Research on improved convolutional wavelet neural network

Jingwei Liu1,2,3*, Peixuan Li1,3, Xuehan Tang1, Jiaxin Li1 & Jiaming Chen2

Artificial neural networks (ANN) which include deep learning neural networks (DNN) have problems such as the local minimal problem of Back propagation neural network (BPNN), the unstable problem of Radial basis function neural network (RBFNN) and the limited maximum precision problem of Convolutional neural network (CNN). Performance (training speed, precision, etc.) of BPNN, RBFNN and CNN are expected to be improved. Main works are as follows: Firstly, based on existing BPNN and RBFNN, Wavelet neural network (WNN) is implemented in order to get better performance for further improving CNN. WNN adopts the network structure of BPNN in order to get faster training speed. WNN adopts the wavelet function as an activation function, whose form is similar to the radial basis function of RBFNN, in order to solve the local minimum problem. Secondly, WNN-based Convolutional wavelet neural network (CWNN) method is proposed, in which the fully connected layers (FCL) of CNN is replaced by WNN. Thirdly, comparative simulations based on MNIST and CIFAR-10 datasets among the discussed methods of BPNN, RBFNN, CNN and CWNN are implemented and analyzed. Fourthly, the wavelet-based Convolutional Neural Network (WCNN) is proposed, where the wavelet transformation is adopted as the activation function in Convolutional Pool Neural Network (CPNN) of CNN. Fifthly, simulations based on CWNN are implemented and analyzed on the MNIST dataset. Effects are as follows: Firstly, WNN can solve the problems of BPNN and RBFNN and have better performance. Secondly, the proposed CWNN can reduce the mean square error and the error rate of CNN, which means CWNN has better maximum precision than CNN. Thirdly, the proposed WCNN can reduce the mean square error and the error rate of CWNN, which means WCNN has better maximum precision than CWNN.

Artificial neural network (ANN)1,2 is a classic machine learning method. ANN is based on a collection of connected units or nodes called artificial neurons, which loosely models the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal from one artificial neuron to another. ANN can learn knowledge and use the learned knowledge to reason about results by the following two modes: Training mode (learning) and Forward calculation mode (reasoning). Convolutional neural network (CNN)3,4 is a method which based on feature extraction of convolution calculation.

The motivation of this study is as follows: Firstly, find a powerful simple neural network which has fast training speed like Back propagation neural network (BPNN)5,6 and Fully connected layers (FCL)7–9. Secondly, solve the local minimum problem according to Radial basis function neural network (RBFNN) method. Thirdly, improve the precision of CNN and verify the better performances of the proposed Convolutional wavelet neural network (CWNN) and Wavelet convolutional neural network (WCNN) based on the two well-known MNISIT and CIFAR-10 datasets.

Existing typical ANN and DNN methods are as follows: BPNN, RBFNN, Wavelet neural network (WNN), CNN, FCL, etc. Each of the above ANN (BPNN, RBFNN, WNN, and CNN are defined as XNN in this study) has advantages and disadvantages as follows: Firstly, the BPNN7–11 is a forward network. It is based on error back propagation12 and gradient descent algorithm. BPNN algorithm is widely used in many commercial applications. Advantages of BPNN are as follows: Firstly, BPNN has strong nonlinear mapping ability and high self-learning ability and adaptability; Secondly, BPNN can be wildly used because it can be well adapted to various samples; Problems and disadvantages of BPNN16 are as follows: Firstly, in the training process, the error of BPNN may drop into the local minimum; Secondly, in the training process, the convergence rate17 is slow. Secondly, RBFNN18,19 is a feedback network. The RBFNN hidden layer is composed of radial basis functions. Advantages of RBFNN are as follows: Firstly, RBFNN has no local minimum

1Information College, Capital University of Economics and Business, Beijing 100070, China. 2Information Department, Beijing University of Technology, Beijing 100124, China. 3These authors contributed equally: Jingwei Liu and Peixuan Li. *email: liujingwei@cueb.edu.cn
problem, which is the biggest problem of BPNN; Secondly, RBFNN has strong mapping ability from input to output and good classification ability. Problems and disadvantages of RBFNN are as follows: Firstly, it is very difficult to find the center of RBFNN hidden nodes; Secondly, it is difficult to determine the number of nodes in the hidden layer of RBFNN. Thirdly, CNN is a deep neural network. A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, activation function, pooling layers, fully connected layers and normalization layers. The advantage of CNN are as follows: CNN has more powerful learning ability for complex learning task. The disadvantage of CNN is that: when the learning object is too simple, the learning complexity of CNN is much bigger than the other ANNs. As a result, the learning speed of CNN may be slower than the other ANNs. Fourthly, FLC is one of the simplest neural networks, which has only two connected layers. The advantage of FLC is that: FLC has a very fast learning speed for simple learning cases than BPNN and RBFNN. The disadvantage of FLC is that: FLC cannot learn complex samples, even cannot complete some learning work which can be complete by BPNN and RBFNN.

Main contributions of this study are as follows: Firstly, the structure of BPNN, the radial basis function of RBFNN and the wavelet function are adopted to implement WNN. Secondly, based on WNN and CNN, CWN is proposed to improve the performances. Thirdly, based on both MINIST and CIFAR-10 datasets, all the above discussed methods are compared.

The rest of this paper is organized as follows: “Results” section addresses the results obtained by three experiments on WNN and CWN. “Methods” section details the methodology of WNN and CWN. “Data” section introduces datasets and “Design of simulation” section introduces more details about all experiments. “Conclusions” section summaries all the simulation results and suggests some directions for further research.

Results
Four experiments are implemented between BPNN, WNN, CWN and WCNN in order to prove the improved effects. Firstly, “feasibility experiment” is designed to verify the feasibility (convergence) of WNN and prove that WNN can solve the problems of BPNN and RBFNN. Secondly, “performances experiment” is designed to verify the best performances (such as maximum precision, minimum error) of BPNN, RBFNN and WNN. Thirdly, “CWN experiment” is designed in order to prove that the performances of the proposed CWN is better than CNN. Fourthly, “WCNN experiment” is designed in order to prove that the performances of the proposed WCNN is better than CNN and CWN.

Definition 1 1 completed simulation process (1CSP) means a completed training process from beginning time 0 to the complete time (the time when the training error is less than the target error).

Definition 2 1 simulation time (1CT) is only one training calculation. 1CSP contains many CTs. \( \Delta w_{jk}^{(3)}, \Delta w_{lj}^{(2)}, \Delta a_j, \Delta b_l \) are calculated once in 1CT.

