Journal of Computers Vol. 35 No. 3, June 2024, pp. 27-38
doi: 10.53106/199115992024063503003

Applying LSTM Model to Predict the Japanese Stock Market with Multivariate Data

Cheng Li[1*], Yu Song[2]

[1] Graduate school of engineering, Fukuoka Institute of Technology,
Fukuoka 8110295, Japan
1chcheng1@163.com

[2] Department of System Management, Fukuoka Institute of Technology,
Fukuoka 8110295, Japan
song@fit.ac.jp

Received 7 October 2022; Revised 25 October 2022; Accepted 26 October 2022

Abstract. Using machine learning methods to analyze and predict time series data is a hotspot issue. Because of its potential profitability, it has attracted a lot of research and investment, particularly in the financial field. Compared with other machine learning prediction models, long short-term memory (LSTM) is very effective for processing time series data, due to its special network structure. In this study, we use three models to predict the Japanese stock market movements. These models can be used to learn and predict multivariate data by adjusting the structure and hyperparameters. The original dataset is made up of NIKKEI 225 and some individual stocks. Subsequently, several well-known technical indicators are calculated and added as a new dataset. Two efforts were also made to improve the quality of the dataset. Multiple sets of numerical experiments are established to examine the impact of increasing the number of features on these models and the impact of lengthening the training data on these models. The results show that lengthening the length of training intervals and increasing the number of features can improve the model performance effectively. The LSTM model has better performance than the encoder-decoder LSTM model and CNN-LSTM model in stock market prediction.

Keywords: machine learning, stock market prediction, LSTM model, technical indicators

1 Introduction

In real life, collecting data for a long or short period is a common activity. Time series data are abundant in research fields such as agriculture, medicine, the military, and finance. It is of great theoretical significance and practical application value to analyze these data to forecast long-term trends or to perform other forms of analysis. Traditional methods for stock market prediction include linear regression, exponential smoothing, autoregressive integrated moving average model (ARIMA model) and others [1]. Recently, machine learning is being used to analyze and forecast time series data, which is a hot research topic right now, particularly in the financial field. Investors and researchers are interested in using machine learning to forecast the stock market. However, some theories and hypotheses claim that the stock market cannot be forecasted because it is dynamic, nonlinear, and influenced by various factors. The random walk theory demonstrated that the changes in stock prices are analogous to the Brownian motion of a molecule [2]. According to the efficient market hypothesis, current prices have accurately, completely, and timely reflected all the information about the intrinsic value of assets [3].

Although some scholars and researchers believe that forecasting stock prices are pointless because stock price fluctuations are random and irregular, many researchers and scholars believe that stock market movements can be forecasted to some extent. People can easily obtain a large amount of information about the stock market thanks to the advancement of computers and the Internet, so the technology to extract and analyze the information to support making investment strategies have become difficult work. With advances in neural networks, it has been demonstrated that certain network models can be used to extract the most valuable information from time series data, including financial data. These characteristics encourage scholars and researchers to investigate whether

* Corresponding Author

neural networks can forecast stock market movement or not. Machine learning methods such as support vector machine (SVM) [4], artificial neural networks [5], and other deep learning methods can also be used for predicting the stock market in the current hot artificial intelligence. There is currently agreement that the recurrent neural network (RNN) is one of the most effective methods for efficiently processing time series data. Moreover, the long short-term memory (LSTM), one RNN variant, not only surmounts the defects of RNN but also improves prediction accuracy.

Many researchers have developed different forecasting methods using machine learning for financial time series forecasting. Experiments are conducted to analyze the forecasting performance, effectiveness and efficiency of the models. Sidra et al. [6] proposed a long short-term memory (LSTM) network to build four deep learning-based regression models to predict the future NIFTY 50 opening price. By optimizing the hyperparameters of the model and adjusting the network structure, they enhanced the predictive ability of the LSTM model to handle data with different structures. The results clearly show that the LSTM-based univariate model is the most accurate model for predicting the next week's opening value of the NIFTY 50 time series using one-week prior data as input. Bacanin et al. [7] conducted a study using the Borsa Istanbul 100 index as an example and proposed a new machine learning approach to predict the stock market index movements. The RMSE, MAPE and correlation coefficient were used as evaluation measure. The study suggested that a hybrid algorithm of a multilayer perceptron and a modified whale optimization algorithm is effective in improving the accuracy of the model. Mahmoodzadeh et al. [8] developed an LSTM model that improved by Gray Wolf Optimization (GWO). The effect of the length of the input time series on the model performance and LSTM model with multiple parameters were investigated using several datasets. Qiu et al. [9] used several meta heuristic algorithms to optimize the parameters of the artificial neural network model and predict the direction of stock market index movement. Jigar et al. [10] applied four prediction models, ANN, SVM, random forest and naive-Bayes with two approaches to two stocks and two stock price indices. The experimental results suggest that the performance of the prediction models can be improved when these technical parameters are represented as trend deterministic data.

