MATEMATIKA, MJIM, Volume 39, Number 2, 149–158

The Long Short-Term Memory (LSTM) Model Combines with Technical Analysis to Forecast Cryptocurrency Prices

Fu Dingyu and Mohd Tahir Ismail*
School of Mathematical Sciences, Universiti Sains Malaysia
11800 USM, Penang, Malaysia
*Corresponding author: m.tahir@usm.my

Article history
Received: 30 October 2022
Received in revised form: 15 June 2023
Accepted: 3 July 2023
Published on line: 1 August 2023

Abstract Cryptocurrency has a considerable market value and massive trading volume. Moreover, it is also known for its extreme volatility. Thus, this paper intends to attempt a new approach to forecast cryptocurrency prices by combining the long short-term memory (LSTM) model and technical analysis. The LSTM model has the advantages of a recurrent neural network and solves the gradient disappearance problem that adjusts weights and biases of long- or short-term memory, which is suitable for processing time series problems. Meanwhile, technical analysis is still a critical price trend analytical method. Overall, the results show that the combined methods get a better effect than only using a single price as a feature. Under the same condition, only using price as features for LSTM model accuracy rate is more than 40% for two different error tolerance, but the model accuracy rate will be improved by more than 60% and 90% if traditional technical indicators are combined as features at the best condition. Moreover, the error rate also reduces for the combined approach compared to the single approach.

Keywords LSTM; Forecasting; Technical analysis; Bitcoin; Modeling

Mathematics Subject Classification 68T01, 91G80.

1 Introduction

Blockchain technology is a significant financial innovation. Since Bitcoin's advent, cryptocurrency trading is already a massive market with rich derivatives. According to the Coin Market Cap website, the overall market cap of cryptocurrencies jumped from approximately $500 billion to $752 billion in the last month of 2020. Today this number changed to $1.27 trillion. Thus, cryptocurrency is highly volatile and has a high risk to investors. Therefore, there is a need to find a scientific way to analyze and forecast cryptocurrency market trading.

There are many methods used for price forecasting. The linear models used in conventional econometric forecastings, such as AR (Autoregression), ARMA (Autoregressive Moving Average), and ARIMA (Autoregressive Integrated Moving Average), are examples of classic econometric models, while nonlinear models such as ARCH (Autoregressive Conditional Heteroskedasticity Model) and GARCH (Generalized Autoregressive Conditional Heteroskedasticity). However, these methods have some flaws, such as when it comes to making long-term predictions, the ARIMA model is not as accurate as it is when making short-term forecasts. In addition, it cannot identify the underlying dynamics present in the many different time series [1-5]. Because these nonlinear models are accurate only if the data fits the chosen distribution model, they limit their applicability by forcing a preconceived distribution on the data [7,8]. The LSTM model, on the other hand, has substantially superior predictive power than the standard ARCH or GARCH model [6].

In recent years, with the acceleration of hardware update speed and the development of computer science, breakthroughs have been made in artificial intelligence; especially in 2012, Google's Cat event marked a new era in machine learning and deep learning. For instance, deep learning models, with their excellent ability of nonlinear fitting and feature automatic learning, have shown advantages in processing complex nonlinear problems in image recognition, speech recognition, and natural language processing. Moreover, the deep learning model does not need to limit the distribution or inherent attributes of the research data, which makes people see its application potential in the field of financial prediction. In addition, the deep learning model can accept matrix input. Many types of time series data, including price data (open, close, high, and low), trading volume, and technical analysis (including EMA at multiple time intervals), can be used as input to the model [9].

Technical analysis has a much longer history, dating back to the 18th century when Japanese traders recorded daily data on rice prices and plotted them to study their rise and fall patterns. This is the origin of the now widely used K-chart technique. According to technical analysis, the point of view of this theory, the price reflects all market information, and price trends and history are often repeated. So, the data of historical stock prices and trading volume are used to analyze and predict the market's future direction through graphs or indicators.

Compared with traditional econometric models, the rich feature quantity and the feature learning ability of deep learning models have greatly improved the prediction accuracy [10,11]. In contrast to traditional linear statistical models (ARMA), machine learning methods allow us to capture the nonlinear properties of highly volatile cryptocurrency prices. Artificial Neural Networks (ANN) has many excellent performances in dealing with large amounts of data and complex problems. However, ANN does not take into account the correlation of neighboring data in dealing with time series data, and the proposed RNN (Recurrent Neural Network) can solve the problem of data correlation well. Nevertheless, RNN has a gradient disappearance problem. So, the LSTM model can solve it. But a single LSTM has its limitations when faced with complex financial data. Therefore, this research attempts to build a quantitative analysis approach for trading cryptocurrencies based on technical analysis indicators and the LSTM model.

