Research on Stock Price Prediction Based on LSTM Neural Network Modeling

Lingling Zeng, Junhong Li

School of Economics, Wuhan University of Technology, Wuhan, China

Abstract. The stock market, as a barometer of the national economy, dynamically reacts to the basic situation of economic operation in real time. Stock price prediction has become even more of a problem for investors, and deep learning algorithms are being widely used in financial instrument price prediction, market trend analysis, investment opportunity determination and portfolio optimization, etc. In this paper, the main use of RNN and LSTM neural network models to predict stock price changes, for this purpose, Ping An Bank, selected from January 1, 2015 to June 1, 2024 A total of 2,287 historical data of stock prices are studied, and it is found that the model predicts the price with a better fit to the real price, and comparing the prediction errors of LSTM, CNN-LSTM, and CNN-BiLSTM models, it is found that the BiLSTM model has the smallest error in predicting the stock price, and it possesses higher prediction accuracy.

Keywords: Stock Price Prediction; Deep Learning; RNN; LSTM; BiLSTM.

1. Introduction

With the rapid development of the market economy, China's financial system has become increasingly sophisticated, leading to a growing number of individuals participating in stock investments with the expectation of achieving returns. However, the high volatility of stocks often deters investors, making the prediction of stock trends a focal point of interest. Accurate predictions not only facilitate informed investment decisions but also optimize asset allocation, enhance capital efficiency, and improve market effectiveness. Although traditional prediction methods have some utility, the inherent uncertainty of stock prices makes it difficult for any single method to achieve precise predictions. Combining multiple methods may improve accuracy.But with advancements in computer technology, machine learning algorithms have been widely applied in stock price prediction, particularly deep learning models, which offer advantages due to their powerful feature extraction capabilities, ability to handle nonlinear relationships, and adaptability to large datasets. Deep learning can automatically extract features, reducing the complexity of manual feature engineering, and through multi-layer nonlinear processing, it can uncover deep-seated patterns that are crucial for prediction. Additionally, deep learning leverages GPUs for efficient parallel computing, employs reasonable design and regularization techniques to avoid overfitting, maintains strong generalization capabilities, adapts to dynamic market changes, and continuously optimizes prediction performance. This paper adopts the LSTM model for stock price prediction, as it can capture long-term dependencies in time series data, which is essential for understanding the dynamic changes in stock prices.

2. Review of Domestic and International Research

In the field of stock price prediction, various scholars have employed different machine learning and deep learning algorithms for research, concluding that deep learning algorithms outperform traditional machine learning methods as well as conventional regression and indicator analysis techniques. Sharma et al. (2012) initiated a comparison between traditional prediction methods and machine learning techniques in sales forecasting, combining the two to achieve distinct predictive outcomes [1]. Alsharef A et al. (2022) asserted that the predictive efficiency of deep learning surpasses that of ordinary linear models, utilizing multiple methods to forecast the price trends of the cryptocurrency Ethereum, thereby aiding investors in making informed decisions [2]. Huang J and Wang Y (2024) compared various models and found that LSTM exhibited higher accuracy in

capturing long-term dependencies and cyclical patterns when predicting stock prices [3]. Kalidindi A R et al. (2023) validated the superiority of deep learning in stock price prediction using data from Tata Consultancy Services [4]. Liu Weilong et al. (2024) leveraged the LSTM model to construct trading strategies, proposing a margin trading portfolio strategy that outperformed the benchmark [5].

Scholars have also made improvements to algorithms such as RNN and LSTM to enhance predictive performance. A Goyal et al. (2023) combined RNN and LSTM to predict stock closing prices [6]. Alsharef A et al. (2020) enhanced the IndRNN model by replacing its default activation function with the more effective Parametric ReLU (PreLU), thereby improving prediction accuracy [7]. Zhang Ying and Li Lu (2024) augmented the LSTM model by incorporating long-term memory peephole connections in the input gate and a unified gate mechanism that couples three gating mechanisms, thereby strengthening long-term memory information transfer and model stability [8]. Xiao Tiantian (2024) integrated K-means clustering with LSTM and discovered that the hybrid K-means-LSTM model outperformed the standalone LSTM model in terms of prediction accuracy and stability [9]. Zhou Zhangyuan and He Xiaoling (2023) optimized the LSTM model using principal component analysis and attention mechanisms, enhancing its predictive capability for the consumer sector index [10]. Gao Yuan and Huang Ren (2024) analyzed trading data from the SSE 50, CSI 300, and CSI 500 indices from 2012 to 2022, finding that the combined CNN-LSTM model achieved higher prediction accuracy than individual CNN or LSTM models [11]. Yin Haiyuan and Yang Qingsong (2022) employed a Bidirectional LSTM (BiLSTM) model to analyze investor sentiment from real-time posts on the East Money Stock Bar, constructing a daily sentiment index to study its impact on stock price bubbles [12]. Wang Jiazeng (2023) improved the BiLSTM model using Multivariate Delay Embedding Tensor (MDT) processing technology, enabling effective prediction of stock price trends and assisting investors in determining optimal entry and exit points in the capital market [13].

3. Theoretical Foundation and Model Construction

3.1. RNN Model

In 1982, American scholar John Hopfield built a neural network with content-addressable memory based on Little's (1974)[14] neuromathematical model using binary nodes, i.e., Hopfield's neural network[15]. 1986, Michael I. Jordan proposed Jordan network under the theory of parallel distributed processing. Jordan proposed the Jordan network under the theory of parallel distributed processing[16]. Later in 1990, Jeffrey Elman proposed the first fully connected RNN, the Elman network[17].