Although scholars have applied various machine learning methods to stock market forecasting, no one has explored the impact of single or multiple feature inputs on the prediction accuracy of model. The overarching goal of this study is to develop LSTM model that can process input data with single or multiple features and predict short-term stock prices. Furthermore, encoder-decoder LSTM and CNN-LSTM are established for comparison experiments based on LSTM model.

It is common in the machine learning process for the model to be unable to be effectively trained due to a lack of training data. Hence, we made two efforts to improve the dataset quality to improve the model prediction accuracy. The original dataset consisted of trading data from the Nikkei 225 index and some individual stocks obtained from Yahoo finance. We calculate and analyze several well-known technical indicators based on historical stock data before developing LSTM model, encoder-decoder LSTM model and CNN-LSTM model to forecast stock price trends in the future. We combine historical trading data and technical indicators as feature engineering, then adjust network structure and hyperparameters to achieve these models from single-feature prediction to multi-feature prediction.

The remaining parts of this paper is divided into four sections. Section 2 surveys why LSTM can extract useful information from verbose data better than RNN. Section 3 describes the experiments and their outcomes. This section also includes information on data processing and feature engineering. Section 4 discusses the model's result. Finally, section 5 summarizes the contributions.

2 Methodology

This section first contrasts RNN and LSTM by using the external input of a single neuron as an example. Then, we briefly discuss how LSTM overcomes RNN's shortcomings. Subsequently, the internal structure, variables, and equations of LSTM neurons are described detailedly.

RNN is one of the neural networks that can retain information and pass it on to the next layer [11, 12]. RNN, unlike ordinary neural networks, is very effective for time series data. This type of information reflects the state or degree of change in a particular thing or phenomenon over time. The information extraction of time series data is realized through the sharing of parameters at different times due to the neuron's memory. Consider, as an example, a single-RNN neuron structure (Fig. 1). x_t represents the input information at the current t moment; y_t represents the output information of the RNN neuron at the current t moment; h_{t-1} represents the state information stored by the neuron at the previous moment; and h_t represents the state information stored by the neuron at

the current t moment. We can see from equation (1) that the state information of neuron h_t is determined by both x_t and h_{t-1}. That is how the RNN remembers the previous information and applies it to the current output calculation.

$$h_t = \tanh\left(w_h \cdot \left[h_{t-1}, x_t\right]\right). \tag{1}$$

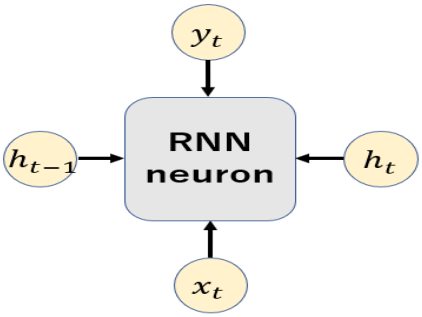

Fig. 1. RNN neuron

However, this ability will be diminished when the outputs require more information from the past. Bengio, et al. [13] investigated this issue in-depth and discovered some fundamental reasons why RNN training is difficult. The RNN will include many accumulative multiplications of activation functions during the iteration. If the tanh or sigmoid function is used as the activation function, many decimals will be multiplied. With the increasing iteration numbers and time series, the accumulative multiplications of decimals cause the gradient to become smaller and smaller until it is close to 0, which is the phenomenon of the vanishing gradient; if the rectified linear unit (ReLU) is used as the activation function, the left derivative of the ReLU function is 0 and the right derivative is always 1, which avoids the occurrence of vanishing. However, the derivative of constant 1 can easily result in an exploding gradient. By improving RNN, LSTM [14, 15] is introduced to eliminate the long-term dependency defects of the RNN.

A cell state C_t was added to LSTM neurons to overcome the vanishing gradient problem compared to input variables outside the RNN neuron. Fig. 2 shows the internal structure of one LSTM neuron to better illustrate how information is transmitted through the LSTM neuron.

