Forecasting stock prices in two ways based on LSTM neural network

Jingyi Du ,Qingli Liu ,Kang Chen ,Jiacheng Wang
College of Electrical and Control Engineering, Xi'an University of Science and Technology, Xi'an, China
924392705@qq.com, 18829347859@163.com, 1932900294@qq.com,763037321@qq.com

***Abstract*—Due to the extensive application of deep learning in processing time series and recent progress, LSTM (Long Short-Term Memory) neural network is the most commonly used and most powerful tool for time series models. The LSTM neural network is used to predict Apple stocks by using single feature input variables and multi-feature input variables to verify the prediction effect of the model on stock time series. The experimental results show that the model has a high accuracy of 0.033 for the multivariate input and is accurate, which is in line with the actual demand. For the univariate feature input, the predicted squared absolute error is 0.155, which is inferior to the multi-feature variable input.**

Keywords—stock price; LSTM; RNN; univariate feature input; multivariate feature input

I. INTRODUCTION

At present, stocks in many cities in China have become a trend. As of September 21, 2018, the number of Chinese stocks has exceeded 143 million and is still growing. With the development of the stock market, new stocks are listed almost every day, and there are more and more stocks that can be traded. Therefore, how to choose a suitable stock (investment) becomes a big problem. In the process of investment, investors are sometimes affected by the surrounding environment, relying on personal experience and their own intuition to analyze and judge, resulting in certain asset losses. Therefore, a scientific and efficient method is needed to replace the traditional stock trading methods to help Guide stock trading [1]. Traditional methods for predicting time series (the types of predictive models in time series) include autoregressive moving average (ARIMA), support vector regression (SVR), and random forest (XGBoot) [2], but because stock price data has Non-linear, high-noise characteristics, these statistical methods are difficult to make high-precision predictions, not applicable to stock prediction. Studies have shown that most of the time series prediction processing uses the LSTM (Long Short-Term Memory) network [3], which is a time recurrent neural network based on RNN (Recurrent Neural Network). Hochreiter & Schmidhuber proposed, and recently improved and promoted by Alex Graves [4], LSTM will have selective memory features on long-term scales, suitable for processing and predicting important events with relatively long intervals and delays in time series.

LSTM is a variant of RNN (Recursive Neural Network). It was developed to solve the problem that RNN easily disappears[5][6]. LSTM can be regarded as a combination of multiple transition gates. It can bypass some units and memorize longer time steps. information. Therefore, LSTM can overcome the gradient disappearance problem to some extent.

II. RELATED THEORY AND TECHNOLOGY

A. RNN neural network

The processing information of the current time of the RNN (Recurrent Neural Network) is not only related to the input of the current moment, but also depends on the sequence information of the previous memory. Because of its special network model structure, the problem of information preservation is solved, so the serialization feature is The tasks are suitable for RNN network structure, such as sentiment analysis [7], keyword extraction [8] and speech recognition[9].

The forward propagation structure of the RNN is shown on the left side of Figure 1. s represents the memory of the RNN network at time t, x represents the sequence information input into the network at time t, o represents the result of network generation at time t, U is the weight between the input layer and the hidden layer, and W is the hidden layer to the hidden layer. Weight, V is the weight of the hidden layer to the output layer. In the unexpanded RNN network structure of Figure 1, the circular arrow pointing from s to s represents that the output information at time t-1 is input to the network with input x at time t. When there are t+1 x inputs into the RNN network, the structure diagram of the RNN is expanded, as shown on the right side of Figure 1.

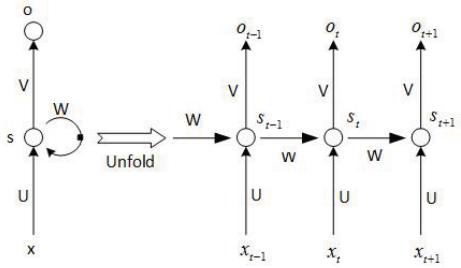

Figure1. RNN forward propagation structure

The internal structure of the RNN can be regarded as a network with a hidden layer. The output of the hidden layer is divided into two parts, one part is transmitted to the next node, and the remaining part is transmitted to its own node. The forward propagation timing diagram of RNN is shown in Figure 2.