Result of feasibility experiment. The dataset of “feasibility experiment” is generated by our designation which is specifically described in the section of “Data”. Results of comparative simulations are discussed by two ways:

Firstly, error descending curves and error surfaces in 1CSP are plotted in Fig. 1.

The average CTs in each CSP of BPNN is 39,802. The maximum error is 0.050000. The mean square error is 0.000319. The error descending curve and error surface are drawn as Fig. 1a.d. The average CTs of RBFNN is 1580. The maximum error is 0.000314. The mean square error is 0.049570. The error descending curve and error surface are drawn as Fig. 1b.e. The average CTs of WNN is 1006. The maximum error is 0.000445. The mean square error is 0.049995. The error descending curve and error surface are drawn as Fig. 1c.f.

Secondly, statistical details of simulations in 10 CSPs are listed in Table 1, which are specifically discussed as follows:

10 CSPs of simulations of BPNN, RBFNN and WNN algorithms are compared to find out the differences of training times, mean square error and maximum error. Columns 2, 5, and 8 (XNN Training times) show how many CTs are required to complete the simulation in each CSP XNN. Columns 3, 6, and 9 (XNN Mean square error) show the final mean square error after each training CSP. Columns 4, 7, and 10 (XNN maximum error) show the final maximum error after each training CSP. The maximum error is expected less than the target error. Therefore, if the training can be completed within 20,000 CTs, the maximum error is less than err_goal = 0.1. The results show that WNN can solve the problems of BPNN and RBFNN with better performance and make preparation for the improvement from CNN to CWN.

According to the above Fig. 1 and Table 1, we can draw the following conclusions: Firstly, all of the BPNN, RBFNN and WNN algorithms are convergent. According to the columns 2, 5, and 8, all the values are less than the max_epoch = 200, 000, which means all the training processes are convergent (all the training errors are lower than the target errors). It is proved that all the algorithms are feasible. Secondly, the average training times of WNN (CTs = 343) is the least, while the average training times of BPNN and RBFNN are 1305 and 2630. The average training times of RBFNN is the most. It is proved that WNN is the fastest algorithm, and RBFNN is the slowest one. Thirdly, the error descending curve of BPNN keeps decreasing, which causes problem of local minimum. BPNN also has problems such as slow convergence speed and local minimum problem. However, the problem of local minimum are solved by RBFNN and WNN because of the structure of network and the active functions of RBFNN and WNN. The error curve of RBFNN in Fig. 1b not only reduces the time but also avoids the local minimum. While the error curve of WNN in Fig. 1c, significant changes (break the process of decreasing) only happen at the very beginning time of the training process.
Result of hyperparameter optimization experiment. The "Hyperparameter optimization experiment" is
designed and implemented in order to verify the best performances (max precision, min error) of BPNN, RBFNN and
WNN, with the comparative results analyzed.

The simulation results of the above 3 algorithms are shown in Table 2, which are specifically discussed as follows.

Columns 2, 5, and 8 (XNN Training times) show the numbers of CTs in each CSP. 20,000 means XNN cannot
complete training within 200,000 CTs (i.e., the square error is not less than the target error with 20,000 training CTs).
Columns 3, 6, and 9 (XNN Mean square error) show the final mean square error after each training CSP. Columns
4, 7, and 10 (XNN Maximum error) show the final maximum error after each training CSP. The maximum error
is expected less than the target error. Therefore, if the training can be completed within 20,000 CTs, the maximum error
is less than $\text{err}_{\text{goal}} = 0.02$.

According to Table 2, we can draw the following conclusions: The success rate of WNN training is the highest (i.e.,
60% of WNN training processes are completed), while only 40% of RBFNN training processes are completed, and 0%
of BPNN training processes are completed. The precision of WNN is the highest because when the target precision

| Number of CSP | BPNN training times | BPNN mean square error | BPNN maximum error | RBFNN training times | RBFNN mean square error | RBFNN maximum error | WNN training times | WNN mean square error | WNN maximum error |
|---------------|---------------------|-----------------------|-------------------|----------------------|------------------------|---------------------|-------------------|---------------------|-------------------|
| 1             | 1530                | 0.002328              | 0.099952          | 672                  | 0.000966               | 0.099939           | 42                | 0.001329            | 0.096115          |
| 2             | 862                 | 0.002457              | 0.099989          | 680                  | 0.001164               | 0.093833           | 224               | 0.001596            | 0.099703          |
| 3             | 1791                | 0.002627              | 0.099962          | 319                  | 0.001948               | 0.080631           | 173               | 0.001659            | 0.099784          |
| 4             | 972                 | 0.002559              | 0.099991          | 985                  | 0.001022               | 0.099876           | 954               | 0.001233            | 0.099798          |
| 5             | 1401                | 0.002637              | 0.099982          | 1417                 | 0.001042               | 0.099998           | 713               | 0.001654            | 0.087587          |
| 6             | 1003                | 0.002255              | 0.099977          | 656                  | 0.001518               | 0.099915           | 461               | 0.001911            | 0.099994          |
| 7             | 845                 | 0.002162              | 0.099929          | 4446                 | 0.001258               | 0.099445           | 222               | 0.001513            | 0.099781          |
| 8             | 2061                | 0.002355              | 0.099973          | 7401                 | 0.000047               | 0.020000           | 111               | 0.001766            | 0.099800          |
| 9             | 1325                | 0.002577              | 0.099991          | 2330                 | 0.001826               | 0.099944           | 478               | 0.001644            | 0.098883          |
| 10            | 1257                | 0.002481              | 0.099934          | 7374                 | 0.001817               | 0.097768           | 53                | 0.001763            | 0.099677          |
| Average       | 1305                | 0.002444              | 0.099968          | 2630 Slowest         | 0.001261               | 0.089135           | 343 Fastest        | 0.001607            | 0.098230          |
| Success rate  | 100%                | 100%                 | 100%              |                      |                        |                     |                   |                     |                   |
errol\_goal is set from 0.1 to 0.02, the success rate of WNN is higher than RBFNN and BPNN. When the errol\_goal is set from 0.02 to 0.005, only WNN can complete the training. The training speed of WNN is the fastest because the average training time of WNN (4936) is less than those of RBFNN (9631) and BPNN (20,000).

Result of CWNN experiment.

