A neural network-based prediction model in water monitoring networks
Xiaohong Ji, Ying Pan, Guoqing Jia and Weidong Fang

ABSTRACT

To improve the prediction accuracy of ammonia nitrogen in water monitoring networks, the combination of a bio-inspired algorithm and back propagation neural network (BPNN) has often been deployed. However, due to the limitations of the bio-inspired algorithm, it would also fall into the local optimal. In this paper, the seagull optimization algorithm (SOA) was used to optimize the structure of BPNN to obtain a better prediction model. Then, an improved SOA (ISOA) was proposed, and the common functional validation method was used to verify its optimization performance. Finally, the ISOA was applied to improve BPNN, which is known as the improved seagull optimization algorithm–back propagation (ISOA–BP) model. The simulation results showed that the prediction accuracy of ammonia nitrogen was greatly improved and the proposed model can be better applied to the prediction of complex water quality parameters in water monitoring networks.

Key words | ammonia nitrogen prediction, back propagation neural network, improved seagull optimization algorithm, neural network, seagull optimization algorithm, water monitoring

HIGHLIGHTS

- The structure of BPNN was optimized to obtain a better prediction model by using the seagull optimization algorithm (SOA).
- We proposed an improving SOA (ISOA) and used the common functional validation method to verify its optimization performance.
- The ISOA was used to improve BPNN, via the improved seagull optimization algorithm–back propagation (ISOA–BP) model.

INTRODUCTION

As an important part of the new generation of information technology, the Internet of Things (IoT) has been widely researched and applied in various scenarios (Fang et al. 2016, 2020a; Khan et al. 2020). Although the emergence of IoT has greatly improved our lives with the increase in the type and number of sensor devices, the amount of data to be processed is also increasing (Taonameso et al. 2019; Fang et al. 2020b, 2020c). How to deal with data efficiently and accurately has become the focus of people’s research. As a method of data fusion, neural networks can abstract human brain neurons from the perspective of information processing and establish models, and compose different networks according to different connections which can be used for analysis and prediction. However, the neural network also has some shortcomings, such as falling into local optimization and poor generalization ability.
As one of the most widely used neural network structures, back propagation neural network (BPNN) is widely used in different fields. However, it is often unable to achieve global convergence and falls into local minima when solving complex problems, resulting in an invalid learning process. Additionally, the learning algorithm converges slowly, especially near the target (Han & Huang 2019). Therefore, in practical applications, the bio-inspired algorithm is often used to optimize its model structure. For example, the particle swarm optimization (PSO) algorithm is used to optimize BPNN to overcome the sensitivity and error fluctuation of the initial value of gradient descent method, and obtain the global optimal initial parameters which make the neural networks converge quickly (Wu et al. 2018). These algorithms also include fruit fly optimization algorithm (FOA) (Wu et al. 2019), genetic algorithm (GA) (Li et al. 2018), as well as mind evolutionary algorithm (MEA) (Wang et al. 2018).

As a newly proposed bio-inspired algorithm, the seagull optimization algorithm (SOA) has been proven to achieve better performance than some traditional algorithms (Dhiman & Kumar 2019). Therefore, in this paper, SOA was used to optimize the structure of BPNN to obtain an improved ammonia nitrogen prediction model. Then, considering the shortcomings of the SOA, an improved algorithm was proposed, and some benchmark functions were used to verify its performance. Finally, the improved SOA (ISOA) was applied to improve the BPNN to obtain a better prediction model. The simulation results verified that the performance of the model is better than that of the traditional model and the model using PSO to optimize BPNN. In other words, the proposed algorithm can be applied to a more complex water quality environment for water quality detection.

Mathematical models of predator migration and attack are discussed. During the migration, the algorithm simulated how a group of gulls moved from one location to another. A seagull must meet the conditions in Equations (1)–(5).

