Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 5
AIDML 2024

Bitcoin prediction and parameter analysis based on LSTM

Ding Wang *

Jurong Country Garden School, Nanjing, China

* Corresponding Author Email: 20202362@stu.hebmu.edu.cn

Abstract. Stock price prediction is currently a research focus in the financial field, especially in blockchain research. The central focus of this research is to forecast Bitcoin's closing price through the integration of deep learning techniques, specifically employing Long Short-Term Memory (LSTM). This study takes into account that Bitcoin is a mainstream virtual currency, and predicting its future price can help investors make better judgments in trading. The goal of this exploration is to identify the most favorable parameter combinations and function prediction applications, ultimately obtaining the most accurate prediction results. The research process includes dataset selection, data processing, model construction, and training. Then adjust and improve the parameters used in the model, and record the process. Finally, test the model and output the test results. And model testing and result output. At the end of the experiment, the effects of different optimizers and parameters on the training results were compared, and the optimal combination was found. The model's predictive accuracy was evaluated through the examination of test data. This study can provide valuable references for researchers and firms.

Keywords: Long Short-Term Memory; Stock price prediction; Deep learning.

1. Introduction

In 2008, Satoshi Nakamoto introduced the idea of Bitcoin as a decentralized digital asset operating on a peer-to-peer network. The Bitcoin Network emerged in 2009 when its creator mined the inaugural Bitcoin block. With a projected issuance limit of approximately 21 million coins expected by 2140, Bitcoin boasts a robust scarcity factor [1]. Compared with other virtual currencies, Bitcoin has a huge advantage. Users can instantly pay and receive any amount of money anytime and anywhere, which means that Bitcoin has a high degree of payment freedom. Furthermore, Bitcoin transactions currently incur minimal or no handling fees. Users have the option to include a fee within their transaction to prioritize processing, leading to quicker confirmation times from the network. Additionally, Bitcoin transactions offer security and irreversibility, devoid of any sensitive or personal customer information [2]. This feature effectively safeguards merchants from potential losses arising from fraud or chargebacks. Therefore, Bitcoin trading is a reliable way of financial management, and the more accurate prediction of the price of Bitcoin has a high reference value for individual users.

The prediction of financial product price is a time series problem. In the late 20th century, Donaldson employed Artificial Neural Networks (ANNs) to forecast the S&P500 stock price, underscoring the comparative advantages of neural networks over conventional methodologies like the weighted least squares method [3]. Similarly, Takeuchi and Lee applied an autoencoder based on a stack-limited Boltzmann machine to identify stock price characteristics and predict monthly stock returns compared to the median. Their model achieved an accuracy of 53% and an annual return rate of 45.93% [4]. Presently, with the proliferation of deep learning models, individuals have access to a multitude of options, including various standalone or hybrid models.A variety of models have been proposed to forecast Bitcoin prices, each offering specific advantages and disadvantages. Among these models are the Long Short-Term Memory (LSTM) model ,the Autoregressive Distributed Lag (ARDL) model,the Autoregressive Integrated Moving Average (ARIMA) model, and several others. For instance, McNally employed Bayesian optimization recurrent neural networks as well as long short-term memory networks. Through a comparative analysis with the widely utilized ARIMA prediction

method, it was observed that the nonlinear deep learning approach exhibited significantly superior performance [5]. Bao utilized wavelet transform in conjunction with an LSTM model to forecast the indices of six stocks, observing superior accuracy compared to both Recurrent Neural Network (RNN) and standard LSTM models [6]. Meanwhile, Wang utilized a linear ARIMA model for time series prediction and implemented support vector regression (SVR) for error series forecasting. Additionally, he integrated the predictions of the ARIMA and SVR models into a deep LSTM model, optimizing its hyperparameters using the Bayesian optimization algorithm. The experimental findings demonstrate that, in comparison to alternative hybrid models, the proposed model effectively enhances prediction accuracy across five distinct time series forecasts [7]. To sum up, LSTM has great advantages over other models in solving the problem of long-term dependence.