CWNN experiments are based on MNIST and CIFAR-10 datasets which are widely recognized and adopted in the comparative experiments. CWNN is better than CNN on MNIST. Results on MNIST are recorded in Table 3. Error rate (effect of method) and Mean square error (MSE, effect of training) are two important indicators. According to the results on MNIST, main findings are as follows: Firstly, classification task can be completed by both CNN and CWNN. The change trend of MSE is declining during the training processes of both CNN and CWNN. Secondly, training accuracy of CWNN is better than CNN. Under the same training frequency, the MSE of CWNN (0.14146) is smaller than CNN (0.16431). Thirdly, classification ability of CWNN is better than CNN. Under the same training frequency, the error rate of CWNN (0.12915) is smaller than CNN (0.16434).

CWNN is better than CNN on CIFAR-10. The results of CIFAR-10 are recorded in Table 4. Error rate and Mean square error are two important indicators. According to the results on CIFAR-10, main findings are as follows: Firstly, classification task can be completed by both CNN and CWNN. The change trend of MSE is declining during the training processes of both CNN and CWNN. Secondly, training accuracy of CWNN is better than CNN. Under the same training frequency, the MSE of CWNN (0.23715) is smaller than CNN (0.29845). Thirdly, classification ability of CWNN is better than CNN. Under the same training frequency, the error rate of CWNN (0.18522) is smaller than CNN (0.20510).

Result of WCNN experiment. WCNN experiments are based on MNIST datasets. WCNN is better than CWNN on MNIST. Results on MNIST are recorded in Table 5. According to the results on MNIST, main findings are as follows: Firstly, classification task can be completed by WCNN. The change trend of MSE is declining during the training
processes. Secondly, training accuracy of WCNN is better than WCNN. Under the same training frequency, the MSE of WCNN (0.0.08850) is smaller than CWNN (0.14146). Thirdly, classification ability of WCNN is better than CWNN. Under the same training frequency, the error rate of WCNN (0.07723) is smaller than CWNN (0.18522).

Methods

Wavelet neural network (WNN). The features, advantages and disadvantages of WNN24,25 are as follows: WNN has activation functions of multi-scaled analysis and scale translation in hidden layers. Advantages of WNN are as follows: Firstly, WNN has high precision and high resolution. Secondly, wavelet transform method is good at analyzing local information of signals26,27 which is also the feature of WNN. Thirdly, wavelet transform can do an excellent job such as function approximation28,29 and pattern classification30. It has been proved that the wavelet neural network is an excellent approximator for fitting single variable function31. Problems and disadvantages of WNN are as follows: WNN cannot complete complex learning task because of structural limitation, etc. Similar problems also exist in BPNN, RBFNN and FCL.

WNN is designed as follows: Firstly, structure of BPNN is adopted as the basic structure of WNN; Secondly, the form of activation function in hidden layers of RBFNN is adopted; Thirdly, the wavelet transform function is adopted as the activation function. The structure of WNN is shown in Fig. 2.

| Numbers of SPs and statistical item | Total numbers of ACs | Error Rate of CNN (accuracy) | MSE of CNN | Error rate of WCNN (accuracy) | MSE of CWNN |
|-----------------------------------|----------------------|----------------------------|------------|-------------------------------|-------------|
| 1                                 | 1000                 | 0.1957                     | 0.267      | 0.175239                      | 0.242       |
| 2                                 | 1000                 | 0.2611                     | 0.4465     | 0.170323                      | 0.325       |
| 3                                 | 1000                 | 0.1959                     | 0.27       | 0.180921                      | 0.257       |
| 4                                 | 1000                 | 0.1888                     | 0.263      | 0.177892                      | 0.2455      |
| 5                                 | 1000                 | 0.1985                     | 0.2805     | 0.212793                      | 0.2875      |
| 6                                 | 1000                 | 0.2093                     | 0.3135     | 0.192254                      | 0.268       |
| 7                                 | 1000                 | 0.2014                     | 0.297      | 0.185923                      | 0.267       |
| 8                                 | 1000                 | 0.1884                     | 0.26       | 0.172283                      | 0.2335      |
| 9                                 | 1000                 | 0.2048                     | 0.2855     | 0.179464                      | 0.25        |
| 10                                | 1000                 | 0.2068                     | 0.3015     | 0.205191                      | 0.2885      |
| Average                           | 1000                 | 0.20510 (0.79489)          | 0.29845    | 0.18522 (0.81477) Smaller      | 0.25715 Smaller |

Table 4. Simulation results of CNN and CWNN on CIFAR-10. Underlined values represent the results with the highest accuracy or the lowest mean square error.

| Numbers of SPs and statistical item | Total numbers of ACs | Error rate of WCNN (accuracy) | MSE of WCNN | Running times of WCNN |
|-----------------------------------|----------------------|----------------------------|------------|-----------------------|
| 1                                 | 6000                 | 0.0478                     | 0.0587     | 222.3220              |
| 2                                 | 6000                 | 0.0566                     | 0.0710     | 239.7420              |
| 3                                 | 6000                 | 0.0977                     | 0.1000     | 245.8560              |
| 4                                 | 6000                 | 0.0501                     | 0.0599     | 259.2540              |
| 5                                 | 6000                 | 0.0680                     | 0.0720     | 265.9660              |
| 6                                 | 6000                 | 0.0473                     | 0.0598     | 266.5450              |
| 7                                 | 6000                 | 0.0366                     | 0.0458     | 280.9490              |
| 8                                 | 6000                 | 0.0397                     | 0.0443     | 295.5480              |
| 9                                 | 6000                 | 0.0701                     | 0.0864     | 512.6480              |
| 10                                | 6000                 | 0.1954                     | 0.1733     | 320.6640              |
| 11                                | 6000                 | 0.0576                     | 0.0671     | 335.1010              |
| 12                                | 6000                 | 0.0654                     | 0.0748     | 351.9140              |
| 13                                | 6000                 | 0.0483                     | 0.0614     | 366.8210              |
| 14                                | 6000                 | 0.2870                     | 0.3465     | 381.8980              |
| 15                                | 6000                 | 0.0763                     | 0.0893     | 404.1410              |
| 16                                | 6000                 | 0.0452                     | 0.0581     | 427.9340              |
| 17                                | 6000                 | 0.0670                     | 0.0756     | 448.3010              |
| 18                                | 6000                 | 0.0573                     | 0.0790     | 476.6720              |
| 19                                | 6000                 | 0.0483                     | 0.0599     | 493.1710              |
| 20                                | 6000                 | 0.0860                     | 0.0871     | 515.0280              |
| Average                           | 6000                 | 0.07723 (0.92277) Smaller  | 0.08850 Smaller | 355.5238              |

Table 5. Simulation results of WCNN on MNIST. Underlined values represent the results with the highest accuracy or the lowest mean square error.
The output neurons of hidden layer $O_j^{-2}(t)$ can be expressed as Eq. (1):

$$O_j^{-2}(t) = \Psi_{a,b}\left(\frac{net_j^{-2}(t) - b_j(t)}{a_j(t)}\right)$$  \hspace{1cm} (1)

In Eq. (1), $net_j^{-2}(t)$ is the input of neurons in the hidden layer, $\Psi_{a,b}(t)$ is the wavelet function (activation function). Parameters $a_j(t)$ and $b_j(t)$ are the scaling parameters of the wavelet function. $t$ is the number of training time.