To avoid collisions between adjacent search agents, we use an additional variable $A$, to calculate the new search agent location.

$$\hat{C}_i = A \times \hat{P}_s(x)$$  (1)

where $\hat{C}_i$ represents the position of the search agent which does not collide with other search agents, $\hat{P}_s$ represents the current position of the search agent, $x$ indicates the current iteration, and $A$ represents the movement behaviour of the search agent in a given search space.

$$A = f_c - (x \times (f_c/\text{Max}_{\text{iteration}}))$$  (2)

where $f_c$ is introduced to control the frequency of employing variable $A$, which decreases linearly from $f_c$ to 0. After avoiding collisions between neighbours, the search agents move towards the direction of the best neighbour.

$$\tilde{M}_s = B \times (\hat{P}_{bs}(x) - \hat{P}_s(x))$$  (3)

where $\tilde{M}_s$ represents the positions of the search agent $\hat{P}_s$ towards the best-fit search agent $\hat{P}_{bs}$. The behaviour of $B$ is randomized and is responsible for balancing between exploration and exploitation properly. $B$ is calculated as:

$$B = 2 \times A^2 \times rd$$  (4)

where $rd$ is a random number in the range of $[0, 1]$.

Lastly, the search agent can update its position with respect to the best search agent by:

$$\tilde{D}_s = |\tilde{C}_i + \tilde{M}_s|$$  (5)

where $\tilde{D}_s$ represents the distance between the search agent and best-fit search agent.

This development is designed to take advantage of the history and experience of the search process. When
attacking prey, the spiral action takes place in the air. This behaviour in the x, y and z planes are represented as follows:

\[ x' = r \times \cos (k) \]  
\[ y' = r \times \sin (k) \]  
\[ z' = r \times k \]  
\[ r = u \times e^{kv} \]

where \( r \) is the radius of each turn of the spiral, \( k \) is a random number in the range \( 0 \leq k \leq 2\pi \), \( u \) and \( v \) are constants to define the spiral shape, and \( e \) is the base of the natural logarithm. The updated position of the search agent is calculated using Equation (6)–(9), and the positions of the other search agents can be calculated using Equation (10):

\[ \bar{P}_s (x) = (\bar{D}_s \times x' \times y' \times z') + \bar{P}_{bs} (x) \]  

where \( \bar{P}_s \) saves the best solution and updates the position of other search agents.

SOA is summarized in Table 1 (Dhiman & Kumar 2019).

### ISOA

Due to the randomness of factor \( B \), the traditional SOA has a poor optimization effect and is prone to fall into local optimization. Therefore, an improved factor \( B \) is expressed as follows:

\[ B = 2 \times A^2 \times n d \times (1 - x / \text{Max}_\text{iteration}) \]  

To solve the problem of weak development ability of the algorithm in the later stage, chaos theory (Wang et al. 2014; Jia et al. 2019; Yue et al. 2019; Zhang et al. 2019a) is adopted to increase the diversity of particles in the later stage and enhance its search ability. The logistic map is introduced, and the basic equation is:

\[ Cx_{i}^{k+1} = \mu Cx_{i}^{k} (1 - Cx_{i}^{k}) \]  

where \( Cx_{i}^{k+1} \) represents the \( i \)th \((i = 1, 2, 3, \cdots, n)\) iteration of the \( k \)th chaotic variable, and \( \mu \) is generally 4. The transformation between the chaotic variable and the original variable is as follows:

\[ Cx_{i}^{k} = (x_{i}^{k} - x_{L,i}) / (x_{U,i} - x_{L,i}) \]  
\[ x_{i}^{k} = x_{L,i} + Cx_{i}^{k} (x_{U,i} - x_{L,i}) \]  