The primary goal of this research is to investigate the influence of various parameters on the prediction outcomes of LSTM. The prediction procedure entails the following steps: Initially, determine the features to be employed, the number of lookback days, and the ratio of the training set to the test set. Subsequently, preprocess the data in accordance with the specified requirements.Next, the parameters of the LSTM model are configured, and the model training methodology is established. Then the LSTM model is trained. Following the training phase, the model will be employed to forecast the data within the test set. Subsequently, the test results alongside the original data will be outputted, with the test results being saved for further analysis. All in all, the results of this research can effectively help Bitcoin investors to assess the risk of trading.

2. Methodology

2.1. Dataset Description and Preprocessing

The dataset utilized in this study is the Bitcoin Price Dataset sourced from Kaggle [8]. The dataset encompasses records spanning from September 17, 2014, to February 19, 2022, comprising daily data points for Bitcoin, including dates, opening prices, closing prices, highest prices, lowest prices, adjusted closing prices, and trading volumes. Except for the volume and date, all other data are kept to six decimal places. Since the closing price is the predicted value in this experiment, other dataset attributes can function as features. The actual closing price serves as reference data for training the model and evaluating the prediction results. For the study, the closing price along with the highest and lowest prices are extracted and employed as feature data. To be specific, 80% of the dataset is assigned for training purposes, while the test set is composed of the remaining 20%. The closing price of each day are predicted by the price data of the previous 29 days as a standard, so the data is divided into a group of 30 days. To ensure the enough stationarity of the data, all the data are processed by first-order difference and converted into floating-point numbers ranging from -1 to 1.

2.2. Proposed Approach

This study aims to investigate various factors influencing the efficiency and accuracy of the model, including parameters within the model, the choice of loss function, and the optimizer. This exploration is conducted by employing the LSTM model to forecast the future price of Bitcoin. In addition, accurate prediction will also be of great help to invest in Bitcoin. Figure 1 shows the Flow of program execution. As depicted in Figure 1, the model parameters undergo adjustment subsequent to data preprocessing. Next, a portion of the data is input into the model to train the model. After training, the model is tested by predicting the test data. Finally, the test results are output by the program in the form of charts.

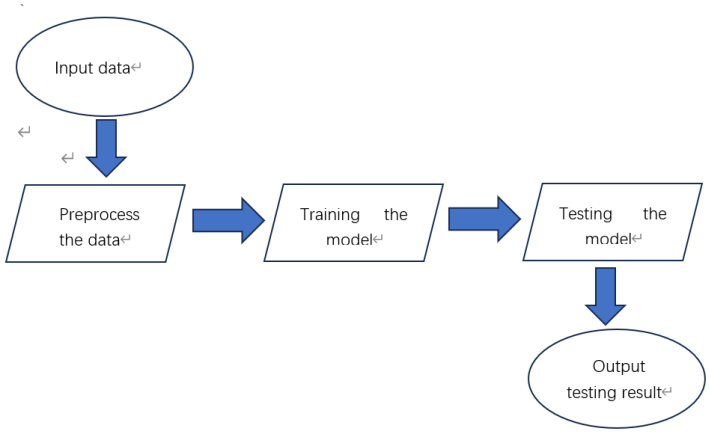

Figure 1. The pipeline of the study

2.2.1. LSTM

The model used in the paper study research is LSTM, which can achieve more accurate results in prediction due to its advantages over other models in solving time series problems with long-term dependence. LSTM, an advanced iteration of the Recurrent Neural Network (RNN), encounters notable hurdles when dealing with distant nodes in time series analysis. This challenge stems from the intricate computation of relationships between distant nodes, entailing multiple Jacobian matrix multiplications, which often result in gradient vanishing or exploding issues. To mitigate these challenges, LSTM adjusts the input threshold, forgetting threshold, and output threshold, enabling variable weights for self-loops. Consequently, the integral scale at different time steps dynamically adjusts while maintaining fixed model parameters, thus circumventing problems related to gradient vanishing or exploding. In LSTM, there are four function layers in each unit, which are forgetting gate, memory gate, memory cell update layer, and output layer. The function of the forget gate is to selectively forget some components of the previous unit state that are not meaningful for subsequent prediction, so as not to let too much memory affect the processing of the current input by the neural network.

