# Choice of the Number of Hidden Layers for Back Propagation Neural Network Driven by Stock Price Data and Application to Price Prediction

# Choice of the Number of Hidden Layers for Back Propagation Neural Network Driven by Stock Price Data and Application to Price Prediction

**Peipei Zhang[1] and Chuanhe Shen[2]**

1. School of Data and Computer Science of Shandong Women's University, Jinan, P.R. China.
2. Institute of Financial Engineering of Shandong Women's University Jinan, P.R. China.
Corresponding Email: schunhe@sina.com; Email: Zhangpeipei0814@163.com

**Abstract.** Since the stock market is dynamic and nonlinear, we adopt the neural network to forecast the stock price. We construct the single hidden layer prediction model firstly, and analyse the effect of prediction accuracy on neurons amount and epochs. To improve the prediction accuracy and operating rate, we then construct the multiple hidden layers prediction model, and provide some theory guide on setting the number of each hidden layer for neural network with multiple hidden layers. Finally, we make a choice of the number of hidden layers by analysing the effect of stock price prediction, and the empirical results obtained demonstrate that the prediction performance of two hidden layers prediction model is better than that of the single hidden layer prediction model. Additionally, the empirical results obtained also demonstrate that the more epochs of training network, the better the results obtained with using the same number of neurons.

## 1. Introduction

Stock investment is a way to obtain higher returns, and has attracted the favour of the majority of investors. Although high returns and high risks in stocks go hand in hand, it doesn't sap stock investors' appetite. Accurately forecasting the stock price is the best way to effectively avoid stock risk, which has attracted many scholars to participate in the stock market research and provide risk hedging tools for the stock market investors. Recently, scholars have put forward many data analysis methods to predict the future price movement of a stock. Of varied methods, the study methodologies are statistical analysis method and machine learning approach.

The statistical analysis methods have been applied to forecast stock price in [1] and [2], include exponential smoothing method, linear regression method, moving average (MA) and autoregressive integrated moving average (ARIMA). Meanwhile, [3] and [4] propose that ARIMA model is a simple and acceptable method to model stationary time series data, and is widely used to analyse and predict linear time series data, and has strong short-term forecasting potential. However, [5] points out that the stock market is dynamic, nonlinear, complex, non-parametric and chaotic in nature. Literatures [6] and [7] show that autoregressive conditional heteroscedasticity (ARCH) and generalized autoregressive conditional heteroscedasticity (GARCH) can handle nonlinear time series data, but these traditional statistics analytical methods are difficult to reveal the internal law of the stock market.

Machine learning can be also used to model the nonlinear statistical data effectively, such as artificial neural network (ANN) [8], support vector machine regression (SVMR) [9], least squares support vector machine regression (LS-SVMR) [10] and so on. Moreover, literature [11] has revealed the superiority of ANN over ARIMA model. Thus, as a machine learning method, ANN is considered in this paper.

ANN is one of artificial system, which simulates structural and functional characteristics of biological neural network resorted to engineering technological means, and has been widely used to deal with financial time series data. For instance, feed forward neural network (FFNN) [12], back propagation neural network (BPNN) [13], radial basis function neural network (RBFNN) [14], general regression neural network (GRNN) [15] and so on. Among these methods, FFNN just passes on a message layer-by-layer with relatively slow convergence rate and comparatively high error, and BPNN adopts back propagation algorithm for improving the development of FFNN. Meanwhile, literature [16] shows that BPNN consistently and robustly outperforms the other models by comparing the forecasting results. However, as a non-convex function, neural network exists multiple extremum points, so neural network may find local minimum value rather than global minimum value. In order to prevent the network falling into local minimum and improve the prediction performance of the neural networks, some researchers come up with several methods. For instance, Levenber-Marquardt algorithm [17], additional momentum method and self-adaptive learning rate adjustment method18, genetic algorithm (GA) [19,20,21], simulated annealing algorithm (SA) [22] and so on. Whereas, particle swarm optimization based on uncertain knowledge and wavelet analysis method are used to optimize SVR in [23] and [24].

On the whole, those studies [17,18,19,20,21,22] mainly pay attention to avoiding falling into local minimum value and further increasing the predictive effectiveness of BPNN by combining other optimization methods, rarely considering its constitution. They adopt the BPNN which have only one hidden layer among the input and output layers. In this paper, we first construct the single hidden layer prediction model to forecast the stock price, but the trend of predicted results is not in accord with that of the actual values at some points. We try to solve this problem by increasing the epochs. However, the more epochs we choose, the longer it takes to train the neural network. Then, we construct the multiple hidden layers prediction model to forecast the stock price, and we find that it can improve the prediction performance without increasing the training time of neural network. Additionally, a kind of design method is presented to provide some theory guide on setting the number of each hidden layer for BPNN with multiple hidden layers, and its validity is confirmed by the experiment.

The organization of this paper is as follows. In section 2, we give the preparation knowledge of BPNN. In section 3, we discuss the forecasting experimental results, including single hidden layer prediction experiment and multiple hidden layers prediction experiment. In Section 4, we present some concluding remarks.

## 2. Preparation Knowledge

ANN does not need to determine the mapping of the inputs to outputs in advance. Through trained, ANN can obtain strong nonlinear mapping ability to learn the data rules and ensure that the actual output closest to the expected output when the input is given. BPNN is a kind of multi-layer feed-forward neural network which is improved by using the error back propagation algorithm (BP algorithm). Therefore, this section presents the content of two aspects, which are the structure of BPNN and the BP algorithm.

### *2.1 The Structure of BPNN*

The typical BPNN consists of one input layer, one or more hidden layers and one output layer. In Figure 1, the BPNN is constructed from $L \in \{3,4,5,6,\ldots\}$ layers, where the first layer is the input layer, $2_{nd} \sim L-1_{th}$ layers are hidden layers, and the $L_{th}$ layer is the output layer. Obviously, BPNN is the single hidden layer prediction model if $L=3$, and it is the multiple hidden layers prediction model if $L \ge 4$. Meanwhile, each note represents a neuron as shown in Figure 1. The input and output layers have $m$ and $n$ neurons respectively, and the $l_{th}$, $l \in \{2,3,\ldots,L-1\}$ hidden layer has $s_l$ neurons.