The RNN model structure includes forward propagation and back propagation.Forward propagation of RNN is the process by which the network gets the output based on the input.As shown in the following figure 1, the RNN adds a feedback connection to the hidden layer, and the hidden layer data is passed horizontally in chronological order internally, in addition to being passed to the output layer. This allows the RNN to retain information from all previous moments, giving it a memory function. Expanding the graph, x_t represents the value input to the RNN network, s_t is the hidden state, indicating that the data generated by the processing of the brother t neurons within the hidden layer will be passed not only to the next layer of neurons, but also to the next neuron within the layer, o_t represents the output value of the hidden layer, which is determined by s_t, and there is also y_t after o_t, which represents the final real output, and in the actual application process In practice it is the predicted value of the target sequence, denoted as $\hat{y}_t$, and after y_t there is L_t, which is the value of the loss function calculated from the predicted value and the true value, and is used as a measure of the loss of the model at the current position.

The forward computation formula for RNN is：

$$s_t = f(Ux_t + Ws_{t-1} + b)\ ; \quad o_t = Vs_t + c\ ; \quad \hat{y}_t = g(o_t)$$

Where f、g is the activation function, U、V、W are the weight parameters, in the RNN network is weight sharing, which reflects the “circular feedback”, but also reduces the amount of operations.

The backpropagation of RNN is to solve the appropriate U、V、W、b、c by gradient descent method, in order to simplify the description, the loss function L is taken as the cross-entropy loss function, in the form of the following:

$$L(\hat{y}, y) = [y\ln\hat{y} + (1-y)\ln(1-\hat{y})]$$

Taking f as the tanh function and g as the softmax function, for the RNN, the final loss L is： $L = \sum_{t=1}^{\tau} L_t$

Since the output $\hat{y}_t = g(o_t) = g(Vs_t + c)$, the gradient calculation is relatively simple for V and c：

$$\frac{\partial L}{\partial c} = \sum_{t=1}^{\tau} \frac{\partial L_t}{c} = \sum_{t=1}^{\tau} \hat{y}_t - \hat{y} \ ; \ \frac{\partial L}{\partial V} = \sum_{t=1}^{\tau} \frac{\partial L_t}{\partial V} = \sum_{t=1}^{\tau} (\hat{y}_t - \hat{y})(s_t)^T$$

The gradient loss of the hidden state s_t consists of two parts: one from o_t in the output layer and the other from s_{t+1} in the next neuron. We define the gradient of s_t as:$\delta_t = \frac{\partial L}{\partial s_t}$

This allows us to recursively deduce $\delta_t$from δ_{t+1}, which is calculated as follows:

$$\begin{aligned}\delta_t &= (\partial o_t/\partial h_t)^T(\partial L/\partial o_t) + (\partial s_{t+1}/\partial s_t)^T(\partial L/\partial s_{t+1}) \\ &= V^T(\hat{y}_t - t_t) + W^T \mathrm{diag}(1-(h_{t+1})^2)\delta_{t+1}\end{aligned}$$

When at position t=τ, which is at the end of the sequence, there are no other items following it, the calculation is as follows:

$$\delta_\tau = (\partial o_\tau/\partial h_\tau)^T(\partial L/\partial o_\tau) = V^T(\hat{y_\tau} - y_\tau)$$

With δ_τ, we can introduce the gradients of W、U and b as follows:

$\frac{\partial L}{\partial W} = \sum_{t=1}^{\tau} \mathrm{diag}(1-s_t^2)\delta_t s_{t-1}^T$; $\frac{\partial L}{\partial U} = \sum_{t=1}^{\tau} \mathrm{diag}(1-s_t^2)\delta_t x_t^T$; $\frac{\partial L}{\partial b} = \sum_{t=1}^{\tau} \mathrm{diag}(1-s_t^2)\delta_t$

RNN advantage is to combine neural networks with time series, using loop feedback connections to realize the information transfer between neurons within the hidden layer, so that the data of the previous moment influences the current moment. However, RNN is prone to gradient vanishing and gradient explosion problems in training, and it is difficult to deal with long sequences. For this reason, LSTM optimization of RNNs by using an approach that improves the delivery method within a layer has become a hot topic of discussion among many scholars.

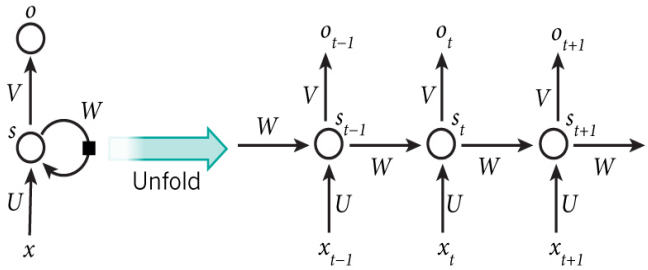

Figure 1. Structure of RNN

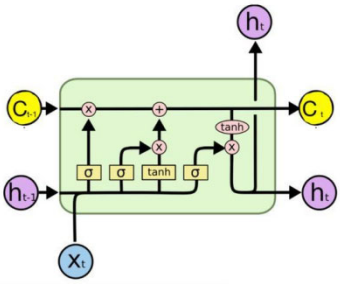

Figure 2. Structure of LSTM cell

3.2. LSTM Model

Long-Short Term Memory (LSTM) was proposed by Jurgen Schmidhuber et al. in 1997[18], which is a kind of recurrent neural network widely used in sequence data modeling also known as the RNN

variant introduced above LSTM solves the problem of long dependencies and has better performance in long time sequences.

The neurons in the hidden layer of RNN contain only one tanh function, which is the activation function f mentioned above, and this structure makes it impossible for RNN to preserve the long-term state.The structure of LSTM neurons is shown in the figure above, which contains two tanh activation functions and three σ activation functions, and in addition to the hidden stateh_t in the RNN, it also adds the cell state c_t, and LSTM can preserve the long-term state through the unitary state to maintain the long-term state of the sequence.