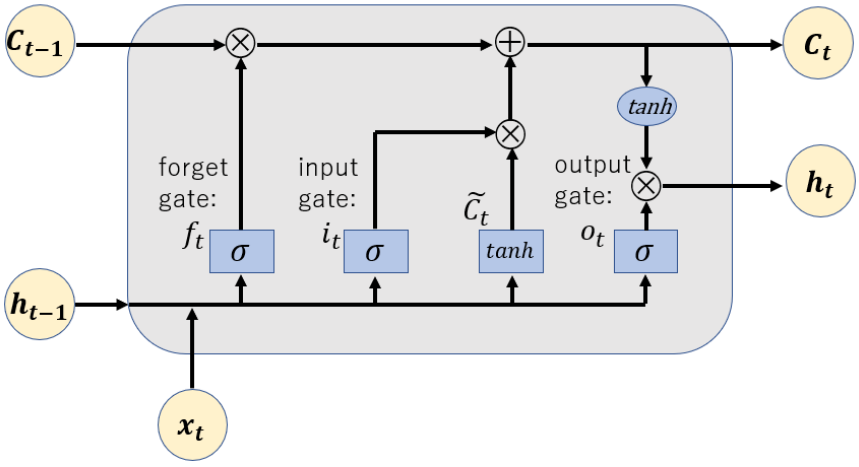

Fig. 2. The internal structure of LSTM neuron

The following section goes into great detail about Fig. 2. Three gates with sigmoid activation functions and one hidden layer with tanh activation functions replace the normally hidden layer in the RNN. The definitions and process steps are shown below. The functions are illustrated as follows.

x_t : input value (vector)

h_t : output value (vector)

C_t : neuron state

$\widetilde{C_t}$: the neuron state being updated

σ : sigmoid function

tanh: hyperbolic tangent function

f_t : a vector that determines how much information discard or remains in forget gate

i_t : a vector that determines what information updates in the input gate

o_t : a vector that determines what information output

Forget gate: The first step in an LSTM neuron is to determine what information must be discarded from the neuron state. This section of the operation is handled by a sigmoid unit known as the forget gate. According to the input information of h_{t-1} and x_t, it outputs a vector between 0 and 1 and the 0~1 value in the vector indicates how much information in the neuron state C_{t-1} is retained or discarded. A value of 0 indicates that no information is reserved, while a value of 1 indicates that all information is reserved. *W* and *b* are weight vector matrices and gate biases, respectively, in the equation (2).

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right]\right) + b_f \ . \tag{2}$$

Input gate and tanh layer: The following step is to decide what new information should be added to the neuron state. This section of the procedure is divided into two steps. First, h_{t-1} and x_t are used to determine which information is updated through a sigmoid unit known as the input gate. Then, through the tanh layer, use h_{t-1} and x_t to generate a new vector that can be updated to the neuron state. The equations (3), (4), (5) are shown below:

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right]\right) + b_i \ . \tag{3}$$

$$\widetilde{C_t} = \tanh\left(W_c \cdot \left[h_{t-1}, x_t\right]\right) + b_c \ . \tag{4}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \ . \tag{5}$$

Output gate: After updating the neuron state, determine what information is output based on h_{t-1}, C_{t-1} and x_t. The input information is routed through a sigmoid unit known as the output gate. Meanwhile, the updating neuron state C_t is processed by the tanh unit to produce a vector between −1 and 1. The vector obtained from the output gate is multiplied by this vector. Finally, the LSTM neuron's output h_t is obtained by equations (6) and (7).

$$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right]\right) + b_o \ . \tag{6}$$

$$h_t = o_t * \tanh\left(C_t\right). \tag{7}$$

With the above three gates, the LSTM can process time series data by discarding or retaining input information.

The long-distance information gradient does not disappear during iteration using the backpropagation algorithm, and the gradient can be well transmitted in the LSTM, greatly reducing the likelihood of a vanishing

gradient. Inspired by the LSTM neural network's superiority, we established an improved LSTM model for forecasting short-term stock prices. The two variants of LSTM, encoder-decoder LSTM and CNN-LSTM are set as comparison experiments. We conduct some numerical experiments to verify and evaluate the validity of these LSTM models by applying these models to the Nikkei 225 and several individual stocks for short-term forecasts. Additionally, we supplement the original dataset with technical indicators. However, training data from both the original and augmented datasets are expanded. In these numerical experiments, we want to see if these two factors can improve the model's prediction accuracy while also improving data quality.