978-1-5386-6243-4/19/$31.00 ©2019 IEEE

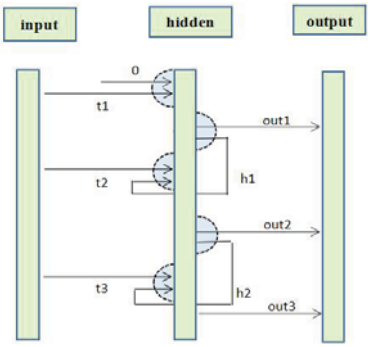

Figure 2. RNN forward propagation timing diagram

In Fig. 2, there are three timings t1, t2, and t3, wherein the time t1 is used as the starting time, and the information input from the input layer at time t1 and the initial value 0 are simultaneously input to the hidden layer, and the output layer outputs the result as out1, out1 at time t1. Multiplying the weight W to generate h1, h1 as the memory information at time t1 and the input information of the input layer at time t2 are used as the total information input hidden layer at time t2, and the input is converted into an overall input through the input weight U and output through the hidden layer. The result of time t2 is out2, the time t3 repeats the above process and h2 is input together with the input weight conversion as the input to generate the output result out3 at time t3.

The network structure of RNN belongs to the forward network, and the backpropagation error is obtained by using the chain-derived algorithm of BackPropagation Through Time (BPTT). However, in the reverse derivation rule, as the number of network layers increases, the value of the reverse transmission error of the activation function becomes smaller or even zero. For the RNN network, the output at time t must be the result of the integrated action of the input at time t=1,...,t-1. That is to say, when updating the model parameters, not only the information input at the current moment affects the model parameters, but the input information at all times before will affect the update of the model parameters. However, once the RNN network has a vanishing gradient problem, the previous input data cannot be affected by the model parameters when the input data is long at the current time [10]. To solve this problem, an LSTM network needs to be introduced [11].

B. LSTM neural network

LSTM is an excellent variant of RNN, inherits the characteristics of most RNN models[12], and solves the problem of gradient disappearance caused by gradient back-propagation. Its structure is shown in Figure 3.

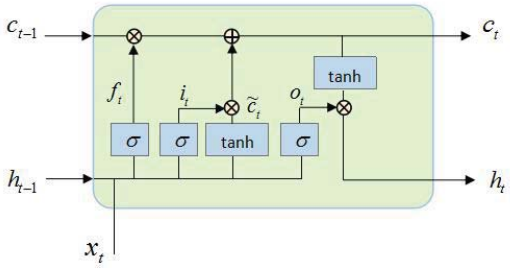

Figure 3. Schematic diagram of LSTM structure

In Fig. 3, ct is the cell state, and the network memory information at time t is stored internally, xt is input information at time t, and ht is information indicating that the next time is output at time t. The core idea of the LSTM structure is to introduce a cell state connection that stores the required memory information, while its internal entry control structure updates and outputs the control information. Compared to the RNN, the LSTM structure is increased. Additional processing of input information. The gate control structure of LSTM consists of three doors, namely the forget door, the input door, and the output door. It is most important to forget the door. The following describes the working principle of the three doors.

Forget the door to determine the information content discarded by the model from the cell state, by reading the previous time output information ht-1 and the current input information xt, outputting a value between 0 and 1 and multiplying each cell state ct-1 , 1 means to completely retain the previous information, and 0 means to completely discard the previous information.

$$f_t = \sigma(W_f \bullet [h_{t-1}, x_t] + b_f) \tag{1}$$

The calculation process of forgetting the gate is as shown in equation (1), where ft is the output of the forgotten gate, and W and bf are the weights and offset matrices of the neural network layer, respectively.

The input gate function can be divided into two parts, one part is the partial discarding of the old information, the other part is the partial reservation of the new information, and the two parts of the result are added to complete the update of the cell state. The specific calculation process is as shown in equations (2), (3), and (4).

$$i_t = \sigma(W_i \bullet [h_{t-1}, x_t] + b_i) \tag{2}$$

$$\tilde{c}_t = \tanh(W_C \bullet [h_{t-1}, x_t] + b_c) \tag{3}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{4}$$

(3) where $\tilde{c}_t$ is a new state value vector created by the tanh network layer, in preparation for cell state update in (4).

The output gate determines the output of the message, the internal sigmoid layer will determine which portion of the information will be output, and then the cell state is processed by the Tanh excitation function and multiplied by the output ot of the sigmoid gate to obtain the final output. The calculation process is Equation (5) and Equation (6).

$$o_t = \sigma(W_O[h_{t-1}, x_t] + b_o) \tag{5}$$

$$t_t = o_t * \tanh(c_t) \tag{6}$$

III. EXPERIMENTAL CONTENT

A. Data acquisition and processing

As a large company established for decades, American Apple's stock data changes are representative. This experiment selects the stock price of Apple's 2008.11.3-2018.11.2 as a data set. 80% of the data is used as a training set and 20% is used as a training set. The data set is shown in Table I:

Table I US APPPLE HISTORICAL STOCK DATA

time	open	lower	higher	closed	Ad-closed	Volume
2008.11.3	15.13	14.98	15.58	15.28	10.26	264484500
2008.11.4	15.71	15.24	15.97	15.85	10.65	349670300
...	...	...	...	...	...	...
2018.11.1	219.05	216.81	216.81	222.22	222.22	58323200
2018.11.2	209.55	213.65	213.65	207.48	207.47	91263400

Table I is 7 columns, the first column is time, and the remaining 6 columns are stock feature attributes. It can be seen from the table that the stock price of 2008 and 2018 is different in magnitude, and the stock volume is different from the other five attributes. The direct analysis with the original data has a great influence on the data analysis results. Depending on the order of magnitude between the data, the data needs to be normalized to eliminate the impact of data magnitude on the results analysis. The raw data after standardization of data is in the same order of magnitude, which is suitable for comprehensive comparative evaluation. In order to achieve a better prediction effect, the original data is uniformly normalized by data, and the data is processed between [0, 1], and the normalization function is as follows.