2 Data

This paper uses the Binance cryptocurrency exchange (www.binance.com) as the data source. Binance provides a convenient and fast application programming interface (API) service; every user can access this API. This paper will use the trading price of the Bitcoin/USD Contract, and we consider last year as the data range from 2021.06.01 to 2022.06.01 while the unit period is in days. Figure 1 shows the Bitcoin price trend.

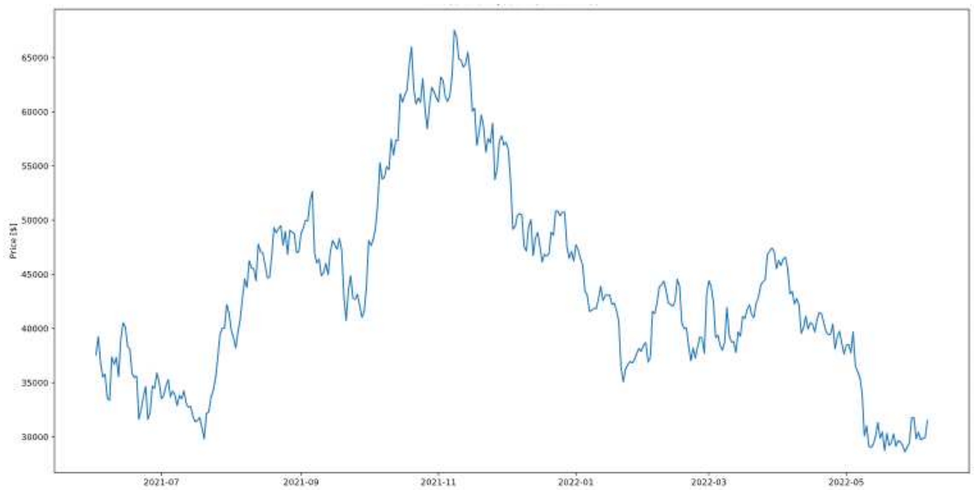

Figure 1: Bitcoin/USDT Contract Price Trend

3 Dataset Structure

We will structure a technical indicators matrix as features input for the LSTM model and combine the daily highest price, lowest price, closing price, and trading volume as a dataset. Table 1 shows the technical indicators and window size used.

Table 1: Technical Indicator's Structure

Name	Window Size (day)
SMA: Simple Moving Average (Fast)	5
SMA: Simple Moving Average (Middle)	10
SMA: Simple Moving Average (Slow)	20
MACD: Moving average convergence divergence (Fast)	12
MACD: Moving average convergence divergence (Slow)	26
RSI: Relative strength index (Fast)	6
RSI: Relative strength index (Middle)	12
RSI: Relative strength index (Slow)	24
KDJ: Stochastic oscillator	9
OBV: On Balance Volume	-
ADX: Average Directional Movement Index	5
ADXR: Smoothed Moving Average of ADX	10
BIAS: Deviation rate (Fast)	5
BIAS: Deviation rate (Middle)	10
BIAS: Deviation rate (Slow)	12

4 Long Short-Term Memory (LSTM) Model

The (long short-term memory) LSTM model can deal effectively with both the preservation of long-term information and the short-term skipping of inputs in latent variable models has long existed. Let's look at a few diagrams to explain the work process. In Figure 2, the LSTM processes the time series data sequentially from left to right. The hidden units, the batch size, and the number of inputs are $\boldsymbol{h}, \boldsymbol{n}, \boldsymbol{d}$, respectively. $\boldsymbol{H_{t-1}}$ is the hidden state at time $\boldsymbol{t-1}$ representing the previous time and $\boldsymbol{X_t}$ is the current time t as input. There are three kinds of gates, the input gate $\boldsymbol{I_t}$, forget gate $\boldsymbol{F_t}$ and the output gate is $\boldsymbol{O_t}$, The structure of these gates represents an operator mechanism, and they will change the state information of the model. The values of these gates are determined when $\boldsymbol{X_t}$, $\boldsymbol{H_{t-1}}$ enters the fully-connected layer with a sigmoid activation function.