The memory gate serves as a control mechanism determining whether the current data should be integrated into the cell state. Within this structure, the first-layer function is responsible for extracting relevant information from the present vector, while the second-layer function regulates the extent to which these memories are incorporated into the unit state. Simultaneously, the output gate, an essential element of the LSTM unit, calculates the output value at the present time step. Additionally, the output layer initially retrieves information from the vector, formed by the current input value and the previous time step's output value, through the first-layer function. Subsequently, it compresses the current unit state to the range (-1,1) through the second-layer function. The output of LSTM at this time can be obtained by multiplying the vectors processed by two layers of functions point to point. There are also different variants of LSTM, such as Tree-LSTM, Bit-LSTM, etc.

2.2.2. Model Operation

Following the preprocessing of the data, the subsequent step involves configuring the parameters of the LSTM model. The number of neurons directly impacts the quantity of conventional hidden units within each memory unit block [10].The input size dictates the number of input features, while the number of layers specifies the count of LSTM layers. Subsequently, the training data is fed into the LSTM model for model training. The training process incorporates parameters such as Epoch, loss function, and optimizer. The Epoch parameter determines the total number of training iterations, the loss function calculates the disparity between the actual and predicted values, and the optimizer guides each parameter of the loss function towards appropriate updates in the correct direction. During each training iteration, following data processing by the LSTM layer, the prediction outcomes for the target days flow into the fully connected layer and are transformed into one-dimensional output

data. Upon completion of training, the model is employed to forecast data within the test set, with the resulting predictions saved and outputted

2.2.3. Loss function

Throughout the training period, the loss function serves as a benchmark for quantifying the disparity between the model's predictions and the ground truth, thereby guiding the optimization process of the model. During the study, the mean square error loss (MSE) function is selected as the designated loss function.Its formula is shown as follows:

$$J_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(yi - \hat{y}i)^2 \quad (1)$$

Where $\hat{y}i$ represents the predicted value, while y signifies the true value.

2.3. Implementation Details

The Implementation details are as follows: The pycharm version used in the experiment is 2023.3.2, the LSTM model is from the pytorch function library, and the test data is from kaggle. Computer system: Windows 10. Model parameters are as follows: number of features 3, lookback 30, number of neurons 32, number of LSTM layers 2, The output dimension is 1. The model training parameters are as follows: the loss function is MSE, the number of training times is 100, and the optimizer is Adam.

3. Results and Discussion

3.1. Influence of Different Optimizers on Training Result

During model training, the choice of optimizer holds significant importance. This study endeavors to examine the influence of three distinct optimizers on the outcomes of predictions. Specifically, the Adam optimizer, the momentum optimizer, and the stochastic gradient descent (SGD) optimizer are under scrutiny.SGD operates by selecting one mini-batch at a time rather than the entire dataset, updating model parameters using gradient descent. While SGD addresses the issue of random small batch sampling, its efficiency is comparatively lower, and it may exhibit instability when dealing with very large datasets, leading to oscillation phenomena. The momentum is an improved algorithm for SGD. It introduces a first order momentum in the SGD basis. The exponentially weighted moving average of the gradient at that time. And Adam adds the second order momentum to the SGD [10,11]. The Figures 2-4 shows the Predicted results after using different optimizers. By comparing Figure 2-4, it can be seen that Adam optimizer is indeed significantly more efficient than SGD and Momentum optimizer under the same training times.