The number of nodes in the hidden layer can be calculated according to the linear correlation theory: Redundant (repetitive or useless) nodes can be found and deleted by the comparison of parameters $\Psi_{a,b}(t)$ in each node in the hidden layer. The wavelet function (activation function) can be selected according to the frame theory. The closer the frame is to the boundary, the better the stability of the wavelet function, but when the frame is closer to the boundary, the problem of data redundancy will occur.

The wavelet function that satisfies the framework conditions is selected in Eq. (2):

$$\Psi_{a,b}(t) = \cos(1.75t) \cdot e^{-\frac{t^2}{2}}$$  \hspace{1cm} (2)

The loss function that we select is mean square error (MSE), $E$ is the mean square error (MSE) of all samples, which can be formulated as Eq. (3). The reasons we use the mean square error (MSE) are as follows: Firstly, the outputs of WNN have negative numbers. Secondly, the cross-entropy loss function includes the logarithm function, which requires non-negative inputs.

$$E = \frac{1}{2} \sum_{n=1}^{N} (\hat{y}_n - y_n)^2$$  \hspace{1cm} (3)

In WNN, the back propagation of input errors ($\delta_k^{-2}$ in the output layer, $\delta_j^{-2}$ in the hidden layer and $\delta_i^{-1}$ in the input layer) can be calculated as Eq. (4) to Eq. (6).

$$\delta_i^{-1} = \frac{\partial E}{\partial net_i^{-1}} = \frac{1}{N} \sum_{n=1}^{N} (\hat{y}_n - y_n)(1 - net_i^{-1})net_i^{-2}$$  \hspace{1cm} (4)

$$\delta_j^{-2} = \frac{\partial E}{\partial net_j^{-2}} = \frac{\partial E}{\partial net_j^{-1}} \cdot \frac{\partial net_j^{-1}}{\partial O_j^{-2}} = \frac{1}{b_j^2} \sum_{k=1}^{size^{-1}} \sum_{j=1}^{size^{-2}} \delta_k^{-1} \Psi \left( \frac{y_n - \hat{y}_n}{b_j} \right) \cdot w_{kj}^{-1}$$  \hspace{1cm} (5)

$$\delta_k^{-3} = \frac{\partial E}{\partial net_k^{-3}} = \frac{\partial E}{\partial O_k^{-3}} \cdot \frac{\partial net_k^{-2}}{\partial O_k^{-3}} = \sum_{j=1}^{size^{-2}} \delta_j^{-2} \cdot w_{kj}^{-2}, k = 1, 2, \ldots, size^{-3}$$  \hspace{1cm} (6)

Gradient descent method is adopted to adjust weights and bias of the neural network. Parameters such as $\Delta w_{ij}^{-2}$, $\Delta a_j^{-2}$, $\Delta b_j^{-2}$, $\Delta w_{kj}^{-1}$, $\Delta b^{-1}$ are adjusted in each training process. $i$, $j$ and $k$ are the numbers of neuron in each layer. $\Delta w_{ij}^{-2}$ represents the changed values of weight between the neurons in the input layer and the hidden layer. $\Delta a_j^{-2}$ and $\Delta b_j^{-2}$ represents the changed values of bias between the input layer and the hidden layer. $\Delta w_{kj}^{-1}$ represents the changed values of weight between the neurons in the hidden layer and the output layer.\n
$\Delta b^{-1}$ represents the changed values of bias between the hidden layer and the output layer. $net_j^{-2}$ represents the input of hidden layer. At the training time $t$, the above parameters can be expressed as Eq. (7) to Eq. (11):
The adjusted results of the above weights and bias are expressed as Eq. (12) to Eq. (16), where $\alpha_{_\text{WNN}}$ is the inertia coefficient of WNN, $\eta_{_\text{WNN}}$ is the learning rate of WNN.

$$
\Delta w_{ij}^{-2} = \frac{\partial E}{\partial w_{ij}^{-2}} = \frac{\partial E}{\partial \text{net}_j} \times \frac{\partial \text{net}_j^{-2}}{\partial w_{ij}^{-2}} = \delta_{ij} \cdot \alpha_{_\text{WNN}}^{-2} \cdot Q_i^{-3}
$$

(7)

$$
\Delta a_{j}^{-2} = \frac{\partial E}{\partial a_{j}^{-2}} = \frac{\partial E}{\partial \text{net}_j} \times \frac{\partial \text{net}_j^{-2}}{\partial a_{j}^{-2}} = \frac{1}{\text{size}^2} \sum_{j=1}^{\text{size}^2} -\delta_{j}^2 \cdot \frac{1}{b_j^{-2}}
$$

(8)

$$
\Delta b_{j}^{-2} = \frac{\partial E}{\partial b_{j}^{-2}} = \frac{\partial E}{\partial \text{net}_j} \times \frac{\partial \text{net}_j^{-2}}{\partial b_{j}^{-2}} = -\frac{1}{\text{size}^2} \sum_{j=1}^{\text{size}^2} \delta_{j}^2 \cdot (\text{net}_j^{-1} - a_j^{-2})
$$

(9)

$$
\Delta w_{kj}^{-1} = \frac{\partial E}{\partial w_{kj}^{-1}} = \frac{\partial E}{\partial \text{net}_k} \times \frac{\partial \text{net}_k^{-1}}{\partial w_{kj}^{-1}} = \delta_{k}^{-1} \cdot Q_j^{-2}
$$

(10)

$$
\Delta b_{k}^{-1} = \frac{\partial E}{\partial b_{k}^{-1}} = \frac{\partial E}{\partial \text{net}_k} \times \frac{\partial \text{net}_k^{-1}}{\partial b_{k}^{-1}} = \delta_{k}^{-1}
$$