Denote

$$x = (x_1, x_2, \cdots, x_i, \cdots, x_n) \quad (1)$$

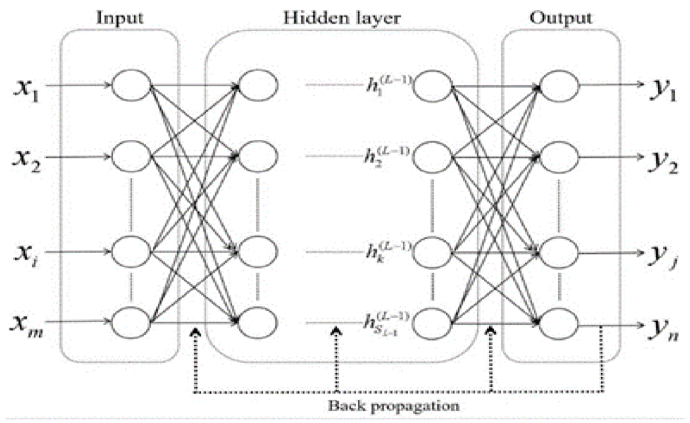


**Figure 1.** The typical BP neural network with multiple hidden layers.

$$y=(y_1,y_2,\cdots,y_j,\cdots,y_m) \tag{2}$$

$$h^{(l)}=(h_1^{(l)},h_2^{(l)},\cdots,h_k^{(l)},\cdots,h_{s_l}^{(l)}) \tag{3}$$

where $x$ is the input vector, $y$ is the output vector of BPNN, and $h^{(l)}$ is the output vector of the $l_{th}$ hidden layer. $x_i$ is the input of $i_{th}$, $i \in \{1,2,\ldots,m\}$ neuron in the input layer, $y_j$ is the output of $j_{th}$, $j \in \{1,2,\ldots,n\}$ neuron in the output layer, and $h_k^{(l)}$ is the output of $k_{th}$, $k \in \{1,2,\ldots,s_l\}$ neuron in the $l_{th}$ hidden layer.

In order to simplify each expression, there is no harm in supposing that $h^{(1)} = x$ and $h^{(L)} = y$. Then, the relations among $x$, $y$ and $h^{(l)}$ can be formulated as

$$h_k^{(l)} = f(net_k^{(l)}), \tag{4}$$

$$net_k^{(l)} = \sum_{j=1}^{s_l\text{-}1} \omega_{kj}^{(l)} h_j^{(l\text{-}1)} + \theta_k^{(l)}, \tag{5}$$

where $l \in \{2,3,\ldots,\mathrm{L}\}$, $k \in \{1,2,\ldots,s_l\}$. $net_k^{(l)}$ is the input of the kth neuron in the $l_{th}$ layer, $\omega_{kj}^{(l)} \in [0,1]$ is the weight between the $k_{th}$ neuron of the lth layer and the $j_{th}$ neuron of the $l-1_{th}$ layer, and $\theta_k^{(l)} \in [0,1]$ is the threshold value of the $k_{th}$ neuron in the lth layer. In this paper, we adopt sigmoid function as the activation function $f(\cdot)$.

Obviously, BPNN is established if $\omega_{kj}^{(l)}$ and $\theta_k^{(l)}$ are determined by using the collection data for $l \in \{2,3,\ldots,\mathrm{L}\}$, $k \in \{1,2,\ldots,s_l\}$ and $j \in \{1,2,\ldots,s_{l-1}\}$. This procedure is called network training, which consists of two processes, one is the signal forward transmission shown by equation (4)-equation (5), and the other is the error back propagation (BP algorithm) displayed in Figure 1.

### *2.2 The Description of BP Algorithm*

Suppose there a training dataset $\{(x(k),z(k)), k=1,2,\ldots,N\}$, where $x(k)$ and $z(k)$ serves as the $k_{th}$ vector of input and actual output, respectively. Let $y(k)$ denote the kth expected output vector of BPNN. Based on equation (1) and equation (2), $x(k)$, $y(k)$ and $z(k)$ can be expressed as

