Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 9362283, 7 pages
https://doi.org/10.1155/2022/9362283

Research Article

Application of Improved Deep Belief Network Based on Intelligent Algorithm in Stock Price Prediction

Hongxia Zhu and Liqiang Fan

College of Science, LangFang Normal College, Langfang 065000, Hebei, China

Correspondence should be addressed to Hongxia Zhu; 1201423@lfnu.edu.cn

Received 17 June 2022; Accepted 21 July 2022; Published 28 August 2022

Academic Editor: Zaoli Yang

In order to improve the prediction accuracy of stock price, an improved model QPSO-DBN-GABP using quantum particle swarm optimization algorithm to optimize deep belief network is proposed. In this model, the quantum particle swarm optimization algorithm is used to find the optimal combination of the number of neurons in each layer of RBM, and the genetic algorithm is used to optimize the initial weight and threshold of BP neural network, so as to obtain the optimized combination prediction model. The prediction results are compared with those of DBN, PSO-DBN, and QPSO-DBN models. Through the comparison of experimental results, it is found that compared with the above three prediction models, the prediction error index RMSE of the model is reduced by about 10.1%, 9.1%, 1.3%, and MAE is reduced by 8.1%, 5.7%, and 0.67%. The prediction accuracy of the model is improved to 96.435%.

1. Introduction

In the stock market, the future trend of the stock price has always been the focus of investors. Therefore, how to predict stock price more accurately has become the research direction of many scholars. At present, the common prediction methods of stock price mainly include statistical analysis method and machine learning method. The statistical analysis method mainly uses the least square to construct various regression models. For example, literature [1] carried on the nationality test and the unit root test to the time series establishes the ARIMA model of the stationary time aligned sequence obtained after the difference of the nonstationary sequence; by analyzing the current exponential fluctuation, the time series model is used to predict the exponential fluctuation in the next year. Literature [2] used a hybrid approach of the adaptive wavelet transform, long short-term memory, and ARIMA-GARCH family models for the stock index prediction. However, because these models assume too many preconditions and have a low ability to deal with nonlinear problems, and the influencing factors of stock price are generally strong nonlinear relationship, the scope of application of these regression models is limited. Due to its strong nonlinear processing ability, the machine learning method has been widely used in stock price prediction in recent years; literature [3] used Long Short-Term Memory in stock prices forecasting in Indonesia during Covid-19. In their experiments, they visualized the data using data science and predicted and simulated the important prices called Open, High, Low, and Closing (OHLC) with various parameters; In paper [4], a data set was created from the Indian stock market and an LSTM model was developed for it. It was then optimized by comparing stateless and fateful models and by tuning for the number of hidden layers. Under the condition of complex influencing factors and a large amount of data, the accuracy of the above model for stock prediction needs to be further improved. Literature [5] used Deep Belief Neural Network (DBN) with Atom Search Optimization optimization and good results were obtained in student performance prediction. Because the prediction result of DBN is affected by the number of neurons in each layer of RBM (Restricted Boltzmann Machine), if the number of RBM neurons is manually selected, it is easy to fall into local optimization due to the large number of selection parameters. Particle swarm and its improved algorithm are widely used in global optimization problems.

This paper proposes an improved model for stock price prediction, which is different from the previous research methods. The model constructed in this paper can automatically select a better number of DBN nodes, which is rare in previous studies. This paper mainly uses MATLAB to create a model for stock forecasting and uses RMSE and MAE as indicators for comparative analysis. Based on traditional DBN, Particle Swarm Optimization (PSO) DBN, Quantum Particle Swarm Optimization(QPSO) DBN [6–8], and Quantum Particle Swarm Optimization(QPSO) DBN at the same time Genetic Algorithm(GA) is used to optimize BP model proposed in this paper. Through the comparative analysis of the above-given evaluation indicators, the better model is selected as the prediction stock price model, so as to reduce the prediction error of stock price.