(11)

According to the above descriptions, pseudocode of WNN method for simulations is shown in Algorithm 1.

| Algorithm 1 Pseudocode of WNN method |
|--------------------------------------|
| **Require:**                          |
| trainingSet: samples described in the Simulation section |
| max_epoch: maximum training times in each CSP (CSP is defined as Definition 1) |
| err_goal: when the current training error is less than target error, the training is finished |
| **Initialization:**                           |
| $w_{ij}^{(3)}$, $w_{ij}^{(2)}$, $a_{j}$, $b_{j}$: weights and scaling parameters are set as random number |
| **Begin:**                                |
| for all trainingSet: |
| $y$ (output of WNN) is calculated by forward calculation |
| end for |
| err (current mean square error) is calculated |
| epoch=1 (current CT, CT is defined as Definition 2) |
| while $err$=err_goal and epoch=max_epoch |
| $\Delta w_{ij}^{(3)}$(epoch), $\Delta w_{ij}^{(2)}$(epoch), $\Delta a_{j}$(epoch), $\Delta b_{j}$(epoch) are calculated |
| $w_{ij}^{(3)}$(epoch), $w_{ij}^{(2)}$(epoch), $a_{j}$(epoch), $b_{j}$(epoch) are calculated |
| for all trainingSet: |
| $y$ is calculated by forward calculation |
| end for |
| err is calculated (updated) |
| epoch++ |
| end while |
| **End**                                    |
Convolutional wavelet neural network (CWN). Based on CNN, the improvement of CWN is that: the fully connected neural network (FCNN) of CNN is replaced by WNN. The structure of CWN has two parts that including convolutional and pooling layer of neural network (CPNN) and WNN. In the hidden layer of the WNN, the activation functions are wavelet scale transformation functions. The structure of CWN is drawn as Fig. 3. Training algorithm of CWN is similar to CNN, while the difference is that FCNN in CNN is replaced by WNN in CWN. The pseudocode of CWN is listed in Algorithm 2.

**Algorithm 2 Pseudocode of CWN method**

| Input: |
| train\_x, train\_y, test\_x and test\_y are set same as those in the pseudocode of CNN |
| Output: |
| wc\_ij, bj;j: weights and bias of CPNN (l = 2, 4, CPNN have 5 layers) |
| w\_ij, a\_ij, bj;j: weights and bias of WNN (WNN have 3 layers) |
| Required parameters: |
| max\_time and target\_error are set same as those in the pseudocode of CNN |
| \( \eta \), CPNN: learning rate of CPNN |
| \( \eta \), WNN, \( \alpha \), WNN: learning rate and inertia coefficient of WNN |
| batch\_size: set the size of batch size for one training cycle |
| Initialization work: |
| r=1 and loss(1) = 1 |
| wc\_ij, bj, w\_ij, a\_ij, bj; weights and bias of CWN (CPNN+WNN) are set as random number. |
| Begin: |
| 1: Set the required parameters and complete the initialization work |
| 2: while \( t < \text{max\_time} \) and loss(\( t \)) > target\_error |
| 3: Get on batched training set (train\_x) and label set (train\_y) according to batch\_size |
| 4: for all trainingSet: |
| 5: train\_p according to train\_x and forward calculation of CWN. |
| 6: end for |
| 7: loss(\( t \)) is calculated as \( \text{loss}(t) = \frac{1}{N} \sum_{n=1}^{N} ((\text{train}\_p(n) - \text{train}\_y(n))^2 \), \( N \) is number of trainingSet. |
| \( \Delta w_{ij}, \Delta a_{ij}, \Delta b_j \) of CPNN and WNN are calculated according to the training algorithms of CPNN. |
| \( \Delta w_{ij}, \Delta b_j \) of CPNN and WNN are calculated according to the training algorithms of CPNN. |
| \( w_{ij}, a_{ij}, b_j \) of WNN are adjusted by the training algorithms of WNN. |
| 9: \( w_{ij}, a_{ij}, b_j \) of CPNN are adjusted according to the training algorithms of CPNN. |
| 10: \( t++ \) |
| 11: end while |
| End |
Wavelet convolutional neural network (WCNN). The improvement of the proposed WCNN is that: the activation function of the convolutional layer in CNN is replaced by the $\Psi()$. The activation function of CNN is sigmoid function, and the $\Psi()$ of WCNN is wavelet scale transformation function.

The structure of proposed WCNN is that: The first part of WCPNN is Wavelet Convolutional Pooling Neural Network (WCPNN), and the second part is Fully Connected Neural Network (FCNN). The structure of WCNN is shown in Fig. 4.

The training process of WCNN is similar to CNN, while the activation function of WCNN is different from CNN. The pseudocode of WCNN is listed in Algorithm 3.

![Figure 4. Structure of WCNN. WCNN is composed of WCPNN and FCNN. The first part of WCPNN is Wavelet Convolutional Pooling Neural Network (WCPNN) and the activation function of WCNN is wavelet scale transformation function.](image-url)
Datasets generation. The data of the first two experiments ("feasibility experiment" and "Hyperparameter optimization experiment") can be generated in the following steps.

The first step is to generate the training set: The training set has two features: x and y. The label of the training set is \( z = f(x, y) \). The relationship between the two-dimensional features and the one-dimensional label can be expressed in Eq. (17):

\[
Z(x, y) = \sin(90 \cdot x) + \cos(90 \cdot y)
\]

According to Eq. (17), features of dataset \( F \) can be generated as Eq. (18). There are 3 different values in \( F \), hence there are \( 3 \times 3 = 9 \) groups of features. Label of dataset \( L \) can be calculated as Eq. (19):

\[
F = \begin{bmatrix}
0 & 0 & 0.5 & 0.5 & 1 & 1 & 1 \\
0 & 0.5 & 1 & 0 & 0.5 & 1 & 0.5 & 1
\end{bmatrix}
\]

\[
L = \begin{bmatrix}
0 & 0.3 & 0.5 & 0.35 & 0.71 & 0.85 & 0 & 0.5 & 1
\end{bmatrix}
\]

The second step is to generate the test set: The features \( F_t \) are designed with 21 different values, hence there are \( 21 \times 21 = 441 \) groups of features. According to Eq. (18), \( F_t \) can be designed as a matrix with 2 rows and 441 columns. Each column of \( F_t \) represent one group of features. The test dataset can be generated by the pseudocode as Algorithm 3.

