According to the error gradient descent method, we modify in turn the correction Δw_{ki} of the weight in the output layer, the correction Δa_k of the threshold in the output layer, the correction Δw_{ij} of the weight in the hidden layer and the correction $\Delta\theta_i$ of the threshold in the hidden layer.

$$\begin{aligned} \Delta w_{ki} &= -\eta\frac{\partial E}{\partial w_{ki}}; \quad \Delta a_k = -\eta\frac{\partial E}{\partial a_k} \\ \Delta w_{ij} &= -\eta\frac{\partial E}{\partial w_{ij}}; \quad \Delta\theta_i = -\eta\frac{\partial E}{\partial \theta_i} \end{aligned} \tag{3}$$

The adjustment formula for the weight in the output layer is given as follows:

$$\Delta w_{ki} = -\eta\frac{\partial E}{\partial w_{ki}} = -\eta\frac{\partial E}{\partial net_k}\frac{\partial net_k}{\partial w_{ki}} = -\eta\frac{\partial E}{\partial o_k}\frac{\partial o_k}{\partial net_k}\frac{\partial net_k}{\partial w_{ki}} \tag{4}$$

The adjustment formula for the threshold in the output layer is written as shown:

$$\Delta a_k = -\eta\frac{\partial E}{\partial a_k} = -\eta\frac{\partial E}{\partial net_k}\frac{\partial net_k}{\partial w_k} = -\eta\frac{\partial E}{\partial o_k}\frac{\partial o_k}{\partial net_k}\frac{\partial net_k}{\partial a_k} \tag{5}$$

The adjustment formula for the weight in the hidden layer is shown below:

$$\Delta w_{ij} = -\eta\frac{\partial E}{\partial w_{ij}} = -\eta\frac{\partial E}{\partial net_i}\frac{\partial net_i}{\partial w_{ij}} = -\eta\frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial net_i}\frac{\partial net_i}{\partial w_{ij}} \tag{6}$$

The adjustment formula for the threshold in the hidden layer is presented as follows:

$$\Delta\theta_i = -\eta\frac{\partial E}{\partial \theta_i} = -\eta\frac{\partial E}{\partial net_i}\frac{\partial net_i}{\partial \theta_i} = -\eta\frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial net_i}\frac{\partial net_i}{\partial \theta_i} \tag{7}$$

In the end, we obtain the following formulas:

$$\Delta w_{ki} = \eta\sum_{p=1}^{P}\sum_{k=1}^{L}(T_k^p - o_k^p)\cdot\psi'(net_k)\cdot y_i \tag{8}$$

$$\Delta a_k = \eta\sum_{p=1}^{P}\sum_{k=1}^{L}(T_k^p - o_k^p)\cdot\psi'(net_k) \tag{9}$$

$$\Delta w_{ij} = \eta\sum_{p=1}^{P}\sum_{k=1}^{L}(T_k^p - o_k^p)\cdot\psi'(net_k)\cdot w_{ki}\cdot\phi'(net_i)\cdot x_j \tag{10}$$

$$\Delta\theta_i = \eta\sum_{p=1}^{P}\sum_{k=1}^{L}(T_k^p - o_k^p)\cdot\psi'(net_k)\cdot w_{ki}\cdot\phi'(net_i) \tag{11}$$

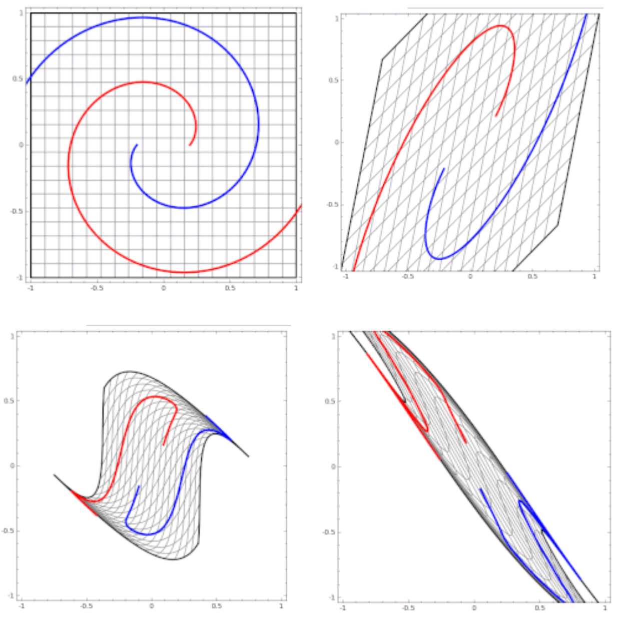


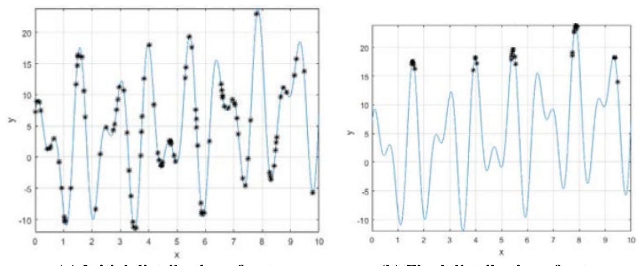
FIGURE 1. Hyperplane segmentation of BP neural network.