2. An Improved Depth Belief Network Model for Stock Price Forecasting

2.1. PSO Principle. PSO is an evolutionary computing technology, which originates from the research on the predation behavior of birds. PSO designs a classless particle to simulate the birds in the bird swarm. The particle has only two attributes: speed and position. Speed represents the speed of movement and position represents the direction of movement. Each particle separately searches for the optimal solution in the search space, records it as the current individual extreme value, shares the individual extreme value with other particles in the whole particle swarm, and finds the optimal individual extreme value as the current global optimal solution of the whole particle swarm. All particles in the particle swarm adjust their speed and position according to the current individual extreme value found by themselves and the current global optimal solution shared by the whole particle swarm.

Suppose a population $G = (G_1\ , G_2\ , \ldots, G_M)$ composed of M particles in a N dimensional search space, in which the ith particle is represented as a N dimensional vector $G_i = (g_{i1}, g_{i2}, \ldots, g_{iN})$, which represents the position of the ith particle in the N dimensional search space and a potential solution of the problem. The fitness value corresponding to the position of each particle G_i can be calculated according to the objective function. The velocity of the ith particle is $V_i = (v_{i1}, v_{i2}, \ldots, v_{iN})$, its individual extreme value is $P_i = (p_{i1}, p_{i2}, \ldots, p_{iN})$, and the population extreme value is $P_i = (p_{i1}, p_{i2}, \ldots, p_{iN})$. In each iteration, particles update their speed and position through individual extreme and group extreme, that is,

$$\begin{aligned} v_{i\,d}^{k+1} &= \omega v_{i\,d}^{k} + c_1 r_1\left(p_{i\,d}^{k} - g_{i\,d}^{k}\right) + c_1 r_2\left(p_{g\,d}^{k} - g_{i\,d}^{k}\right),\\ g_{i\,d}^{k+1} &= g_{i\,d}^{k} + v_{i\,d}^{k+1}, \end{aligned} \tag{1}$$

where ω is inertia weight; $d = 1, 2, \ldots, N$; $i = 1, 2, \ldots, M$, k is the current number of iterations; $v_{i\,d}^{k}$ is the velocity of particles; c_1, c_2 is a nonnegative constant, called the acceleration factor, r_1, r_2 is a random number distributed in an interval $[0, 1]$.

2.2. QPSO Principle. As an optimization algorithm of particle swarm algorithm, a quantum particle swarm optimization algorithm in quantum space based on quantum theory is an uncertain search algorithm, which can more likely search for the best advantage that is not near the optimal solution of the ith particle from the beginning of the iteration to the current number of iterations and the current optimal solution of the particle population. Verified by several benchmark functions, the performance of the algorithm is obviously better than the ordinary PSO algorithm in all aspects [9].

In the QPSO algorithm, the population size is set as M, in the process of evolution, the particles add or subtract with a certain probability to update the position of each particle and generate a new particle population, which is determined by formulas (2)–(5):

$$p = a * \text{pbest}(i, :) + (1 - a) * \text{gbest}, \tag{2}$$

$$\text{mbest} = \frac{\text{sum}(\text{pbest})}{\text{popNum}}, \tag{3}$$

$$b = 1 - \frac{\text{step}}{\text{Maxstep}} * 0.5, \tag{4}$$

$$\begin{aligned} \text{pop}(i, :) &= p + b * \text{abs}(\text{mbest} - p) * \log\left(\frac{1}{u}\right)\\ &\quad * (1 - 2 * (u \geq 0.5)), \end{aligned} \tag{5}$$

where a and u are random numbers between 0 and 1, step is the current iteration number, Maxstep is the maximum iteration number, popNum is the population size, $m$best is the average value of the historical extreme value of contemporary particle swarm, b is the contraction and expansion coefficient, which decreases linearly in the convergence process of QPSO, $p$best$(i, :)$ is the historical extreme value of the ith particle, $g$best is the global extreme value of particle swarm, and step is the current evolutional generation, sum represents a function that sums all components in a vector, abs stands for finding the absolute value and log represents natural logarithm.

2.3. GA. The basic principle of GA is to imitate the evolutionary law of "natural selection and survival of the fittest" in the biological world. Professor J. Holland of Michigan University proposed the algorithm in 1967.