$$x(k) = (x_1(k), x_2(k), \ldots, x_i(k), \ldots, x_m(k)),$$

$$y(k) = (y_1(k), y_2(k), \ldots, y_j(k), \ldots, y_n(k)),$$

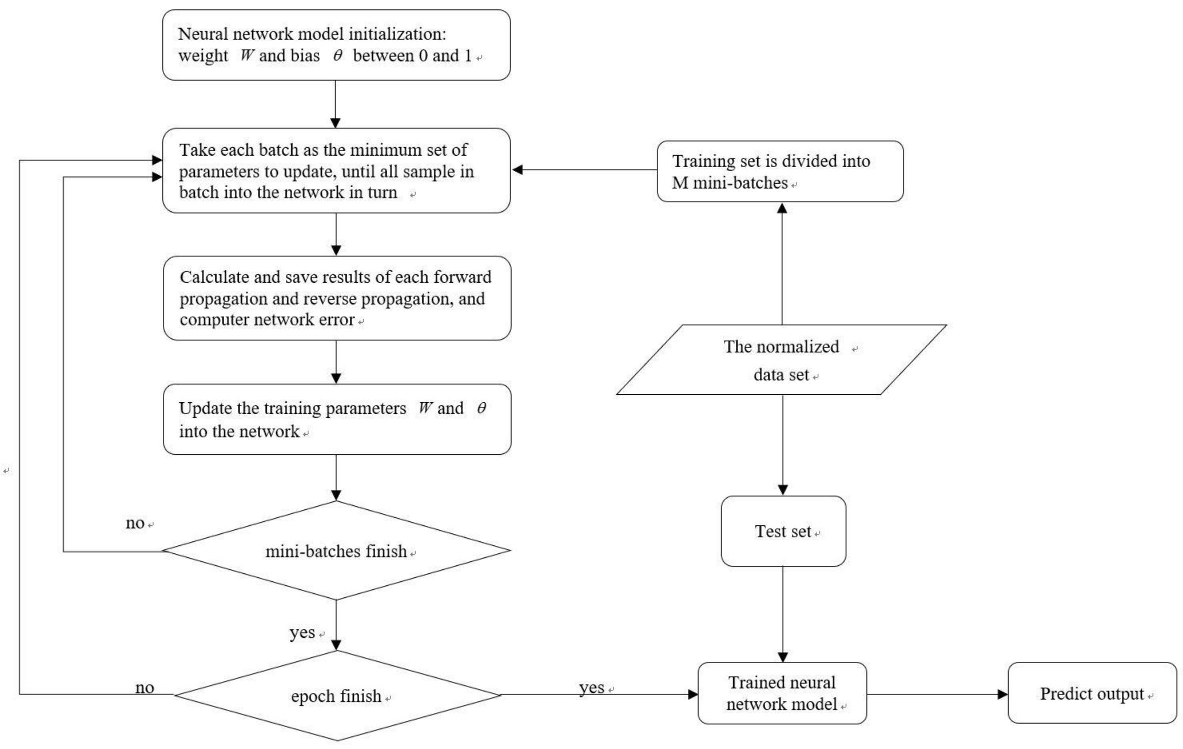

Neural network model initialization: weight W and bias θ between 0 and 1
Take each batch as the minimum set of parameters to update, until all sample in batch into the network in turn
Training set is divided into M mini-batches
Calculate and save results of each forward propagation and reverse propagation, and computer network error
The normalized data set
Update the training parameters W and θ into the network
no
mini-batches finish
Test set
yes
no
epoch finish
yes
Trained neural network model
Predict output

**Figure 2.** The flow diagram of training BP neural network with using mini-batch gradient descent method. It should be noted that $W$ and $\theta$ respectively consist of all connecting weights and threshold values used in equations (4)-(5).

$$z(k)=(z_1(k),z_2(k),\ldots,z_j(k),\ldots,z_n(k)).$$

Therefore, the mean square error between the actual output and the expected value is calculated via

$$E=\frac{1}{N}\sum_{K=1}^{N}E(k), \tag{6}$$

where $E(k)$ is the error of the $k_{th}$ training sample, that is

$$E(k)=\frac{1}{2}\sum_{j=1}^{n}[z_j(k)-y_j(k)]^2 .$$

The basic idea of BP algorithm is gradient descent method, which is used to minimize the mean square error (6) through regulating variable weight connection and threshold value. Thus, for each iteration, each weight and threshold value are updated by using

$$\omega_{kj}^{(l)}(t+1)=\omega_{kj}^{(l)}(t)+\Delta\omega_{kj}^{(l)}(t)=\omega_{kj}^{(l)}(t)-\alpha\,\partial E\big/\partial\omega_{kj}^{(l)}(t), \tag{7}$$

$$\theta_{k}^{(l)}(t+1)=\theta_{k}^{(l)}(t)+\Delta\theta_{k}^{(l)}(t)=\theta_{k}^{(l)}(t)-\alpha\,\partial E\big/\partial\theta_{k}^{(l)}(t), \tag{8}$$

where the arguments $t+1$ and $t$ are on behalf of the next and the current training step, respectively, $\alpha\in(0,1)$ is learning rate, $-\partial E\big/\partial\omega_{kj}^{(l)}(t)$ and $-\partial E\big/\partial\theta_{k}^{(l)}(t)$ indicate that the change along the gradient descent direction of mean square error (6).

However, BPNN is easy to fall into local minimum. In order to solve this problem, additional momentum method is used when we construct the single hidden layer prediction model and the multiple hidden layers prediction model [27], $\Delta\omega_{kj}^{(l)}(t)$ and $\Delta\theta_{k}^{(l)}(t)$ of equation (7)-equation (8) can be revised by

$$\Delta\omega_{kj}^{(l)}(t+1) = -\alpha\, \partial E \big/ \partial\omega_{kj}^{(l)}(t) + \eta\Delta\omega_{kj}^{(l)}(t), \tag{9}$$

$$\Delta\theta_{k}^{(l)}(t+1) = -\alpha\, \partial E \big/ \partial\theta_{k}^{(l)}(t) + \eta\Delta\theta_{k}^{(l)}(t), \tag{10}$$

where $\eta$ is a positive momentum constant, and it typically ranges from 0.1 to 0.8.

Equations (7)-(10) suggest that all $N$ samples of the training dataset are needed to be used for each parameter updating, which will greatly slow down the training speed along with the increase of training sample size, although it makes the algorithm to be converged to global optimal solutions. Thus, we consider using mini-batch gradient descent (MBGD) method [25, 26] to accelerate the convergence.

Comparing gradient descent method to MBGD method, the major difference is MBGD only extracts $M$ $(1 < M < N)$ samples at random from the training set in every training. In other words, when we applying MBGD method to conduct the $i_{th}$ training network, corresponding mean square error $E^{(i)}$ is calculated by using

$$E^{(i)} = \frac{1}{M}\sum_{k=1}^{M} E^{(i)}(k),$$

$$E^{(i)}(k) = \frac{1}{2}\sum_{j=1}^{n} [z^{(i)}{}_j(k) - y^{(i)}{}_j(k)]^2,$$

where superscript $i$ indicates that the used data are extracted from the $i_{th}$ random sampling. That is to say, $E$ needs to be replaced with $E^{(i)}$ in formulas (7)-(10) if we adopt MBGD to update each parameter.

## 3. Experiment Results

The aim of this paper is that comparing the single hidden layer prediction model to the multiple-hidden-layers prediction model, the detailed data analysis procedure can be decomposed into five steps.