3 Experiments and Results

3.1 Preliminary Work

Yahoo finance was used in this study to collect publicly available market information on stocks. The dataset consists of the Nikkei 225 and several individual stocks, including open, high, low, close prices and trade volume. Data cleaning and data standardization are two major steps in data preprocessing before entering data into the model [16]. Data cleaning is primarily used to fill in missing values and remove outliers from a dataset. Data standardization is the scaling of data values into a specific interval (0~1). Due to the different nature of each evaluation index in the multi-index evaluation system, it usually has different orders of magnitude and units. When the values of various indicators vary widely, if the original data are directly used for analysis, the indicators with higher values will play a larger role in the analysis, while the effects of indicators with lower values will be weakened. Therefore, standardizing the original data must ensure the reliability of the results.

After preprocessing the stock data, we compute some technical indicators that are commonly used in the stock market as feature engineering. Scholars generally agree on one point: data and features determine the upper limit of machine learning, and models and algorithms only approximate this upper limit. Therefore, feature engineering is critical for improving data quality in data preprocessing. Feature engineering is the process of extracting features from original data that can be used to better represent the actual problems dealt with by the prediction model. Good features can simplify the model while also improving its robustness and generalization. They can also improve the model's convergence speed and prediction accuracy at the same time. In this study, the 10-day moving average (MA10), 10-day relative strength index (RSI10), on balance volume (OBV), and price movement direction are calculated as features.

The MA10 is the average closing price of a stock in the market over the previous 10 days, and it is significant because it reflects the stock's 10-day average price. It is a critical indicator line for determining stock trends [17]. The stock trading operation reflects the aggregate results of various factors. Ultimately, price changes are determined by the relationship between supply and demand. The RSI indicator calculates the percentage of the total increase in stock prices to the average of the total change in stock prices over a given period using the supply and demand balance principle. The RSI value ranges from 0 to 100. The stronger the uptrend is, the higher the RSI value, and vice versa. The trend of changes in trading volume is used to infer the trend of stock prices. The direction of price movement is calculated by subtracting today's close value from the close value one day ahead. If the difference value is positive, we label price movement with 1. Conversely, if the difference value is negative, we label price movement with −1.

The prediction results of NIKKEI 225 and three individual stocks will be presented in tabular form in this section. We use root mean square error (RMSE) to estimate how well the model predicted for the model evaluation. It is the most popular measure for evaluating the accuracy of the model. RMSE is defined mathematically as follows:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}\left(X_i - x_i\right)^2}{n}}\,. \tag{8}$$

3.2 Model

We conduct some numerical experiments to verify and evaluate the validity of these LSTM models by applying these models to the Nikkei 225 and several individual stocks for short-term forecast. First, we build a basic LSTM model on the Nikkei 225 and several individual stocks for short-term forecast. The structure diagram of the LSTM model is illustrated in Fig. 4. These results are used as a benchmark for comparison. Among the time series data processing, the time series prediction problem with series as input and series as output is a more challenging prediction problem. The encoder-decoder LSTM has good performance in dealing with this type of problem, so one encoder layer and one decoder layer are added to the basic LSTM model to test the performance of encoder-decoder LSTM in predicting short-term stock market movements. We then combined the CNN with the basic LSTM model to test whether the CNN could improve the prediction results of the model.

Fig. 3 is the loss function of LSTM model during parameter tuning. This LSTM model is trained with Nikkei 225 and ALL dataset (see the details in second paragraph of sec. 3.3). As shown in the figure, after epoch=100, there is no longer a significant decrease in the loss function, so the epoch is set as 100 for the efficiency of model training and preventing overfitting.