GA starts with an initial population, which is composed of a certain number of individuals encoded by genes. After the initial population is generated, the genetic operator is used to generate the offspring population representing the new solution set, and the better approximate solution is obtained through generation-by-generation evolution. In each generation, individuals are selected according to the fitness of individuals in the solution space. The greater the chance of individuals with high fitness to be selected into the next generation, and the offspring population representing the new solution set is generated through crossover and mutation operations.

The basic steps of a genetic algorithm are as follows:

Step 1. Coding: GA should first encode the actual problem and express the problem in string. This string is equivalent to a chromosome in genetics. The sum of individual strings generated by each generation is called the group. For the convenience of computer implementation, the string length is usually fixed, and the character is 0 or 1.

Step 2. Generation of initial population: randomly generate N string structure data, each string structure data becomes an individual, and N individuals constitute the initial population of the genetic algorithm. The genetic algorithm starts evolution with this initial population as the initial point.

Step 3. Fitness evaluation: the fitness function needs to be defined according to specific problems. The larger the adaptation, the better the individual, and the smaller the individual, the worse.

Step 4. Selection: selection is also called replication. Replication is the basic operator of GA. It reproduces excellent individuals in the next generation of a new population, reflecting the natural selection principle of "survival of the fittest." Common selection methods include roulette selection, tournament selection, and random competition.

Step 5. Crossover: crossover operator, also known as gene recombination, plays a central role in GA. Cross operation is performed on two individuals who have undergone a selection operation. The crossover method adopted shall be able to transfer the characteristics of the parent string to the child string. The substring should be able to partially or completely inherit the structural characteristics and effective genes of the parent string. Cross reflects the idea of information exchange.

Step 6. Variation: firstly, an individual is randomly selected in the population. For the selected individual, the value of a string in the string structure data is randomly changed with a certain probability. As in the biological world, the probability of variation in GA is very low, and the value is usually very small.

Step 7. The initial population obtains the next generation population through the above steps and then judges whether the end condition is met. If so, a better approximate solution is output, If not, repeat steps 3– 7.

2.4. BP Neural Network. BP neural network is the learning process of error back propagation algorithm. The learning process is mainly composed of two processes: forward transmission of information and back propagation of error. BP neural network is a three-layer network.

Input layer: each neuron in the input layer is responsible for receiving input information from the outside and transmitting it to each neuron in the middle layer.

Hidden layer: the middle layer is the internal information processing layer, which is responsible for information transformation. According to the requirements of information changeability, the middle layer can be designed as a single hidden layer or multihidden layer structure. The information transmitted from the last hidden layer to the neurons of the output layer is further processed to complete a learning forward propagation process.

Output layer: the output layer outputs information processing results to the outside world.

When the actual output is inconsistent with the expected output, it enters the back propagation stage of error. The error passes through the output layer, modifies the weight and threshold of each layer in the way of error gradient descent, and transmits it back to the hidden layer and the input layer. The repeated process of information forward transmission and error reverse transmission is not only the process of continuously adjusting the weight and threshold of each layer but also the process of neural network learning and training. This process continues until the error of network output is reduced to an acceptable level or the preset learning times. Because the initial weight and threshold of the BP neural network are given randomly, and the output error of the network will affect the initial weight and threshold; GA is used to find a better initial weight and threshold and assign it to the BP network, so as to achieve small error and improve the accuracy of prediction. The process of optimizing the BP neural network by GA is shown in Figure 1.

2.5. DBN. DBN is a probability generation model. Compared with the neural network of the traditional discrimination model, the generation model establishes a joint distribution between observation data and labels, and evaluates both P (observation|label) and P (label|observation), while the discrimination model only evaluates the latter, that is, P (label|observation). The whole DBN regression prediction model is composed of several superimposed RBMs and a layer of the BP network. Each RBM network has two layers. The first layer is called the visual layer, which is generally the input layer, and the other layer is the hidden layer, that is, the feature extraction layer we generally refer to. The structure of the DBN regression prediction model is shown in Figure 2.