Datasets of MNISIT and CIFAR-10. MNISIT and CIFAR-10 are widely recognized and adopted in the comparative experiments. MNISIT is well known from the National Institute of Standards and Technology. In MNIST, the training set consists of 250 digits handwritten from different people, and test set consists of the same proportion of digits data. CIFAR-10 is widely used in lots of image classification research. In CIFAR-10, there are 50,000 \( 32 \times 32 \) images in the training set and 10,000 \( 32 \times 32 \) images in the test set. To meet the need of comparison, the size of samples in CIFAR-10 is reshaped to \( 28 \times 28 \). All samples are divided into 10 classes, with 6000 samples in each class. Images in the two datasets are shown in Fig. 5.

Design of simulation

Architecture of WNN. "Feasibility experiment" is designed as follows: Common parameters for all the algorithms are set according to the same rules as follows: Firstly, the initial parameters such as \( w_{jk}, w_{ij} \) are set with the same randomization rules. Secondly, the target precision is set as a very low value (i.e., the target error is set as a very big value), so all the above algorithms are very easy to converge. Thirdly, the training time is set as a large value, so all the above algorithms have enough training time to complete the training. The initialization work for specific parameters is as follows: Firstly, maximum training time limitation is set as a very large number \( (\text{max\_epoch} = 20,000) \) to ensure that BPNN, RBFNN, WNN have enough time to complete the training. Secondly, target error is set as a very large number \( (\text{err\_goal} = 0.1) \) to make the training easy to complete. Thirdly, training processes of each algorithm are repeated 10 times (10 CSPs, CSP is defined in definition 2, i.e., \( \text{case\_repeat} = 10 \)) to observe statistical characteristics. Fourthly, the learning efficiency parameter is set as \( \text{lr} = 0.2 \). Fifthly, the inertia coefficient parameter is set as \( \text{la} = 0.3 \). Many values for \( \text{lr} \) and \( \text{la} \) were tested, where 0.2 and 0.3 were optimal for \( \text{lr} \) and \( \text{la} \), respectively. The termination condition is that: Firstly, when the current training error is less than the target error, the training process is completed. Secondly, when the current training time exceeds the maximum training time limit, the training is stopped.

"Hyperparameter optimization experiment" is designed as follows: Parameters for this simulation are set as follows: Firstly, the initial parameters such as \( w_{jk}, w_{ij} \) were set with the same randomization rules. Secondly, the target precision was set very high to find the highest precision of BPNN, RBFNN, and WNN. Thirdly, the training time was set as a large value, so all the above algorithms have enough training time to complete the training. The initialization work for specific parameters is as follows: Firstly, maximum training time limitation was set as a very large number \( (\text{max\_epoch} = 20,000) \). Secondly, maximum error was set as a very small number \( (\text{err\_goal} = 0.02) \). Thirdly, simulations of BPNN, RBFNN, and WNN were repeated for 10 CSPs \( (\text{case\_repeat} = 10) \). Fourthly, the learning efficiency parameter was set as \( \text{lr} = 0.2 \). Fifthly, the inertia coefficient parameter was set as \( \text{la} = 0.3 \).
values of \( lr \) and \( la \) were tested in different simulations, and the values of \( lr = 0.2 \) and \( la = 0.3 \) listed above are the best. The termination condition is that: Firstly, when the current training error is less than the target error, the training process is completed. Secondly, when the current training time exceeds the maximum training time limit, the training is stopped.

**Architecture of CWNN.** Simulation of CWNN is designed as follows: The network structures and parameters of CNN and CWNN simulation are listed in Table 6. Different values of the learning rate \( \eta \) and coefficient of inertia \( \alpha \) are tested in repetitive simulations, and the values of \( \eta \) and \( \alpha \) listed in Table 6 are the best ones.

**Architecture of WCNN.** Simulation of WCNN is designed as follows: The network structures and parameters of WCNN simulation are listed in Table 7. Different values of the learning rate \( \eta \) and other parameters are tested in repetitive simulations, and the values of \( \eta \) in Table 7 are the best ones.

**Conclusions**

In this paper, WNN, CNN are implemented and CWNN, WCNN are proposed, all of them are simulated and compared. Conclusions are as follows:

Firstly, both the structure of BPNN, the form of activation function in hidden layers of RBFNN and the wavelet transform functions are adopted to the design of WNN. The comparative results of BPNN, RBFNN and WNN are shown in Table 8.

According to Table 8, the following conclusions can be drawn: the mean MSE and mean error rate of WNN are lowest, and the training speed of WNN is the fastest, and the WNN method has no local minimum issue.

Secondly, CWNN is proposed. The fully connected neural network (FCNN) of CNN is replaced by WNN. WCNN is proposed. The activation function of the convolutional layer in CNN is replaced by the wavelet scale transformation function. The comparative simulations between CNN, CWNN and WCNN are shown in Table 9.

According to Table 9, the following conclusions can be drawn: All of CNN, CWNN and CWNN can complete the task of classification on MNIST.

Fifthly, training accuracy of CWNN is higher than CNN and classification ability of CWNN is better than CNN. Fifthly, training accuracy of WCNN is higher than CNN and classification ability of WCNN methods are better than CWNN.

There are still some limitations of our methods although we've made some improvements based on CNN. Firstly, limitations of convolution layer. Back propagation algorithm is not a very efficient learning method, because these algorithms need the support of large-scale data sets. In back propagation algorithm, the parameters near the input layer will be adjusted very slowly when the layers are too deep. Secondly, limitations of pooling layer. A lot of valuable information such as information between the local and the whole will be lost in the pooling layer. Finally, the features extracted from each convolution layer cannot be explained, because neural network is a black box model which is difficult to be explained.