| Table 1 | SOA procedure |
|---------|---------------|
| **Seagull optimization algorithm** |
| Input: seagull population \( \bar{P}_s \) |
| Output: optimal search agent \( \bar{P}_{bs} \) |
| 1: procedure SOA |
| 2: Initialize the parameters \( A, B, \text{and Max}_\text{iteration} \) |
| 3: Set \( f_L \leftarrow 2 \) |
| 4: Set \( u \leftarrow 1 \) |
| 5: Set \( v \leftarrow 1 \) |
| 6: while \((x < \text{Max}_\text{iteration})\) do |
| 7: \( \bar{P}_{bs} \leftarrow \text{ComputeFitness}(\bar{P}_s) \) |
| 8: \( rd \leftarrow \text{Rand}(0, 1) \) |
| 9: \( k \leftarrow \text{Rand}(0, 2\pi) \) |
| 10: \( r \leftarrow u \times e^{kv} \) |
| 11: Calculate the distance \( \bar{D}_s \) using Equation (11) |
| 12: \( P \leftarrow x' \times y' \times z' \) |
| 13: \( \bar{P}_s (x) \leftarrow (\bar{D}_s \times P) + \bar{P}_{bs} (x) \) |
| 14: \( x \leftarrow x + 1 \) |
| 15: end while |
| 16: return \( \bar{P}_{bs} \) |
| 17: end procedure |
| 1: procedure ComputeFitness(\( \bar{P}_s \)) |
| 2: for \( i \leftarrow 1 \) to \( n \) do |
| 3: \( \text{FIT}_i[i] \leftarrow \text{FitnessFunction}(\bar{P}_s (\bar{T}, i)) \) |
| 4: end for |
| 5: \( \text{FIT}_{\text{best}} \leftarrow \text{BEST}(\text{FIT}_i[i]) \) |
| 6: return \( \text{FIT}_{\text{best}} \) |
| 1: Procedure \( \text{BEST}(\text{FIT}_i[i]) \) |
| 2: \( \text{Best} \leftarrow \text{FIT}_i[0] \) |
| 3: for \( i \leftarrow 1 \) to \( n \) do |
| 4: if \( \text{FIT}_i[i] < \text{Best} \) then |
| 5: \( \text{Best} \leftarrow \text{FIT}_i[i] \) |
| 6: end if |
| 7: end for |
| 8: return \( \text{Best} \) |
| 9: end procedure |
where \( x_{L,i} \) and \( x_{U,i} \) are the search upper and lower bounds of the \( i \)th dimension variable, respectively, and \( y_i^k \) is the value obtained by transforming the \( i \)th chaotic variable \( Cx_i^k \) into the optimization variable after chaotic mapping.

The idea of the improved SOA is to use chaotic after iteration to conduct chaotic iteration on the location of a seagull with the best fitness and increase its diversity. First, the original variables are mapped to chaotic variables using Equation (13), and then transformed using Equation (12). Finally, the original spatial position value is returned by Equation (14). If the position after the chaos is better than before the chaos, save it; otherwise, save the position before the chaos.

### Functions test

In this section, the improved algorithm is tested on some unimodal and multimodal benchmark functions (Dhiman & Kumar 2019). The information of these functions is shown in Table 2.

Here, the PSO, the traditional SOA and the ISOA are used for comparison. The parameter settings of each algorithm are shown in Table 3, and the \( \text{Max}_{\text{iterations}} \) are 500, the number of seagulls is 100 and the dimension of seagulls is 30.

The optimization results of the improved algorithm on different functions are shown in Figures 1–4.

Table 4 compares the optimization results of each function.

As can be seen from the figures and the tables, the optimization effect of the SOA is better than that of the PSO, and the convergence speed is faster. When