The training process in DBN can be divided into two steps: the first is the pretraining stage, which trains each layer of the RBM network in an unsupervised way to ensure that the feature vectors are mapped to different feature spaces and retain the feature information as much as possible. Then, there is the fine-tuning stage. In this section, the last layer of DBN is the BP network, and the output eigenvector of the last layer of RBM is used as its input eigenvector for supervised training.

2.6. Process of Optimizing DBN. The number of neuron nodes in each hidden layer in the DBN network has an important impact on the experimental results. On the one hand, too few hidden layer nodes will not fully learn the

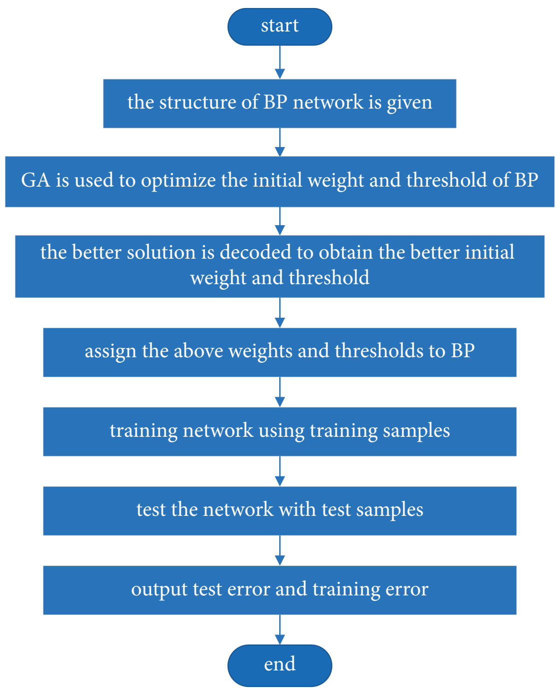
start
the structure of BP network is given
GA is used to optimize the initial weight and threshold of BP
the better solution is decoded to obtain the better initial weight and threshold
assign the above weights and thresholds to BP
training network using training samples
test the network with test samples
output test error and training error
end

Figure 1: The process of optimizing BP neural network by GA.

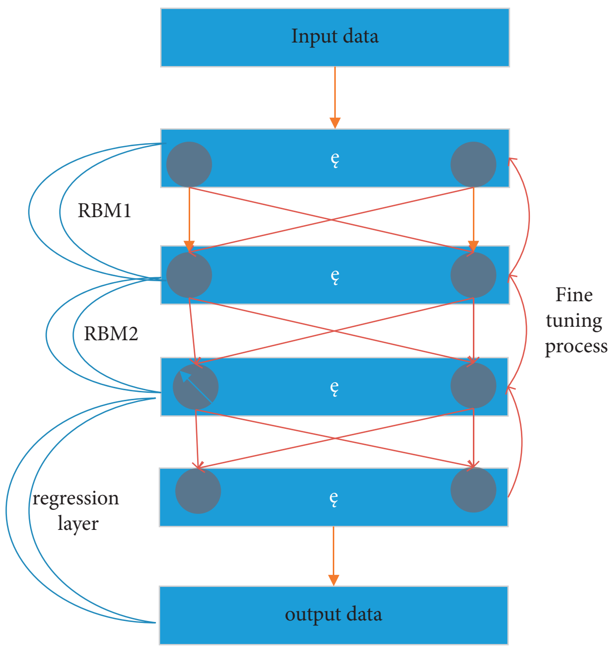

Figure 2: The network structure of DBN.

nonlinear relationship between input and output, in which case the prediction result error is large; on the other hand, too many hidden layer nodes will increase the learning time of the model and may even cause the model to collapse, unable to obtain the optimal solution of the model. Usually, the number of neurons in each layer is artificially set according to experience through experimental comparison. Obviously, this method is not only time-consuming and laborious but also not easy to find the optimal solution. In