- Step 1: Select data.

The daily trading data (fromJune1, 2016 to June 26, 2018) of Industrial and Commercial Bank of China (ICBC) shares (601398) are obtained by using Tushare, which is a Python package specially providing financial data.

- Step 2: Pre-process data.

In order to highlight the function and effect of BPNN, we eliminate the impact of units of measurement and trends embedded in the data by using the formula (11).

$$\bar{u}_t = \ln\frac{u_t}{u_{t-1}}, \tag{11}$$

where $u_t$ denotes one of the opening price, highest price, lowest price, closing price and trading volume at time $t$, and $t$ ranges from June 2, 2016 to June 26, 2018. Obviously, $u_t = 0$ if $t$ is June 1, 2016, and $\{\bar{u}_t\}$ represents the logarithmic return series when $\{u_t\}$ is closing price series.

- Step 3: Train BP neural network.

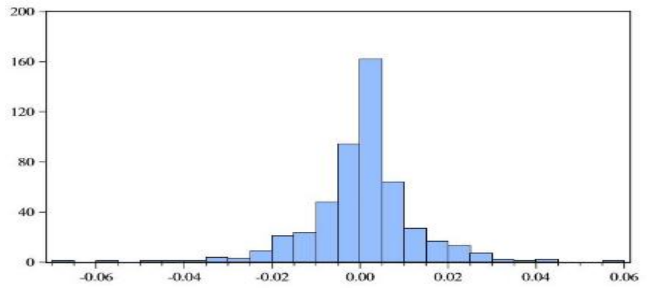
The detail procedure is shown in Figure 2. The training dataset consists of data from June 1, 2016 to June 4, 2018. We fetch the value of L from the set $\{3,4,5,6,7\}$. The opening prices, highest prices, lowest prices, closing prices and trading volumes of each stock for a third day are taken as an input vector, and the closing price of following day is taken as the output.

- Step 4: Test BP neural network.

The test dataset consists of data from June 5, 2018 to June 26, 2018. The test effects can be illustrated by their prediction accuracy, and they are measured by using mean absolute error (12).

$$\text{MAE} = \frac{1}{T}\sum_{t=1}^{T} |z_t - y_t|, \tag{12}$$

where T is the number of forecasting period, $z_t$ and $y_t$ respectively denote the actual closing price and expected closing price at period $t$, $t \in \{1, 2, \ldots, T\}$.



| | |
|---|---|
| Series: ICBC | |
| Sample 6/01/2016 6/26/2018 | |
| Observations 504 | |
| Mean | 0.000441 |
| Median | 0.000000 |
| Maximum | 0.059138 |
| Minimum | -0.065265 |
| Std. Dev. | 0.012052 |
| Skewness | -0.385998 |
| Kurtosis | 7.810897 |
| Jarque-Bera | 498.5549 |
| Probability | 0.000000 |

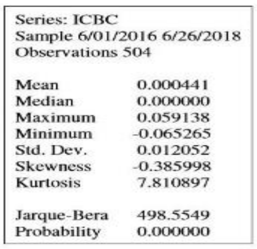


**Figure 3.** The histogram and statistical index values of probability density of the logarithmic return series

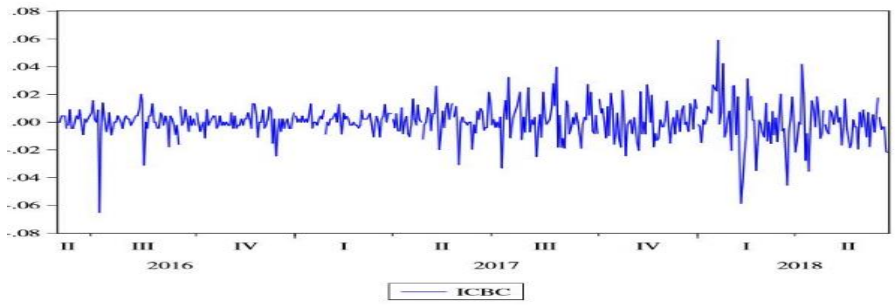


**Figure 4.** The time series chart of the logarithmic return series.

- Step 5: Make a comparison.

Using test results, we compare the single hidden layer prediction model to multiple hidden layers prediction model.

### *3.1 Data Description*

Figure 3 shows that Skewness and Kurtosis of the logarithmic return series of ICBC value −0.385998<0 and 7.810897>0, which means that these two sets of data respectively exhibit left-skewed peak and right-skewed peak distribution. That is, they have obvious peaks and fat tails. Figure 4 depicts that the logarithmic return series of these two sets of data have agglomeration features. Figure 3 and Figure 4 show that the logarithmic return series of these two sets of data have non-normal, non-stationary and non-linear characteristics. Thus, we need BPNN with different layers to process these data.

### *3.2 Single-Hidden-Layer Prediction Model*

Here, we choose $L$ as $L=3$ and aim to analyse the effect of different numbers of neurons and epochs for single hidden layer prediction model. According to [27], the number of neurons in hidden layer can be determined by

$$s_2 = \sqrt{mn}\,, \tag{13}$$

$$s_2 = \log_2 m\,, \tag{14}$$

$$s_2 = \sqrt{m+1} + \alpha\,, \tag{15}$$

where $m$ and $n$ respectively denote the neurons count of the input and output layers, and $\alpha \in [0,10]$. Combining equations (13), (14) and (15), we can find that the quantitative range of neurons in the hidden layer should enlarge with increasing the value of $m$.

Since the opening prices, highest prices, lowest prices, closing prices and trading volumes of each stock for a third day are taken as an input vector, which indicates that the input layer consists of fifteen neurons, this is, $m=15$. Meanwhile, the closing price of following day is taken as the output, so we have $n=1$. Based on the formulas (13) and (14), we can find $3 < s_2 < 4$, and formula (15) suggests that $s_2$ can range from 4 to 14. From the above, we can fetch $s_2$ in the set $\{3,4,\ldots,14\}$.