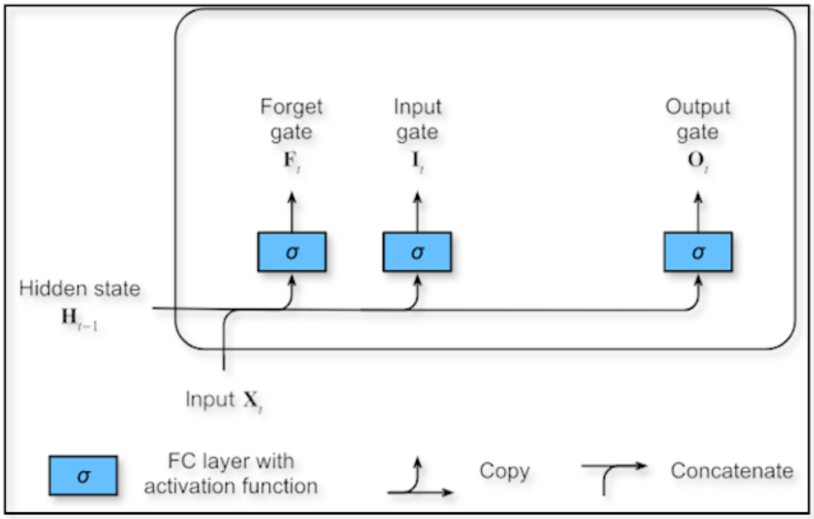

Figure 2: Gate Structure

$$\begin{aligned}
\mathbf{I}_t &= \sigma(\mathbf{X}_t\mathbf{W}_{xi} + \mathbf{H}_{t-1}\mathbf{W}_{hi} + \mathbf{b}_i),\\
\mathbf{F}_t &= \sigma(\mathbf{X}_t\mathbf{W}_{xf} + \mathbf{H}_{t-1}\mathbf{W}_{hf} + \mathbf{b}_f),\\
\mathbf{O}_t &= \sigma(\mathbf{X}_t\mathbf{W}_{xo} + \mathbf{H}_{t-1}\mathbf{W}_{ho} + \mathbf{b}_o).
\end{aligned} \tag{1}$$

Equation (1) is for calculating the input gate I_t, $\mathbf{W}_{xi} \in \mathbb{R}^{d\times h}$ and $\mathbf{W}_{hi} \in \mathbb{R}^{h\times h}$ are weight parameters, $\mathbf{b}_i \in \mathbb{R}^{1\times h}$ is bias parameter, the forget gate $\boldsymbol{F_t}$, $\mathbf{W}_{xf} \in \mathbb{R}^{d\times h}$ and $\mathbf{W}_{hf} \in \mathbb{R}^{h\times h}$ are weight parameters, $\mathbf{b}_f \in \mathbb{R}^{1\times h}$ is bias parameter, and the output gate $\boldsymbol{O_t}$ weight and bias is $\mathbf{W}_{xo}$, $\mathbf{W}_{ho} \in \mathbb{R}^{d\times h}$, $\mathbf{b}_o \in \mathbb{R}^{1\times h}$.

The cell state is an important concept in LSTM, and those gates introduced above are designed to change the state information of the cell. The candidate memory cell $\mathbf{C}^{\sim}{}_t \in \mathbb{R}^{n\times h}$, input data into the input gate I_t to compute the values that govern how much new data we consider through $\mathbf{C}^{\sim}{}_t$. The memory cell $\mathbf{C}_{t-1} \in \mathbb{R}^{n\times h}$, The forget gate, denoted by F_t determines the percentage of the previous memory cell's content, $\mathbf{C}_{t-1} \in \mathbb{R}^{n\times h}$, that is kept. (refer to Figure 3).

$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{X}_t\mathbf{W}_{xc} + \mathbf{H}_{t-1}\mathbf{W}_{hc} + \mathbf{b}_c), \tag{2}$$

$$\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \tilde{\mathbf{C}}_t, \tag{3}$$

where $\odot$ is a Hadamard product. In (2) $\mathbf{W}_{xc}$ and $\mathbf{W}_{hc}$, $\mathbf{b}_c$ denote the weight and bias.