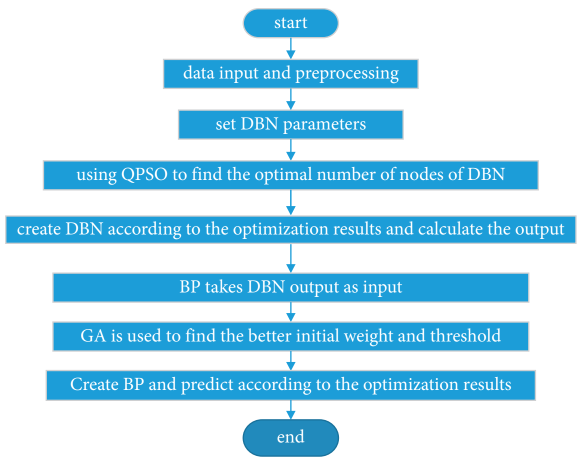
start
data input and preprocessing
set DBN parameters
using QPSO to find the optimal number of nodes of DBN
create DBN according to the optimization results and calculate the output
BP takes DBN output as input
GA is used to find the better initial weight and threshold
Create BP and predict according to the optimization results
end

Figure 3: Flow-process diagram of the QPSO-DBN-GABP.

this paper, by setting the selection range of the number of neuron nodes in 3 to 5 hidden layers, the better number of layers is selected by comparison; Then, the global optimization ability of the quantum particle swarm optimization algorithm is used to find the combination of the number of neurons that make the output error of the model smaller; that is, QPSO is used to optimize the combination of the number of nodes in each layer of DBN with the selected number of layers. At the same time, a genetic algorithm is used to optimize the connection weight and threshold of the regression layer, that is, the BP layer. This optimization model is abbreviated as QPSO-DBN-GABP. The optimization process is shown in Figure 3. The specific steps are as follows:

Step 1. Setting of DBN parameters: the maximum number of iterations of training RBM, the number of blocks of RBM training samples, the random sample size each time, and the value range of the number of neuron nodes in the hidden layer of DBN.

Step 2. In the QPSO algorithm, a random solution x is generated between the maximum and minimum values of the hidden layer set in equation (1), and the reciprocal of the mean square error tested in the DBN model determined by x is calculated as a fitness function, i.e.,

$$f(x) = \frac{1}{\sum_{i=1}^{N} \left(y_i - \widehat{y}_i\right)^2 / N}, \tag{6}$$

where N is the total number of samples, y_i and $\widehat{y}_i$ are the actual and predicted values of the ith sample.

Step 3. Calculate the fitness of each particle in the particle swarm, record the individual extreme value and global extreme value of the particle swarm respectively, and update the position pop (i,) of each particle in the current particle swarm according to equations (2)–(5).

Step 4. The fitness of each particle in the new particle swarm is calculated according to the fitness function formula, and the individual extremum and global extremum are updated; judge whether the iteration conditions for exiting the loop are met. If so, exit the loop, otherwise, return to step 3.

Step 5. The optimal solution of the problem is output and assigned to the DBN network.

Step 6. BP network takes the output of the DBN network as the input, takes finding the best parameters through GA as the initial weight and threshold to predict the test data, and outputs the prediction performance evaluation index.

3. Stock Price Forecast and Analysis

3.1. Data Source and Preprocessing. Stock data selects the historical trading data of a listed company in China over a period of time (the data comes from the Forecaster Network). A total of 2786 data were obtained, according to the principle that the number of training samples is between 2/3 and 4/5 [10], the training set is provided with 2100 data and the testing set is provided with 686 data, and for the sake of ensuring the randomness, the data set is randomly divided into the training set and test set. According to literature [11, 12], the input variables are preliminarily set as opening price, highest price, lowest price, closing price, transaction amount, business volume, turnover rate, rise or fall on the nth day, and closing price in n to $n-5$ days. Then, using the calculation of the Spearman coefficient in the feature selection method, we know that the correlation coefficients of these variables and output variables are 0.0098, −0.2002,00846, −0.09, which are far less than 0.9, so these can not be used as input variables. Finally, the selected input variable are opening price, highest price, lowest price, and closing price in n to 4/5 days. The output variable is the closing price of the stock on the day $n+1$.

