Electronic Research Archive

ERA, 32(11): 6364–6378.
DOI: 10.3934/era.2024296
Received: 15 August 2024
Revised: 12 November 2024
Accepted: 15 November 2024
Published: 22 November 2024

https://www.aimspress.com/journal/ERA

*Research article*

# Load forecasting of microgrid based on an adaptive cuckoo search optimization improved neural network

**Liping Fan[1,2,*] and Pengju Yang[1,2]**

[1] Key Laboratory of Collaborative Control and Optimization Technology of Liaoning province, Shenyang 110142, China
[2] College of Information Engineering, Shenyang University of Chemical Technology, Shenyang 110142, China

* **Correspondence:** Email: fanliping@syuct.edu.cn; Tel: +86-24-89386011; Fax: +86-24-89386011.

**Abstract:** Load forecasting is an important part of microgrid control and operation. To improve the accuracy and reliability of load forecasting in microgrid, a load forecasting method based on an adaptive cuckoo search optimization improved neural network (ICS-BP) was proposed. First, a load forecasting model in microgrid based on a neural network was designed. Then, a novel adaptive step adjustment strategy was proposed for cuckoo search optimization, and an adaptive position update law based on loss fluctuation was designed. Finally, the weights and biases of the forecasting model were optimized by the improved cuckoo search algorithm. The results showed that the BP network optimized by the improved cuckoo search optimization enhanced the global search ability, avoided the local optima, quickened the convergence speed, and presented excellent performance in load forecasting. The mean absolute percentage error (MAPE) of the ICS-BP forecasting model was 1.13%, which was very close to an ideal prediction model, and was 52.3, 32.8, and 42.3% lower than that of conventional BP, cuckoo search improved BP, and particle swarm optimization improved BP, respectively, and the root mean square error (RMSE), mean absolute error (MAE), and mean square error (MSE) of ICS-BP were reduced by 75.6, 70.6, and 94.0%, respectively, compared to conventional BP. The proposed load forecasting method significantly improved the forecasting accuracy and reliability, and can effectively realize the load forecasting of microgrid.

**Keywords:** microgrid; BP neural network; cuckoo search optimization; load forecasting; adaptive

## 1. Introduction

Load forecasting is an important component of microgrid control and operation. It is also the basis for optimizing distributed energy sources such as energy storage and photovoltaics. It directly affects the stability and integrity of the entire power system [1,2]. Accurate load forecasting has become an important tool for energy market competition [3–5].

There are various techniques and methods for load forecasting, such as time series analysis, modal decomposition, Fourier analysis, etc. [6]. There is serious dynamic uncertainty and nonlinearity in electrical network environment, and the high degree of uncertainty caused by different consumers in load consumption patterns and the highly inconsistent energy consumption patterns of the same consumer need to be considered in load forecasting [7], which makes load forecasting in microgrid still a complex problem [8]. Traditional data-driven prediction methods have two difficult problems to solve: First, they cannot fully utilize the correlation between measurement data and load data. Second, they do not support extraction patterns independent of fixed pattern lengths [9]. Therefore, there are still some significant challenges in load forecasting of microgrid.

In recent years, neural networks have gradually been proven to be an effective prediction method [10]. However, the application of neural networks relies on the strategy called empirical risk minimization [11]. In this case, the total cost representing the error between the actual value and the predicted value usually has different trends in the training and testing samples, even if the neural network performs well on the training samples, it may still lead to poor performance in new situations. So, some advanced machine learning algorithms such as artificial rabbit optimization, swarm intelligence optimization, balancing composite motion optimization, gradient boosting decision trees algorithms, etc. are often used to improve these conventional prediction methods [12–15].

An improved cuckoo search algorithm was designed to optimize the neural network in this paper. By improving the cuckoo search algorithm, the optimal neural network parameter configuration can be automatically searched, thereby achieving the minimization of load prediction errors in microgrids.

## 2. Materials and methods

### *2.1. Design of BP neural network*

The back propagation (BP) neural network is a multilayer feedforward network trained according to the error BP algorithm, which is widely used in nonlinear approximation problems [16]. The structure of BP the neural network is shown in Figure 1.

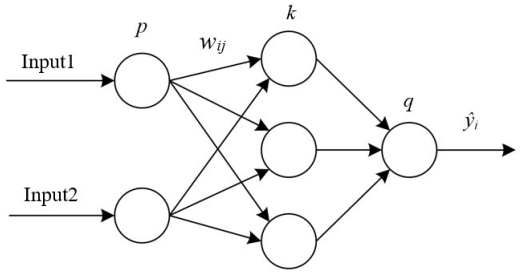


**Figure 1.** BP neural network structure.

A BP neural network typically consists of three layers: input layer, hidden layer, and output layer [17]. The number of hidden layer nodes in a BP neural network has a significant impact on its prediction accuracy. When the number of nodes is small, the training frequency increases and the training accuracy is not high; when there are many nodes, the training time increases and is easy to overfit. Usually, the number $r$ of hidden layer units can be designed based on experience as [18]:

$$r = d + \sqrt{p+q} \tag{1}$$

where, $p$ is the number of input layer units; $q$ is the number of output layer units; and $d$ is an integer ranging from 1 to 10.

The main steps of the learning process of the BP neural network are as follows:

Step 1: Initialize the weights and biases of the neural network.