B. NONLINEAR TRANSFORMATION AND LINEAR SEPARABILITY

Several neurons of the BP neural network are interconnected, and among them, one can accept several input signals and convert them to output signals according to certain rules. Because of the complex connection relationships among neurons and the nonlinear methods of the neurons that transmit signals, various relationships can be built among the input and output signals, thereby creating a black-box model, with rules that cannot be accurately described with a mechanistic model but which are objective, deterministic or fuzzy between the input and the output. The neural network learns how to use the linear transformation of the matrix plus the nonlinear transformation of the activation function [17].

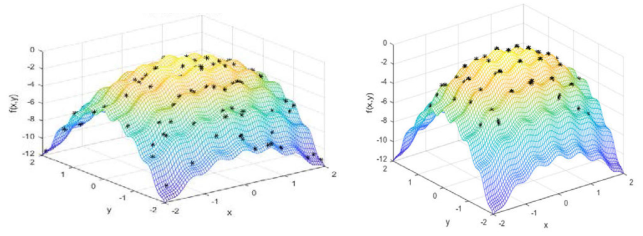
In the 1D case and with classification as an example, if we require division into three classes, namely, positive, negative and zero, the straight line in 1D space can find two hyperplanes to segment these three classes into the subspace one dimension lower than the current space. If the current space is a straight line, the hyperplane is a point. In the 2D situation, the four quadrants of the plane are also linearly separable. After magnifying, translating, rotating and distorting the original 2D space, the spatial transformation of the neural network can find a hyperplane to segment the red and blue lines in 3D space. Finally, we find a hyperplane to segment the space [18], and the spatial change of the neural network is shown in Figure 1.

Supervised learning gives a large number of training samples to the neural network so that it can learn how to transform the space from the training samples. The weights in each layer control how to transform the space, and the end goal is a well-trained weight matrix for all layers. To distort the spatially and linearly separable angle, the neural network learns how



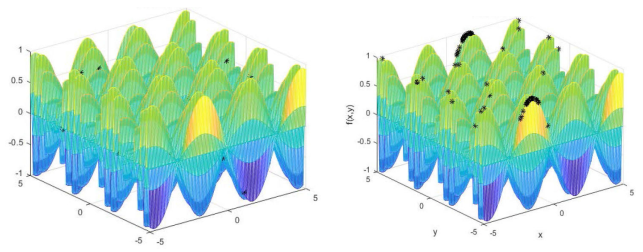
(a) Initial distribution of ants (b) Final distribution of ants

FIGURE 2. Function $f1$.



(a) Initial distribution of ants (b) Final distribution of ants

FIGURE 3. Function $f2$.



(a) Initial distribution of ants (b) Final distribution of ants

FIGURE 4. Function $f3$.

to use the linear transformation of the matrix plus the nonlinear transformation of the activation function and places the original input space into the linearly separable/sparse space for classification/regression. We increase the number of nodes and increase the number of dimensions, i.e., increase the linear transformation ability. To increase the number of layers, we increase the number of activation functions, i.e., increase the number of nonlinear transformations [19], [20].

IV. PARAMETER AND OPTIMIZATION PERFORMANCE ANALYSIS OF ANT COLONY OPTIMIZATION

In the real living environment, an ants in a colony leave a substance on the path that they pass through as they search for sources, and we call this substance the pheromone. The ants perceive the pheromone left by others, and this matter determines the behaviors of the ants to a certain extent. The concentration of pheromone indicates the length of the path. The more concentrated the pheromone, the shorter the path, and the less concentrated the pheromone, the longer the path.

Generally, ants prioritize the path with a greater amount of the pheromone (indicating that other ants have passed this path many times) and simultaneously release the pheromone to enhance the concentration along this path to finally constitute a positive feedback loop. Over time, the pheromone concentration is continuously attenuated. When an ant makes a choice, the pheromone left by others suffers certain volatilization.

To combine the ACO with optimization problems, it needs to build a solution space, i.e., all solutions to the problem to be optimized. The basic application idea is described as follows: every path that the ants pass is a solution, and all paths form the solution space. In the end, we select the optimal solution from the space.

A. TRAVELLING SALESMAN PROBLEM (TSP)

The travelling salesman problem (TSP) is described as follows: There are N cities, and starting from the initial position, the salesman travels to every one of the N cities, does not travel through a city that has already been visited, returns to the departure point at the end, and minimizes the total cost, i.e., finds the shortest path.

It might not be difficult to describe TSP, but it is also not easy to obtain the exact solution because there are $(n-1)!/2$ feasible paths. For this reason, the academic circle has referred to the TSP as an NP-hard problem. The TSP is a NP-complete problem. If one type of algorithm works well for the TSP application, it usually achieves good results if used in other applications. Therefore, it is quite reliable to verify whether an improved algorithm is effective via the TSP.