LSTM consists of one or more LSTM cells, there are three types of gating in LSTM, namely: input gates, forget gates and output gates, the gating can be regarded as a fully-connected layer, and the storage and updating of information by LSTM is realized by these gating. The gating is realized by sigmoid function and dot product operation, the general form of which is:

$$g(x) = \sigma(Wx + b)$$

Where the sigmoid function is: $\sigma(x) = \frac{1}{1+e^{-x}}$, which is a nonlinear activation function that maps a real value to the interval 0-1 and is used to characterize how much information passes. When the gate output value is 0 it means that none of the information passed and a value of 1 means that all of the information passed. w denotes the weight matrix of the network, b denotes the bias vector and tanh is the hyperbolic tangent function.

The input vector of the hidden layer is x_t, the output vector is h_t and the memory cell is c_t.

Input gate: used to decide how much new information to add to the memory cell c_t, the output ranges between 0 and 1, and is computed by the formula:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_t)$$

Oblivion Gate: is a key component of the LSTM unit that controls the self-connecting unit and decides which part of the information is to be discarded and which part is to be retained. The output ranges from 0 to 1 and is calculated as:

$$f = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$

The information in the previous moment's memory cell c_t also affects the current memory cell c_{t-1}:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

Output gate: controls the effect of the memory cell c_t on the output value h_t, i.e., at time step t, which part of the memory cell will be output. The output gate is calculated as follows:

$$O_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$

The output of the LSTM cell at moment t is h_t,

$$h_t = o_t \odot \tanh(c_t)$$

With the control of these gates, LSTM can effectively deal with long-term dependencies in sequence data, making it outstanding in sequence modeling.

3.3. BiLSTM Neural Network

Bidirectional Long Short-Term Memory (BiLSTM) As an improvement of the LSTM model, Bi-LSTM is a method of processing sequential data in the forward and backward directions by using two independent LSTM networks, which can preserve both future and past information. The forward LSTM is responsible for processing sequences sequentially, while the backward LSTM is responsible for the opposite order. Finally, the outputs of the two networks are connected to produce a final prediction as shown in the figure 3.You can see that the forward hidden layer output at the moment x_t and the reverse hidden layer output are spliced to get the hidden layer output at the current moment y_t.

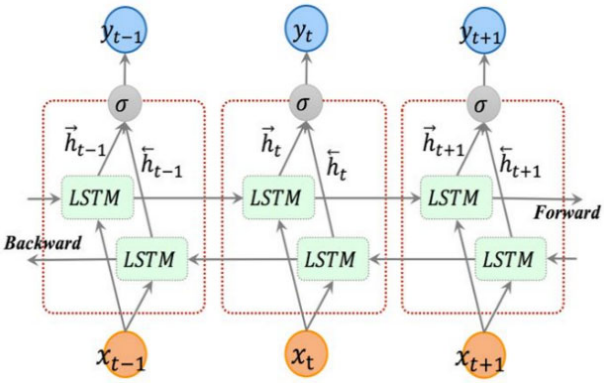

Figure 3. Structure of BiLSTM model

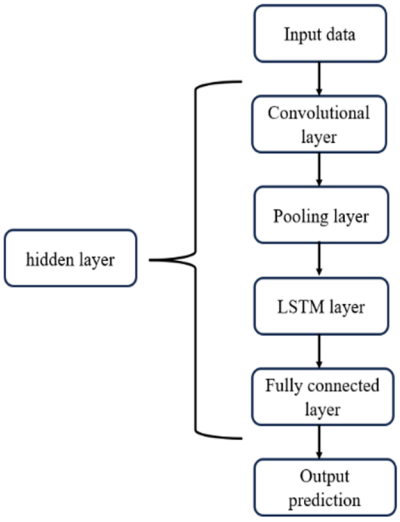

Figure 4. Structure of CNN-LSTM neural network model

3.4. CNN-LSTM Neural Network Model Structure

As Convolutional Neural Networks (CNN) model has better data feature extraction advantages, it can extract the high-dimensional feature information that LSTM model can't extract, and dig out the law of stock price changes; LSTM model has unique advantages for time series data, it can grasp the importance of the data in different periods and give different weights to avoid the early data gradually disappearing during training, and more accurately predict the stock price. The LSTM model has a unique advantage for time series data, which can grasp the importance of data in different periods and assign different weights to avoid the gradual disappearance of early data during training, so as to predict the stock price more accurately. In this paper, a CNN-LSTM combined neural network model is constructed, which consists of three modules: the input layer, the hidden layer and the fully connected layer (output layer), and the structure of the CNN-LSTM combined neural network model is as shown in the figure 4 above.

4. Stock Forecasting Application

4.1. Experimental Data and Preprocessing

4.1.1. Data Source

The data in this paper comes from the Tushare module in Python software, and a stock listed and traded on the main board of the Shanghai Stock Exchange ------ Ping An Bank (stock code 000001) is selected, and the stock's timeframe is from January 1, 2015 to June 1, 2024, with a total of 2,287 pieces of data. The data obtained from the Tushare module contains a total of 9 main variables.The

data obtained is organized by time. The acquired data are arranged in order of time from far to near, in units of days. Among the many indicators for stock price prediction, this paper chooses the classic OHLV indicator as the input variables of the model, i.e., opening price, high price, low price, turnover, and volume.

4.1.2. Data Preprocessing

Data preprocessing is divided into two steps: first, the raw data are initially processed to check whether there are problems such as missing values. The second step needs to standardize the data to eliminate the effect of magnitude and make the data in the same magnitude. In this paper, the Min-Max standardization method is used. This method can maximize the retention of the original characteristics of the stock price series, and its trend graph is also similar to the shape of the stock trend curve.