Step 2: Input the learning sample to the network, and obtain the network output by weighting and mapping. The weighted input design for the $i^{\text{th}}$ neuron is:

$$z_i^{(l)} = \sum_{j=1}^{n^{(l-1)}} (w_{ij}^{(l)} a_j^{(l-1)}) + b_i^{(l)} \tag{2}$$

in which $n^{(l\text{-}1)}$ is the number of neurons in layer $l$-1, $a_j^{(l\text{-}1)}$ is the output of the $j^{\text{th}}$ neuron in the previous layer, $w_{ij}^{(l)}$ is the weight connecting the $j^{\text{th}}$ neuron and the $i^{\text{th}}$ neuron, and $b_i^{(l)}$ is the bias term in layer $l$.

The activation function is represented as:

$$a_i{}^{(l)} = \sigma(z_i{}^{(l)}) \tag{3}$$

Step 3: Calculate the loss based on the output of BP network and the real labels, that is:

$$J = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 \tag{4}$$

where $J$ is the loss function, $m$ is the number of samples, $y_i$ is the true label, and $\hat{y}_i$ represents the predicted value of the neural network.

Step 4: Calculate the gradient of the loss for each weight and bias in the network. This process propagates back from the output layer to the input layer, using the chain rule for gradient calculation.

The error term of the output layer is calculated as follows:

$$\delta_i{}^{(o)} = \nabla a_i{}^{(o)} J \odot \sigma(z_i{}^{(o)}) \tag{5}$$

where $\delta_i^{(o)}$ is the error term of the $i^{\text{th}}$ neuron in the output layer, $\nabla a_i^{(o)} J$ represents the gradient of the loss function with respect to the output, $\odot$ represents element by element multiplication, $\sigma'(z_i{}^{(o)})$ represents the derivative of the activation function of the output layer, and $z_i^{(o)}$ represents the weighted input of the neurons in the output layer.

The error term of the hidden layer is calculated layer by layer according to the following formula, that is:

$$\delta^{(l)} = ((W^{(l+1)})^T \delta^{(l+1)}) \odot \sigma'(z^{(l)}) \tag{6}$$

where $l$ is the layer index, $W^{(l+1)}$ is the weight matrix of the next layer, and T stands for matrix transpose.

Step 5: Update the weights and biases of the network according to the gradient information to

reduce the difference between the predicted results and the true values. The weight update law of the output layer is:

$$w_{ij}^{(o)}(k+1) = w_{ij}^{(o)}(k) - \alpha\delta_j^{(o)}(k)z_i^{(o-1)}(k) \tag{7}$$

The weight update law of the hidden layer is:

$$w_{ij}^{(l)}(k+1) = w_{ij}^{(l)}(k) - \alpha\delta_j^{(l)}(k)z_i^{(l-1)}(k) \tag{8}$$

where $w_{ij}^{(o)}$ is the mapping weight from the $i^{\text{th}}$ neuron in the hidden layer to the $j^{\text{th}}$ neuron in the output layer, $w_{ij}^{(l)}$ is the mapping weight from the $i^{\text{th}}$ neuron of layer $l$ to the $j$ neuron of layer $l + 1$, $\alpha$ is the learning law, and $k$ is the number of iterations.

The deviation update law of both the output layer and the hidden layer is designed as:

$$b_i^{(l)}(k+1) = b_i^{(l)}(k) - \alpha\delta_i^{(l)}(k) \tag{9}$$

Step 6: Repeat Steps 2 to 5 until the loss converges or the predetermined number of iterations is reached.

There are a lot of parameters in a neural network, including the number of layers, the number of neurons, the weight, and so on. These parameters have a significant impact on the prediction results. To achieve satisfactory prediction results, it is necessary to optimize these parameters appropriately, which makes the training of neural networks complex and time-consuming.

### *2.2. Improvement of cuckoo search algorithm*

The cuckoo search (CS) algorithm is a metaheuristic optimization algorithm proposed based on simulating the parasitic breeding behavior of cuckoo [19]. In this algorithm, a set of nests is considered as a population, and the eggs in each nest are considered as a feasible solution to the optimization problem [20]. During breeding, cuckoos randomly find parasitic nests through Levy flights and lay their own eggs (candidate solution) in the nests, and the parasitic nest with the best egg (highest fitness value) is preserved for the next generation. In order to obtain the best parasitic nest, a nest replacement mechanism is introduced to the CS algorithm to preserves high-quality solutions in each iteration and randomly replace poor solutions. After multiple iterations, the optimal solution will be finally found in a vast solution space [21]. Compared with other algorithms, the CS algorithm has advantages such as wide applicability, accurate optimization results, and short calculation time [22,23], providing convenience for achieving good global search. At present, the CS algorithm has been applied to scientific and engineering problems such as image processing, feature selection, and fault diagnosis [24–26]. Some variant cuckoo algorithms based on parameter adaptation, population dividation, and other strategies have also emerged, further enhancing the robustness and adaptability of the algorithm in global search tasks and making some progress in the application of complex optimization problems [27]. However, the CS algorithm has certain limitations in terms of search ability, parameter sensitivity, and high-dimensional problems, as well as lacks strict convergence guarantee.

