American Journal of Engineering Research (AJER) ***2022***
American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-11, Issue-04, pp-32-40
www.ajer.org
Research Paper Open Access

Prediction of Jakarta Composite Index Volatility Using Long Short Term Memory

Lis Wahyuni[1], Sobri Abusini[2], Mila Kurniawaty[3]
[1]*Master Program of Mathematics, Faculty of Mathematics and Natural Sciences, University Brawijaya, Malang, Indonesia*
[2,3]*Department of Mathematics, Faculty of Mathematics and Natural Sciences, University of Brawijaya, Malang, Indonesia*

ABSTRACT: *This paper discusses the determination of the model and the results of JCI (Jakarta Composite Index) volatility prediction using LSTM (Long Short Term Memory). The LSTM models were tested in several different scenarios using various hyperparameters. The volatility prediction performance on LSTM is measured by RMSPE and RMSE values, the best model is the model that has the smallest RMSPE and RMSE of all models. Based on test results, it is found that LSTM models can predict JCI volatility with good accuracy. RMSPE and RMSE of all models used have small values.*
KEYWORDS: *Prediction, Volatility, LSTM, RMSPE, RMSE*

Date of Submission: 01-04-2022 Date of acceptance: 10-04-2022

I. INTRODUCTION

Stocks are financial instruments that are highly considered in the capital market because they can provide high returns. There is another thing that must be understood, namely that stock investment belongs to the category of investment with high risk due to high fluctuations, so shareholders may experience a very large capital loss. According to [1], the capital loss is a loss due to the difference between the purchase price and the selling price which is negative. Therefore, it is necessary to do a deeper analysis before deciding to buy a stock. If the price of the purchased shares continues to decline, various considerations are needed to predict various possibilities to avoid losses.

The movement of stock prices is indeed very fast-changing, but it is still possible to make predictions or forecasts with various methods for consideration for shareholders. Stock price fluctuations can be seen from the volatility. High and low volatility can describe the risks that must be considered by investors. The higher the volatility, the higher the risk, and vice versa. The newer research on stock volatility continues to emerge to address the various shortcomings or problems that existed in previous studies to obtain the best method for predicting stock volatility. The research carried out continues to develop until now. Some of the problems that arise in forecasting or predicting stock volatility include the accuracy of the prediction model, the period of application of the model is used for prediction because some can only be for a short time while what is needed is a longer period or prediction error problems. Based on some of these problems, a method is needed to predict stock volatility with high accuracy, a fairly long period of application, or a small error so that the prediction results obtained are accurate. For prediction models, the lower the error value, the better the model.

The model that is also being developed and used for prediction purposes is deep learning. Deep learning is part of machine learning [9]. Yang, et al. [11] research states that recent research has shown that deep learning models are better than traditional machine learning models at predicting financial markets. To predict more accurately, the LSTM (Long Short Term Memory) architecture is used. LSTM has a memory cell that can associate the memory of previous events with the input of new events, making it suitable for predicting time series financial data such as stocks. According to [7] LSTM is a reliable choice for the needs of high data accuracy and low data variance. Based on consideration of the capabilities possessed by LSTM, in this study, LSTM will be used to predict the volatility of the JCI (Jakarta Composite Index) shares on the IDX (Indonesia Stock Exchange) using the RMSE (Root Mean Square Error) and RMSPE (Root Mean Square Percentage Error) performance measures. The RMSPE measure can help diagnose if a consistent estimation occurred in a particular experiment [5]. The RMSE measure is used because RMSE is sensitive to outliers [4], and this is

appropriate because the data used has outliers and these outliers do not want to be ignored. The volatility prediction here aims to minimize existing risks. Stocks have high heteroscedasticity and variance, minimizing errors as small as possible will greatly help obtain more accurate accuracy. The JCI measures the price performance of all stocks listed on the Main Board and Development Board of the IDX [2], therefore the JCI is an index that is monitored by many people, including investors and other capital market players. The JCI continues to be evaluated regularly, so that issuers on the JCI may change. The important role of the JCI in the capital market makes this index chosen as the data for this research. The identification of the problem in this study, namely very dynamic stock price fluctuations cause stocks to become financial instruments that have high risk, stock price volatility will greatly affect capital market participants in making decisions.