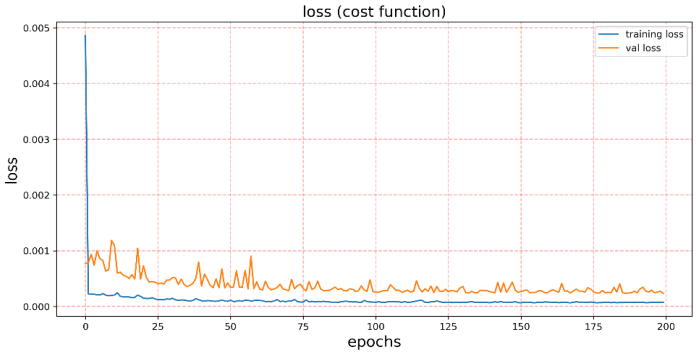

Fig. 3. The loss function of NIKKEI 225

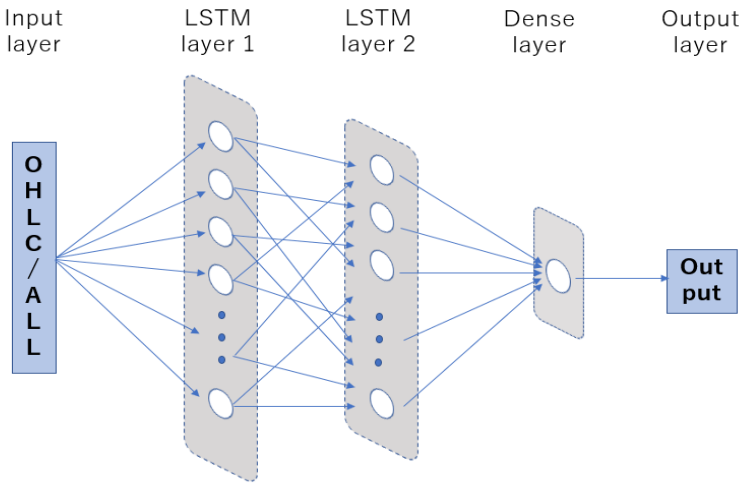

Fig. 4. The structure diagram of the LSTM model

Table 1. Parameters of the basic part of models

Timestamp	20
Epochs	100
Batch size	16
Unit of the first LSTM layer	64
Unit of the second LSTM layer	32
Dense layer	1
Optimizer	adam

3.3 Experiments

The dataset is approximately 2,800 trading days long and was collected from January 2011 to June 2022. As an example, consider the NIKKEI 225 historical data chart (Fig. 5). Since the LSTM network can extract useful information from verbose data without being constrained by long-term dependency defects, the dataset is divided into four parts to train this LSTM independently to forecast the short-term stock market. The training period is set at 600, 400, and 200 days respectively. The test period is a fixed 30-day period that follows each training period. These numerical experiments are used to investigate the effect of different length train periods on the performance of these LSTM models.

Fig. 5. The historical data of NIKKEI 225

Furthermore, some contrast experiments are conducted to verify if these technical indicators are effective in improving the prediction accuracy of models. A general process of our method is described as follows. First, we apply the LSTM model to the dataset that consisted of closing price, OHLC or ALL to predict stock market index and individual stocks. Second, we implement encoder-decoder LSTM model and CNN-LSTM model to the same datasets as comparison groups for LSTM respectively. Third, it is necessary to ensure the reliability of the results and the adequacy of models. As mentioned above, the RMSE is used as performance measure.

The detailed experimental procedure is as follows. We begin to use the dataset with only one dimension feature "close price" to train three models to verify the performance of these models in processing one dimension input data. These results are used as a benchmark for comparison. Subsequently, we use the dataset which includes open, high, low, and close prices (OHLC for short) as the input data to explore the performance and predictive ability of the models in processing multiple dimension input data. Finally, the OHLC dataset and technical indicators are combined into a new dataset called "ALL" for training and predicting. Based on these results, we use ALL dataset to explore whether the selected technical indicators can improve the prediction performance of the models.

In order to compare the performance of the models, we use the same dataset to train the LSTM model and its two variants (encoder-decoder LSTM and CNN-LSTM) separately. Except for the special layers of the model itself (encoding layer, decoding layer, convolution layer), the rest of the parameter settings are the same. The epoch is 100. The batch size is 16. The mean squared error is chosen as loss function. The optimizer is adam [18]. As shown in the Table 1. The input data and prediction interval are also set to be the same at the same period. These three groups of experimental results are summarized in section 3.4.

3.4 Results

The experiment results of Nikkei 225 are illustrated in Table 2 to Table 5. The second column shows the length of the various training periods. The last three columns show the results of different LSTM models respectively. As previously stated, the test period is a fixed 30-day period that follows each training period. The model's prediction results using closing price as input data are shown in the row of "RMSE of closing price" in Table 2. The model's prediction results using OHLC as input data and the model's prediction results when ALL is used as input data are shown in the rows of "RMSE of OHLC" and "RMSE of ALL" respectively.