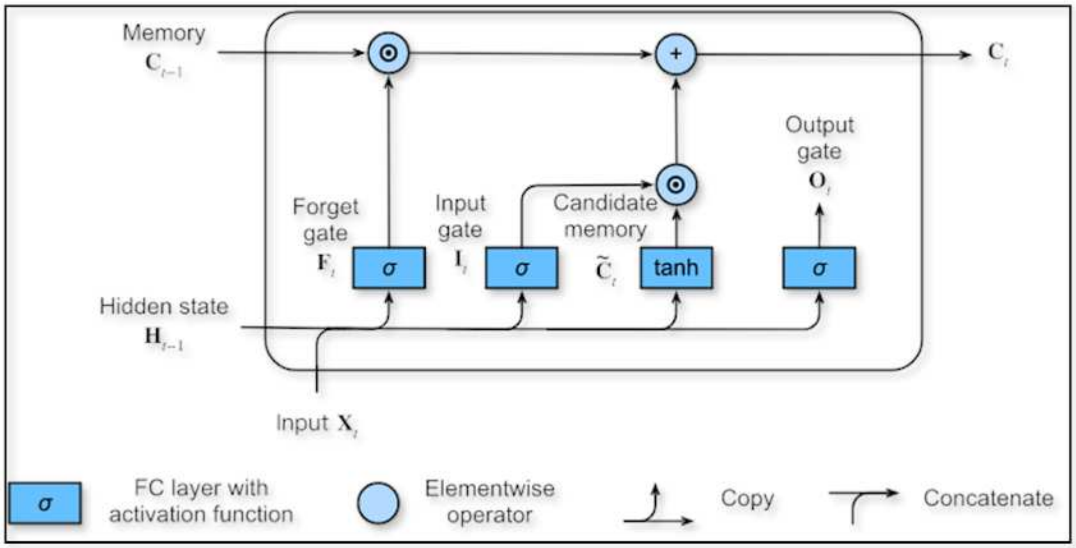

Figure 3: Memory Cell

If the forget gate is nearly 1 all the time and the input gate is approximately 0 all the time, then the previous memory cells $\mathbf{C}_{t-1}$ will be saved over time and passed on to the current time step. This architecture addresses the vanishing gradient issue and enhances the ability to identify long-range relationships inside sequences. Finally, as shown in Figure 4, for the purpose of computation, we must first define the hidden state $\mathbf{H}_t \in \mathbb{R}^{n\times h}$. Throughout this case, the output gate $\boldsymbol{O}_t$ is useful. For situations when the output gate is close to 1, all memory data is effectively passed on to the predictor; otherwise, all data is kept within the memory cell itself and no additional processing is done.