The modification in this study is the use of realized volatility as an LSTM input. The architecture of the LSTM modeling has been modified, which includes the use of the Lambda layer and the determination of various hyperparameters. The existence of the Lambda layer has a function so that arbitrary expressions can be used as layers when constructing functional and sequential API (Application Programming Interface) models [10].

II. VOLATILITY PREDICTION DESIGN USING LSTM

The following is a design for JCI volatility prediction using LSTM.

- Taking the JCI dataset from Yahoo Finance from January 31, 2011, until January 29, 2021.
- Performing data cleaning which includes deleting unused data, as well as detecting and deleting NaN values.
- The price used here is only the close price, then making a plot for the close price of the dataset.
- Calculating the log returns of the close price and making the plot of log returns.
- Plotting the distribution of log returns and comparing the plots with the standard normal distribution.
- Calculation of observation values, mean, standard deviation, minimum, median, maximum, skewness and kurtosis of log returns.
- Testing the stationarity of log returns using the ADF (Augmented Dickey Fuller) test. If the result of this test is stationer then calculating daily realized volatility from log returns, if the result of this test is nonstationary then using differencing to help getting stationer data.
- Determination of current (realized) and future volatility.
- Cleaning of NaN values on current and future volatility.
- Splitting data log returns, current and future volatility into 2 parts, including train and test data according to Table 1.

Table 1. Splitting dataset

Type Data	Numbers	Percentage
Train data	2139 days	89,55%
Test data	252 days	10,45%

- Normalization of current volatility and future volatility with min-max scaling.
- Using the current volatility of the training dataset as the LSTM input.
- Determination of various hyperparameters (hidden layer neuron units, batch sizes, epochs).
- Overfitting prevention: use of APIs (*EarlyStopping* and *ModelCheckpoint*).
- Determination of the validation_split argument for splitting the training dataset into training and validation datasets.
- LSTM modeling.
- Visualization of the learning curve on the training dataset and validation of each LSTM model.
- Prediction the volatility of each LSTM model.
- Normalization of predictive volatility with min-max scaling.
- Visualization of the comparison of prediction volatility with futures volatility on the testing dataset.
- Calculation of RMSPE and RMSE of all LSTM models.
- Summing up the best LSTM model.

III. RESULTS AND DISCUSSION

The JCI dataset from January 31, 2011, to January 29, 2021, which has been cleaned up, then plots the daily close prices. The plot of daily close prices is shown in Figure 1. Figure 1 shows that the JCI data used during the observation period has very fluctuating price movements. The highest price for the JCI in this period was IDR 6,689.00 on 19 February 2018. The JCI experienced two very significant declines in September 2015 and March 2020. According to [3] negative sentiment on global issues such as the Greek debt crisis, rising interest rates Fed interest rates, falling commodity prices and the slowdown in China's economy lowered the JCI to a price of IDR 4,121.00, there were also domestic factors that influenced the decline in the JCI in 2015

such as Indonesia's declining economic growth, the weakening of the rupiah, declining financial performance, and economic conditions. Indonesian politics are not harmonious.

Fig. 1. Plot of daily close prices

In March 2020, when the JCI fell to a price of IDR 3,938.00, a temporary suspension of trading was imposed (trading halt) due to the COVID-19 pandemic that occurred and greatly affected the Indonesian economy [8]. According to [3] conditions in 2020 are very uncertain and not very good, but at the beginning of 2021, it will be a recovery phase, which is indicated by the occurrence of stable conditions and an increase in the number of companies experiencing profits. From the close prices obtained, the log returns can be calculated. The plot of the log returns calculation is given in Figure 2, and the log returns distribution is presented in Figure 3.