Table 2. Results of first prediction interval (NIKKEI 225)

	Length of training interval (days)	LSTM	Encoder-Decoder LSTM	LSTM-CNN
RMSE of closing price	600	0.0125	0.0351	0.0618
	400	0.0172	0.0368	0.0649
	200	0.0233	0.0587	0.0541
RMSE of OHLC	600	0.0198	0.0347	0.0698
	400	0.0200	0.0407	0.0758
	200	0.0246	0.0561	0.0554
RMSE of ALL	600	0.0136	0.0224	0.0677
	400	0.0132	0.0429	0.0682
	200	0.0183	0.0346	0.0615

Table 3. Results of second prediction interval (NIKKEI 225)

	Length of training interval (days)	LSTM	Encoder-Decoder LSTM	LSTM-CNN
RMSE of closing price	600	0.0107	0.0346	0.0678
	400	0.0147	0.0364	0.0741
	200	0.0178	0.0413	0.0916
RMSE of OHLC	600	0.0089	0.0363	0.0635
	400	0.0103	0.0388	0.0654
	200	0.0159	0.0391	0.0769
RMSE of ALL	600	0.0138	0.0247	0.0406
	400	0.0150	0.0302	0.0456
	200	0.0172	0.0614	0.0496

Table 4. Results of third prediction interval (NIKKEI 225)

	Length of training interval (days)	LSTM	Encoder-Decoder LSTM	LSTM-CNN
RMSE of closing price	600	0.0138	0.0218	0.0283
	400	0.0158	0.0225	0.0366
	200	0.0185	0.0240	0.0495
RMSE of OHLC	600	0.0145	0.0227	0.0287
	400	0.0168	0.0213	0.0333
	200	0.0191	0.0220	0.0425
RMSE of ALL	600	0.0111	0.0196	0.0201
	400	0.0125	0.0183	0.0277
	200	0.0169	0.0193	0.0227

Table 5. Results of fourth prediction interval (NIKKEI 225)

	Length of training interval (days)	LSTM	Encoder-Decoder LSTM	LSTM-CNN
RMSE of closing price	600	0.0131	0.0162	0.0284
	400	0.0100	0.0191	0.0395
	200	0.0114	0.0209	0.0563
RMSE of OHLC	600	0.0106	0.0271	0.0655
	400	0.0116	0.0451	0.0704
	200	0.0473	0.0288	0.1134
RMSE of ALL	600	0.0083	0.0167	0.0224
	400	0.0098	0.0183	0.0262
	200	0.0100	0.0206	0.0432

Due to limited space, we summarize the prediction results of the experiments of Itochu, Toyota and Sony respectively and present the RMSE values in the form of intervals for the experimental results, as shown in Table 6 to Table 8. The first column is the length of training interval. The second column is the training model. The prediction results of different features are shown in last three columns.

Table 6. Prediction results of Itochu

Length of training interval (days)	Model	Features		
		CLOSE	OHLC	ALL
600	LSTM	0.0054~0.0086	0.0060~0.0082	0.0048~0.0069
	Encoder-Decoder LSTM	0.0079~0.0164	0.0089~0.0168	0.0090~0.0164
	CNN-LSTM	0.0190~0.0334	0.0163~0.0317	0.0153~0.0259
400	LSTM	0.0057~0.0089	0.0050~0.0097	0.0062~0.0072
	Encoder-Decoder LSTM	0.0067~0.0157	0.0071~0.0164	0.0086~0.0166
	CNN-LSTM	0.0189~0.0328	0.0162~0.0308	0.0168~0.0273
200	LSTM	0.0056~0.0120	0.0057~0.0110	0.0064~0.0097
	Encoder-Decoder LSTM	0.0085~0.0211	0.0098~0.0178	0.0091~0.0174
	CNN-LSTM	0.0144~0.0327	0.0151~0.033	0.0178~0.0289

Table 7. Prediction results of Toyota

Length of training interval (days)	Model	Features		
		CLOSE	OHLC	ALL
600	LSTM	0.0117~0.0943	0.0104~0.0242	0.0084~0.0153
	Encoder-Decoder LSTM	0.0095~0.1247	0.0141~0.0467	0.0133~0.0506
	CNN-LSTM	0.0486~0.0608	0.0217~0.0859	0.0190~0.0982
400	LSTM	0.0144~0.1030	0.0114~0.0241	0.0110~0.0192
	Encoder-Decoder LSTM	0.0169~0.1301	0.0119~0.0751	0.0148~0.0525
	CNN-LSTM	0.0404~0.0919	0.0267~0.1818	0.1458~0.0527
200	LSTM	0.0111~0.1155	0.0107~0.0167	0.0078~0.0135
	Encoder-Decoder LSTM	0.0208~0.1817	0.0090~0.0583	0.0133~0.0684
	CNN-LSTM	0.0521~0.1101	0.0373~0.1158	0.0337~0.1658