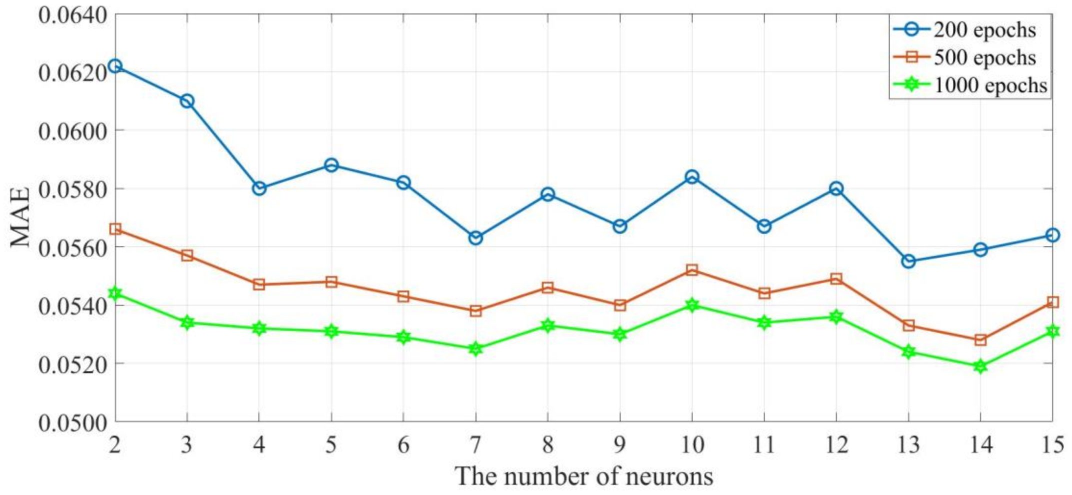


**Figure 5.** Forecast error of single hidden layer prediction model with different numbers of neurons and epochs.

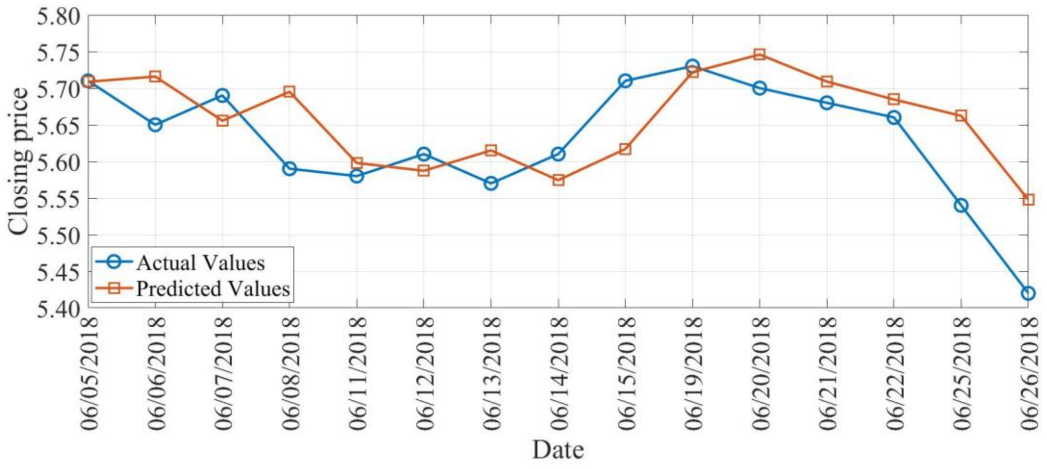


**Figure 6.** Compared actual values with predicted values of single hidden layer prediction model.

Figure 5 shows that the more times of the network is trained, the better the results obtained by using the same number of neurons. To obtain good prediction effect, we should train network by reaching 1000 epochs and even more. Figure 5 also displays that increasing the number of neurons cannot always assure the improvement of predictive effect when we fix epochs count of training network. As shown in Figure 5, the number of neurons should be respectively chosen as $s_2 = 14$ for realizing the optimal prediction effect of ICBC if we train the neural network up to 1000 epochs, and these forecasting results are represented by Figure 6.

As shown in Figure 6, the trend of predicted results is not in accord with that of the actual values a few points. For overcoming this problem, the most direct idea is to increase the epochs. Considering the realization of BP algorithm, more than two hundred parameters are needed to be updated in every training of BPNN at this moment. Then, it is not difficult to find that the more epochs you choose, the longer it takes to train the neural network. Thus, we consider to verify whether increasing the number of hidden layers can improve the prediction performance and not increase the training time.

### *3.3 Multi-Hidden-Layers Prediction Model*

For BP neural network with multiple hidden layers, there is no clear theoretical guidance for setting the number of each hidden layer, namely, the number of layers in a neural network is set as L⩾4. Here, a kind of design method is presented. Meanwhile, experimental results are given to verify our method when the neural network contains two hidden layers.

For $i \in \{1,2,\ldots,L\}$, we use $D^{(i)}$ to denote the $i_{th}$ layer of original BPNN, which indicates that $D^{(1)}$ is the input layer, $D^{(L)}$ is the output layer, and $D^{(2)}$, $D^{(3)}$, …, $D^{(L-1)}$ are hidden layers. Then, we divide the original BPNN to $L-2$ new networks. For $r \in \{1,2,\ldots,L-2\}$, $D^{(r)}$, $D^{(r+1)}$ and $D^{(r+2)}$ are the input layer,