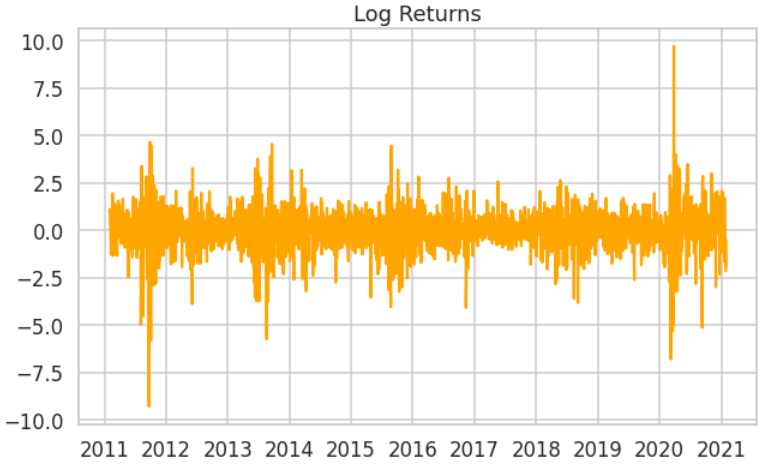

Fig. 2. Plot of log returns

Figure 3 shows positive kurtosis (leptokurtic) seen that the peak of log returns are higher and the tail is thicker than the standard normal distribution. For descriptive analysis of log returns during the observation period, it can be seen in Table 2.

Table 2. Descriptive analysis of log returns

Observation	2433
Mean	0.023093
Standard Deviation	1.094169
Minimum	-9.299684
Median	0.086737
Maximum	9.704219
Skewness	-0.533845
Kurtosis	8.137918

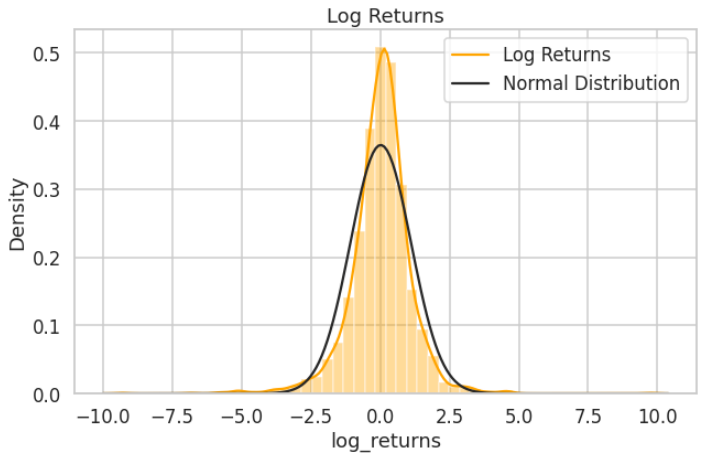

Fig. 3. Plot comparison of log returns distribution and standard normal distribution

- **Stationarity Test**

Log returns need to be checked for stationarity. Stationarity can be checked using the ADF test with the following hypothesis.

H_0: time series data has a unit root, and is not stationary.

H_1: time series data has no unit root and is stationary.

If the data is not stationary, it is necessary to do a differencing process until the data is stationary. The results of the ADF test on the log returns data are presented in Table 3.

Table 3. ADF test results of log returns

ADF Statistics	11.9235843
p-value	$4.9665607 \times 10^{-22}$
Critical Value 5%	-2.8627

From Table 3, it can be seen that log returns have p-value < 0.05, meaning that H_0 is rejected and H_1 is accepted, so it can be concluded that the log returns do not have a unit root and are stationary.

- **Realized Volatility Calculation**

To obtain the daily realized volatility at certain intervals, in this case, it can be in the form of weekly (5 days), monthly (21 days), annual (252 days) intervals, scaling is carried out as follows.

$$RV_{daily} = \sqrt{\sum_{i=1}^{T} r_i^2} \cdot \sqrt{\frac{1}{n-1}},$$

where n is the interval of the number of days used. In the following Figure 4, a plot of daily realized volatility is presented using different windows intervals.

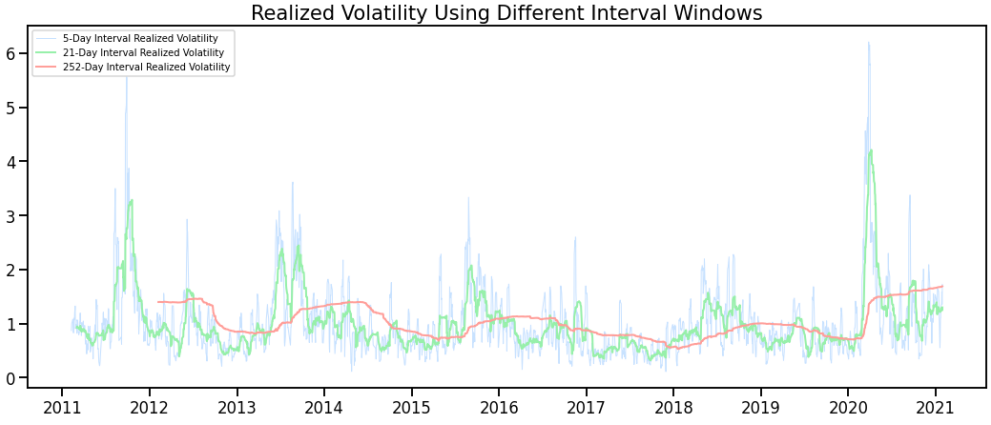

Fig. 4. Realized volatility at intervals of windows 5, 21, and 252 days.