The mathematical model of the TSP is given as follows: let $G=(V,E)$ be a weighted graph, where $V=\{1,2,\cdots,n\}$ is the vertex set, and $E=\left\{e_{ij}=(i,j)\,|i,j\in V, i\neq j\right\}$ is the frontier set. With $d_{ij}=d_j(i,j\in V, i\neq j)$ as the frontier set, $d_{ij}\{i,j\in V, i\neq j\}$ records the distance from vertex i to vertex j. When $d_{ij}>0$ and $d_{ij}\neq\infty$, $d_{ij}=d_{ji}(i,j\in V)$, and the mathematical model formula of TSP is written as follows.

$$\min F=\sum_{i\neq j} d_{ij}\times x_{ij}$$

$$x_{ij}=\begin{cases}1, & e_{ij}\text{ side on the optimal path}\\ 0, & e_{ij}\text{ side not on the optimal path}\end{cases} \tag{12}$$

$$\begin{cases}\sum\limits_{i\neq j} x_{ij}=1, & i\in V, \quad (a)\\ \sum\limits_{i\neq j} x_{ij}=1, & j\in V, \quad (b)\\ \sum\limits_{i\neq j} x_{ij}=1, & j\in V, \quad (c)\end{cases} \tag{13}$$

$$\sum_{i,j\in s} x_{ij}=|S| \tag{14}$$

$$x_{ij}=\begin{cases}1, & e_{ij}\text{ side on the optimal path}\\ 0, & e_{ij}\text{ side not on the optimal path}\end{cases} \tag{15}$$

B. ANT COLONY OPTIMIZATION ALGORITHM MODEL

To simulate ant behaviors, the following definitions are given: n : number of cities; m : number of ants, C: set of all cities, d_{ij} : distance between city i to city j, α : importance of the pheromone, β : importance of the heuristic factor, ρ : volatilization of the pheromone, $\eta_{ij}(t)=\frac{1}{d_{ij}}$: expectation for the ants to transfer from city i to city j, τ_{ij} : pheromone intensity on path (i,j), and $p_{ij}^k(t)$: state transition probability for ant k to go from city i to city j at t, where j is the unvisited city.

We set the parameters for the specific process of ant colony optimization (ACO): τ_{ij} is 0, and the number of ants is m. Every choice made by the kth $(k=1,2,\ldots,m)$ ant is determined by the pheromone concentration, and its state transition probability is $p_{ij}^k(t)$, as shown by Formula (16).

$$p_{ij}^k(t)=\begin{cases}\dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}(t)\right]^{\beta}}{\sum\limits_{s\in a_k}\left[\tau_{is}(t)\right]^{\alpha}\left[\eta_{is}(t)\right]^{\beta}}, & j\in a_k\\ 0, & \text{Others}\end{cases} \tag{16}$$

where $a_k=\{1,2,\ldots,n\}-tabu_k$ refers to the set of all cities to be chosen the next time, $tabu_k$ is the cities that ant k has passed (which undergoes certain changes over the operation of ACO), i represents the city in which the ant is located, j is a member of set $tabu_k$, $p_{ij}^k(t)$ is the probability that ant k travels from city i to the next city j, $\eta_{ij}(t)=\frac{1}{d_{ij}}$ is the expectation that the ant goes from city i to city j, and τ_{ij} is the pheromone concentration on path (i,j). It can be learned from Formula (16) that $p_{ij}^k(t)$ is proportional to $\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}(t)\right]^{\beta}$, and therefore, when an ant chooses a path, it prioritizes the city with higher pheromone concentration.

ACO also has a significant characteristic, i.e., pheromone volatilization. In other words, as time passes, the pheromone concentration is persistently attenuated when an ant visits the cities and releases the pheromone. This characteristic makes the pheromone update itself continuously and avoids the full influence of the heuristic factor due to excessive pheromone concentration. Therefore, the pheromone adjustment formula for path (i,j) is given as follows:

$$\tau_{ij}(t+n)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij} \tag{17}$$

$$\Delta\tau_{ij}=\sum_{k=1}^{n}\Delta\tau_{ij}^k \tag{18}$$

In Formula (17) and Formula (18), τ_{ij}^k represents the amount of pheromone released by the kth ant when transferring from city i to city j, and $\Delta\tau_{ij}$ is the sum of pheromone left by all ants when going from city i to city j.

C. IMPACT OF KEY PARAMETERS IN ANT COLONY OPTIMIZATION

1) ANALYSIS OF THE NUMBER OF ANTS m IN THE ANT COLONY

When ACO searches the optimal solution, it needs every ant to release the pheromone and for other members to participate in team collaboration and exchange of information by perceiving the pheromone left by members to achieve complex and orderly behaviors. With the TSP as an example, after an ant finishes an iteration, a solution is generated, and this solution records all paths that have been passed and belong to the feasible solution set. When there are m ants, their paths after finishing an iteration become one of the many sets, and the superset is the problem solution set. Thus, the larger the number of ants in the ant colony, the larger the subsets, thus improving the algorithm stability and the search ability of the ant colony.

In this initial stage of operation, the larger the number of ants m, the better the implementation of ACO and the more accurate the obtained optimal solution. However, when the number exceeds a certain limit, it is not as effective in improving the algorithm. In fact, the performance no longer improves, the convergence slows, and the efficiency falls. Too many ants adversely affect the positive feedback information on which ACO depends, and it will take a longer time to run.

2) ANALYSIS OF INFORMATION HEURISTIC FACTOR α

The heuristic factor α in ACO it represents the relative importance of the impact that the pheromone plays in leading the ants to search paths in the entire operation. When α is larger, the ant colony is relatively stable in selection of the entire path, and the result is not as random, and thus the ants more easily choose the path selected by others. When α too small, the process is easily trapped in a local optimal solution.

For ACO, α has a great influence. If the value of α is too small, it is easy to get stuck in a local optimum too early in the process and thus difficult to obtain the global optimum, i.e., prematurity occurs. However, if the value of $\alpha\beta$ is too large, it means that the pheromone has a tremendous impact on the ants making a choice of path. In the end, when time passes, the local positive feedback mechanism comes into play and thus, ACO is trapped in a local optimal solution.