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{7}$$

In order to ensure that the predicted value of the network output is the same as the true value, the prediction result is inverse normalized, and the inverse normalization function is as follows.

$$\hat{y}_i = y_i \bullet (x_{\max} - x_{\min}) + x_{\min} \tag{8}$$

In equation (8), y_i is the network output value and $\hat{y}_i$ is the predicted value.

B. Model building

This experiment uses Python language programming and builds a model based on the Tensorflow deep learning framework. The model network structure is a 2-layer LSTM cyclic network with 12 hidden layer nodes per layer. Smaller batch training is used because of the larger data set and the generalization ability of the enhanced model. Because the learning rate is too large, the loss may not even converge, so the learning rate is set to 0.03, and the number of iterations is 1000.

This experiment uses two methods to train the model to verify the effect of the LSTM network on stock time series prediction. In order to analyze and compare the two methods, the parameters of the model are consistent.

Method 1: The univariate input feature, that is, using the historical data of the stock closing price, predicts the stock closing price of the next day, because the time step is set to 20, which can be understood as predicting the data of the following day with the previous 20 days of data.

Method 2: Multi-factor input characteristics, based on the lowest price, highest price, opening price, closing price, trading volume and other factors in the stock historical data, predict the closing price of the next day's stock.

C. Evaluation index

In this experiment, the mean square error (MSE) is used as the loss function for each training of the model. The mean absolute error (MAE) is used to evaluate the prediction results. The mean square error function and the mean absolute error function are respectively Formula (9) and formula (10).

$$\mathrm{MSE} = (\hat{y}_i, y_i) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{9}$$

$$\mathrm{MAE} = (\hat{y}_i, y_i) = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - y_i|}{y_i} \tag{10}$$

IV. ANALYSIS OF RESULTS

A. Model training effect analysis

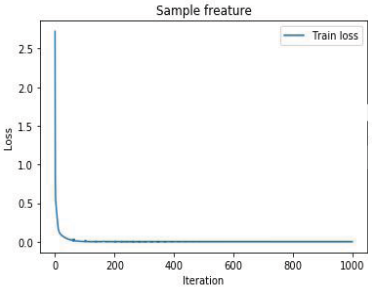

Figure 4 . single variable input loss

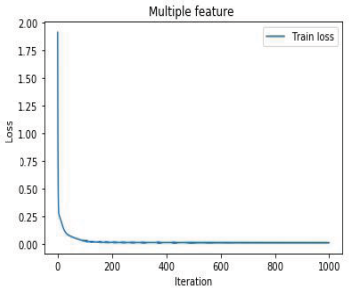

Figure 5 . multivariate input loss

TABLE II MODEL TRAINING LOSS

Feature input	MSE
Univariate feature input	0.035
Multivariate feature input	0.024

Figure 4 and Figure 5 are the univariate feature input loss function graph and the multivariate feature input loss function graph. The abscissa is the number of iterations and the ordinate is the loss value of each time. It can be seen from the figure that the initial mean square error of the univariate feature input is about 3, while the multi-variable feature input has an initial mean square error of about 2, which is smaller than the univariate mean square error. The MSE values for univariate and multivariate in Table II are the average MSE of 1000 iterations, the average MSE for a single variable is 0.035, and the average MSE for a multivariate is 0.024, which is also small compared to a single variable. The comparison betwee

the two shows that the training model of multivariate features is better than the training model of univariate features. In Figures 4 and 5,the loss function value converges quickly within the (0,0.1) interval around the model iteration 80 times, and then stabilizes at a lower level, indicating that the model is fast and stable, which is conducive to the analysis and forecast of volatile and unstable stock prices.