Because different units or orders of magnitude of input and output data may cause problems such as slower model convergence and longer training time, it is necessary to normalize the sample data. The normalization formulas are as follows:

$$P = \frac{p - p_{\min}}{p_{\max} - p_{\min}}, \tag{7}$$

where P is the normalized value, $p_{\min}$ is the minimum value in the dataset to be normalized, $p_{\max}$ is the maximum value in the dataset to be normalized, and p is any value in the dataset to be normalized.

3.2. Model Evaluation Index. To evaluate the performance of different models, two evaluation indexes, root mean square error (RMSE) and mean absolute error (MAE), were used to evaluate the model performance. The calculation formulas for RMSE and MAE are as follows:

$$\mathrm{RMSE} = \sqrt{\sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i)^2}{n, \mathrm{MAE} = \frac{1}{n} \sum_{i=1}^{n} \left| y_i - \hat{y}_i \right|,} \right.} \tag{8}$$

Table 1: Error comparison of implicit layer 3.

Number of nodes	RMSE	MAE
[20, 30]	0.12438	0.07692
[20, 50]	0.11778	0.07421
[20, 70]	0.11036	0.07117

Table 2: Error comparison of implicit layer 4.

Number of nodes	RMSE	MAE
[20, 30]	0.12306	0.07705
[20, 50]	0.12271	0.07659
[20, 70]	0.11776	0.07514

Table 3: Error comparison of implicit layer 5.

Number of nodes	RMSE	MAE
[20, 30]	0.15111	0.09775
[20, 50]	0.13611	0.08408
[20, 70]	0.12777	0.08243

where $\hat{y}_i$ is the predicted value of the test set, y_i is the true value of the test set, and n is the number of test sets.

3.3. Experimental Results. The configuration of the experimental computer: CPU is Intel i7-3770H, internal memory is 8 GB, algorithm model operation software is MATLAB 2016a. There are 9 neurons in the input layer of DBN. Considering the efficiency and accuracy, when the value of the hidden layer is in three to five layers, and the range of the number of neurons in the hidden layer is [20, 30], [20, 50], [20, 70], the stock prediction algorithm based on DBN is selected by experimental comparison, and then the prediction accuracy of DBN, PSO-DBN, QPSO-DBN, and QPSO-DBN-GABP is compared. The main parameters of PSO, QPSO, and GA are as follows: the population size of PSO and QPSO is 40, the number of iterations is 50; GA has a population size of 30 and a genetic algebra of 10. First, the average error results of 10 DBN runs at different levels are given, as shown in Tables 1–3.

Based on the results of the previous three tables, we can see that when the number of RBM is 3 layers, the error evaluation index of RMSE and MAE with the value range of RBM node number in [20, 70] is 0.11036 and 0.07117 which are smaller than other hidden layer node number [20, 30] and [20, 50]. When the number of RBM is 4 or 5, each error evaluation index increases sharply, the smallest one is 0.11776 (RMSE) and 0.07514(MAE), which is about 7% and 5% larger than that of the 3 layers with the value range of RBM node number in [20,70] so the prediction of true value cannot be achieved. So the hidden layer of the model is 3 and the number of hidden layer nodes ranges from [20,70].

From the above-given data processing results, and in order to compare the prediction performance of the QPSO-DBN-GABP model, the same set of data is modeled and