3) ANALYSIS OF EXPECTED HEURISTIC FACTOR β

In ACO, β is an expected heuristic factor indicating the relative importance of the heuristic pheromone in instructing the ants to search the path. The final effect of a priori and certain other factors are closely related to the value of β. When the value of β is large, the probability that ants select the local minimum path at a certain point correspondingly increases, and the algorithm convergences at a faster speed, but weakened randomness makes the ACO easily trapped in a local optimal solution.

For the efficiency of ACO, β plays a large role. If the value of β is too large, the optimal path obtained worsens, but if it is too small, the algorithm becomes a random search process, and it is not easy to find the optimal solution.

4) ANALYSIS OF PHEROMONE VOLATILIZATION COEFFICIENT ρ

As time goes by, the pheromone released by the visited ants volatizes. Similar to other bionic optimization algorithms, ACO also has weaknesses, i.e., ease of trapping in a local optimal solution and slow convergence speed. The value of ρ directly affects the convergence speed and global search ability of ACO.

TABLE 1. Test function formulas.

Function No.	Formula	Value of Independent Variable
$f1$	$f(x)=9\sin(5x)+7\cos(8x)+x$	[0,10]
$f2$	$f(x,y)=-(x^2+2y^2-0.3\cos(3\pi x)$ $-0.4\cos(4\pi y)+0.6)$	[-2,2]
$f3$	$f(x,y)=\sin(x)\sin(y+\pi)\cos^5(4\pi y)$	[-5,5]
$f4$	$f(x,y)=10+x^2+y^2-10(\cos(2\pi x)$ $+\cos(2\pi y))$	[-4,4]

5) ANALYSIS OF PHEROMONE INTENSITY FACTOR Q

The total amount of pheromone released by the ant after one cycle is Q. Generally, as the value of Q increases, the increase in speed of the total pheromone along the path travelled by the ants accelerates to a certain extent, which helps enhance the positive feedback of ACO to expedite its convergence speed and facilitate the search for the optimal solution.

The impact of Q on ACO is not only related to the abovementioned α, β and ρ but also to the selection of the pheromone update model. Three common models are used in ACO, namely, ACS, AQS and ADS. In the ACS and AQS models, parameter Q is quite different in the effect it has while it changes over the quantity included in the problem in regard to the impact Q plays on the ACS model. The research result shows that Q has no significant impact on the efficiency of the ACS model. Therefore, there is no explicit standard when choosing a value for Q. In other words, Q can be any value, and no special circumstance is considered.

D. REALIZATION OF FUNCTION OPTIMIZATION BASED ON ANT COLONY OPTIMIZATION

1) TEST FUNCTION

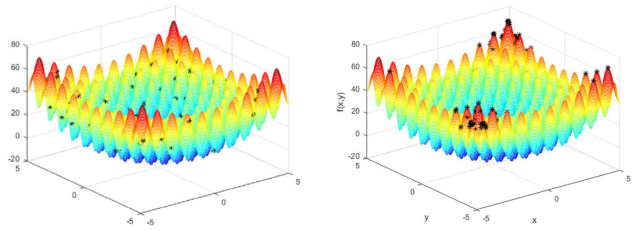
Additionally, $f1$ is a function of one variable, and it produces a 2D image with the shape of an irregular wave, different peak heights, and numerous local minimums and maximums; $f2$ is a 3D image in the space and has 4 global minimums and several local maximums; $f3$ is also a 3D image with flaked peaks as well as several local minimums and maximums; and $f4$ is a multimodal test function and has several local maximums and minimums.

2) TEST EXPERIMENT AND OPTIMIZATION PERFORMANCE ANALYSIS

We select 4 test functions for optimization. Among them, one test function is a 2D multimodal function, and its peaks have the following characteristics: equal distance but unequal height, equal height but unequal distance, and unequal distance and unequal height. The remaining 3 test functions are 3D multimodal functions with different complexity. The purpose of the experiment is to find the maximum value of the test function. The experimental parameters are set as follows: number of ants $m = 100$, number of iterations Echo $= 50$, pheromone volatilization coefficient $\rho = 0.8$,

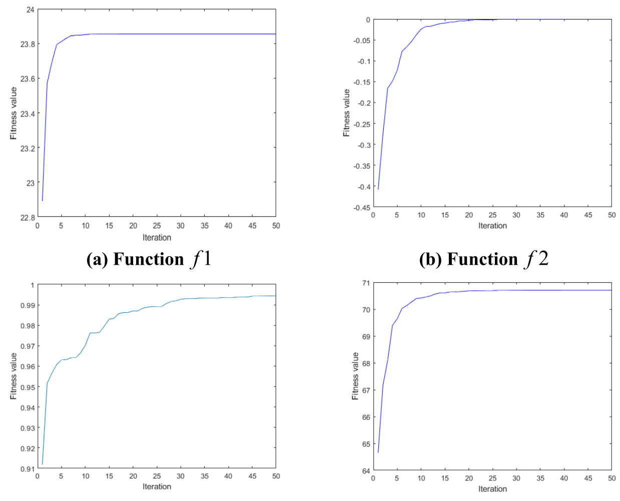
TABLE 2. Optimization implementation of ant colony optimization (ACO).