4.2. Model training and parameter settings

4.2.1. RNN Model Structure

The closing price is chosen for stock price prediction, and the Sequential model in Keras is used to build the network, with the first SimpleRNN layer and and the second SimpleRNN layer having a number of 100 neurons.Dropout=0.1 in both layers, 10% of the neuron outputs will be randomly zeroed in each training iteration. The Adam algorithm is used as the optimizer of the model, and the learning rate parameter passed to the Adam optimizer is 0.001, which is a hyperparameter controlling the magnitude of weight update during the model training process. The loss function loss is set to mean_squared_error, the batch size is set to batch_size=64, the number of training rounds is set to epochs=100, and the validation frequency is set to validation_freq=1, which specifies that validation is performed once at the end of each epoch.

4.2.2. LSTM Model Structure

For the neural network model, Keras, a deep learning library in Python, was mainly used for the construction and training of the model. The parameters of LSTM are set as follows: total sample size 2287, number of training sets 1824, number of test sets 458; time step is 5, which means that the closing price of the past 5 days is used to predict the closing price of the next 1 day; batchsize = 4; the dropout ratio of 0 is a regularization technique, which prevents overfitting by randomly dropping a portion of the activation values of the neurons during training. neuron's activation value during training to prevent overfitting. In the first LSTM layer, dropout=0.4: means 40% of neurons may be dropped in each training iteration, and in the second LSTM layer, dropout=0.5, means 50% of neurons may be dropped in each training iteration; the number of nodes in the first fully-connected layer is 32, and the number of nodes in the second fully-connected layer is 16, and the number of training rounds is 30.

4.3. Evaluation Metric Selection

In order to evaluate the prediction effect of the model, four evaluation indexes are used: mean square error (MSE), root mean square error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE). MAE is the weighted average of the absolute value of the difference between the predicted value and the true value, which avoids the errors from canceling each other out and accurately reflects the magnitude of the prediction error; MAPE describes the average deviation between the predicted value and the true value; the smaller the values of MSE, RMSE, MAE and MAPE are, the higher the prediction accuracy of the model is.

4.4. Analysis of Model Prediction Results and Performance Comparison

4.4.1. Analysis of RNN Model Prediction Results

Line plots of the changes in Training Loss and Validation Loss during the training process of RNN:

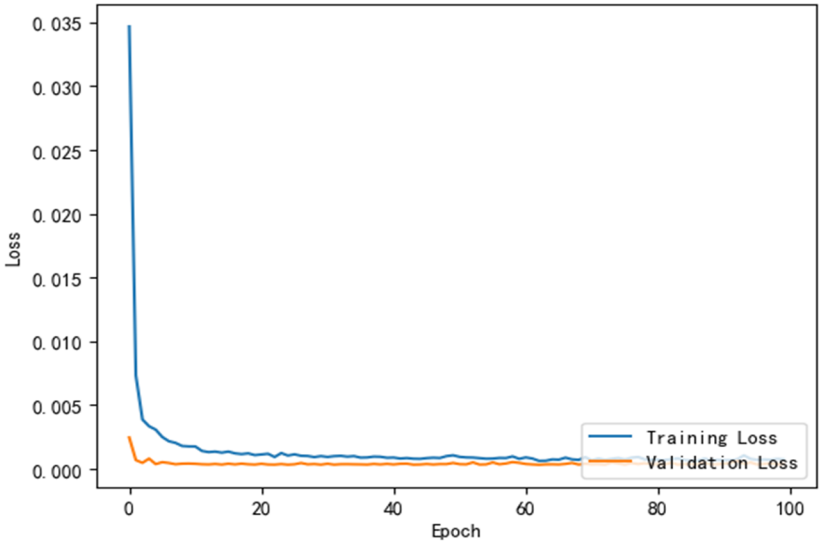

Figure 5. Loss variation graph for batch_size=16

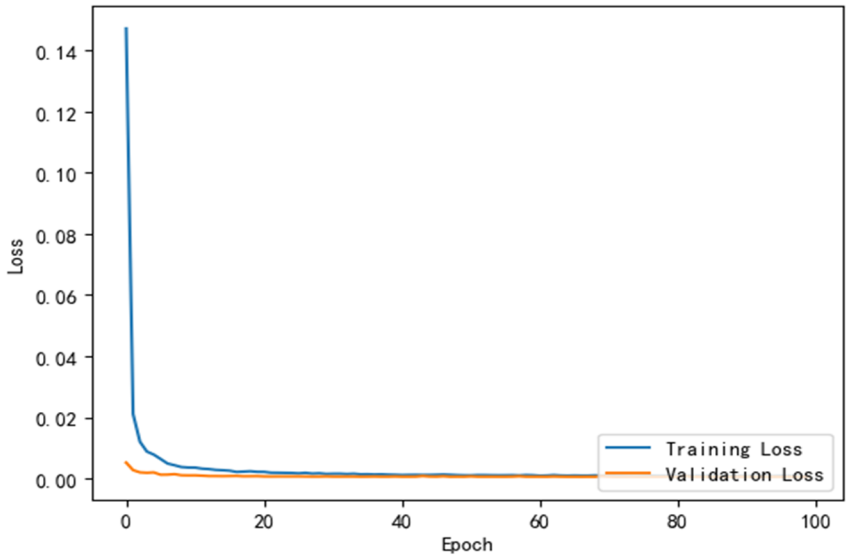

Figure 6. Loss variation graph for batch_size=64

The X-axis represents the number of training rounds (Epochs), ranging from 0 to 100, and the Y-axis represents the loss value, ranging from 0 to 0.14, where the lower the loss value, the better the model performance. In general, the training loss decreases with the number of training rounds, reflecting the model's performance improvement on the training data. The validation loss also decreases with the number of iterations, but the validation loss may rise after a certain node if the model is overfitting the training data. The double line in the figure has a significant gap at the beginning of training, implying that the model learns too fast on the training set and underperforms on the complex validation set. As the number of iterations increases, the double lines gradually approach and decrease simultaneously, indicating that the model performs well on both the training and validation sets without overfitting. If the training loss continues to fall while the validation loss starts to rise, it may indicate that the model is overfitting.