In the conventional CS algorithm, the objective function only utilizes the distance information

between anchor nodes and unknown nodes, while the distance information between unknown nodes is not utilized, which to some extent limits its localization accuracy [28]. On the other hand, the random step size in the cuckoo algorithm can also lead to problems such as algorithm instability, slow convergence speed, and difficulty in parameter adjustment [29,30]. To improve the search performance, an improved CS optimization algorithm that can adjust search direction and search step size in real-time is proposed in this paper. During the search process, the improved CS algorithm automatically calculates the difference between the current position of each nest and the current optimal nest position, and the future search direction is then determined based on this position difference. Meanwhile, the step size adjustment law and position update law based on a time-varying loss function are constructed to realize the adaptive update of the nest positions. When the search performance is good, the step size will automatically decrease to explore the solution space more finely, while when the search performance is poor, the search step size will automatically increase to expand the search range. Through such an adaptive adjustment mechanism, the improved CS algorithm can flexibly adjust the search step size and direction in different stages, thereby ensuring higher search speed and accuracy. The improved CS algorithm only requires adjusting the step size and position update law in each iteration, without involving complex matrix operations or high-dimensional data processing, thus significantly reducing the training time required for the model and allowing for quick load forecasting.

Suppose the value of the loss function corresponding to the $k^{\text{th}}$ iteration is $J(k)$ and the iteration step is $S$, then the change in the loss function is expressed as:

$$\Delta J = J(k) - J(k-1) \tag{10}$$

The adjustment of step $S$ is designed as:

$$S(k+1) = S(k) \times \beta \tag{11}$$

where $\beta$ represents the adjustment coefficient. If $\Delta J > 0$, then select $\beta = 1.05$ to increase the step size; while if $\Delta J < 0$, then make $\beta = 0.95$ to reduce the step size.

If the position of the $i^{\text{th}}$ nest is $P_i$ and the best position of nests is $P_{\text{best}}$, then at the $k^{\text{th}}$ iteration, the position difference $d_c$ between the current nest and the best nest can be calculated as follows:

$$d_c(k+1) = P_{\text{best}}(k) - P_i(k) \tag{12}$$

So, the search direction during the $k+1$ iteration can be adjusted according to the following equation:

$$d(k+1) = d(k) + d_c(k+1) \tag{13}$$

where $d$ is the adjustment of the iterative search direction. Accordingly, the update law of nest position is designed as:

$$P_i(k+1) = P_i(k) + S(k+1) \times d(k+1) \tag{14}$$

Through the design of the step size adjustment law and position update law, the step size and search direction can be adjusted adaptively in the iterative search process, which is convenient to reach the optimal solution more accurately and quickly, and can also suppress the violent oscillation caused by random step size to the conventional CS process.

The main process of improving the CS algorithm is shown in Figure 2. The optimization process of the improved CS was described as follows:

Setp 1: Initialize parameters such as the number of nests, the maximum number of iterations, the

search step size, and the discovery probability.

Setp 2: Evaluate each nest and calculate its fitness value, and select the nest with the highest fitness as the current optimal solution. The highest fitness corresponds to the smallest loss function value.

Setp 3: The bird nest is updated according to the adaptive step adjustment and position update formula to generate a new generation of bird nest positions.

Setp 4: Determine whether the end condition is met. When the maximum number of iterations is reached, the calculation stops and the optimal solution is output. Otherwise, go back to Step 2 and continue the loop iteration.

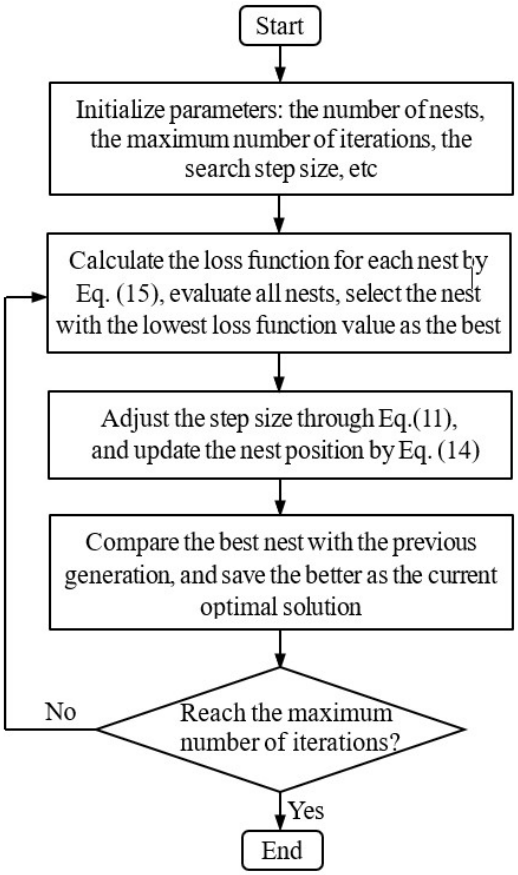


**Figure 2.** Flowchart of the improved cuckoo algorithm.

The improved CS algorithm can enhance the global search ability, improve the accuracy and speed of finding the global optimal solution, and avoid the problem of falling into the local optimal solution because of its self-adaptive ability of search step size.

### *2.3. Optimization of neural network based on improved CS*

In order to improve the accuracy of load forecasting, the improved CS algorithm was used to optimize the neural network model. A microgrid load forecasting algorithm based on the improved CS algorithm optimized BP neural network was designed. The specific implementation process is as follows:

Step 1: Create BP neural network prediction model.

Step 2: Set the number of nests.

Step 3: Set the maximum number of iterations.

Step 4: Set step parameters such as the initial step size, step decline rate, and minimum allowed step size.

Step 5: Randomly generate the initial locations of nests in the search space.

Step 6: Calculate the fitness of each nest.

Step 7: Find the nest with the best objective function value (that is, the smallest loss function value).

Step 8: Update the parameters of BP neural network. Take the best nest location found in the previous step as the current global best location, and use it to update the weight and deviation of the neural network.