$$\boldsymbol{H}_t = \boldsymbol{O}_t \odot \tanh(\boldsymbol{C}_t) \tag{4}$$

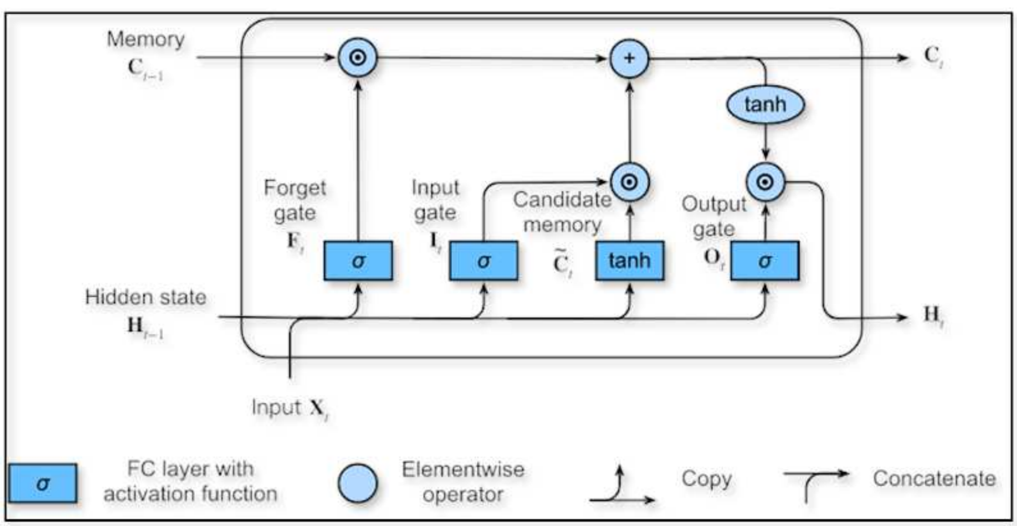

Figure 4: Hidden State

5 Experimental Settings

The first thing that we did was divide the dataset into three different groups: the training set (60 percent), the test set (20 percent), and the validation set (20 percent). The LSTM neural network forecasting model and its general structure should then be composited in the second stage, as shown in Figure 5. The dataset input layer is the initial layer in the structure. The second layer is an LSTM layer with 512 neurons with tanh activation function. The third layer is referred to as a Dropout layer, and its function is to prevent overfitting. The parameter for this layer is set to 0.2. The fourth layer is an LSTM layer, and studies have shown that a multi-layer LSTM model can considerably improve a model's ability to fit data and make accurate predictions. [12,13]. Hence, we adopted a dual LSTM layer structure with 512 neurons with tanh activation function. The fifth layer is a Dropout layer. Its purpose is the same as the third layer to prevent overfitting, where the parameter is 0.2. The sixth layer is a fully-connected layer with 256 neurons. While the last layer is also fully connected, it produces a one-dimensional output to predict prices. The optimizer adopted Adam [14,15] and the batch size is 4, epochs is 300.

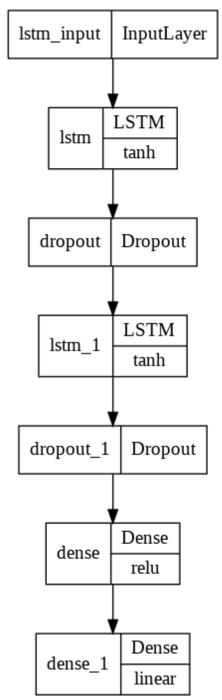

Figure 5: LSTM Neural Network Model Structure

The mean square error (MSE) is loss function calculation formula as shown in (5),

$$\text{MSE}=\frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2. \tag{5}$$

The root mean square error (RMSE) formula is shown in (6),

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2}. \tag{6}$$

The mean absolute error (MAE) formula is shown in (7).

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - p_i), \tag{7}$$

where y_i refers to the true value and p_i refers to the predicted value.

The accuracy rate with the error-tolerant formula is shown in (8),

$$\text{Accuracy rate} = \frac{\text{COUNT(ABS}\,(y - p) < T)}{\text{COUNT}(y)} * 100, \tag{8}$$

where y refers to the true value dataset, p refers to the predicted value dataset, and N is the total number of predictions. While T is an error-tolerant rate value, and in this experiment, we adopt 5% and 10% values.

We designed experiments with six different conditions for comparison

- Model 1: LSTM (include only the Bitcoin's daily highest price, lowest price, closing price, and trading volume).
- Model 2: LSTM + All Indicators (including Bitcoin's daily data and Table 1 indicators).
- Model 3: LSTM-MACD (Model 2 without the MACD indicator).
- Model 4: LSTM - MACD -ADX – ADXR (Model 3 without ADX and ADXR indicators).
- Model 5: LSTM - MACD -ADX -ADXR -KDJ (Model 4 without KDJ indicator).
- Model 6: LSTM – MACD – ADX – ADXR -KDJ – RSI (Model 5 without RSI indicator).

6 Results and Findings

As discussed in the previous section, six experiment models have been suggested to evaluate the performance of LSTM and technical indicators. Starting with Model 1, which includes only Bitcoin price, and Model 2, with Bitcoin price adding all the technical indicators. While Model 3 until Model 6 are the model where some technical indicators are excluded. Table 2 and Figure 6 present the results based on error measurements, while Table 3 and Figure 7 show the results based on accuracy measurements.

As portrayed in Table 2 and Figure 6, the model that includes the technical indicator (Model 2 – Model 6) has a lower error measurement than the model without the technical indicator (Model 1). Thus, adding the technical indicators improved the forecast performance. When removing some technical indicators, some models get further optimized. It happened to Model 3 (removing MACD) and Model 4 (removing MACD, ADX, and ADXR), where the error measurement values were reduced. However, when further reductions are made to Model 5 (removing MACD, ADX, ADXR, and KDJ) and Model 6 (removing MACD, ADX, ADXR, KDJ, and RSI), the error measurement values start to increase but are still lower compared to Model 1 (without all technical indicators). Therefore based on the error measurement results (MSE and RMSE), the best performance model is Model 4.