We tuned the batch_size parameter, which defines the number of samples in each training batch and directly affects the number of samples used in each iteration to update the model weights. A smaller batch_size (e.g., 16) helps the model learn data diversity, but may lead to slower training and tend to fall into local optima. A larger batch_size (e.g., 64) speeds up training, but may increase memory consumption and reduce training accuracy. By comparing the loss function images of

batch_size=16 and batch_size=64, it is found that the loss value is lower for batch_size=16 (0-0.035), while the loss value is higher for batch_size=64 (0-0.14), but the two loss lines of the latter converge to 0 faster and show better performance. Therefore, batch_size=64 was chosen, i.e., 64 samples were used per iteration to update the model weights.

The normalized data is divided into a training set and a test set with a ratio of 80% and 20%, respectively. The training set is used for RNN model training, and the test set is used for analyzing the model prediction results.

Figure 7. Stock Price Forecast Chart

Figure 8 Stock price test set prediction effect

The RNN model is trained, and the loss function is derived as, the predicted data obtained by substituting the model using the test set and the original real data are visualized, as shown in the figure above, the predicted stock price is very close to the actual stock price, indicating that the prediction effect is good.

4.4.2. Analysis of LSTM Model Prediction Results

Here the loss values and test set predictions of the four models (LSTM, BiLSTM, CNN-LSTM, CNN-BiLSTM) will be compared separately. Firstly, the loss values of the LSTM model on the training set.

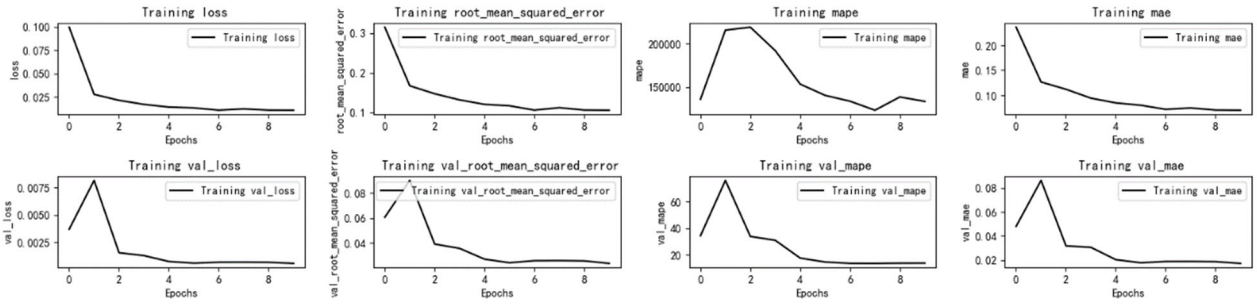

Figure 9. Loss metrics on the training set (LSTM)

The X-axis represents the number of training rounds (Epochs), here Epochs are 10, which is the number of times the model has been iterated over the dataset.The Y-axis represents the different error metrics, each of which has its own range of measurement. Each line in the graph represents the trend of a metric with the number of training rounds. It can be seen that these error metrics are decreasing as the training progresses, which indicates that the model's performance on the training data is improving, and the error basically tends to 0 when Epochs = between 8 and 10.

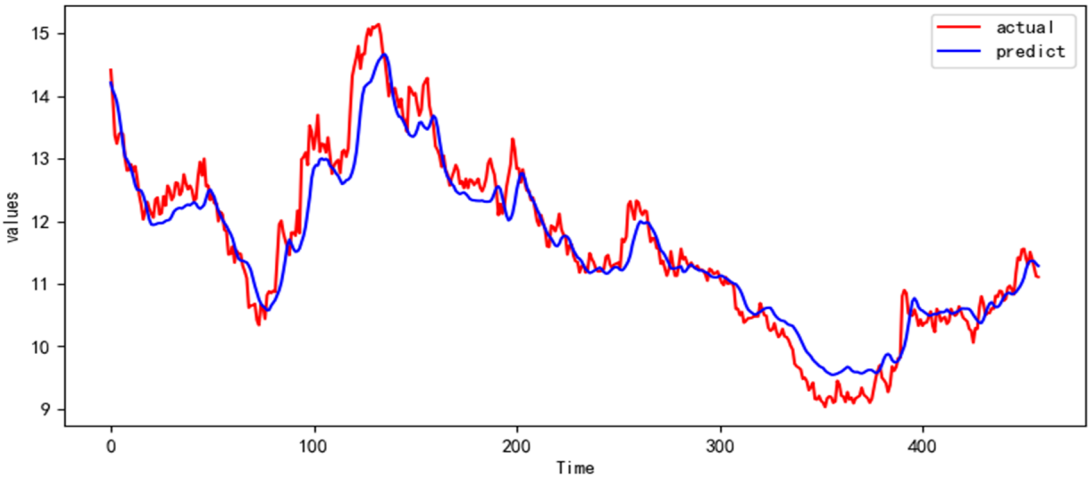

Figure 10. Comparison of LSTM model predicted stock prices

The overall trend of the predicted data in the test set is consistent with the true values, but there is not much overlap. At this point, the MSE value is 0.15, the RMSE value is 0.388, the MAE value is 0.284, and the MAPE value is 0.024.