Step 9: Update the location of each nest with step size adaptive adjustment and position adaptive update laws.

Step 10: Check the iterations. If the predetermined maximum number of iterations is reached, the iteration will be stopped. Otherwise, return to Step 6 and continue iterating.

To avoid overfitting in the model, the LASSO (least absolute shrinkage and selection operator) regularization was employed to enhance the generalization ability of the final prediction model [31]. By introducing the sum of the absolute values of the model parameters as a penalty term into the loss function, the loss function is transformed to:

$$J = \frac{1}{2m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{n}|w_{ij}| \tag{15}$$

where $\lambda$ is the hyper-parameter that controls the strength of the regularization, and $w_{ij}$ represents the weights of the corresponding layer in the BP neural network. After processing in this way, not only the generalization ability of the model is enhanced, but also the risk of overfitting the training data is reduced.

## 3. Results and discussion

### *3.1. Data acquisition and parameter settings*

Microgrid is a small power system composed of photovoltaic cell arrays, wind turbines, fuel cells and other distributed energy and energy storage systems, which can operate either grid-connected or in island mode [32,33]. The simplified structure of a certain microgrid is shown in Figure 3. The validation of the load forecasting scheme was carried out based on the load data of the microgrid from February 1, 2023 to May 10, 2023, and the sampling interval of sample data was 1 h. The data of the microgrid from February 1, 2023 to May 9, 2023 was selected as the training set for the model, with the 24-hour load data on May 10, 2023 as the test set. This dataset covers operational data under different conditions such as seasons and weather changes, thus reflecting the impact of different operational conditions on load characteristics, which are crucial for the accuracy of load forecasting. Meanwhile, the data samples used in the prediction experiment are sufficient and contain enough data to support effective model training, ensuring the reliability of the prediction results and the generalizability of the model.

During operation, the initial positions of the 30 nests were randomly generated within the search space. The input of the BP neural network was time and load, i.e., inputsize = 2; and the load of the

microgrid was the predicted value, i.e., outputsize = 1. Based on the number of inputs and outputs of the BP network, the number of hidden layer nodes was calculated according to Eq (1) and the number of hidden layer nodes was 3, that is, hidenLayerSizee = 3. The main parameter settings for the CS algorithm were: the nest number was nests = 20; the maximum iteration number was maxGeneration = 200; the initial search step size was initialAlpha = 0.1; the step size decrease rate was alphaDecayRate = 0.05; and the minimum allowable step size was minAlpha = 0.01.

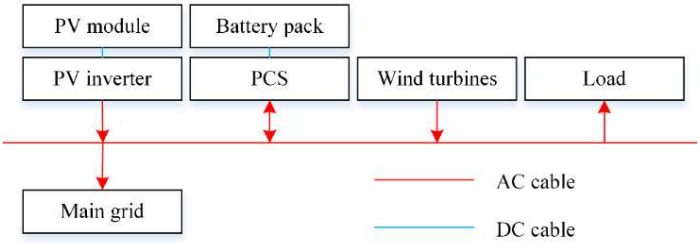


**Figure 3.** Simple structure diagram of a microgrid.

Due to the large difference between load data, the sampling data was processed by Min-Max normalization. Normalization ensures a fair contribution of each feature to the model. By standardizing the dataset, differences between features can be eliminated, the convergence speed of optimization algorithms can be accelerated, model performance and prediction accuracy can be improved, and the instability of numerical calculations can be reduced. The Min-Max normalization is calculated by:

$$x_s = \frac{x - min(x)}{max(x) - min(x)} \tag{16}$$

where $x$ is the original feature data and $x_s$ is the data after normalization.

### *3.2. Evaluation indexes*

The evaluation indexes commonly used in load forecasting of microgrid include mean absolute percentage error (MAPE), mean absolute error (MAE), root mean square error (RMSE) and mean square error (MSE), which can be calculated by:

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{y_i} \times 100\% \tag{17}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{18}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{19}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{20}$$

where $N$ is the total number of samples, $y_i$ is the true value, and $\hat{y}_i$ is the predictive value.

### *3.3. Comparative analysis*

To verify the effectiveness of the proposed improved CS optimized BP neural network (ICS-BP) for microgrid load prediction, simulations were conducted using Matlab simulation software. The results were compared with those obtained from traditional BP, conventional CS optimized BP (CS-BP), chaos sequence improved CS optimized BP (CCS-BP), and particle swarm optimization improved BP (PSO-BP) forecasting methods.

The load forecasting results of microgrids obtained by four different prediction methods are shown in Figure 4. From the predicted results in Figure 4, it can be seen that although the conventional BP neural network prediction method can predict the trend of load changes in microgrids, there was a significant difference between the predicted microgrid load and the actual load value, and the prediction error was relatively large. After using the CS algorithm to improve the BP network, the load prediction values were basically consistent with the actual load values, especially after using the adaptive variable step size ICS algorithm to optimize the parameters of the BP network, the load prediction results obtained were closer to the actual values, and the prediction accuracy was further improved. The prediction performance of the PSO improved BP prediction model was not ideal. The prediction accuracy of the CCS algorithm was higher than that of the standard CS algorithm, but it was still lower compared to the variable step size ICS algorithm.