Function	Mean Fitness Value	Optimal Solution	Worst Solution	Run Time (S)	Mean Run Time (S)	Variance of Mean Fitness Value
$f1$	23.8233	23.8547	18.4868	0.2995	0.3556	0.0203
$f2$	-0.0300	-1.7464E-06	-0.7193	0.5348		0.0057
$f3$	0.9833	0.9992	0.8109	0.4397	0.5747	2.4793E-04
$f4$	70.3437	70.7066	55.8280	0.6764		1.0959



(a) Initial distribution of ants (b) Final distribution of ants

FIGURE 5. Function $f4$.



(a) Function $f1$ (b) Function $f2$

(c) Function $f3$ (d) Function $f4$

FIGURE 6. Convergence curves of four test functions.

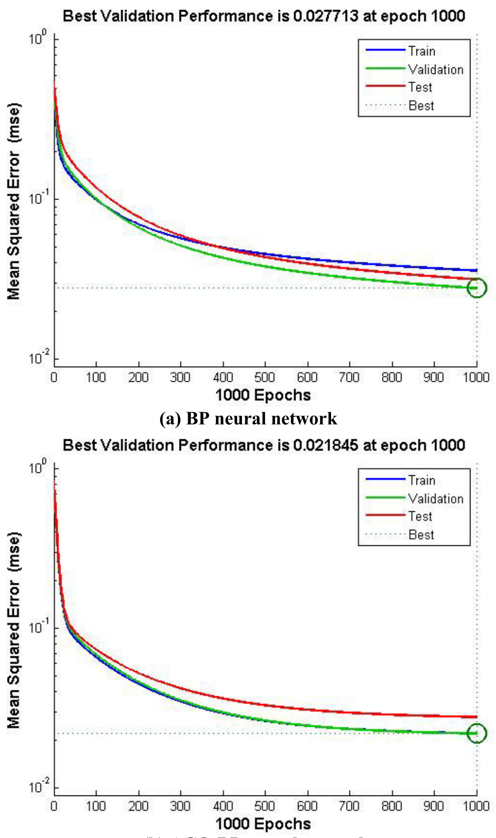
transfer probability $\rho0 = 0.2$, and total pheromone $Q = 1$. The stride is set as 0.05, and every group operates 20 times independently.

It can be observed from Fig. 6 that the value of the convergence curve of $f1$ is close to 23.85, that of $f2$ is close to -1.383e-04, that of $f3$ is close to 0.9944, and that of $f4$ is close to 70.71.

3) EXPERIMENTAL RESULTS AND ANALYSIS

We tidy up the simulation test data and keep four decimal places, as shown in Table 2.

ACO can achieve good solutions in optimization of both 2D and 3D multimodal functions. It can be observed from the above that with a faster convergence speed in searching for the maximum value of the function, ACO needs fewer iterations and has high efficiency. ACO runs for a short time in function optimization with high efficiency. Table 2 shows that the shortest and longest times for ACO in function optimization are 0.2995 s and 0.5090s respectively, and thus it is obvious that it has high efficiency. For multimodal functions, ACO takes a shorter time in optimization of 2D functions than for 3D functions. Moreover, ACO has better stability in operation of function optimization, and the variance of the mean fitness value is small, i.e., the data fluctuate slightly and are more stable. In the search for the maximum value of the function, the curve of mean fitness gradually increases and finally converges to a relatively stable state.



(a) BP neural network

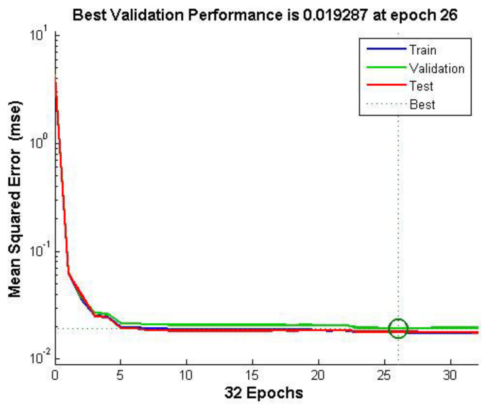
(b) ACO-BP neural network

FIGURE 7. MSE curve of traingd function.

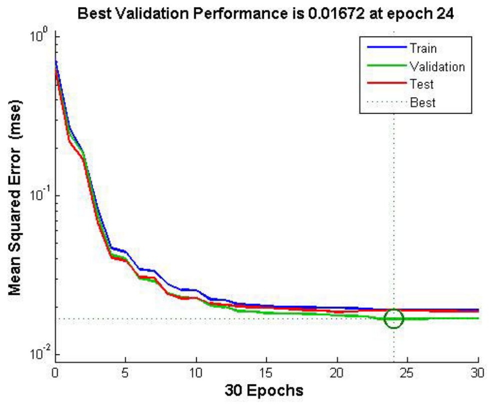
V. PERFORMANCE ANALYSIS OF OPTIMIZED BP NEURAL NETWORK BY ACO

This paper adopts the BP neural network and ACO-BP neural network to solve the scheduling problems of urban operation and management. First, this approach builds the scheduling model via the neural network and initializes its structure. Second, it uses ACO in the weight and threshold operations, improves its convergence and calculation accuracy, and makes dispatch with minimizing the make span as the goal.