B. Analysis of prediction results

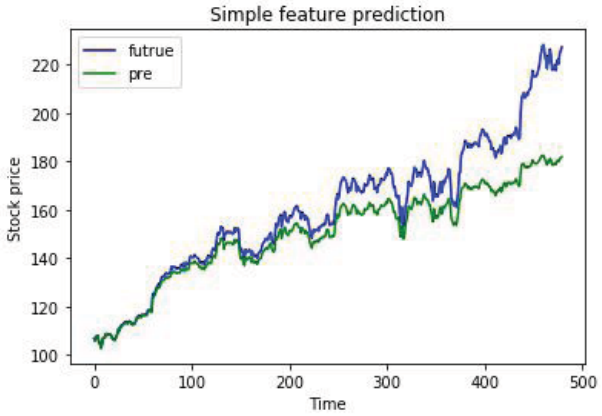

Figure 6. single variable prediction

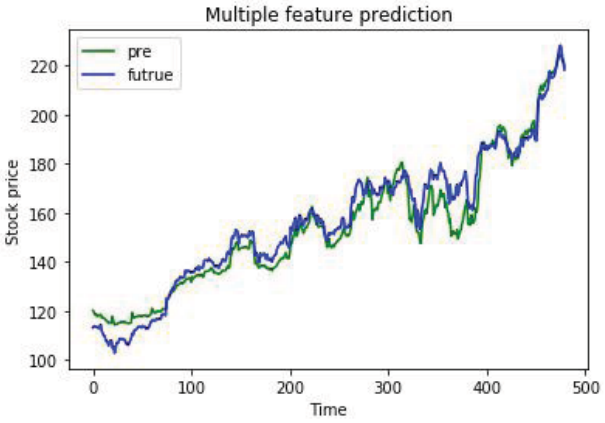

Figure7. multivariate variable prediction

TABLE III MODEL PREDICTION ERROR

Feature input	MAE
Univariate feature input	0.155
Multivariate feature input	0.033

Figure 6 and Figure 7 are the univariate feature input prediction effect graph and multivariate prediction effect graph. The abscissa is the number of prediction data (one value for each time point), and the ordinate is the stock price. The blue line in the figure is the real value, the green line is the model prediction value, and there are 480 prediction data. The two data in Table III are the MAE (squared absolute error) predicted by the univariate input and the multivariate input model, respectively.

In Fig. 6, the predicted values of the first 400 data models are close to the real values, and the error of 280 numbers between the predicted and actual values of the last 80 data is gradually increased. From Table III, the average absolute error is 0.155. The predicted value of the overall data in Figure 7 is close to the real value, the latter 80 data are basically coincident, and the average absolute error of multi-feature input is 0.033. Compared with the single-variable feature input prediction effect, the model has high fitness and good prediction effect.

V. CONCLUSIONS

The experimental LSTM neural network uses two input methods to predict the stock price of Apple through training and learning, which verifies that the model has better prediction effect on multivariate feature input and meets the actual demand. However, due to limited data, only Apple's single stock data was selected for research and forecast, which is difficult to represent the overall stock market forecast effect.

REFERENCES

[1] Deng Fengxin, Wang Hongliang. Application of LSTM Neural Network in Stock Price Trend Forecast——Based on the Research of Stock Market Data in US and Hong Kong Stock Markets[J].Financial Economy,2018(14):96-98.

[2] Ma Chaoqun, Wang Xiaofeng. Forecast of Vegetable Sales Based on LSTM Network Model[J].Modern Computer (Professional Edition), 2018 (23): 26-30.

[3] Bao W, Yue J, Rao Y. A deep learning framework for financial time series using stacked autoencoders and long-short term memory[J]. Plos One, 2017,12(7).

[4] Zachary C L, John B, Charles E. A Critical Review of Recurrent Neural Networks for Sequence Learning[M]. Computer Science,2015.

[5] Hochreiter S， Schmidhuber J.Long short-term memory[J].Neural Computation，1997，9（8）：1735-1780.

[6] Lipton Z C，Berkowitz J，Elkan C.A critical review of recurrent neural networks for sequence learning[J].ar Xiv preprint ar Xiv：1506.00019，2015.

[7] Liang Jun,Chai Yumei,Yuan Huibin, et al. Sentiment Analysis Based on Polar Transfer and LSTM Recursive Network[J].Journal of Chinese Information Processing,2015,29(5):152-160.

[8] Bao Zhiqiang,Zhou Yipeng.Study on the extraction of fresh comment keywords based on LSTM[J].Fujian Computer,2018,34(10):90-92.

[9] Soltau H, Liao H, Sak H. Neural Speech Recognizer: Acoustic-to-Word LSTM Model for Large Vocabulary Speech Recognition[J]. 2016:3707-3711.

[10]Glorot X,Bordes A,Bengio Y. Deep Sparse Rectifier Neural Networks[C]. International Conference on Artificial Intelligence and Statis-tics，2011.

[11]BAHDANAU D, CHO K, BENGIO Y. Neural Machine Translation by Jointly Learningto Align and Translate[J/OL].[2016-05-19].https://arxiv.org/pdf/1409.0473.pdf.

[12]Chen Zesi,HuWenxin.Simplified speech synthesis of LSTM[J].Computer Engineering and Applications,2018,54(03):131-135.