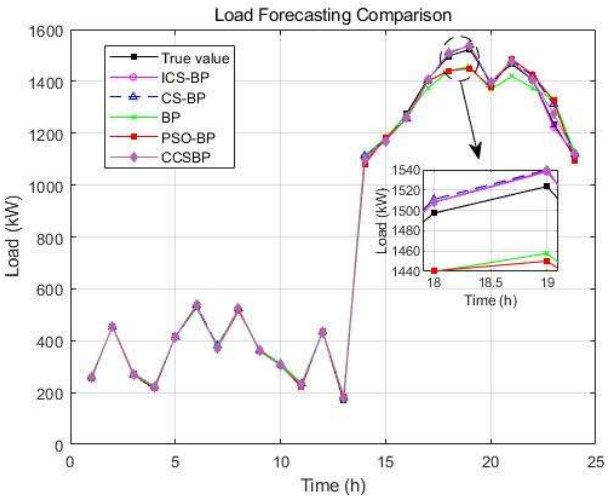


**Figure 4.** Results of load forecasting.

The absolute error variation curves of sample tracking for three different load forecasting methods are shown in Figure 5.

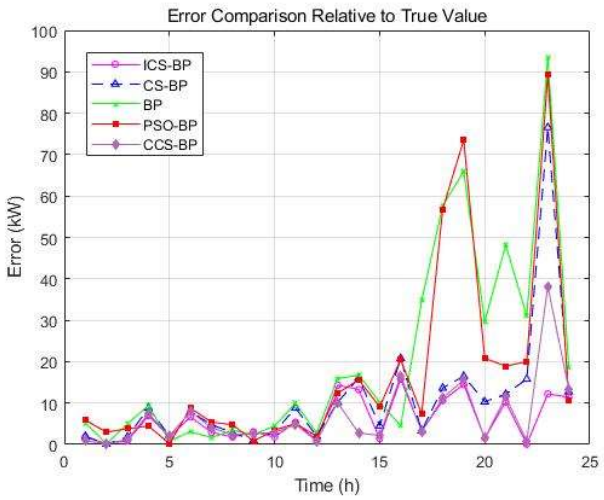


**Figure 5.** Comparison of prediction errors.

From the error curves shown in Figure 5, it can be observed that when the load changed very little, the prediction errors of the five forecasting methods were relatively small. However, when encountering severe load fluctuations, the BP forecasting model brought large prediction errors. The ICS algorithm significantly reduces the prediction errors generated by the BP network. The load forecasting process based on the ICS-BP prediction model produced the smallest prediction error and there was no significant fluctuation in prediction error, which proves that the ICS-BP prediction model has good adaptability and can effectively resist the influence of external disturbances such as load changes. In contrast, when the traditional CS algorithm was used to improved the BP network, although it reduced load prediction errors to some extent, the error curve of CS-BP had significant fluctuations, indicating that the CS-BP algorithm shows unstable performance in microgrid load forecasting. The prediction error of the CCS-BP algorithm has also been reduced, while the prediction error of PSO-BP was also relatively large, which was close to that of the simple BP network, indicating that PSO has little effect on improving the prediction performance of BP network.

The main comparison data of prediction accuracy and speed of five different prediction methods are shown in Table 1. In the table, $t_c$ represents the computational time of the prediction algorithm.

**Table 1.** Comparison of prediction accuracy and speed.

| Method | MAPE (%) | RMSE (kW) | MAE (kW) | MSE [(kW)$^2$] | $t_c$/s |
|---|---|---|---|---|---|
| BP | 2.3712 | 31.0170 | 19.7878 | 962.0565 | 0.062 |
| CS-BP | 1.6832 | 18.5078 | 11.0681 | 342.5403 | 0.103 |
| PSO-BP | 1.9618 | 28.2056 | 16.733 | 796.0526 | 0.118 |
| CCS-BP | 1.2886 | 11.3959 | 7.7894 | 129.8668 | 0.153 |
| ICS-BP | 1.1304 | 7.5880 | 5.8215 | 57.5775 | 0.068 |

From the error indicators presented in Table 1, it is evident that among the five forecasting methods, the ICS optimized BP network exhibited lowest prediction error compared to the BP network, the conventional CS optimized BP network, the CCS optimized BP, and the PSO optimized BP network, while the BP network has the largest error among the five prediction methods. The MAPE of ICS-BP

load forecasting decreased by 52.3, 32.8, 12.3, and 42.3%, respectively, compared to conventional BP, CS-BP, CCS-BP, and PSO-BP. Meanwhile, the RMSE, MAE, and MSE indicators of the ICS-BP model were 75.6, 70.6, and 94.0% lower than those of the traditional BP model, respectively, fully demonstrating the excellent load forecasting performance of ICS-BP. In addition, the MAPE value of the ICS-BP forecasting method can be considered as approximately zero, indicating that the improved forecasting model is very close to a perfect prediction model.

Analyzing the computational time of the five prediction methods in Table 1, it can be found that the computational time of the ICS-BP model was slightly longer than that of BP, but was much shorter than the CS-BP, CCS-BP, and PSO-BP models, which indicates that the introduction of ICS increased the complexity of the calculation, but the impact was not too obvious. The computational time of CS-BP, CCS-BP, and PSO-BP was much longer than that of BP and ICS-BP, indicating that the introduction of CS, CCS, and PSO has a great impact on the computing speed of the BP prediction network. The ICS-BP prediction model can guarantee the real-time performance of load forecasting.

The comparison data between predicted values and true values are shown in Table 2.