In this experiment, the neural network is set as a three-layer structure: one input layer, one hidden layer and one output layer. The input layer selects four attribute values for training according to the scheduling characteristics. In this

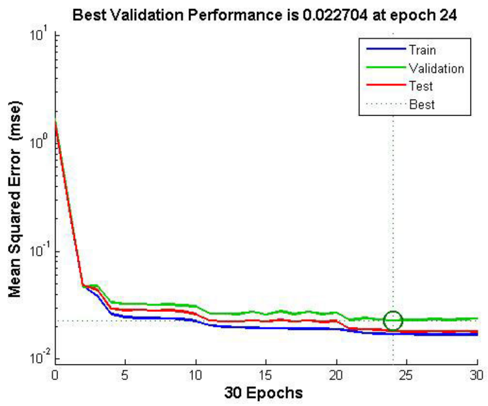


(a) BP neural network

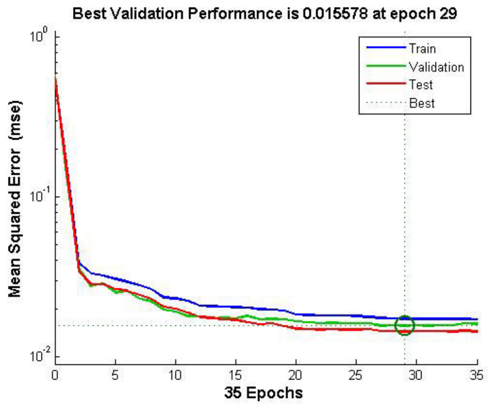


(b) ACO-BP neural network

FIGURE 8. MSE curve of trainscg function.



(a) BP neural network



(b) ACO-BP neural network

FIGURE 9. MSE curve of trainbfg function.

layer, the number of neurons is determined by the number of parameters of the input attribute. The hidden layer is the intermediate computational processing layer, and the number of its neurons is determined according to different problems.

We use MATLAB software to train the normalized input data and determine the training function in the training process. This paper determines the training function via experimental comparison, i.e., with the same remaining parameters, we apply the following four methods to train the neural network: the gradient descent method (traingd), the scaled conjugate gradient method (trainscg), the quasi-Newton method (trainbfg), and the LM algorithm (trainlm). For the conventional BP neural network and ACO-BP neural network, we complete the number of iterations and training error in every training function, find the training function corresponding to the lowest training error by comparing the number of iterations and training errors, and determine it as the final training function. Finally, we show the training error and result and also compare the training results of two different neural networks.

The result comparison of the different training functions of two neural networks is shown in Table 3. The corresponding mean square error (MSE) curves for four training functions are shown in Figure 7 through Figure 10. For comparison in the same neural network, traingd function has the larger MSE, and its number of iterations is much greater than that of

TABLE 3. Comparison of results of different training functions.

	BP Neural Network		ACO-BP Neural Network	
	Epoch	Performance	Epoch	Performance
Negative Gradient Descent Algorithm (traingd)	1000	0. 0286	1000	0. 0223
Quantitative Conjugate Gradient Algorithm (trainscg)	25	0. 0189	24	0. 0156
Bfgs Quasi-Newton Algorithm (trainbfg)	23	0. 0235	29	0. 0177
Lm Algorithm (trainlm)	29	0. 0148	10	0. 0139

other functions. This function is forced to terminate because it reaches the maximum number of iterations, and its convergence speed is far slower than that of the remaining functions. Except for the MSE of trainbfg function, which is greater than 0.02, the other values are all less than 0.02 with a number of iterations lower than 30. Observing the three functions trainscg, trainbfg and trainlm, it can be found that trainlm function has the lowest MSE, a smaller number of iterations and the fastest convergence speed. Therefore, trainlm is chosen as the training function.

After the trainlm function is determined as the training function, because the BP neural network has certain restrictions, to avoid getting trapped in a local minimum and failing to obtain the global optimal solution, this paper selects ACO to optimize the BP neural network and search for the optimal

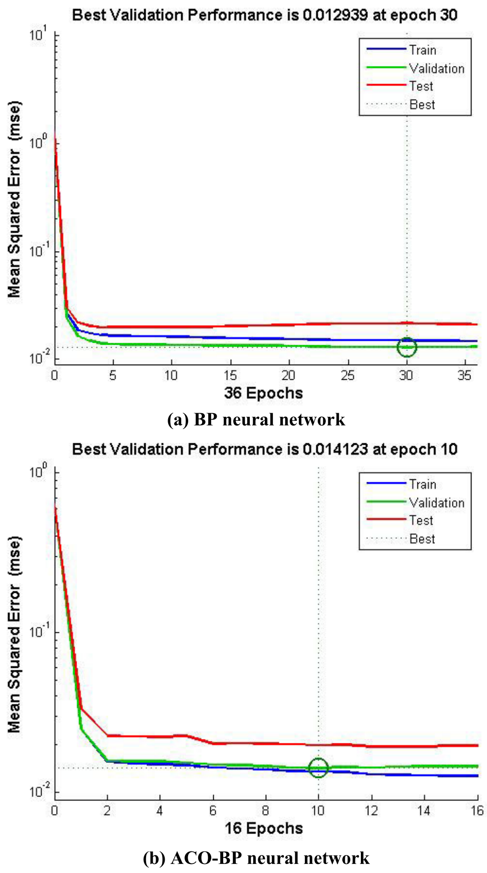


FIGURE 10. MSE curve of trainlm function.