Table 2: Error Result Comparison

Model	MSE	MAE	RMSE
1	1.380	0.850	1.180
2	0.630	0.695	0.800
3	0.613	0.662	0.790
4	**0.580**	**0.590**	**0.760**
5	0.710	0.700	0.850
6	0.790	0.560	0.890

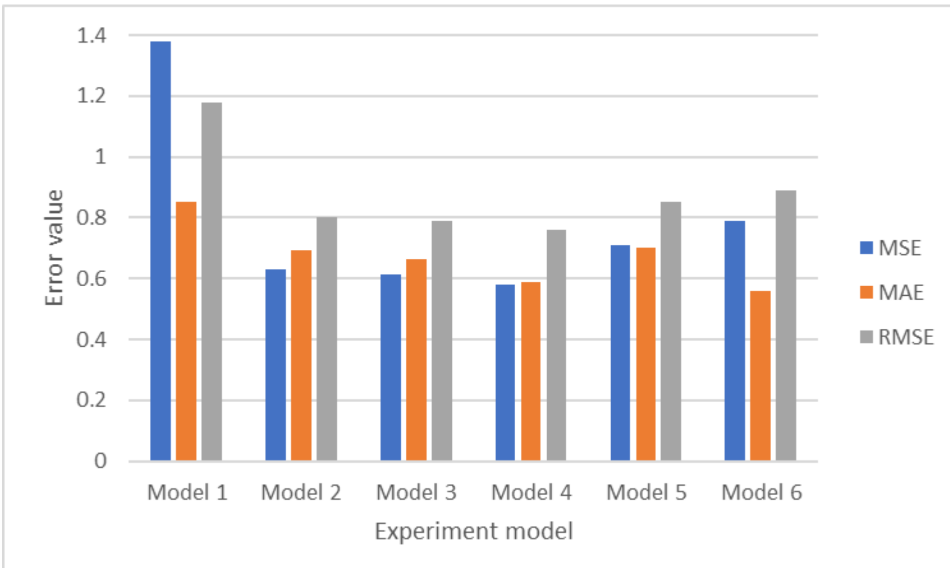

Figure 6: Error Result in Comparison between Models

Next, when examining Table 3 and Figure 7, in terms of accuracy at different error tolerance levels, the model with technical indicators (Model 2-Model 6) is more accurate compared to the model not including technical indicators (Model 1). It appears that Model 2 until Model 6 have more than 60% and 90% accuracy for 5% and 10% error tolerant. While for Model 1, only more than 40% accuracy for both error tolerant. Furthermore, looking closely between Model 2 until Model 6, it seems that omitting some technical indicators improves the accuracy of Model 3 (taking out MACD) and Model 4 (taking out MACD, ADX, and ADXR). Conversely, the improvement did not happen to Model 5 (taking out MACD, ADX, ADXR, and KDJ) and Model 6 (taking out MACD, ADX, ADXR, KDJ, and RSI) because it showed the accuracy dropped. As a result, regarding the accuracy measurement, Model 4 has the highest accuracy compared to other models.

7 Conclusion

The LSTM model offers features that handle both short-term input skipping and long-term information preservation. It can be used to solve the gradient disappearance problem effectively. This paper presents the study of LSTM combined with technical indicators to forecast the Bitcoin price. Overall

Table 3: Accuracy Rate with Error-tolerant Result Comparison

Model	Error Tolerant	Accuracy	Error Tolerant	Accuracy
1	5%	0.466	10%	0.421
2	5%	0.636	10%	0.925
3	5%	0.653	10%	0.941
4	**5%**	**0.676**	**10%**	**0.963**
5	5%	0.641	10%	0.938
6	5%	0.638	10%	0.921