**Table 2.** Comparison between true and predicted values.

| True (kW) | Forecasted (kW) | | | | | Forecastied error (kW) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | BP | CS-BP | PSO-BP | CCS-BP | ICS-BP | BP | CS-BP | PSO-BP | CCS-BP | ICS-BP |
| 258.150 | 263.338 | 260.130 | 252.3 | 257.26 | 259.900 | 5.1880 | 1.9800 | 5.8500 | 0.8900 | 1.7500 |
| 452.794 | 452.953 | 452.685 | 455.77 | 452.998 | 452.998 | 0.1590 | 0.1090 | 2.9800 | 0.2040 | 0.2040 |
| 267.762 | 272.752 | 269.569 | 271.57 | 266.543 | 268.628 | 4.9900 | 1.8070 | 3.8100 | 1.2190 | 0.8660 |
| 214.758 | 224.179 | 223.563 | 219.22 | 222.160 | 221.692 | 9.4210 | 8.8050 | 4.4600 | 7.4020 | 6.9340 |
| 416.092 | 416.881 | 412.021 | 416.26 | 413.056 | 414.272 | 0.7890 | 4.0710 | 0.1700 | 3.0360 | 1.8200 |
| 528.622 | 531.718 | 536.556 | 537.4 | 536.556 | 535.265 | 3.0960 | 7.9340 | 8.7800 | 7.9340 | 6.6430 |
| 376.723 | 378.362 | 381.522 | 371.32 | 372.651 | 373.786 | 1.6390 | 4.7990 | 5.4000 | 4.0720 | 2.9370 |
| 520.13 | 523.542 | 522.954 | 515.36 | 522.454 | 525.927 | 3.4120 | 2.8240 | 4.7700 | 2.3240 | 5.7970 |
| 362.257 | 364.686 | 364.224 | 363.09 | 359.662 | 359.212 | 2.4290 | 1.9670 | 0.8300 | 2.5950 | 3.0450 |
| 307.441 | 311.915 | 310.515 | 303.96 | 302.260 | 305.638 | 4.4740 | 3.0740 | 3.4800 | 5.1810 | 1.8030 |
| 226.159 | 236.108 | 234.953 | 221.09 | 230.953 | 231.533 | 9.9490 | 8.7940 | 5.0700 | 4.7940 | 5.3740 |
| 432.954 | 435.746 | 434.963 | 434.94 | 432.086 | 431.945 | 2.7920 | 2.0090 | 1.9900 | 0.8680 | 1.0090 |
| 170.419 | 186.318 | 185.593 | 182.77 | 183.563 | 184.743 | 15.8990 | 15.1740 | 12.3500 | 13.1440 | 14.3240 |
| 1096.79 | 1113.56 | 1112.53 | 1081.15 | 1079.64 | 1110.06 | 16.7700 | 16.0400 | 15.6400 | 17.1500 | 13.2700 |
| 1172.07 | 1182.26 | 1176.56 | 1181.17 | 1169.581 | 1170.65 | 10.1900 | 4.4900 | 9.1000 | 2.4890 | 1.4200 |
| 1276.23 | 1271.78 | 1255.452 | 1255.73 | 1259.612 | 1260.45 | 4.4500 | 20.7780 | 20.5000 | 16.6180 | 15.7800 |
| 1407.64 | 1372.93 | 1404.32 | 1400.25 | 1404.623 | 1404.48 | 34.7100 | 3.3200 | 7.3900 | 3.0170 | 3.1600 |
| 1497.3 | 1439.62 | 1510.93 | 1440.47 | 1508.583 | 1507.9 | 57.6800 | 13.6300 | 56.8300 | 11.2830 | 10.6000 |
| 1523.6 | 1457.69 | 1540.09 | 1450.06 | 1539.156 | 1531.09 | 65.9100 | 16.4900 | 73.5400 | 15.5560 | 7.4900 |
| 1398.52 | 1368.84 | 1388.21 | 1377.71 | 1395.15 | 1396.99 | 29.6800 | 10.3100 | 20.8100 | 3.3700 | 1.5300 |
| 1467.09 | 1418.93 | 1479.25 | 1486.02 | 1478.651 | 1477.36 | 48.1600 | 12.1600 | 18.9300 | 11.5610 | 10.2700 |
| 1405.05 | 1374.04 | 1420.84 | 1425.06 | 1404.32 | 1404.98 | 31.0100 | 15.7900 | 20.0100 | 0.7300 | 0.0700 |
| 1235.98 | 1329.53 | 1312.51 | 1325.2 | 1274.154 | 1223.77 | 93.5500 | 76.5300 | 89.2200 | 38.1740 | 12.2100 |
| 1106.68 | 1125.24 | 1119.43 | 1096.12 | 1120.014 | 1118.09 | 18.5600 | 12.7500 | 10.5600 | 13.3340 | 11.4100 |

From the data in Table 2, it is evident that there were significant differences in the absolute error

values between the predicted and actual values under the five different forecasting methods. The maximum errors generated by the ICS-BP, CS-BP, CCS-BP, PSO-BP, and conventional BP network were 15.78, 76.5, 38.1, 89.22, and 93.55 kW, respectively. The maximum prediction error of the ICS-BP network was 83.1, 79.4, 58.7, and 82.4% lower than that of conventional BP, CS-BP, CCS-BP and PSO-BP, respectively. The ICS optimized BP network model achieved more accurate and reliable results in microgrid load forecasting.