### Table 2 | Information of benchmark functions

| Type       | Name          | Expression                  | Domain of definition | Global optimum | Optimal value |
|------------|---------------|----------------------------|----------------------|----------------|---------------|
| Unimodal   | Sphere        | \( F_1(x) = \sum_{i=1}^{n} x_i^2 \) | \([-100, 100]^n\)   | \(0^n\)       | 0             |
|            | Schwefel’s 2.22 | \( F_2(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i| \) | \([-10, 10]^n\) | \(0^n\)       | 0             |
|            | Schwefel’s 1.2 | \( F_3(x) = \sum_{i=1}^{n} \left( \sum_{j=1}^{i} x_j \right) \) | \([-100, 100]^n\) | \(0^n\)       | 0             |
|            | Schwefel’s 2.21 | \( F_4(x) = \max_i \{|x_i|, 1 \leq i \leq n\} \) | \([-100, 100]^n\) | \(0^n\)       | 0             |
|            | Noise         | \( F_5(x) = \sum_{i=1}^{n} x_i^4 + \text{random}[0, 1) \) | \([-1.28, 1.28]^n\) | \(0^n\)       | 0             |
| Multimodal | Rastrigin     | \( F_6(x) = \sum_{i=1}^{n} [x_i^2 - 10 \cos(2\pi x_i) + 10] \) | \([-5.12, 5.12]^n\) | \(0^n\)       | 0             |
|            | Ackley        | \( F_7(x) = -20 \exp(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + e \) | \([-32, 32]^n\) | \(0^n\)       | 0             |
|            | Griewank      | \( F_8(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \frac{n}{n} \cos \left( \frac{x_i}{\sqrt{n}} \right) + 1 \) | \([-600, 600]^n\) | \(0^n\)       | 0             |

### Table 3 | Parameter settings of each algorithm

| Algorithm | Parameter | Value |
|-----------|-----------|-------|
| ISOA      | \( u \)   | 1     |
|           | \( v \)   | 0.1   |
|           | \( f_c \) | 2     |
| SOA       | \( u \)   | 1     |
|           | \( v \)   | 0.1   |
|           | \( f_c \) | 2     |
| PSO       | \( c_1 \) | 1.49445 |
|           | \( c_2 \) | 1.49445 |
|           | \( \omega \) | 0.5  |
the number of iterations is about 100 times, the optimization effect of the PSO is better than that of the PSO for 500 iterations. Furthermore, the improved ISOA has better optimization results, faster convergence speed and a better effect than the traditional SOA algorithm.
There are many kinds of neural networks, among which BPNN is one of the most widely used. It has the advantages of simple structure, self-learning, self-organization, self-adaptation, fast training speed, local approximation and global convergence. It is generally composed of the input layer, the hidden layer and the output layer. It has been widely used in the field of prediction. The main idea of BPNN is to divide learning into forwarding the propagation of signal and back propagation of error. Specifically, in the learning process, the sample input is input through the input layer, and then transferred to the output layer through the operation of hidden layer neurons. Then, the error between the actual data and the predicted data of the output layer is calculated, and the error is put into the stage of back propagation. In the process of back propagation, the connection weights between each layer of neurons are constantly adjusted based on the gradient descent strategy until the deviation between the final predicted value and the actual value is minimized (Yang & Wang 2018; You et al. 2018; Zhang et al. 2019b). The model of BPNN is shown in Figure 5.

Suppose there are $d$ neurons in the input layer of BPNN model, a hidden layer with $g$ neurons and $l$ neurons in the output layer. The input is $X_i = [X_{i1}, X_{i2}, \cdots, X_{iN}]^T$, where $i = 1, 2, \cdots, d$. $N$ represents the number of samples and the output is $Y_j$, where $j = 1, 2, \cdots, l$. The connection weight of $i$th neuron in the input layer to $h$th neuron in the hidden layer is calculated as $W_{ih}$. The output of the hidden layer is $V_i = f(W_{ih}X_i + b_i)$, where $f$ is the activation function and $b_i$ is the bias.

### Table 4 | Optimization results of each function

| Function /algorithm | ISOA   | SOA    | PSO   |
|---------------------|--------|--------|-------|
| F1                  | $1.867 \times 10^{-33}$ | $2.888 \times 10^{-18}$ | 0.01132 |
| F2                  | $3.964 \times 10^{-15}$ | $6.757 \times 10^{-7}$ | 0.618  |
| F3                  | $9.142 \times 10^{-16}$ | $2.076 \times 10^{-15}$ | 0.2642 |
| F4                  | $5.201 \times 10^{-17}$ | $1.576 \times 10^{-8}$ | 0.1465 |
| F5                  | 0.000192 | 0.005356 | 0.1678 |
| F6                  | 0      | 0      | 5.386  |
| F7                  | $6.839 \times 10^{-14}$ | $5.152 \times 10^{-11}$ | 0.1454 |
| F8                  | 0      | 0      | 0.001802 |
layer is $V_{ih}$, and the threshold value of $h$th neuron in the hidden layer is $\theta_h$. The connection weight of $h$th neuron in the hidden layer and $j$th neuron in the output layer is $W_{hj}$ and the threshold is $\theta_j$. The input of the $h$th neuron in the hidden layer is $A_h$ and $C_j$ is the input of the $j$th neuron in the output layer.