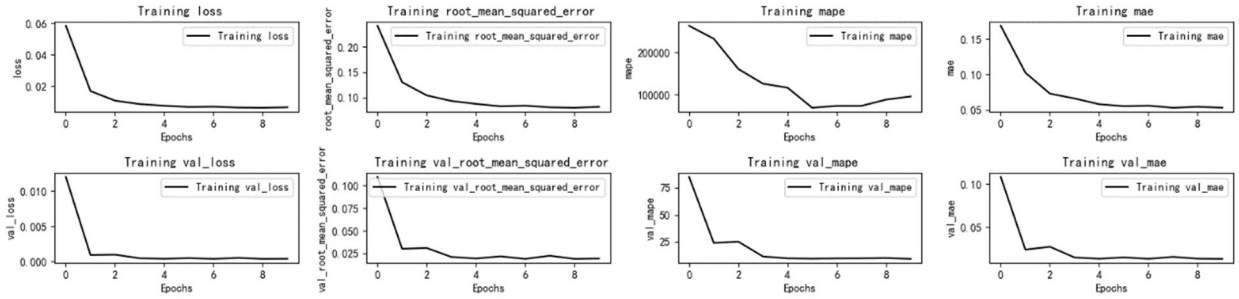

Figure 11. Loss metrics on the validation set (BiLSTM)

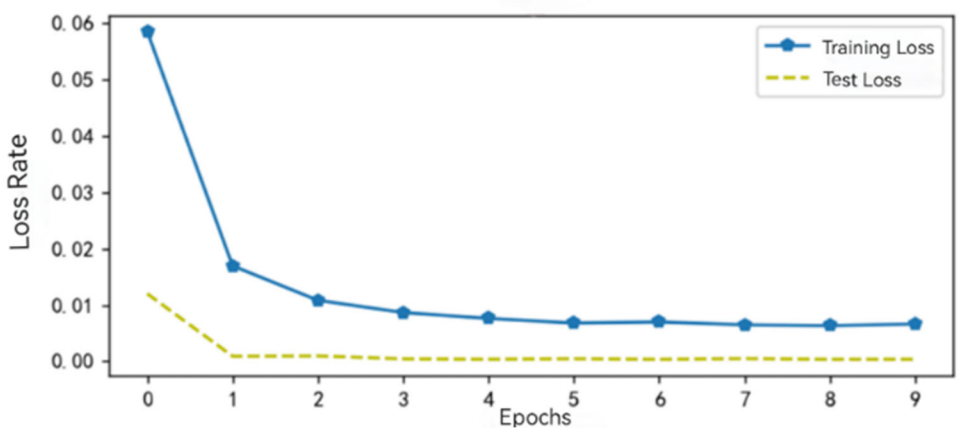

Figure 12. Loss variation on the training and test sets of the BiLSTM model

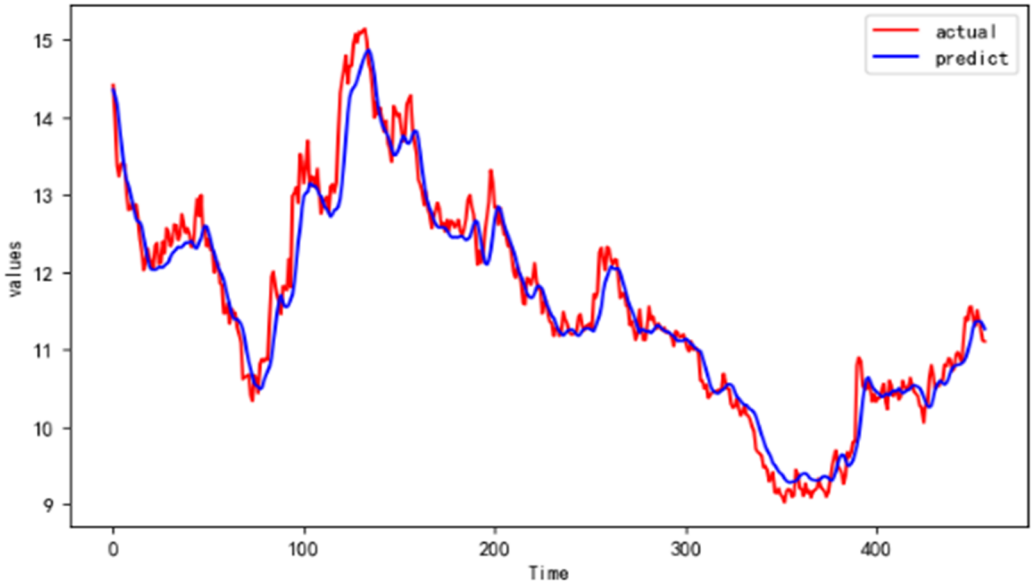

Figure 13. Comparison of BiLSTM model predicted stock prices

It can be seen that the overall trend of the predicted data in the test set is more consistent with the true value under the BiLSTM model prediction effect. At this time, the MSE value is 0.101, the RMSE value is 0.317, the MAE value is 0.223, and the MAPE value is 0.019, and the error values are all lower than the previous LSTM model.

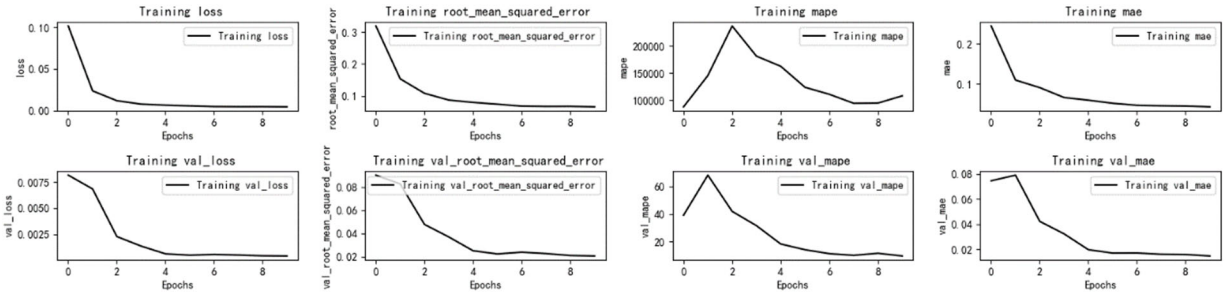

Figure 14. Loss indicators on the validation set (CNN-LSTM)