The window interval used in this study is 21 days which represents about 1 month of stock trading time. The 21-day interval was used because the 5-day interval was too hectic to observe meaningful patterns or information, while the 252-day interval reduced volatility too significantly. Time series prediction models predict future values based on previously observed values. In this study, the realized volatility value becomes the current volatility value, then the future volatility value that will be used as the target is obtained by shifting the volatility (current) backward by 1 index. It can be said that for yesterday, today is the future so if today's volatility is shifted 1 day back, it can be used as the output future targeted yesterday, this value is then used for training and evaluation of model performance. Visualization of current and future volatility is given in Figure 5.

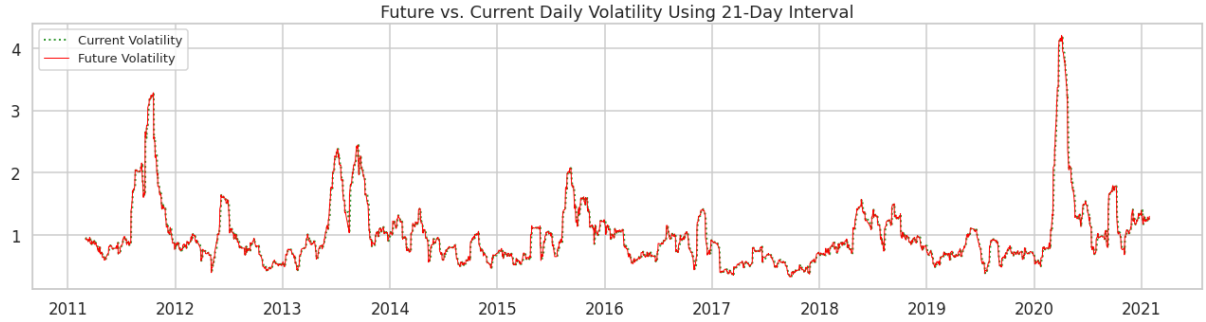

Fig. 5 Future and current volatility with 21 days windows interval.

- **Hyperparameter Tuning**

There is no mandatory rule in determining the number of hyperparameters, so the hyperparameters in this study were first tested and then seen in detail what hyperparameters produced the best model for predicting stock volatility of JCI data by looking at the smallest RMSPE and RMSE values. In this study, the details of the hyperparameters used in LSTM are given in Table 4.

Table 4. Details of the hyperparameters used.

No.	Hyperparameter	Numbers
1	Unit neurons in hidden	16, 32, 64
2	Batch sizes	16, 32, 64
3	Epochs	100, 200, 500, 1000