For the further research: Firstly, try to improve the learning ability and learning speed of CNN, WCNN and CWNN by changing the network structure. Secondly, use WCNN and CWNN as neurons to build a more lager and powerful neural network. Thirdly, design more experiments to prove the feasibility and verify the performances of the improved methods and of the work above.
| No. | Parameter type       | Parameter name     | CNN          | CWNN          |
|-----|----------------------|--------------------|--------------|--------------|
| 1   | 1st to 5th layers    | First type of NN   | CPNN         | CPNN         |
| 2   | 2nd, 4th layers      | Activation function of convolutional layer | Sigmoid | Sigmoid |
| 3   | 1st layer            | Dimension of the 1st layer | $28 \times 28$ | $28 \times 28$ |
| 4   | 2nd layer            | Dimension of 1st convolutional layer | $28 \times 28$ | $28 \times 28$ |
| 5   | 3rd layer            | Dimension of 1st pooling layer | $24 \times 24$ | $24 \times 24$ |
| 6   | 2nd, 3rd layers      | Number of features | 6           | 6            |
| 7   | 4th layer            | Dimension of 2nd convolutional layer | $12 \times 12$ | $12 \times 12$ |
| 8   | 5th layer            | Dimension of 2nd 1 pooling layer | $8 \times 8$ | $8 \times 8$ |
| 9   | 4th, 5th layers      | Number of features | 12          | 12           |
| 10  | – 1st, – 2nd, – 3rd layers | Second type of NN | FCNN         | WNN          |
| 11  | – 3rd layer          | Dimension of -3rd input layer | 192         | 192         |
| 12  | – 2nd layer          | Dimension of -2nd hidden layer | None | 50 |
| 13  | – 2nd layer          | Activation function of hidden layer | None | Wavelet |
| 14  | – 1st layer          | Dimension of – 1st output layer | 10            | 10            |
| 15  | – 1st layer          | Activation function of output layer | Sigmoid | Sigmoid |
| 16  | Hyperparameters      | Learning rate $\eta$ | 0.1          | 0.1          |
| 17  | Hyperparameters      | Coefficient of inertia $\alpha$ | None | 0.2 |
| 18  | Hyperparameters      | max_SPs            | 10           | 10           |
| 19  | Hyperparameters      | max_ACs            | 6000         | 6000         |
| 20  | Hyperparameters      | target_err         | 0.0000001    | 0.0000001    |
| 21  | Hyperparameters      | BatchSize          | 10           | 10           |

Table 6. Configurations of CNN, CWNN experiments.

| No. | Parameter type       | Parameter name     | WCNN          |
|-----|----------------------|--------------------|--------------|
| 1   | 1st to 5th layers    | First type of NN   | WCPNN         |
| 2   | 2nd, 4th layers      | Activation function of convolutional layer | Wavelet |
| 3   | 1st layer            | Dimension of the 1st layer | $28 \times 28$ |
| 4   | 2nd layer            | Dimension of 1st convolutional layer | $28 \times 28$ |
| 5   | 3rd layer            | Dimension of 1st pooling layer | $24 \times 24$ |
| 6   | 2nd, 3rd layers      | Number of features | 6            |
| 7   | 4th layer            | Dimension of 2nd convolutional layer | $12 \times 12$ |
| 8   | 5th layer            | Dimension of 2nd 1 pooling layer | $8 \times 8$ |
| 9   | 4th, 5th layers      | Number of features | 12           |
| 10  | – 1st, – 2nd, – 3rd layers | Second type of NN | FCNN         |
| 11  | – 3rd layer          | Dimension of -3rd input layer | 192         |
| 12  | – 2nd layer          | Dimension of -2nd hidden layer | None | None |
| 13  | – 2nd layer          | Activation function of hidden layer | None | None |
| 14  | – 1st layer          | Dimension of -1st output layer | 10            |
| 15  | – 1st layer          | Activation function of output layer | Sigmoid |
| 16  | Hyperparameters      | Learning rate $\eta$ | 0.1          |
| 17  | Hyperparameters      | Coefficient of inertia $\alpha$ | None |
| 18  | Hyperparameters      | max_SPs            | 10           |
| 19  | Hyperparameters      | max_ACs            | 6000         |
| 20  | Hyperparameters      | target_err         | 0.0000001    |
| 21  | Hyperparameters      | BatchSize          | 10           |

Table 7. Configurations of WCNN experiments.
Table 8. Comparative results of BPNN, RBFNN and WNN on generation dataset.

| Statistics       | BPNN    | RBFNN   | WNN    |
|------------------|---------|---------|--------|
| Maximum MSE      | 0.00707 | 0.068413| 0.002192|
| Minimum MSE      | 0.00061 | 0.00005 | 0.00003 |
| Mean MSE         | 0.00061 | 0.011274| 0.000351|
| Maximum error rate| 0.083616| 0.762847| 0.109309|
| Minimum error rate| 0.052301| 0.015604| 0.019316|
| Mean error rate  | 0.070413| 0.220663| 0.036454|
| Mean training times | No case success | 9631 | 4963 |

Table 9. Comparative results of CNN, CWNN and WCNN on MNIST dataset.

| Statistics       | CNN     | CWNN    | WCNN   |
|------------------|---------|---------|--------|
| Maximum MSE      | 0.1759  | 0.201   | 0.3456 |
| Minimum MSE      | 0.1532  | 0.125   | 0.03456|
| Mean MSE         | 0.1643  | 0.1414  | 0.0885 |
| Maximum error rate| 0.1807  | 0.2453  | 0.287  |
| Minimum error rate| 0.1499  | 0.1068  | 0.0366 |
| Mean error rate  | 0.1643  | 0.1291  | 0.0772 |