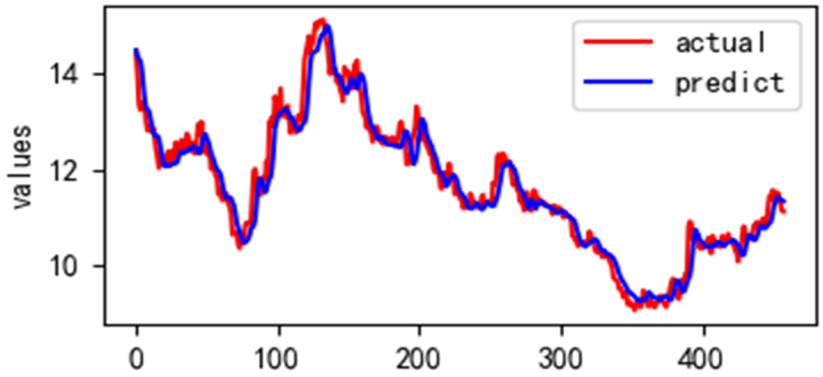

Figure 15. Comparison of CNN-LSTM models predicting stock prices

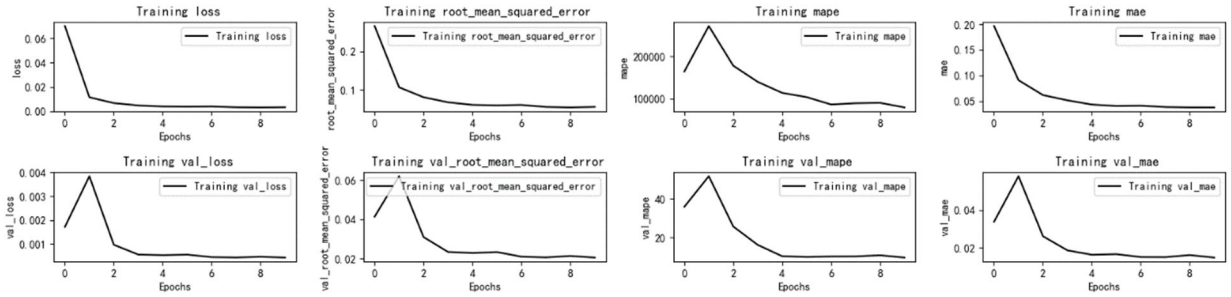

Figure 16. Loss indicators on the validation set (CNN-BiLSTM)

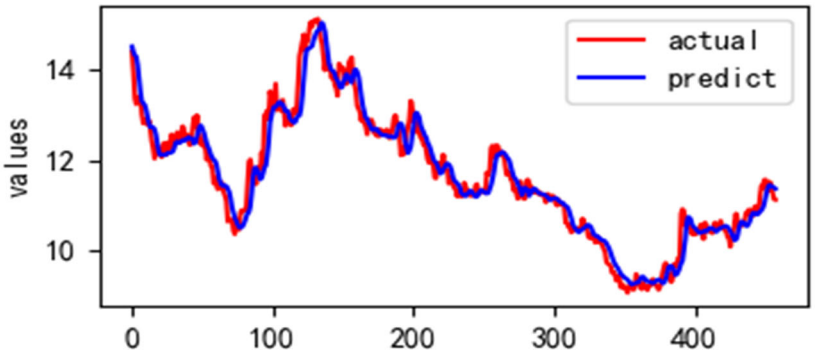

Figure 17. Comparison of CNN-BiLSTM models predicting stock prices

With the predictive effect of the hybrid CNN-LSTM model, the overall trend of the predicted data in the test set is closer to the true value. At this time, the MSE value is 0.114, the RMSE value is 0.338, the MAE value is 0.242, and the MAPE value is 0.02, which is higher than the loss value of the previous BiLSTM model.

Finally, there is a graph of the prediction effect of the hybrid CNN-BiLSTM model, and the overall trend of the predicted data in the test set is more consistent with the real value. At this time, the MSE value is 0.114, the RMSE value is 0.337, the MAE value is 0.244, and the MAPE value is 0.021, which is very close to the loss value of the previous model, CNN-LSTM model, and it can be seen that the prediction effect of these two hybrid models is relatively close.