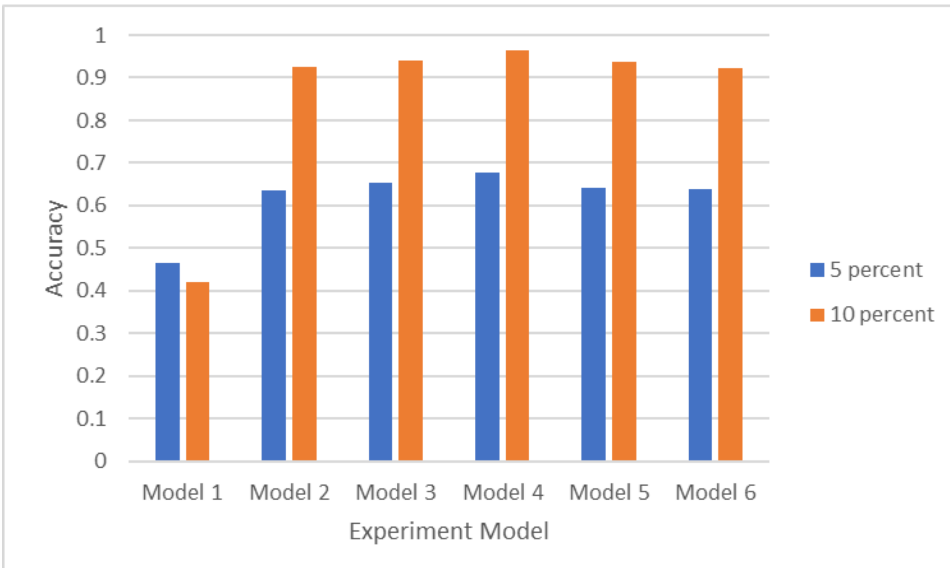

Figure 7: Accuracy Comparison between Models

the results show that the forecasting performance is better when the LSTM model is linked with the technical indicators where the error is reduced, but accuracy is increased. However, it is interesting that not all the technical indicators contributed to the forecasting performance. Results show that Model 4 is the best model for forecasting Bitcoin price, where this model only includes five technical indicators (SMA, RSI, KDJ, OBV, and BIAS). As there are many technical indicators in the literature, perhaps, which indicators can improve the model more effectively will be our further work.

References

[1] J. Hao, X. Li, Y. Lv and N. Xiao. A Comparative Study on SSE 50 Index Prediction Based on ARIMA Model and LSTM Neural Network. In *2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Dec 3*. IEEE. 2021. 216-219.

[2] Ariyo AA, Adewumi AO, Ayo CK. Stock price prediction using the ARIMA model. In *2014 UKSim-AMSS 16th international conference on computer modelling and simulation, Mar 26*. IEEE. 2014. 106-112.

[3] Yu, P., & Yan, X. Stock price prediction based on deep neural networks. *Neural Computing and Applications*. 2019. 32: 1609–1628.

[4] Mohd Tahir Ismail, Remal Shaher Al-Gounmeein. Overview of Long Memory for Economic and Financial Time Series Dataset and Related Time Series Models: A Review Study. *IAENG International Journal of Applied Mathematics*. 2022. 52(2): 1-9.

[5] Hiransha M, Gopalakrishnan EA, Menon VK, Soman KP. NSE stock market prediction using deep-learning models. *Procedia computer science*. 2018. 132: 1351-62.

[6] Liu R, Jiang Y, Lin J. Forecasting the Volatility of Specifc Risk for Stocks with LSTM. *Procedia Computer Science*. 2022. 202: 111-4.

[7] Davis N, Raina G, Jagannathan K. A framework for end-to-end deep learning-based anomaly detection in transportation networks. *Transportation research interdisciplinary perspectives*. 2020. 5: 100112.

[8] Wang J, Cui Q, Sun X, He M. Asian stock markets closing index forecast based on secondary decomposition, multi-factor analysis and attention-based LSTM model. *Engineering Applications of Artificial Intelligence*. 2022. 113: 104908.

[9] Ma C, Yan S. Deep learning in the Chinese stock market: the role of technical indicators. *Finance Research Letters*. 2022. 103025.

[10] Sheng Y, Fu K, Wang L. A PCA-LSTM Model for Stock Index Forecasting: A Case Study in Shanghai Composite Index. In *2022 7th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), Apr 22*. IEEE. 412-417.

[11] Tran DT, Magris M, Kanniainen J, Gabbouj M, Iosifidis A. Tensor representation in high-frequency financial data for price change prediction. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI), Nov 27*. 2017. IEEE. 1-7.

[12] Chang Z, Yuan W, Huang K. Remaining useful life prediction for rolling bearings using multi-layer grid search and LSTM. Computers and Electrical Engineering. 2022. 101:108083.

[13] Chi D. Research on electricity consumption forecasting model based on wavelet transform and multi-layer LSTM model. Energy Reports. 2022. 8: 220-8.

[14] Kingma DP, Ba J. Adam: A method for stochastic optimization. 2014. arXiv preprint arXiv:1412.6980.