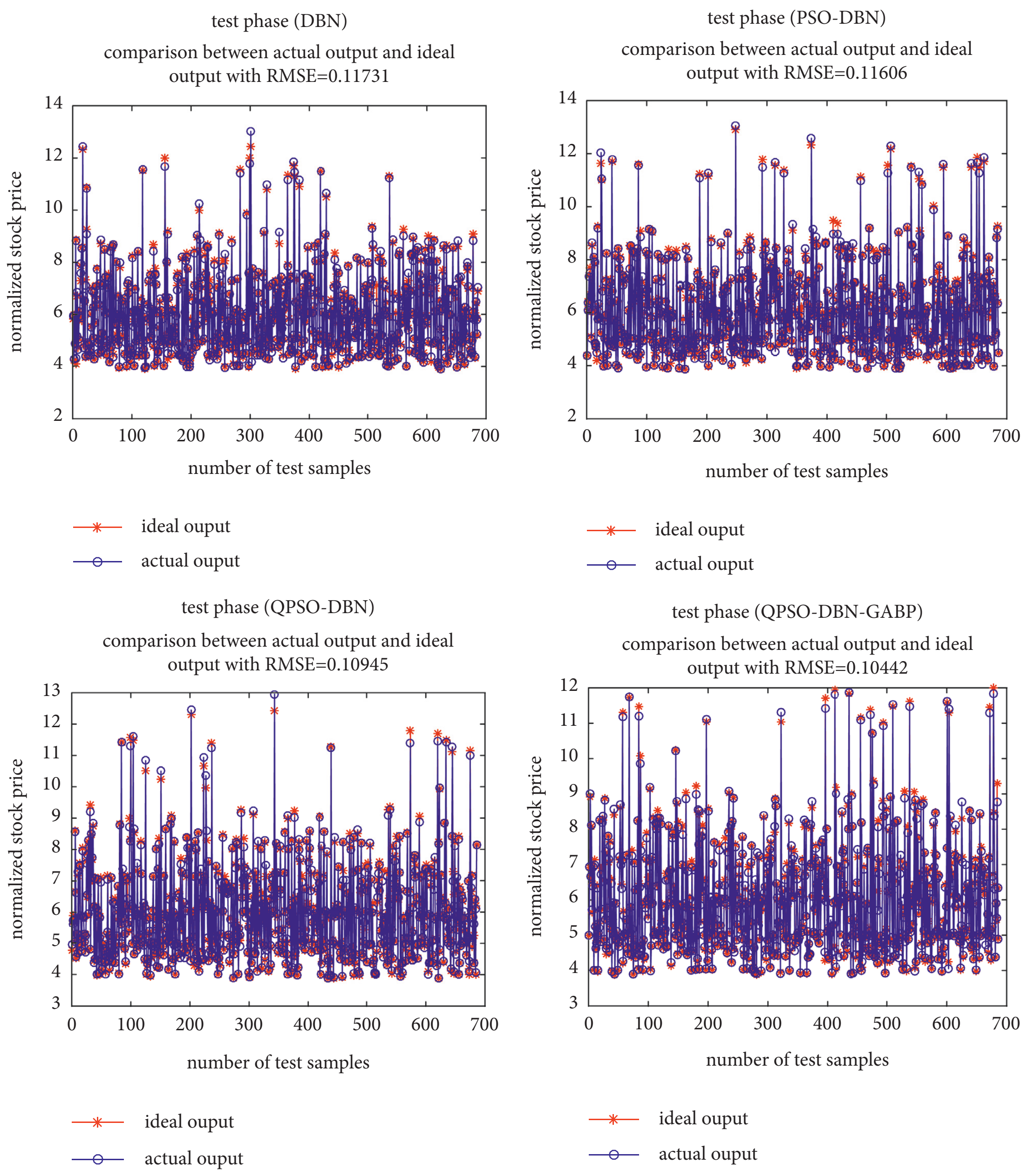

Figure 4: Comparison of prediction results of each model.

Table 4: The error evaluation index.

Evaluating indicator	RMSE	MAE
DBN	0.1163	0.0746
PSO-DBN	0.1153	0.07292
QPSO-DBN	0.10706	0.06946
QPSO-DBN-GABP	0.10566	0.069

predicted using DBN, PSO-DBN, and QPSO-DBN. The prediction curves of each model are shown in Figure 4. The prediction simulation results (Table 4) of the above four models show that the QPSO-DBN-GABP model performs best: compared with DBN, PSO-DBN, and QPSO-DBN model. The RMSE of QPSO-DBN-GABP model predictions decreased about by 10.1%, 9.1%, 1.3%, and MAE by 8.1%, 5.7%, and 0.67%, respectively.