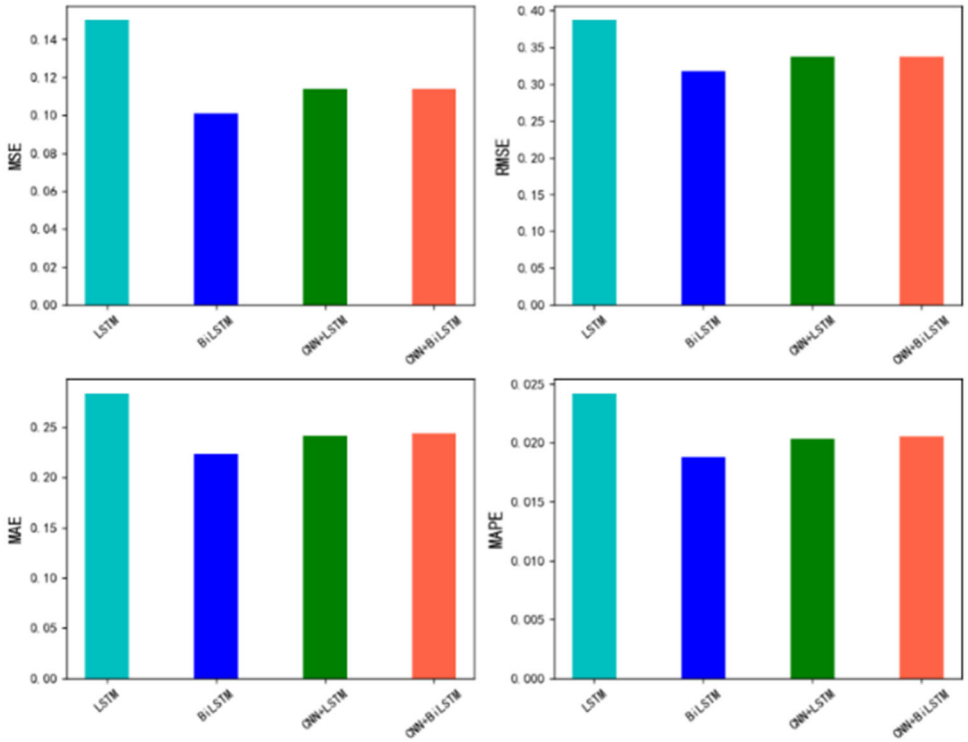

Figure 18. Comparison of the errors of the four models

4.4.3. Model Prediction Result Comparison

The figure 18 shows the performance of the four evaluation indicators in different models.The horizontal axis represents the four different models, which are LSTM, BiLSTM, CNN-LSTM, and CNN-BiLSTM models. The vertical axis represents the error values, and by comparing the bar lengths of the different models, it is possible to visualize which models perform better. The shorter the bar, the smaller the error of the model and the better the performance, it can be seen that whether it is MSE、RMSE、MAE and MAPE, the bar of BiLSTM represented by the blue bar graph is the shortest, then it is inferred that the BiLSTM has the best fitting effect in this stock price prediction.

5. Summary and Outlook

In this paper, for the stock price prediction problem, RNN, LSTM, BiLSTM, CNN-LSTM and CNN-BiLSTM models are used to predict the stock price of Ping An Bank, and the performance of the models is evaluated by four error functions. The results show that the BiLSTM model has the smallest prediction error (MSE, RMSE, MAE, MAPE) and the highest fit between the prediction results and the real value, which provides reference value for investors' decision-making. However, the research in this paper has the following shortcomings: Firstly,only single-feature LSTM is studied, and the full comparison of multi-feature LSTM and hyper-parameters (such as the number of iterations, neuron random discard rate, and optimizer) is not involved; Secondly,in terms of the model structure, the hyper-parameters of LSTM can be optimized by combining with Kmeans clustering, genetic algorithms, or the introduction of Transformer model in the future; Lastly, the impacts of macro-economics, financial conditions on stock prices are not considered.

Acknowledgment

(Supported by the China Society of Higher Education under the project “Research on the identification and prevention of financial risks in countries along the Belt and Road under the new pattern of financial liberalization” (202401hx0209))

References

[1] Sharma,Suresh Kumar and Vinod Sharma.“Comparative Analysis of Machine Learning Techniques in Sale Forecasting.” International Journal of Computer Applications 53 (2012): 51-54.

[2] Alsharef A,Sonia,Arora M,et al.Predicting Time-Series Data Using Linear and Deep Learning Models—An Experimental Study[J]. 2022.

[3] Huang J,Wang Y.Comparative Analysis of Different Machine Learning Techniques in Forecasting Stock Price.International Conference on Artificial Intelligence Innovation[J].2024.

[4] Kalidindi A R,Ramisetty N S,Kruthiventi S S,et al.Comparative Analysis of RNN Variants Performance in Stock Price Prediction[J]. 2023.

[5] Liu Weilong,Zhang Yong,Yang Xingyu.Online financing and bond portfolio trading strategy based on LSTM prediction information[J].Systems Engineering Theory and Practice,2024.

[6] Goyal A,Choudhary A,Malik D,et al.Implementing andAnalysis ofRNN LSTM Model forStock Market Prediction[J]. 2023.

[7] Alsharef A,Bhuyan P,Ray A.Predicting Stock Market Prices using Fine-Tuned IndRNN[J].International Journal of Innovative Technology and Exploring Engineering, 2020, 9(7):7.

[8] Zhang Y,Li L.RF-MIP-LSTM stock price prediction model[J]. Computer Engineering and Applications:1-14[2024-06-18].

[9] Xiao Tiantian.Stock price prediction of securities based on K-means-LSTM model[J]. Science and Industry,2024,24(03):210-215.

[10] Ziongyuan,He Xiaoling.Stock price prediction method based on optimized LSTM model[J]. Statistics and Decision Making,2023,39(06):143-148.

[11] Gao Yuan,Huang Y.Stock price index prediction based on deep learning[J]. Software Engineering, 2024, 27(05): 7-13.

[12] Yin Haiyuan,Yang Qingsong.The impact of investor sentiment on stock price bubbles in stock bars based on Bi-LSTM model mining[J]. Journal of Management,2022,19(12):1874-1885.

[13] Wang Jiazeng. Research on stock price prediction and stock portfolio based on improved LSTM neural network[D].Xi'an University of Architecture and Technology,2023.2.2

[14] Little, W.A., 1974.The existence of persistent states in the brain. In From High-Temperature Superconductivity to Microminiature Refrigeration (pp. 145-164). Springer, Boston, MA.

[15] Hopfield, J.J.,1982.Neural networks and physical systems with emergent collective computational abilities. Proceedings of the national academy of sciences, 79(8), pp.2554-2558.

[16] Jordan, M.I.,1986. Serial order: A parallel distributed processing approach (Tech. Rep. No. 8604). San Diego: University of California, Institute for Cognitive Science.

[17] Elman, J.L.,1990. Finding structure in time. Cognitive science, 14(2), 179-211.

[18] Hochreiter S,Schmidhuber J.Long Short-term Memory[J].Neural Computation,1997,9(8):1735-1780.