The hidden layer and the output layer in turn. This indicates that each hidden layer of original BPNN can be seen as the input layer for a new network. Then, according to equations (3) and (13)-(15), the neurons counts of the neighbouring hidden layers have the following relations

$$s_{l+1} = \sqrt{s_l s_{l+2}}\,, \tag{16}$$

$$s_{l+1} = \log_2 s_l\,, \tag{17}$$

$$s_{l+1} = \sqrt{s_l + 1} + \alpha\,, \tag{18}$$

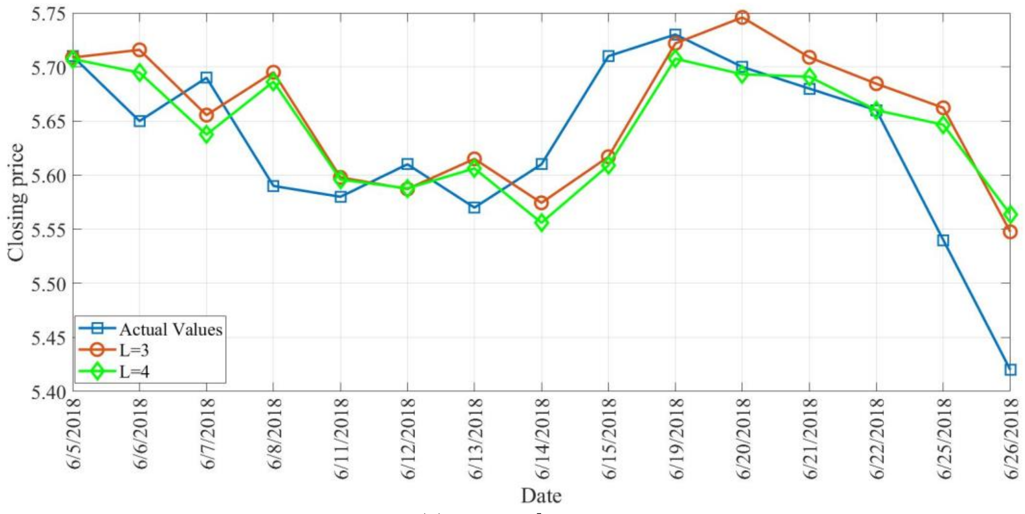
where $l \in \{2,3,\ldots,L-2\}$.

### *3.4 Choices of Number of Hidden Layers*

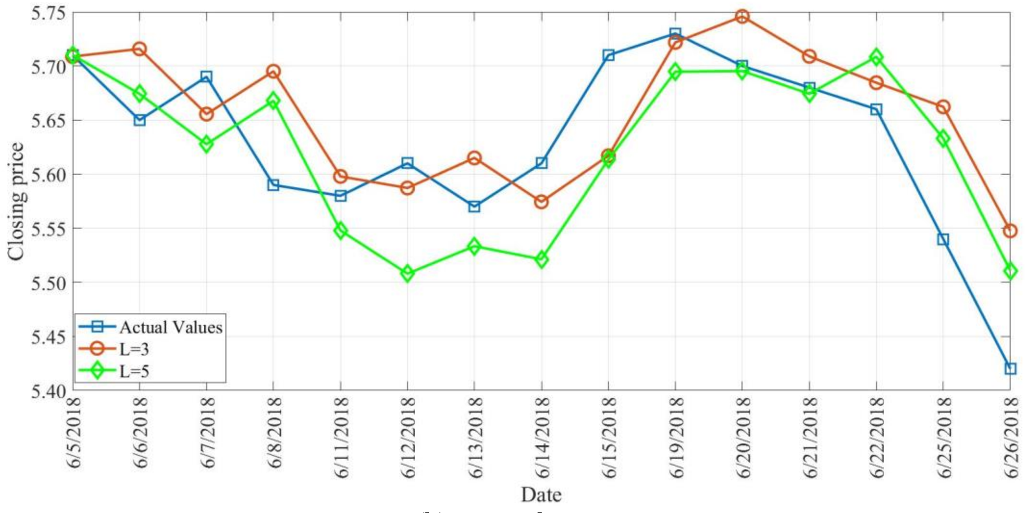
Let $L \in \{3,4,5,6,7\}$, and we obtain the relative optimal BPNN architectures through repeated experiments. It is worth mentioning that the network structure is deemed to achieve the best prediction accuracy if it returns the smallest MAE. Table 1 and Figure 7 present the empirical results of ICBC, and the neurons amounts of the corresponding neural networks are respectively $(m,s_2,n)$ =(15,14,1), $(m,s_2,s_3,n)$ =(15,4,10,1), $(m,s_2,s_3,s_4,n)$ =(15,5,4,4,1), $(m,s_2,s_3,s_4,s_5,n)$ = (15,5,5,5,5,1). After a simple calculation, we can find that only about 150 parameters are needed to be updated in every training of BPNN with multiple hidden layer. In contrast to more than 200 parameters needed to be updated in every training of BPNN with single hidden layer, it takes the shorter time to train networks with multiple hidden layers than to train networks with a single hidden layer if we choose same training epochs.

**Table 1.** Sample results of L-layer BPNN of ICBC, where forecast errors are obtained by calculating the difference between actual values and predicted values.