Assuming the actual value of $j$th neuron in the output layer is $R_j$, the total error of network output is:

$$E = \frac{1}{2} \sum_{j=1}^{l} (R_j - Y_j)^2$$

(15)

BPNN adopts the gradient descent method to adjust the weights and thresholds of the network to obtain better output. The following equation can be obtained by expanding the error to the hidden layer and then to the input layer:

$$E = \frac{1}{2} \sum_{j=1}^{l} (R_j - f(C_j - \theta_j))^2$$

$$= \frac{1}{2} \sum_{j=1}^{l} (R_j - f(\sum_{h=1}^{g} W_{hj}(A_h - \theta_h) - \theta_j))^2$$

(16)

where $f(\cdot)$ is represented as an S function. It can be seen from the above that error $E$ is related to the thresholds and weights. Therefore, the final error can be changed indirectly by changing the weights and thresholds.

The related equations of the weights and thresholds are as follows:

$$W_{hi}(k + 1) = W_{hi}(k) + \Delta W_{hi}$$

(17)

$$\theta_j(k + 1) = \theta_j(k) + \Delta \theta_j$$

(18)

$$V_{ih}(k + 1) = V_{ih}(k) + \Delta V_{ih}$$

(19)

$$\theta_h(k + 1) = \theta_h(k) + \Delta \theta_h$$

(20)

where $k = 1, 2, \ldots, N - 1$ is the number of samples.

ISOA–BP

The main idea of the ISOA–BP hybrid programming is to optimize the weights and thresholds of the back propagation network based on the ISOA. The main steps are as follows.

Step 1: Set the parameters of ISOA, including the number of seagulls. The weights and thresholds of the BPNN that need to be optimized are encoded as the initial seagull population.

Step 2: Initialize the position of seagulls and use Equation (7) to change their position so as to avoid collisions.

Step 3: Calculate the fitness of all seagulls at present, and find the best one as the best seagull in this iteration.

Step 4: Update the position of each seagull according to Equations (1)–(11).

Step 5: Chaotic algorithm (Equations (12)–(14)): logistic mapping is used to map the individual extremum of particles to $(0, 1)$ for chaotic iteration. After the iteration, the inverse mapping is returned to the spatial range of the original solution. Calculate the fitness value of the current solution, and output the new solution when the new solution is better than the old one.

Step 6: To determine whether the iteration times or required accuracy are reached, output the final position as the optimal seagull position; otherwise, return to step 3.

Step 7: Decode the optimal output into the initial weights and thresholds of BPNN, and train the neural network until it meets the requirements.

Data pre-processing

Data from May to August in 2016 for a river in Qinghai province were collected once a day, including water temperature ($^\circ$C), pH, dissolved oxygen (mg/L), conductivity (µs/cm), turbidity (nephelometric turbidity units, NTU), permanganate index (mg/L) and ammonia nitrogen (mg/L). A total of 123 groups of data were collected. The first 100 groups of data were used to train the network and the last 23 groups of data were used to verify the network performance.

Before using the sample data for training, it is necessary to process the sample data and pre-process the missing or wrong data. After that, the following equation was used for
normalization:

\[ y_i = (0.8 - 0.2) \times \left( \frac{x_i - x_{i\min}}{x_{i\max} - x_{i\min}} \right) + 0.2 \]  

(21)

where \( x_i \) and \( y_i \) are the data before and after normalization, and \( x_{i\max} \) and \( x_{i\min} \) are the maximum and minimum values of data before normalization.