Table 8. Prediction results of Sony

Length of training interval (days)	Model	Features		
		CLOSE	OHLC	ALL
600	LSTM	0.0030~0.0090	0.0033~0.0096	0.0028~0.0086
	Encoder-Decoder LSTM	0.0043~0.0148	0.0038~0.0187	0.0034~0.0122
	CNN-LSTM	0.0076~0.0366	0.0075~0.0304	0.0064~0.0294
400	LSTM	0.0027~0.0076	0.0036~0.0091	0.0025~0.0081
	Encoder-Decoder LSTM	0.0042~0.0137	0.0037~0.0148	0.0039~0.0138
	CNN-LSTM	0.0080~0.0481	0.0102~0.0401	0.0073~0.0344
200	LSTM	0.0046~0.0117	0.0040~0.0122	0.0038~0.0098
	Encoder-Decoder LSTM	0.0055~0.0237	0.0070~0.0245	0.0051~0.0184
	CNN-LSTM	0.0059~0.0484	0.0080~0.0320	0.0062~0.0394

4 Discussion

These datasets are divided into four training intervals and prediction intervals. The four intervals prediction results of NIKKEI 225 are shown in Table 2 to Table 5 in order. The results demonstrate the effect of different length train periods on the performance of different input from the top down. Comparing the rows of “RMSE of closing price”, “RMSE of OHLC” and “RMSE of ALL” in the four tables, we can see that these additional features are effective at improving the prediction accuracy of these models. At the last three columns of the tables, we can explicitly compare the performance of LSTM, encoder-decoder LSTM and CNN-LSTM models from left to right.

As revealed in the results, the RMSE of the model trained with ALL is slightly better than that of the model trained with OHLC in most cases. This suggests that adding MA10, RSI10, OBV, and direction as features to the threes LSTM models will improve them. Although the encoder-decoder LSTM and CNN-LSTM occasionally perform better than the LSTM, most of the time, the LSTM is the best performing one. The prediction results of fourth prediction interval have the best performance in NIKKEI 225 dataset. Table 5 displays that the best performance is 0.0083. Furthermore, the three LSTM models have significantly improved performance while increasing the training period.

Since RMSE is sensitive to outliers, the prediction error will be amplified. Hence, we conducted multiple rounds of experiments to reduce the influence and took the average of five experiment results as the final result. In Table 6 to Table 8, although the prediction results are displayed simply, the findings also hold true. These results can be extended to the prediction of individual stocks, as the experiment results show that the three LSTM

models also have good prediction performance for the three individual stocks. Regarding the prediction of individual stocks, the best prediction results for Itochu, Toyota and Sony are 0.0048, 0.0078 and 0.0025 respectively. These prediction results are all derived from the LSTM model trained on the ALL dataset.

Lai and Chen used an LSTM model to analyze some Taiwan stocks [19] and the best results is 0.013. Li and Shen apply an attention-based multi-input LSTM to the Chinese stock exchange [20]. Their model's prediction result is 0.996. To forecast the S&P500 and the Korea Composite Stock Price Index 200, Baek and Kim developed an overfitting prevention LSTM [21]. The model's best performance is 0.6928. The prediction accuracy of the LSTM model in this study is superior to other studies.

Combining all the results, the LSTM model outperforms the encoder-decoder LSTM and CNN-LSTM models. This may be due to the fact that the encoder-decoder framework is more suitable for machine translation of Seq2Seq, while the output variables of all models in this paper are one-step prediction, which does not involve the multi-step semantic sequencing problem and does not maximize the role of decoder layer and encoder layer. The basic idea of CNN-LSTM is to use convolutional neural network to do feature extraction of images and LSTM is used to generate the image description, while the model prediction in this paper can be regarded as the prediction of regression problem. Therefore, compared with encoder-decoder LSTM and CNN-LSTM models, the prediction results of this LSTM model are stable and perform well when compared to NIKKEI 225 and other individuals. However, in terms of overall results, even with the same model and dataset, the RMSE will vary to some extent when the model predicts different periods. We believe this is due to the stock market's volatility.