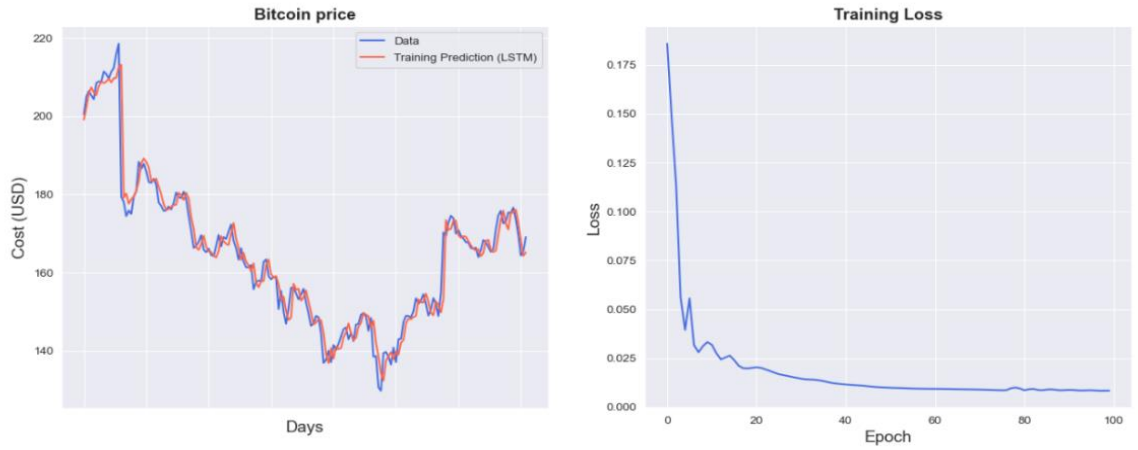

Figure 2. Predicted results after using Adam optimizer

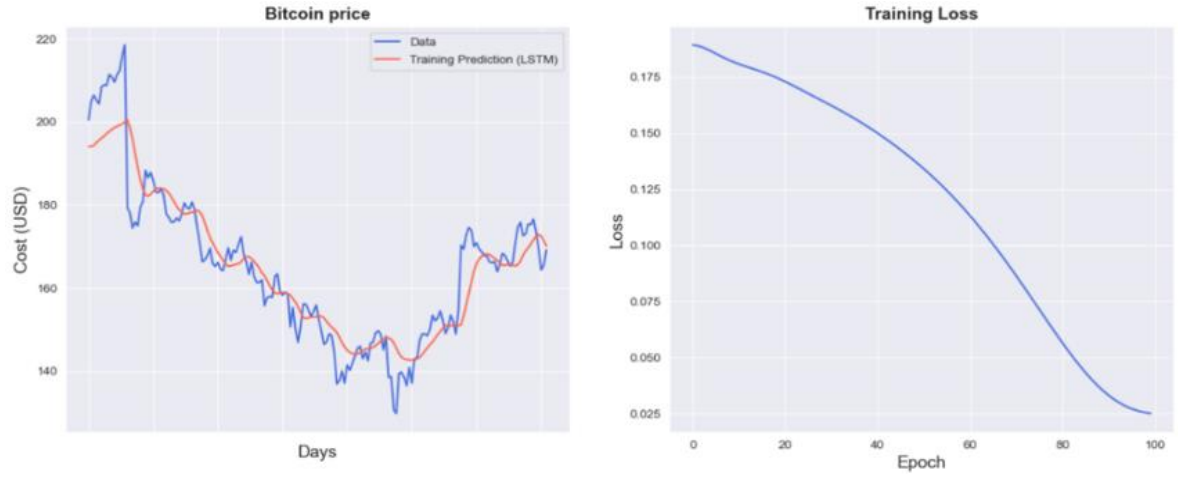

Figure 3. Predicted outcomes after the implementation of Momentum optimizer

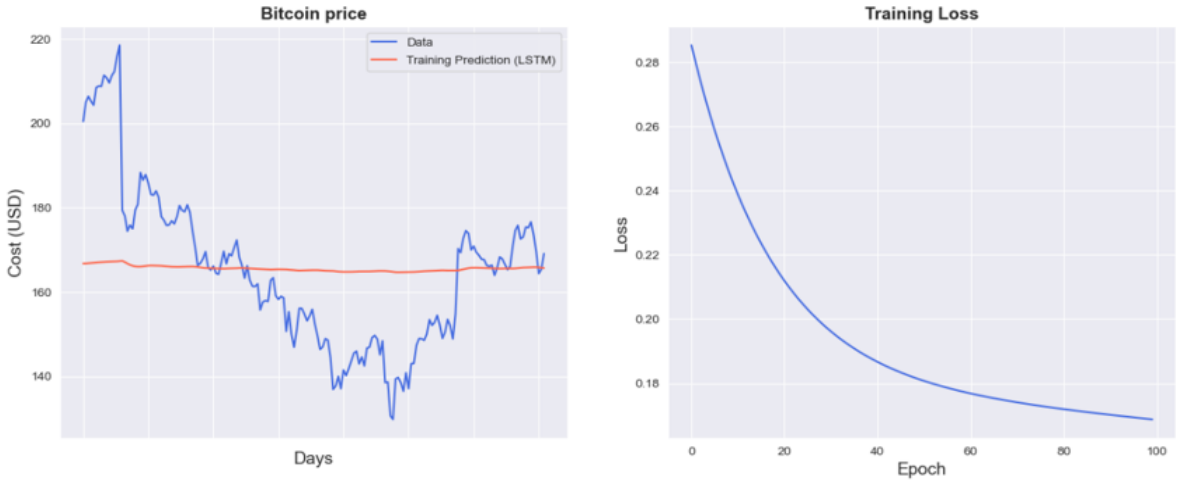

Figure 4. Predicted outcomes after the implementation of SGD optimizer

3.2. Influence of Different Neuron numbers on Training Result

In the subsequent process, diverse neuron quantities were employed within the LSTM model. The evaluation of the disparity between the predicted outcomes and the actual values was conducted through the utilization of the root mean square error (RMSE) calculation.The following is the influence of different neurons on the predicted result under the same number of training. By comparing the visual training results of Figure 5-8 with RMSE, it can be concluded that different numbers of neurons have no significant effect on the efficiency of training.