References

1. Zhu, N. et al. Application of artificial neural networks to predict multiple quality of dry-cured ham based on protein degradation. Food Chem. https://doi.org/10.1016/j.foodchem.2020.125886 (2021).
2. Di Piazza, A., Di Piazza, M. C., La Tona, G. & Luna, M. An artificial neural network-based forecasting model of energy-related time series for electrical grid management. Math. Comput. Simul. 184, 294–305. https://doi.org/10.1016/j.matcom.2020.05.010 (2021).
3. Apostolopoulos, I. D. & Mpesiana, T. A. Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks. Phys. Eng. Sci. Med. 43, 635–640. https://doi.org/10.1017/s1355612620000865 (2020).
4. Zhang, X., Zhao, Z., Zheng, Y. & Li, J. Prediction of taxi destinations using a novel data embedding method and ensemble learning. IEEE Trans. Intell. Transp. Syst. 21, 68–78. https://doi.org/10.1109/tits.2018.2885887 (2020).
5. Li, J., Yao, X., Wang, X., Yu, Q. & Zhang, Y. Multiscale local features learning based on BP neural network for rolling bearing intelligent fault diagnosis. J. Clean. Prod. https://doi.org/10.1016/j.jclepro.2019.118671 (2020).
6. Sun, W. & Huang, C. A carbon price prediction model based on secondary decomposition algorithm and optimized back propagation neural network. IEEE Trans. Power Electron. https://doi.org/10.1109/tpele.2020.3064357 (2020).
7. Langer, S. Analysis of the rate of convergence of fully connected deep neural network regression estimates with smooth activation function. J. Multivar. Anal. https://doi.org/10.1016/j.jmva.2020.104695 (2021).
8. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. & Yuille, A. L. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans. Pattern Anal. Mach. Intell. 40, 834–848. https://doi.org/10.1109/t pami.2017.2699184 (2018).
9. Kamnitas, K. et al. Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. Med. Image Anal. 36, 61–78. https://doi.org/10.1016/j.media.2016.10.004 (2017).
10. He, Z. et al. Deep transfer multi-wavelet auto-encoder for intelligent fault diagnosis of gearbox with few target training samples. Knowl. Based Syst. https://doi.org/10.1016/j.knosys.2019.105313 (2020).
11. Sabir, Z., Wahab, H. A., Umar, M., Sakar, M. G. & Raja, M. A. Z. Novel design of Morlet wavelet neural network for solving second order Lane-Emden equation. Math. Comput. Simul. 172, 1–14. https://doi.org/10.1016/j.matcom.2020.01.003 (2020).
12. Jin, X., Liu, Q. & Long, H. Impact of cost-benefit analysis on financial benefit evaluation of investment projects under back propagation neural network. J. Comput. Appl. Math. https://doi.org/10.1016/j.cam.2020.113172 (2021).
13. Li, Y. et al. Clinicopathological features of fibrosarcomatous dermatofibrosarcoma protuberans and the construction of a back-propagation neural network recognition model. Orphanet J. Rare Dis. https://doi.org/10.1186/s13023-021-01698-4 (2021).
14. Wang, T., Gao, H. J. & Qiu, J. B. A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control. IEEE Trans. Neural Netw. Learn. Syst. 27, 416–425. https://doi.org/10.1109/tnnls.2015.2411671 (2016).
15. Baghaee, H. R., Mirsaleim, M. & Gharehpetian, G. B. Power calculation using RBF neural networks to improve power sharing of hierarchical control scheme in multi-DER microgrids. IEEE J. Emerg. Select. Top. Power Electron. 4, 1217–1225. https://doi.org/10.1109/jestpe.2016.2581762 (2016).
16. Shi, B. G., Bai, X. & Yao, C. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. 39, 2298–2304. https://doi.org/10.1109/tpami.2016.2646371 (2017).
17. Shi, P., Li, F. B., Wu, L. G. & Lim, C. C. Neural network-based passive filtering for delayed neutral-type semi-Markovian jump systems. IEEE Trans. Neural Netw. Learn. Syst. 28, 2101–2114. https://doi.org/10.1109/tnnls.2016.2573853 (2017).
18. Smolik, M. & Skala, V. Radial basis function and multi-level 2D vector field approximation. Math. Comput. Simul. 181, 522–538. https://doi.org/10.1016/j.matcom.2020.10.009 (2021).
19. Baklacioglu, T. Predicting the fuel flow rate of commercial aircraft via multilayer perceptron, radial basis function and ANFIS artificial neural networks. Aeronaut. J. 125, 453–471. https://doi.org/10.1017/aer.2020.119 (2021).
20. Armenteros, J. J. A. et al. SignalP 5.0 improves signal peptide predictions using deep neural networks. *Nat. Biotechnol.* **37**, 420. https://doi.org/10.1038/s41598-019-0036-z (2019).

21. Hannun, A. Y. et al. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nat. Med.* https://doi.org/10.1038/s41591-018-0268-3 (2019).

22. Huang, F. M. et al. A deep learning algorithm using a fully connected sparse autoencoder neural network for landslide susceptibility prediction. *Landslides* **17**, 217–229. https://doi.org/10.1007/s10346-019-01274-9 (2020).

23. Yang, Q., Zhang, J. H., Wang, X. F. & Wei-Guo, L. I. Wind speed and generated wind power forecast based on wavelet-neural network. *Power Syst. Technol.* **1**, 1–10 (2009).

24. Qiao, W. & Yang, Z. Forecast the electricity price of US using a wavelet transform-based hybrid model. *Energy* **193**, 511–530. https://doi.org/10.1016/j.energy.2019.116704 (2020).

25. Rezamand, M., Kordestani, M., Carriere, R., Ting, D. S. K. & Safi, M. A new hybrid fault detection method for wind turbine blades using recursive PCA and wavelet-based PDF. *IEEE Sens. J.* **20**, 2023–2033. https://doi.org/10.1109/jSEN.2019.2948997 (2020).

26. Sharma, V., Yang, D., Walsh, W. & Reidl, T. Short term solar irradiance forecasting using a mixed wavelet neural network. *Renew. Energy* **90**, 481–492. https://doi.org/10.1016/j.renene.2016.01.020 (2016).

27. Yang, H.-J. & Hu, X. Wavelet neural network with improved genetic algorithm for traffic flow time series prediction. *Optik* **127**, 8103–8110. https://doi.org/10.1016/j.ijleo.2016.06.017 (2016).

28. Bai, Y. et al. Air pollutants concentrations forecasting using back propagation neural network based on wavelet decomposition with meteorological conditions. *Atmos. Pollut. Res.* **7**, 557–566 (2016).

29. Bai, Y., Li, Y., Wang, X., Xie, J. & Li, C. Air pollutants concentrations forecasting using back propagation neural network based on wavelet decomposition with meteorological conditions. *Atmos. Pollut. Res.* **7**, 557–566. https://doi.org/10.1016/j.apr.2016.01.004 (2016).

30. Wang, B., Gu, X., Ma, L. & Yan, S. Temperature error correction based on BP neural network in meteorological wireless sensor network. *Int. J. Sens. Netw.* **23**, 265–278. https://doi.org/10.1504/ijssnet.2017.083532 (2017).

31. Fang, Y., Fei, J. & Ma, K. Model reference adaptive sliding mode control using RBF neural network for active power filter. *Int. J. Electr. Power Energy Syst.* **73**, 249–258. https://doi.org/10.1016/j.ijepes.2015.05.009 (2015).

Acknowledgements

This work was supported by Basic Research Funds of Capital University of Economics and Business of Beijing universities; Scientific Research Project of Beijing Municipal Education Commission [Grant No. SM202110038009].

Author contributions

J.L. proposed the methodology. P.L., X.T. and J.C. conducted experiments and analyzed the experimental results. J.L. wrote the main manuscript text. All authors reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to J.L.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2021