## 4. Conclusions

The BP neural network optimized by the ICS algorithm can effectively improve the accuracy of load forecasting in microgrid. The adaptive step size adjustment law and adaptive position update law are designed for the CS algorithm, which solves the problems of slow search speed and iterative oscillation caused by random step size of the conventional CS algorithm, and improves the search accuracy and speed of the CS algorithm. The ICS algorithm applied to the parameter optimization of the BP neural network solves the problems such as slow training speed and difficult parameter optimization of conventional BP network, and significantly improves the accuracy and reliability of load forecasting in microgrid. Compared with the traditional BP network, the conventional CS, the chaos sequence, and the PSO optimized BP network, the MAPE value of the ICS algorithm optimized BP network decreased by 52.3, 32.8, 12.3, and 42.3%, respectively. Additionally, the other indicators of the ICS algorithm optimized BP network are also significantly better than those of the other three methods. Compared with the traditional BP network, the RMSE, MAE, and MSE of ICS-BP decreased by 75.6, 70.6, and 94.0%, respectively, and the computational speed of ICS-BP was 42.4, 55.6, and 34.0% faster than PSO-BP, CCS-BP, and CS-BP, respectively. The proposed ICS-BP pediction model possesses good robustness and generalization capability, making it suitable for various load forecasting scenarios. Future studies can expand the application of this model to solve a wider range of process prediction problems and to promote its application in the actual microgrid systems.

## Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

## Acknowledgments

This work was supported by the Intercollegiate Cooperation Project of Colleges and Universities in Liaoning Province (Grant [2020] 28), and the Key Project of Liaoning Provincial Department of Education (LJKZZ20220057).

## Conflict of interest

The authors declare there is no conflict of interest.

## References


1. S. Goswami, S. Malakar, B. Ganguli, A. Chakrabarti, A novel transfer learning-based short-term solar forecasting approach for India, *Neural Comput. Applic.*, **34** (2022), 16829–16843. https://doi.org/10.1007/s00521-022-07328-9
2. B. Deepanraj, N. Senthilkumar, T. Jarin, A. E. Gurel, L. Sundar, A. Anand, Intelligent wild geese algorithm with deep learning driven short term load forecasting for sustainable energy managementin microgrids, *Sustainable Comput.: Inf. Syst.*, **36** (2022), 100813. https://doi.org/10.1016/j.suscom. 2022.100813
3. A. Rafati, M. Joorabian, E. Mashhour, H. R. Shaker, Machine learning-based very short-term load forecasting inmicrogrid environment: evaluating the impact of high penetration of PV systems, *Electr. Eng.*, **104** (2022), 2667–2677. https://doi.org/10.1007/s00202-022-01509-4
4. S. Wang, F. Wu, P. Takyi-Aninakwa, C. Fernandez, D. Stroe, Q. Huang, Improved singular filtering-Gaussian process regression-long short-term memory model for whole-life-cycle remaining capacity estimation of lithium-ion batteries adaptive to fast aging and multi-current variations, *Energy*, **284** (2023), 128677. https://doi.org/10.1016/j.energy.2023.128677
5. S. Wang, Y. Fan, S. Jin, P. Takyi-Aninakwa. C. Fernandez, Improved anti-noise adaptive long short-term memory neural network modeling for the robust remaining useful life prediction of lithium-ion batteries, *Reliab. Eng. Syst. Safe*, **230** (2023), 108920. https://doi.org/10.1016/j.ress.2022.108920
6. A. Jahani, K. Zare, L. M. Khanli, Short-term load forecasting for microgrid energy management system using hybrid SPM-LSTM, *Sustainable Cities Soc.*, **98** (2023), 104775. https://doi.org/10.1016/j.scs.2023.104775
7. A. A. Muzumdar, C. N. Modi, G. M. Madhu, C. Vyjayanthi, Designing a robust and accurate model for consumer-centric short-term load forecasting in microgrid environment, *IEEE Syst. J.*, **16** (2022), 2448–2459. https://doi.org/ 10.1109/JSYST.2021.3073493
8. F. Mohammad, C. Kimy, Energy load forecasting model based on deep neural networks for smart grids, *Int. J. Syst. Assur. Eng.*, **11** (2020), 824–834. https://doi.org/10.1007/s13198-019-00884-9
9. R. Wazirali, E. Yaghoubi, M. S. Abujazar, R. Ahmad, H. A. Vakili, State-of-the-art review on energy and load forecasting in microgrids using artificial neural networks, machine learning, and deep learning techniques, *Electr. Pow. Syst. Res.*, **225** (2023), 109792. https://doi.org/10.1016/j.epsr. 2023.109792
10. H. Xu, Y. Chang, Y. Zhao, F. Wu, A novel hybrid wind speed interval prediction model based on mode decomposition and gated recursive neural network, *Environ. Sci. Pollut. R.*, **29** (2022), 1–17. https://doi.org/10.1007/s11356-022-21904-5
11. B. B. Yong, H. Liang, F. C. Li, J. Shen, X. Wang, Q. G. Zhou, A research of Monte Carlo optimized neural network for electricity load forecast, *J. Supercomput.*, **76** (2020), 6330–6343. https://doi.org/10.1007/ s11227-019-02828-3
12. L. Nguyen-Ngoc, Q. Nguyen-Huu, G. D. Roeck, T. Bui-Tien, M, Abdel-Wahab, Deep neural network and evolved optimization algorithm for damage assessment in a truss bridge, *Mathematics*, **12** (2024), 2300. https://doi.org/10.3390/math12152300
13. Y. F. Li, H. L. Minh, M. S. Cao, X. D. Qian, M. A. Wahab, An integrated surrogate model-driven and improved termite life cycle optimizer for damage identification in dams, *Mech. Syst. Signal Pr.*, **208** (2024), 110986. https://doi.org/10.1016/j.ymssp.2023.110986