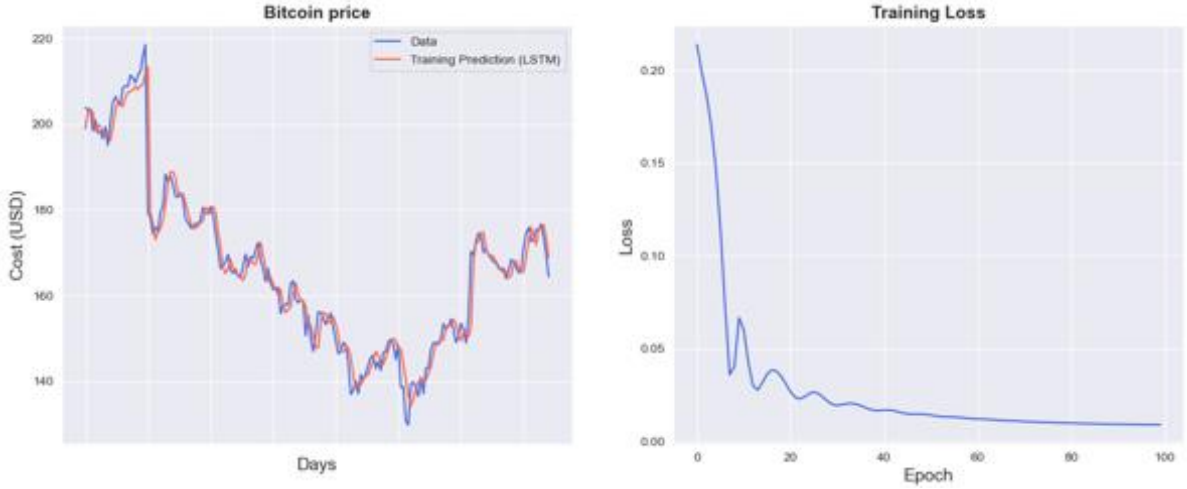

Figure 5. Neurons number:16, RMSE: 4.03

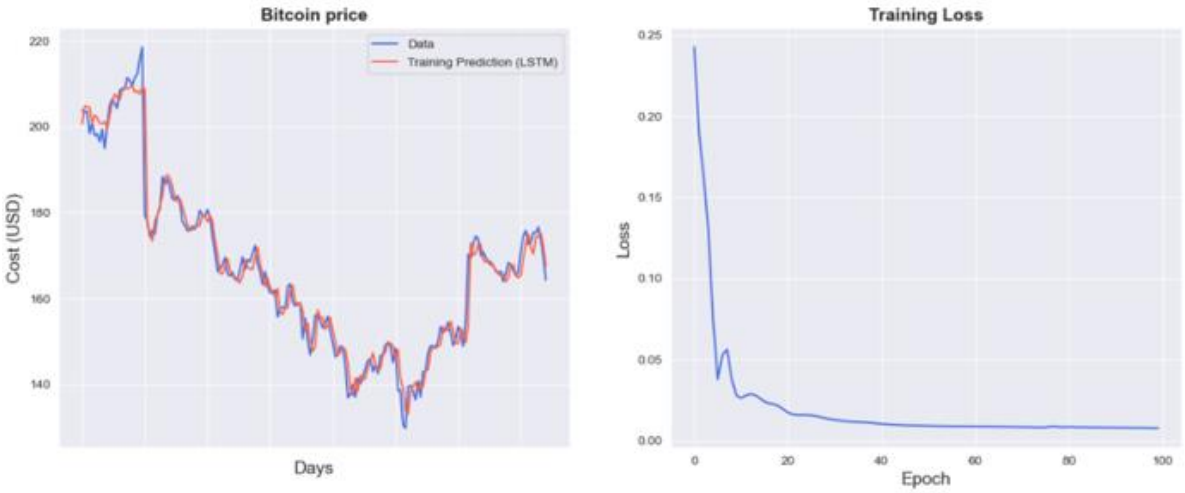

Figure 6. Neurons number:32, RMSE: 4.03

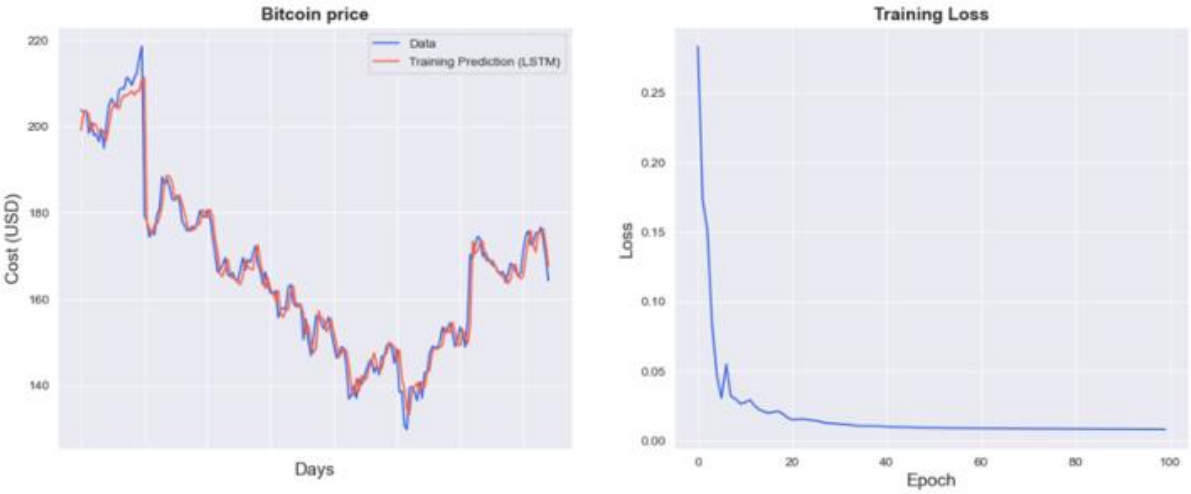

Figure 7. Neurons number:64, RMSE: 4.01

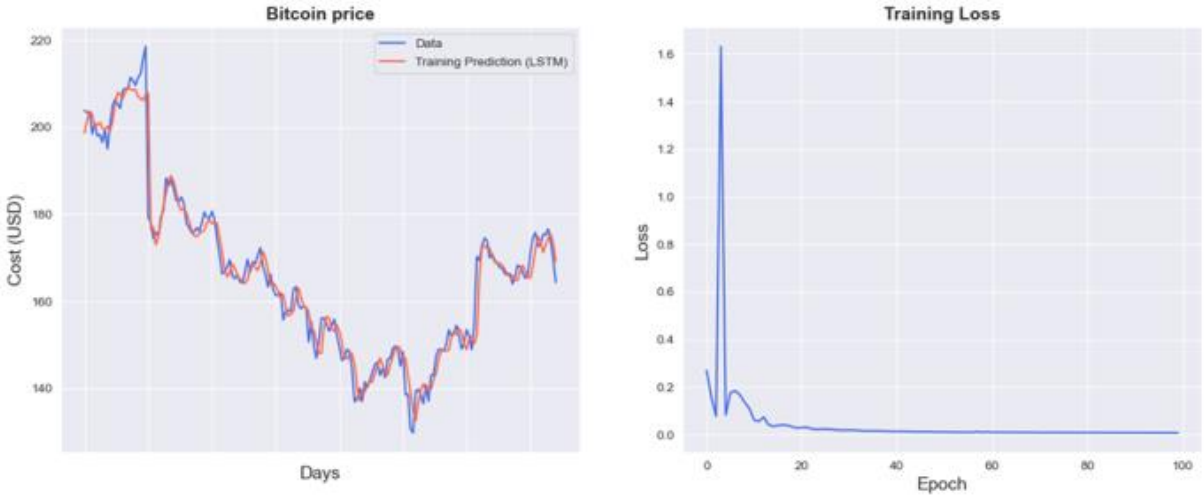

Figure 8. Neurons number:128, RMSE: 4.01

3.3. Influence of First Difference on Prediction Result

It is apparent that the LSTM model primarily relies on previous-day values when forecasting a specific date, resulting in forecast lines that closely mirror fluctuations in actual prices. This pattern could stem from the non-stationarity inherent in the time series data, underscoring the importance of addressing time series stationarity. Approaches to address this issue encompass logarithmic

transformation, smoothing, differencing, and decomposition techniques such as Empirical Mode Decomposition (EMD) and wavelet transform. The way used to perform first-order differencing on the data. So that the lack of stationarity of the data is improved [12]. By comparing the training results of Figures 9,10, it can be found that the first-order difference effectively solves the lag problem of the prediction results. At the end of the study, the author uses the test data to test the model. It is obvious that the fitting degree of prediction is higher in the time period with less price fluctuation. However, in the period of large price fluctuations, although the overall trend of the forecast results is correct and the lag is small, there is a big gap with the true value. Therefore, the LSTM model still needs to be improved.