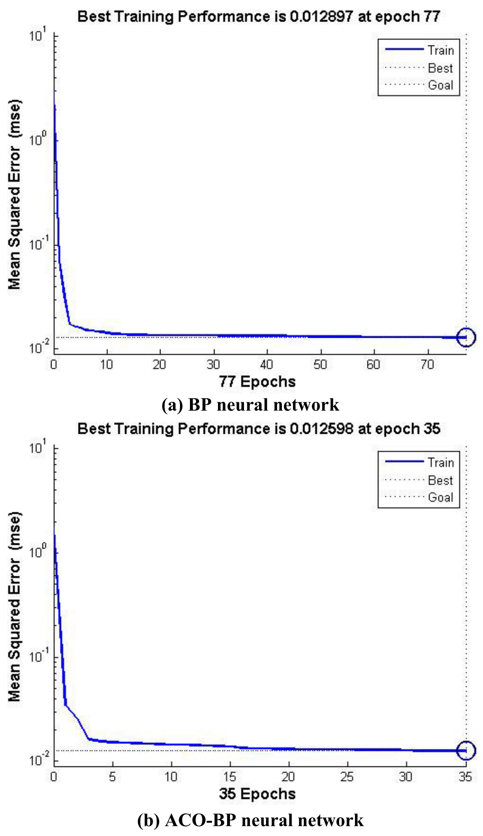


FIGURE 11. Comparison of training results and simulation results between ACO-BP neural network and BP neural network.

TABLE 4. Number of weights and thresholds to be optimized.

Connection Weight in Input Layer And Hidden Layer	Threshold in Hidden Layer	Connection Weight in Output Layer And Hidden Layer	Threshold in Output Layer
65	5	5	1

TABLE 5. Results comparison before and after ACO optimization.

		BP Neural Network	ACO-BP Neural Network
Training Sample	Error	0. 012798	0. 012616
Testing Sample	Error	0. 021547	0. 019513
	Grade Matching Degree	43.4%	42.38%

solution to the problem space in a broader scope. Therefore, after combining the BP neural network with ACO, the ACO-BP neural network can be obtained, which is highly effective in global searching and has strong local optimization ability and a fast convergence speed.

The specific steps are listed as follows. First, we determine the structure of the BP neural network and initialize its weight and threshold. For the ACO component, we encode the data, perform the ACO optimization operations and calculate the optimal fitness value. We repeat the above steps and judge whether the current generation or the optimal fitness value has reached the convergence condition. If not, we return to the optimization operation, and if so, it indicates that the optimal BP neural network has been found. Thus, we obtain its weight and threshold, update the corresponding weight and threshold to the neural network and calculate its error.

If there are 13 neurons in the input layer, 1 neuron in the output layer, and 5 nodes in the hidden layer, the BP neural network structure is 13-5-1. According to this structure, it can be calculated that there are 65 weights from the input layer to the hidden layer and 5 weights from the hidden layer to the output layer, for a total of 70 weights. For the thresholds, there are 5 weights in the hidden layer and 1 in the output layer, for 6 altogether. Therefore, there are 76 variables to be optimized. We take the MSE between the actual output value and the expected output value as the individual fitness value. Table 4 and Table 5 list the weights and thresholds to be optimized and the result before and after ACO. A comparison of the training result and simulation result produced by the ACO-BP neural network and BP neural network is shown in Figure 10.

According to the above results, it can be observed from the training process that the ACO-BP neural network has a far smaller number of iterations than the BP neural network, actually only half of the number of the BP neural network. Additionally, the ACO-BP has a faster convergence speed, and its MSE is also smaller than that of the BP neural network. Compared with the simulation result, for most sample points, the ACO-BP neural network has an error smaller than that of

BP neural network, and its total MSE is also slightly lower than that of the BP neural network. Therefore, the ACO-BP neural network is better than the BP neural network with respect to performance.

VI. CONCLUSION

The scale of the neural network structure affects its learning ability and generalization capacity. If the structure is too small, the learning ability is not sufficient, and if it is too large, the generalization capacity is weakened. If the parameters of the neural network are too small, it is likely that the output of each layer of the network is close to 0, and if the parameters are too large, the output of each layer will be large, i.e., each neuron is saturated, and when performing backpropagation, the gradient to the activation function will approach zero, which will also cause the gradient to disappear. An intelligent optimization algorithm is designed to make the neural network learn a proper structure and meet the requirements of learning ability while preserving the best generalization capacity. Intelligent calculation that simulates biological behaviors is an excellent optimization method. This paper combines the neural network with ACO to optimize its structure, parameters, and stability. This paper also solves high-dimensional complex problems in practice and proves the effectiveness of the method in this paper via optimization problems in urban operation and management. The research achievements of this paper pave the way toward practical applications of neural networks and offer an effective reference.