Simulation results

In this subsection, the back propagation (BP), PSO–BP, SOA–BP and ISOA–BP models are compared to verify the performance of BPNN optimized by the ISOA. The parameters of each model are shown in Table 5, and the \( Max_{\text{iterations}} \) are 1,000, train function is trainrp, number of seagulls is 100, iteration times of seagulls is 30 and the structure of BP is 6-7-1.

The convergence comparison of different models is shown in Figure 6. As can be seen from Figure 6, the proposed ISOA–BP model has a smaller convergence value, which is better than that of the BP, PSO–BP and SOA–BP models.

As can be seen from Figure 7, the proposed ISOA–BP model has a faster convergence speed, which converges to the optimal value faster, and its convergence value is better than that of the BP, PSO–BP and SOA–BP models. In other words, the predicted value of the proposed algorithm is closer to the actual value. The simulation results of 23 groups of validation data from different models are shown in Figure 7.

As can be seen from Figure 8, the average error of the predicted value of the proposed algorithm is the smallest,

| Algorithm | Parameter | Value |
|-----------|-----------|-------|
| ISOA      | \( u \)   | 1     |
|           | \( v \)   | 0.1   |
|           | \( f_c \) | 2     |
| SOA       | \( u \)   | 1     |
|           | \( v \)   | 0.1   |
|           | \( f_c \) | 2     |
| PSO       | \( c_1 \) | 1.49445 |
|           | \( c_2 \) | 1.49445 |
|           | \( \omega \) | 0.5 |
which indicates that the predicted value is closer to the actual value. The predicted value of the proposed ISOA-BP prediction model is closer to the actual value, which shows more accurate prediction accuracy than the traditional BP, PSO-BP or SOA-BP prediction models.

In Figure 8 shows the error comparison of the four models, and the value is the absolute value of the error. It can be seen from the figure that the proposed model has the lowest error value for the verification samples, the lowest average error and the highest prediction accuracy.

The following two evaluation methods are used to evaluate the prediction accuracy of different models (Yu & Bai 2018).

1: Root Mean Square Error (RMSE)

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - p_i)^2}
\]  

(22)

2: Nash–Sutcliffe Efficiency (NS)

\[
NS = 1 - \frac{\sum_{i=1}^{n} (x_i - p_i)^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2}
\]  

(23)

where \(x_i\) represent the actual output of the test data, \(p_i\) is the predicted output of the test data, \(\bar{x}\) is the average of the actual output of the test data, and \(n\) is the number of samples. For the first evaluation method, the lower the value, the better the effect. For the second evaluation, the closer the value to 1, the better.

The results of the above two evaluation algorithms calculations are shown in Table 6.

As can be seen from Table 6, compared with the traditional single neural network prediction model and PSO or SOA model, the proposed improved optimization model, namely the ISOA model, has a higher prediction accuracy.

**CONCLUSIONS**

Since the traditional BPNN is easily limited to local optimization, which leads to the low accuracy of ammonia nitrogen prediction, in this paper, the SOA is adopted with strong optimization performance to optimize the weights and thresholds of BPNN. Because of the shortcomings of the SOA, this paper proposes an improved algorithm to optimize BPNN by using chaos. The simulation results show that the prediction accuracy of the proposed model is higher than that of the traditional BPNN, PSO and SOA models. The prediction accuracy of the new model is higher and the effect is better, which can be applied to predict ammonia nitrogen in more complex water environments.

**ACKNOWLEDGEMENTS**

This work is supported by the National Natural Science Foundation of Qinghai Province, China (No. 2020-ZJ-724).

**DATA AVAILABILITY STATEMENT**

All relevant data are included in the paper or its Supplementary Information.

**REFERENCES**

Dhiman, G. & Kumar, V. 2019 Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems* 165, 169–196.

Fang, W., Zhang, C., Shi, Z., Zhao, Q. & Shan, L. 2016 BTRES: beta-based trust and reputation evaluation system for wireless sensor networks. *Journal of Network and Computer Applications* 59 (1), 88–94.