4. Conclusions

As a probability generating model, DBN has a wide range of applications. When using this network to predict the stock price, it is composed of multiple RBMs and a BP at this time. However, the prediction result is related to the number of RBMs, the number of nodes in each layer, and the selection of BP's initial weight and threshold. In order to improve the accuracy of prediction, this paper proposes an improved DBN model. Firstly, for the data used this time, the better RBM layer is selected as three layers through the experimental results. The value range of the number of nodes in each layer is [20,70]. Secondly, the quantum particle swarm optimization algorithm is used to seek a better combination of the number of nodes in each layer, and the DBN is established by using the selected combination of the number of nodes. Then, the genetic algorithm is used to search for the better initial weight and threshold of the BP department. Finally, the obtained parameter combination model is established by using the training data set, and the test data set is used for testing. The simulation results show that this combined forecasting model has a smaller error and higher accuracy than DBN, PSO-DBN, and QPSO-DBN when forecasting stock prices.

Data Availability

The authors confirm that the data supporting the findings of this study are available within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by Langfang Science and Technology Research and Development Self Financing Project (Grant no. 2019011008)] and Langfang Education Big Data Analysis Laboratory and Langfang Key Laboratory of Education Big Data Analysis and Key Technology.

References

[1] X. Zhang, R. Zhai, and W. Gao, “Analysis and economic prediction of stock index based on index tracking and ARIMA model,” *Journal of Physics: Conference Series*, vol. 1903, no. 1, pp. 012015–12024, 2021.

[2] M. Zolfaghari and S. Gholami, “A hybrid approach of adaptive wavelet transform, long short-term memory and ARIMA-GARCH family models for the stock index prediction,” *Expert Systems with Applications*, vol. 182, no. 1, p. 115226, 2021.

[3] W. Budiharto, “Data science approach to stock prices forecasting in Indonesia during Covid-19 using Long Short-Term Memory (LSTM),” *Journal of Big Data*, vol. 8, no. 1, pp. 47–49, 2021.

[4] A. Yadav, C. K. Jha, and A. Sharan, “Optimizing LSTM for time series prediction in Indian stock market,” *Procedia Computer Science*, vol. 167, no. C, pp. 2091–2100, 2020.

[5] S. Surenthiran, R. Rajalakshmi, and S. S. Sujatha, “Student performance prediction using Atom search optimization based deep belief neural network,” *Optical Memory & Neural Networks*, vol. 30, no. 2, pp. 157–171, 2021.

[6] Xi Zhang, J. Zhang, Y. Hu, T. Tang, J. Yang, and Y. Zhang, “Structural damage recognition based on the finite element method and quantum particle swarm optimization algorithm,” *IEEE Access*, vol. 8, no. 1, pp. 184785–184792, 2020.

[7] Y. Chen and D. Wang, “Forecasting by designing Mamdani general type-2 fuzzy logic systems optimized with quantum particle swarm optimization algorithms,” *Transactions of the Institute of Measurement and Control*, vol. 41, no. 10, pp. 2886–2896, 2019.

[8] C. Deng, X. Zhang, Y. Huang, and Y. Bao, “Equipping seasonal exponential smoothing models with particle swarm optimization algorithm for electricity consumption forecasting,” *Energies*, vol. 14, no. 13, p. 4036, 2021.

[9] J. Luo, Y Shao, X Liao, J. Liu, and J. Zhang, “Complex permittivity estimation for cloths based on QPSO method over 40-50 GHz,” *IEEE Transactions on Antennas and Propagation*, vol. 69, no. 1, pp. 600–605, 2021.

[10] L. Xiang, “A short-term prediction about the stock price of China’s pharmaceutical manufacturing industry based on back propagation neural network,” in *Proceedings of the 2018 International Symposium on Social Science and Management Innovation(SSMI 2018)*, vol. 68, pp. 198–206, January 2019.

[11] N. Thampanya, J. Wu, M. A. Nasir, and J. Liu, “Fundamental and behavioural determinants of stock return volatility in ASEAN-5 countries,” *Journal of International Financial Markets, Institutions and Money*, vol. 65, no. 1, Article ID 10, 2020.

[12] A. Do, R. Powell, J. Yong, and A. Singh, “Time-varying asymmetric volatility spillover between global markets and China’s A, B and H-shares using EGARCH and DCC-EGARCH models,” *The North American Journal of Economics and Finance*, vol. 54, no. 1, Article ID 101096, 2020.