Figure 9. Before first-order differencing.

Figure 10. After first-order differencing

4. Conclusion

This study aims to improve the precision of Bitcoin price forecasting through the implementation of the LSTM model. So as to play a reference role in Bitcoin trading, and also to explore the different efficiency of LSTM under different parameters and training methods. After the test of different parameters, the optimizer and the treatment of data stationarity have the most significant impact on the experimental results, in which the training efficiency of Adam optimizer is significantly higher

than Momentum, SGD and other optimizers, and the first-order difference effectively solves the lag of the test results. But changing the number of neurons in each cell did not optimize the training results. In the future, more models will be used to predict the price of Bitcoin, such as RNN, CNN and so on. In addition, some variants of LSTM will be tried in the future. More combinations of parameters will be tried until the most efficient parameter setting is found.

References

[1] H. Chen. Economic analysis of Bitcoin. Zhejiang University, 2015.

[2] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2024.

[3] R. G. Donaldson, M. Kamstra M. Neural network forecast combining with interaction effects. Journal of the Franklin Institute, 336(2), 1999, pp: 227-236.

[4] L. Takeuchi L, Y. Lee. Applying deep learning to enhance momentum trading strategies in stocks, Technical Report. Stanford, CA, USA: Stanford University, 2013.

[5] S. McNally, J. Roche, S. Caton S. Predicting the price of bitcoin using machine learning, 2018 26th euromicro international conference on parallel, distributed and network-based processing (PDP). IEEE, 2018 pp: 339-343.

[6] W. Bao, J. Yue, Y. Rao. A deep learning framework for financial time series using stacked autoencoders and long-short term memory, PloS one, 12(7), 2017 p: e0180944.

[7] Y. W. Wang, S. C. Ma. Time series prediction based on a hybrid model of ARIMA and LSTM, Computer Applications and Software, 38 (2), 2021, pp: 291-298

[8] Information on: https://www.kaggle.com/code/meetnagadia/

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

[10] D. P. Kingm, J. Ba. Adam: A method for stochastic optimization, arXiv preprint arXiv: 1412.6980, 2014.

[11] S. Ruder. An overview of gradient descent optimization algorithms, arXiv preprint arXiv: 1609.04747, 2016.

[12] H. G. Yu, D. Gu, H. Xu, A LSTM dam deformation safety monitoring and prediction model considering variable, Journal of Anhui University of Technology, 40 (1), 2023 pp: 89-96.