REFERENCES

[1] S. Heo and J. H. Lee, "Parallel neural networks for improved nonlinear principal component analysis," *Comput. Chem. Eng.*, vol. 127, no. 4, pp. 1–10, Aug. 2019.
[2] A. Choi, R. Wang, and A. Darwiche, "On the relative expressiveness of Bayesian and neural networks," *Int. J. Approx. Reasoning*, vol. 113, pp. 303–323, Oct. 2019.
[3] W. Jia, S. Qin, and X. Xue, "A generalized neural network for distributed nonsmooth optimization with inequality constraint," *Neural Netw.*, vol. 119, pp. 46–56, Nov. 2019.
[4] J. Kim, H. Kim, S. Huh, J. Lee, and K. Choi, "Deep neural networks with weighted spikes," *Neurocomputing*, vol. 311, no. 15, pp. 373–386, Oct. 2018.
[5] Z. Ye and M. K. Kim, "Predicting electricity consumption in a building using an optimized back-propagation and Levenberg-Marquardt back-propagation neural network: Case study of a shopping mall in China," *Sustain. Cities Soc.*, vol. 42, pp. 176–183, Oct. 2018.
[6] T. Akter and S. Desai, "Developing a predictive model for nanoimprint lithography using artificial neural networks," *Mater. Des.*, vol. 160, pp. 836–848, Dec. 2018.
[7] D. J. Hemanth, J. Anitha, and L. H. Son, "Brain signal based human emotion analysis by circular back propagation and deep Kohonen neural networks," *Comput. Elect. Eng.*, vol. 68, pp. 170–180, May 2018.
[8] R. Tang, S. Fong, S. Deb, V. A. Vasilakos, and R. C. Millham, "Dynamic group optimisation algorithm for training feed-forward neural networks," *Neurocomputing*, vol. 314, no. 7, pp. 1–19, Nov. 2018.
[9] Z. Zhang, A. Li, and S. Yu, "Finite-time synchronization for delayed complex-valued neural networks via integrating inequality method," *Neurocomputing*, vol. 318, no. 27, pp. 248–260, Nov. 2018.
[10] A. Taheri-Garavand, V. Meda, and L. Naderloo, "Artificial neural Network?Genetic algorithm modeling for moisture content prediction of savory leaves drying process in different drying conditions," *Eng. Agricult., Environ. Food*, vol. 11, no. 4, pp. 232–238, Oct. 2018.
[11] S. Heo and J. H. Lee, "Fault detection and classification using artificial neural networks," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 470–475, 2018.
[12] C. Cheng, X. Cheng, N. Dai, X. Jiang, and W. Li, "Prediction of facial deformation after complete denture prosthesis using BP neural network," *Comput. Biol. Med.*, vol. 66, pp. 103–112, Nov. 2015.
[13] J. Wang, Y. Wen, Z. Ye, L. Jian, and H. Chen, "Convergence analysis of BP neural networks via sparse response regularization," *Appl. Soft. Comput.*, vol. 61, pp. 354–363, Dec. 2017.
[14] J. Wang, Y. Wen, Y. Gou, Z. Ye, and H. Chan, "Fractional-order gradient descent learning of BP neural networks with Caputo derivative," *Neural Netw.*, vol. 89, pp. 19–30, May 2017.
[15] Z. Sun, Y. Chen, X. Li, X. Qin, and H. Wang, "A Bayesian regularized artificial neural network for adaptive optics forecasting," *Opt. Commun.*, vol. 382, no. 1, pp. 519–527, Jan. 2017.
[16] Y. Yang, Y. Chen, Y. Wang, C. Li, and L. Li, "Modelling a combined method based on ANFIS and neural network improved by DE algorithm: A case study for short-term electricity demand forecasting," *Appl. Soft Comput.*, vol. 49, pp. 663–675, Dec. 2016.
[17] M. Sivatte-Adroer, X. Llanas-Parra, I. Buj-Corral, and J. Vivancos-Calvet, "Indirect model for roughness in rough honing processes based on artificial neural networks," *Precis. Eng.*, vol. 43, pp. 505–513, Jan. 2016.
[18] F.-H. Hsiao, "Neural-network based approach on delay-dependent robust stability criteria for dithered chaotic systems with multiple time-delay," *Neurocomputing*, vol. 191, no. 26, pp. 161–174, May 2016.
[19] L. M. L. De Campos, R. C. De Oliveira, and M. Roisenberg, "Optimization of neural networks through grammatical evolution and a genetic algorithm," *Expert Syst. Appl.*, vol. 56, no. 1, pp. 368–384, Sep. 2016.
[20] J. Moreno-Valenzuela and C. Torres-Torres, "Adaptive chaotification of robot manipulators via neural networks with experimental evaluations," *Neurocomputing*, vol. 182, no. 19, pp. 56–65, Mar. 2016.

QI NA received the B.S. degree in food science and engineering from Northeast Agricultural University, Harbin, China, in 2007, and the Ph.D. degree in management science and engineering from Harbin Engineering University, Harbin, in 2018.

He is currently a Researcher with the Centre for Big Data and Business Intelligence, a Lecturer with the College of Computer Science and Technology, and a Research Assistant with the School of Economics and Management, Harbin Engineering University. His main research areas include data mining, enterprise operation management, and big data technology.

GUISHENG YIN received the B.S. degree in computer application, the M.S. degree in computer application technology, and the Ph.D. degree in control theory and control engineering from Harbin Engineering University, Harbin, China, in 1986, 1989, and 2000, respectively.

He is currently a Professor with the College of Computer Science and Technology, Harbin Engineering University. His main research areas include data mining, computational intelligence, and information security.

ANG LIU received the B.A. degree in scientific English and the Ph.D. degree in management science and engineering from Harbin Engineering University, Harbin, Heilongjiang, China, in 2006 and 2018, respectively.

He is currently a Lecturer with the School of Management, Heilongjiang University of Science and Technology, Harbin. His main research interests include network simulation, knowledge management, and game calculation.