14. V. T. Tran, T. K. Nguyen, H. Nguyen-Xuan, M. A. Wahab, Vibration and buckling optimization of functionally graded porous microplates using BCMO-ANN algorithm, *Thin Wall Struct.*, **182** (2023), 110267. https://doi.org/10.1016/j.tws.2022.110267
15. B. L. Dang, H. Nguyen-Xuan, M. A. Wahab, An effective approach for VARANS-VOF modelling interactions of wave and perforated breakwater using gradient boosting decision tree algorithm, *Ocean Eng.*, **268** (2023), 113398. https://doi.org/10.1016/j.oceaneng.2022.113398
16. T. Li, Y. Li, L. S. Ren, A method of using back propagation neural network to estimate orbital life time of LEO satellites, *Adv. Space Res.*, **72** (2023), 1961–1969. https://doi.org/10.1016/j.asr.2023.05.026
17. Y. M. Xie, W. Li, C. Liu, M. Du, K. Feng, Optimization of stamping process parameters based on improved GA-BP neural network model, *Int. J. Precis. Eng. Man.*, **24** (2023), 1129–1145. https://doi.org/10.1007/s12541-023-00811-w
18. A. J. Thomas, S. J. Walters, S. M. Gheytassi, R. E. Morgan, M. Petridis, On the optimal node ratio between hidden layers: a probabilistic study, *Int. J. Mach. Learn. Comput.*, **6** (2016), 241–247. https://doi.org/10.18178/IJMLC.2016.6.5.605
19. M. Braik, A. Sheta, H. Al-Hiary, S. Aljahdali, Enhanced cuckoo search algorithm for industrial winding process modeling, *J. Intell. Manuf.*, **34** (2023), 1911–1940. https://doi.org/10.1007/s10845-021- 01900-1
20. Z. P. Hou, M. Zhou, C. Roberts, H. Dong, Cuckoo search approach for automatic train regulation under capacity limitation, *Sci. China Inf. Sci.*, **66** (2023), 149204. https://doi.org/10.1007/s11432-020-3254-0
21. H. J. Zheng, Y. Peng, J. Guo, Y. Chen, Course scheduling algorithm based on improved binary cuckoo search, *J. Supercomput.*, **78** (2022), 11895–11920. https://doi.org/10.1007/s11227-022-04341-6
22. L. Z. Duan, S. Q. Yang, D. B. Zhang, Multilevel thresholding using an improved cuckoo search algorithm for image segmentation, *J. Supercomput.*, **77** (2021), 6734–6753. https://doi.org/10.1007/s11227-020-03566-7
23. M. G. Mortazavi, M. H. Shirvani, A. Dana, M. Fathy, Sleep-wakeup scheduling algorithm for lifespan maximization of directional sensor networks: a discrete cuckoo search optimization algorithm, *Complex Intell. Syst.*, **9** (2023), 6459–6491. https://doi.org/10.1007/s40747-023-01078-4
24. J. T. Cheng, Y. Xiong, Parameter control based Cuckoo Search Algorithm for numerical optimization, *Neural Process. Lett.*, **54** (2022), 3173–3200. https://doi.org/10.1007/s11063-022-10758-0
25. H. Khan, S. Jamal, M. Hazzazi, M. Khan, I. Hussain, New image encryption scheme based on Arnold map and cuckoo search optimization algorithm, *Multimed. Tools Appl.*, **82** (2023), 7419–7441. https://doi.org/10.1007/s11042-022-13600-w
26. R. Cristin, B. Kumar, C. Priya, K. Karthick, Deep neural network based Rider-cuckoo search algorithm for plant disease detection, *Artif. Intell. Rev.*, **53** (2020), 4993–5018. https://doi.org/10.1007/s10462-020-09813-w
27. M. Gupta, M. Abouhawwash, S. Mahajan, A. K. Pandit, Cuckoo search algorithm, its variants and applications: A review, in *AIP Conference Proceeding*, **2495** (2023), 020072. https://doi.org/10.1063/5.0122753

28. X. Qin, B. Xia, T. Ding, L. Zhao, An improved cuckoo search localization algorithm for UWB sensor networks, *Wireless Networks*, **27** (2020), 527–535. https://doi.org/10.1007/s11276-020-02465-2
29. J. Wei, H. Y. Niu, A ranking-based adaptive cuckoo search algorithm for unconstrained optimization, *Expert Syst. Appl.*, **204** (2022), 117428. https://doi.org/10.1016/j.eswa.2022.117428
30. B. She, A. Fournier, M. Yao, Y. Wang, G. Hu, A self-adaptive and gradient-based cuckoo search algorithm for global optimization, *Appl. Soft Comput.*, **122** (2022), 108774. https://doi.org/10.1016/j.asoc.2022.108774
31. S. Sadik, M. Et-tolba, B. Nsiri, A modified adaptive sparse-group LASSO regularization for optimal portfolio selection, *IEEE Access*, **12** (2024), 107337–107352. https://doi.org/10.1109/ACCESS.2024.3438125
32. M. Uddin, H. Moa, D. Dong, S. Elsawah, J. Zhu, M. Guerrero, Microgrids: A review, outstanding issues and future trends, *Energy Strateg. Rev.*, **49** (2023), 101127. https://doi.org/10.1016/j.esr.2023.101127
33. A. Solat, G. B. Gharehpetian, M. S. Naderi, A. Anvari-Moghaddam, On the control of microgrids against cyber-attacks: A review of methods and applications, *Appl. Energy*, **353** (2024), 122037. https://doi.org/10. 1016/j.apenergy.2023.122037


AIMS Press