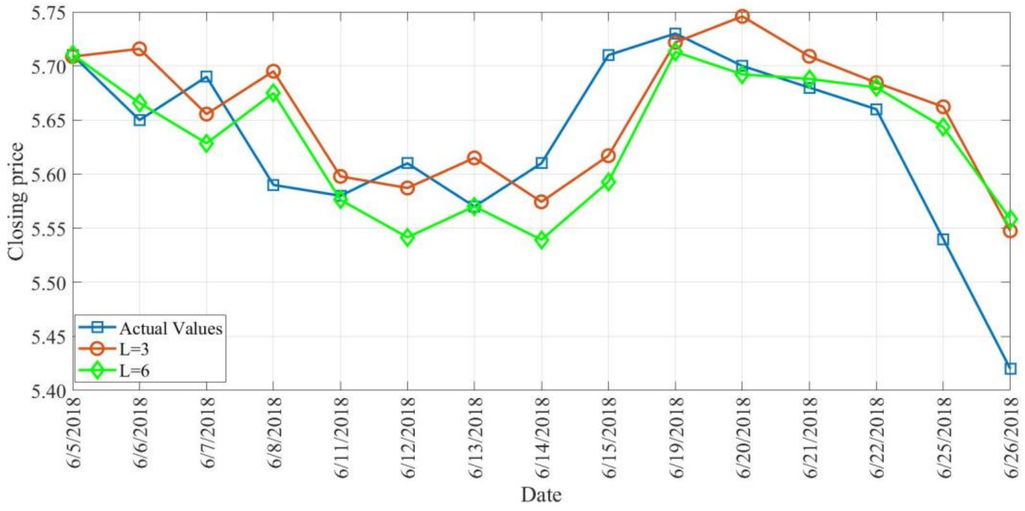
| Date | Actual Values | Predicted Values | | | | |
|---|---|---|---|---|---|---|
| | | L=3 | L=4 | L=5 | L=6 | L=7 |
| 06/05/2018 | 5.71 | 5.7087 | 5.7074 | 5.7102 | 5.7105 | 5.7090 |
| 06/06/2018 | 5.65 | 5.7158 | 5.6947 | 5.6742 | 5.6658 | 5.7081 |
| 06/07/2018 | 5.69 | 5.6555 | 5.6375 | 5.6277 | 5.6285 | 5.6357 |
| 06/08/2018 | 5.59 | 5.6951 | 5.6864 | 5.6678 | 5.6749 | 5.6802 |
| 06/11/2018 | 5.58 | 5.5979 | 5.5959 | 5.5479 | 5.5762 | 5.5850 |
| 06/12/2018 | 5.61 | 5.5872 | 5.5878 | 5.5080 | 5.5416 | 5.5505 |
| 06/13/2018 | 5.57 | 5.6150 | 5.6062 | 5.5334 | 5.5702 | 5.5971 |
| 06/14/2018 | 5.61 | 5.5744 | 5.5560 | 5.5213 | 5.5393 | 5.5523 |
| 06/15/2018 | 5.71 | 5.6170 | 5.6090 | 5.6135 | 5.5929 | 5.5938 |
| 06/19/2018 | 5.73 | 5.7218 | 5.7076 | 5.6947 | 5.7130 | 5.6899 |
| 06/20/2018 | 5.70 | 5.7458 | 5.6932 | 5.6953 | 5.6922 | 5.7274 |
| 06/21/2018 | 5.68 | 5.7089 | 5.6909 | 5.6743 | 5.6880 | 5.6823 |
| 06/22/2018 | 5.66 | 5.6845 | 5.6600 | 5.7085 | 5.6801 | 5.6607 |
| 06/25/2018 | 5.54 | 5.6622 | 5.6466 | 5.6330 | 5.6434 | 5.6286 |
| 06/26/2018 | 5.42 | 5.5476 | 5.5637 | 5.5108 | 5.5586 | 5.5654 |



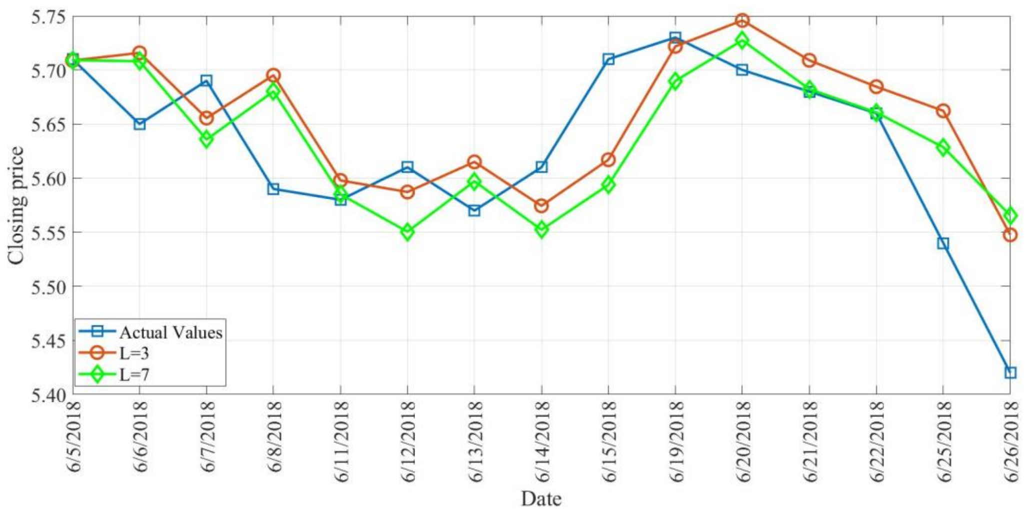
(a) L=3 and L=4



(b) L=3 and L=5



(c) L=3 and L=6



(d) L=3 and L=7

**Figure 7.** Numerical comparison with single hidden layer prediction model and multiple hidden layers prediction model for ICBC.

From Figure 7, we can observe that the prediction accuracy level of the single hidden layer prediction model compared with that of the multiple-hidden-layers prediction model is not quite significant. It can be considered that both models have achieved good prediction results based on the prediction mean square errors of both models. Because the experimental data selected in this study are weak nonlinearity, the network prediction effect of two hidden layers is good compared with that of more hidden layers as shown in Figure 7. Additionally, the performance of the two hidden layers prediction model is better than the single hidden layer prediction model in terms of the difference between the predicted values and the actual values as shown in Figure 7 (a). Combined with the prediction accuracy and running speed of the model, there are good reasons to believe that the network with multiple hidden layers is superior to the network with a single hidden layer in stock price prediction.

## 4. Conclusions

In this paper, the stock data is analysed by using the Python software, and the data treatment results are applied to evaluate the effect of BPNN with different hidden layers for stock price prediction. Based on the experiment results, we find that the more times of training network, the better the results obtained by using the same number of neurons. Meanwhile, both the single hidden layer prediction model and the multiple hidden layers prediction model can achieve good prediction in terms of comparing their mean absolute errors, but some actual values are closer to the predicted values of the multiple-hidden-layers prediction model. In our future studies, hybrid approaches of statistical analysis method and machine learning can be considered to improve the prediction performance of existing predictive models, which is similar to that reported in [29].