5 Conclusion

LSTM, encoder-decoder LSTM and CNN-LSTM models for predicting short-term stock prices were proposed in this study. The experiment results lead to the following conclusions.

First, as the length of the training data increased, the RMSE decreased significantly. We can conclude that, while the information density of the dataset used in this study is low, it can be compensated by increasing the number of sample data.

Second, as the number of data features increased, the RMSE decreased. It can be concluded that in addition to increasing the amount of data, increasing the number of features (data quality) will also have an impact on the training results of these models.

Third, LSTM model performs better than encoder-decoder LSTM model and CNN-LSTM model in most cases. In addition, one of the characteristics of the stock market is volatility, and RMSE will vary greatly with different training periods. The experimental results show that while RMSE does not decrease linearly as the number of features increases, it performs well in both Nikkei 225 and individual stocks. The MA10, RSI10, OBV and the direction of stock movement demonstrated that these indicators can effectively improve the performance of these models.

Finally, when compared with other studies, the results in this study are superior.

References

[1] J.-H. Wang, J.-Y. Leu, Stock market trend prediction using ARIMA-based neural networks, in: Proc. International Conference on Neural Networks, 1996.
[2] J.C. Horne, G.G. Parker, The random-walk theory: An empirical test, Financial Analysts Journal 23(6)(1967) 87-92.
[3] E.F. Fama, Efficient market hypothesis, PhD Thesis, Chicago: University of Chicago Press, 1960.
[4] S. Mukherjee, E. Osuna, F. Girosi, Nonlinear prediction of chaotic time series using support vector machines, in: Proc. the 1997 IEEE Signal Processing Society Workshop, 1997.
[5] E. Turban, R. Trippi, Neural networks finance and investment: using artificial intelligence to improve real-world performance, 1995.
[6] S. Mehtab, J. Sen, A. Dutta, Stock price prediction using machine learning and LSTM-based deep learning models, in: Proc. Symposium on Machine Learning and Metaheuristics Algorithms, and Applications, 2020.
[7] N. Bacanin, M. Zivkovic, L. Jovanovic, M. Ivanovic, T.A. Rashid, Training a Multilayer Perception for Modeling Stock Price Index Predictions Using Modified Whale Optimization Algorithm, in: Proc. Computational Vision and Bio-Inspired Computing, 2022.
[8] A. Mahmoodzadeh, H.R. Nejati, M. Mohammadi, H.H. Ibrahim, S. Rashidi, T.A. Rashid, Forecasting tunnel boring machine penetration rate using LSTM deep neural network optimized by grey wolf optimization algorithm, Expert

Systems with Applications 209(2022) 118303.
[9] M. Qiu, Y. Song, Predicting the direction of stock market index movement using an optimized artificial neural network model, PloS one 11(5)(2016) e0155133. DOI:10.1371/journal.pone.0155133.
[10] J. Patel, S. Shah, P. Thakkar, K. Kotecha, Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques, Expert systems with applications 42(1)(2015) 259-268.
[11] K. Funahashi, Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks, Neural Networks 6(6)(1993) 801-806.
[12] J.L. Elman, Finding structure in time, Cognitive science 14(2)(1990) 179-211.
[13] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE transactions on neural networks 5(2)(1994) 157-166.
[14] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 1(1)(1997) 1735-1780.
[15] F.A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: Continual prediction with LSTM, Neural Computation 12(10)(2000) 2451-2471.
[16] G.W. Milligan, M.C. Cooper, A study of standardization of variables in cluster analysis, Journal of Classification 5(2) (1988) 181-204.
[17] V.H. Shah, Machine learning techniques for stock prediction, Foundations of Machine Learning 1(1)(2007) 6-12.
[18] S. Ruder, An overview of gradient descent optimization algorithms. <https://arxiv.org/abs/1609.04747>, 2016 (accessed 01.10.2022).
[19] H. Li, Y. Shen, Y. Zhu, Stock price prediction using attention-based multi-input LSTM, in: Proc. Asian Conference on Machine Learning, 2018.
[20] C.-Y. Lai, R.-C. Chen, R.E. Caraka, Prediction stock price based on different index factors using LSTM, in: Proc. 2019 International conference on machine learning and cybernetics, 2019.
[21] Y. Baek, H.Y. Kim, ModAugNet: A new forecasting framework for stock market index value with an overfitting prevention LSTM module and a prediction LSTM module, Expert Systems with Applications 113(2018) 457-480.