Fang, W., Cui, N., Chen, W., Zhang, W. & Chen, Y. 2020a A trust-based security system for data collecting in smart city. *IEEE
Transactions on Industrial Informatics. doi: 10.1109/TII.2020.3006137 (Early Access).
Fang, W., Zhang, W., Chen, W., Liu, Y. & Tang, C. 2020b TMSRS: trust management-based secure routing scheme in industrial wireless sensor network with fog computing. Wireless Networks 26 (5), 3169–3182.
Fang, W., Zhang, W., Chen, W., Pan, T., Ni, Y. & Yang, Y. 2020c Trust-based attack and defense in wireless sensor networks: a survey. Wireless Communications and Mobile Computing 2020, 20, Article ID 2643546.
Han, Z. Y. & Huang, X. G. 2019 GA-BP in thermal fatigue failure prediction of microelectronic chips. Electronics 8 (542), 1–14.
Jia, H. M., Xing, Z. K. & Song, W. L. 2019 A new hybrid seagull optimization algorithm for feature selection. IEEE Access 7, 49614–49631.
Khan, P., Zhu, W., Huang, F., Gao, W. & Khan, N. A. 2020 Micro-nanobubble technology and water-related application. Water Supply 20 (6), 2021–2035.
Li, X., Sha, J. & Wang, Z. L. 2017 Chlorophyll-A prediction of lakes with different water quality patterns in China based on hybrid neural networks. Water 9 (2), 1–13.
Taonameso, S., Mudau, L. S., Traoré, A. N. & Potgieter, N. 2019 Borehole water: a potential health risk to rural communities in South Africa. Water Supply 19 (1), 128–156.
Wang, G. G., Guo, L. H., Gandomi, A. H., Hao, G.-S. & Wang, H. 2014 Chaotic krill herd algorithm. Information Sciences 274, 17–34.
Wang, W. X., Tang, R. C., Li, C., Liu, P. & Luo, L. 2018 A BP neural network model optimized by mind evolutionary algorithm for predicting the ocean wave heights. Ocean Engineering 162, 98–107.

Wu, J., Li, Z. B., Zhu, L. & Li, G. Y. 2018 Optimized BP neural network for dissolved oxygen prediction. IFAC Papers on Line 51 (17), 596–601.
Wu, L., Yang, Y. W., Maheshwari, M. & Li, N. 2019 Parameter optimization for FPSO design using an improved FOA and IFOA BP neural network. Ocean Engineering 175, 50–61.
Yang, Z. S. & Wang, J. 2018 A hybrid forecasting approach applied in wind speed forecasting based on a data processing strategy and an optimized artificial intelligence algorithm. Energy 160, 87–100.
You, L. J., Tan, Q. G., Kang, Y. L., Xu, C. & Lin, C. 2018 Reconstruction and prediction of capillary pressure curve based on particle swarm optimization-back propagation neural network method. Petroleum 4, 268–280.
Yu, T. T. & Bai, Y. 2018 A comparative study of extreme learning machine, least squares support vector machine, back propagation neural network for outlet total phosphorus prediction. In 2018 Prognostics and System Health Management Conference (PHM-Chongqing), Chongqing, China, pp. 717–722.
Yue, Y. G., Cao, L., Hu, J., Cai, S. & Hang, B. 2019 A novel hybrid location algorithm based on chaotic particle swarm optimization for mobile position estimation. IEEE Access 7, 58541–58552.
Zhang, Z. X., Yang, R. N. & Li, H. Y. 2019a Antlionoptimizer algorithm based on chaos search and its application. Journal of Systems Engineering and Electronics 30, 352–365.
Zhang, W. B., Han, G. J., Wang, J. & Liu, Y. 2019b A BP neural network prediction model based on dynamic cuckoo search optimization algorithm for industrial equipment fault prediction. IEEE Access 6, 11736–11746.

First received 19 November 2020; accepted in revised form 1 February 2021. Available online 15 February 2021