## 5. Acknowledgements

This work was supported by Social Science Research Foundation of Ministry of Education of China (15Y-JA790051), National Social Science Fund Project of China (17BGL058), Shandong Province Natural Science Foundation (ZR2016GM20), Shandong Women's University Youth Scientific Research Project (2016ZD04), and Science and technology plan of Shandong provincial department of education (J18KA236).

## 6. References

[1] J. P. C. Kleijnen, G. A. Mithram, "Statistical techniques in simulation," IEEE Transactions on System Man & Cybernetics, 1974, 7(9), pp. 680-680.

[2] J. J. Wang, J. Z. Wang, Z. G. Zhang, et al, “Stock index forecasting based on a hybrid model,” Omega, 2012, 40(6), pp. 758-766.
[3] B. U. Devi, D. Sundar, D. P. Alli, “An effective time series analysis for stock trend prediction using ARIMA model for nifty midcap-50,” International Journal of Data Mining & Knowledge Management Process, 2013, 3(1), pp. 65-78.
[4] A. A. Adebiyi, A. O. Adewumi, C. K. Ayo, “Stock price prediction using the ARIMA model,” 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, 2014, pp. 105-111.
[5] Y. S. A. Mostafa, A. F. Atiya, “Introduction to financial forecasting,” Applied Intelligence, 1996, 6(3), pp. 205-213.
[6] R. F. Engle, “Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation,” Econometrics, 1982, 50(4), pp. 987-1007.
[7] T. Bollerslev, “Generalized autoregressive conditional heteroskedasticity,” Journal of Econometrics, 1986, 31(3), pp. 307-327.
[8] H. D. Qin, “Based on neural network forecasting stock market trend,” Chinese dissertation database, 2005.
[9] L. Yu, H. H. Chen, S. Y. Wang, et al, “Evolving least squares support vector machines for stock market trend mining,” IEEE Transaction on Evolutionary Computation, 2009, 13(1), pp. 87-102.
[10] T. V. Gestel, J. A. K. Suykens, D. E. Baestaens, et al, “Financial time series prediction using least squares support vector machines within the evidence framework,” IEEE Transaction on Neural Networks, 2001, 12(4), pp. 809-821.
[11] A. A. Adebiyi, A.O.Adewumi, C. K. Ayo, “Comparison of ARIMA and artificial neural networks models for stock price prediction,” Journal of Applied Mathematics, 2014.
[12] F. W. O. Landt, “Stock price prediction using neural networks,” Maters Thesis, Leiden University, 1997.
[13] J. L. Yu, Z. Q. Sun, V. Kroumov, “Modeling and decision making in stock market based on BP neural network,” Theory and Practice of System Engineering, 2003.
[14] W. Shen, X. P. Guo, C. Wu, et al, “Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm,” Knowledge-Based Systems, 2011, 24(3), pp. 378-385.
[15] M. T. Leung, A. S. Chen, H. Daouk, “Forecasting exchange rates using general regression neural networks,” Computers & Operations Research, 2000, 27(11-12), pp. 1093-1110.
[16] Y. G. Song, Y. L. Zhou, R. J. Han, “Neural networks for stock price prediction,” arXiv preprint arXiv: 1805.11317, 2018.
[17] F. Li, C. Liu, “Application study of BP neural network on stock market prediction,” 2009 Ninth International Conference on Hybrid Intelligence Systems, 2009, pp. 174-178.
[18] W. M. Ma, Y. Y. Wang, N. F. Dong, “Study on stock price prediction based on BP neural network,” 2010 IEEE International Conference on Emergency Management and Management Sciences (ICEMMS), 2010, pp. 57-60.
[19] K. J. Kim, I. Han, “Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index,” Expert Systems with Applications, 2000, 19(2), pp. 125-132.
[20] X. Z. Lu, D. Q. Ye, M. Nan, “Forecasting stock prices based on the genetic algorithms and neural network,” Computer Development & Applications, 2010, 23(2), pp. 61-62.
[21] M. Göçken, M. Özçalıcı, A. Boru, et al, “Integrating metaheuristics and artificial neural networks for improved stock price prediction,” Expert Systems with Applications, 2016, 44, pp. 320-331.
[22] Z. Ming, W. Yan, “Research and application of BP neural network improvement algorithm,” Microcomputer Information, 2009, 25(6), pp. 193-195.
[23] J. Xin, Y. H. Kang, K. Y. Zhang, “Stock price prediction using SVM optimized by particle swarm optimization based on uncertain knowledge,” International Journal of Digital Content Technology and Its Applications, 2012, 6(23), pp. 216-221.

[24] L. J. Kao, C. C. Chiu, C. J. Lu, et al, “A hybrid approach by integrating wavelet-based feature extraction with MARS and SVM for stock index forecasting,” Decision Support Systems, 2013, 54(3), pp. 1228-1244.
[25] A. Krizhevsky, I. Sutskever, G. E. Hinton, “Image-net classification with deep convolutional neural networks,” Advances in Neural Information Processing Systems, 2012, pp. 1097-1105.
[26] M. Dixon, D. Klabjan, J. H. Bang, “Classification-based financial markets prediction using deep neural networks,” Algorithmic Finance, 2016, pp. 1-11.
[27] T. M. Zurada, “Introduction to artificial neural systems,” St. Paul, West Publishing Company, 1992.
[28] L. Q. Han, “Artificial neural network tutorial,” Beijing University of Posts and Telecommunications press, 2006.
[29] E. Hadavandi, H. Shavandi, A. Ghanbari, “Integration of genetic fuzzy systems and artificial neural networks for stock price forecasting,” Knowledge-Based Systems, 2010, 23(8), pp. 800-808.