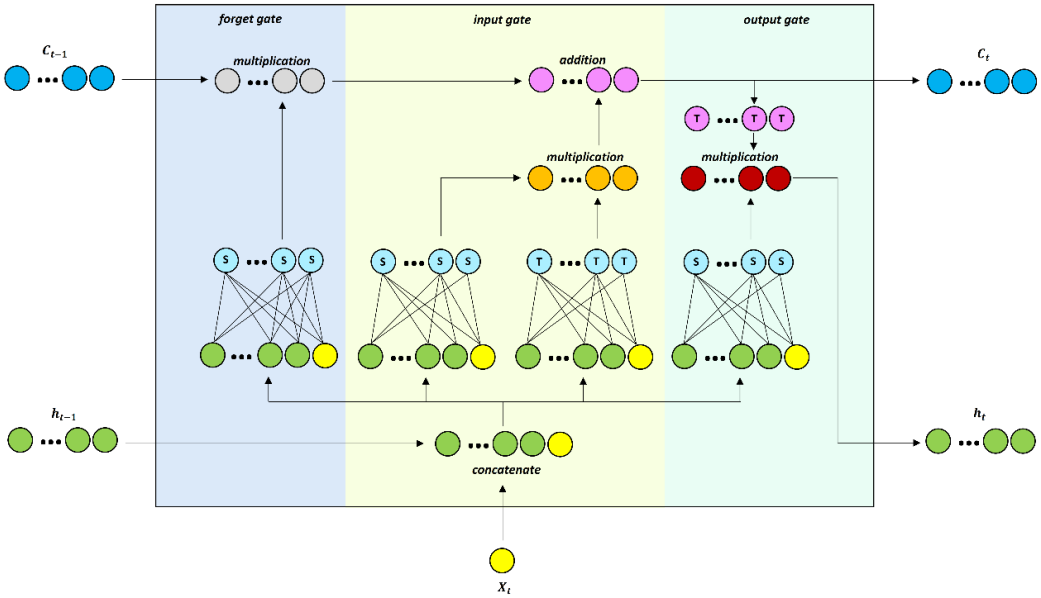

Fig. 6 LSTM cell.

In this study, validation split=0.2 is used, so that the training dataset is split automatically by the **fit()** function into training and validation datasets, with the percentage of training datasets used for validation datasets of 0.2. The performance measure that is used as a monitor to end the training is RMSPE validation, so the training will be stopped if the RMSPE validation is no longer increasing. The final model after training is

stopped by *EarlyStopping* may not be the best model in the validation dataset, therefore an additional callback is used that can store the best model during training, namely *ModelCheckpoint* with its monitor also using RMSPE validation. In LSTM modeling, the Lambda layer is used.

In LSTM modeling, the LSTM layer is used, inside the LSTM layer, there are LSTM cells. Based on [6], the LSTM cell used in this study is given in Figure 6. In Figure 6, 3 inputs enter the LSTM cell, namely the hidden state value in the previous time step $t-1$ (h_{t-1}), the cell state value in the previous time step $t-1$ (C_{t-1}), and the input value at the current time step t (x_t),. It can be seen that the LSTM has 4 FFNNs in Figure 6, namely 1 on the forget gate, 2 on the input gate, and 1 on the output gate. The size of the input and output tensors (dimensions) is symbolized by a circle that is given a special color. In Figure 6, hidden state h_{t-1} and cell state C_{t-1} are vectors that have dimension d which is determined from the number of hyperparameter units of neurons in the hidden layer in the LSTM cell, in this study d can be worth 16, 32, or 64. The dimensions of the vector h_{t-1} and C_{t-1} should be the same. The dimensions of the vectors h_{t-1} and C_{t-1}, as well as h_t and C_t, must be the same. The input to Figure 6 is a vector having 1 dimension which is determined by the number of features. This study uses 1 feature as an input to the LSTM, namely realized volatility.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lambda (Lambda)             (None, None, 1)           0

 lstm (LSTM)                 (None, 16)                1152

 dense (Dense)               (None, 1)                 17

=================================================================
Total params: 1,169
Trainable params: 1,169
Non-trainable params: 0
```

Fig. 7. LSTM parameters (16 units).

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lambda (Lambda)             (None, None, 1)           0

 lstm (LSTM)                 (None, 32)                4352

 dense (Dense)               (None, 1)                 33

=================================================================
Total params: 4,385
Trainable params: 4,385
Non-trainable params: 0
```

Fig. 8. LSTM parameters (32 units).

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lambda (Lambda)             (None, None, 1)           0

 lstm (LSTM)                 (None, 64)                16896

 dense (Dense)               (None, 1)                 65

=================================================================
Total params: 16,961
Trainable params: 16,961
Non-trainable params: 0
```

Fig. 9. LSTM parameters (64 units).

The summary of the architecture along with the number of parameters of the LSTM (16 units), LSTM (32 units), and LSTM (64 units) models respectively in Figures 7, 8, and 9. The Lambda layer has no neuron units, so the parameter in the Lambda layer is zero. The LSTM modeling architecture is shown in Figure 10.

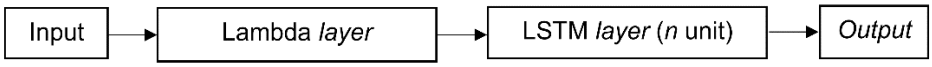

Fig. 10. The LSTM modeling architecture.

Table 5. RMSPE and RMSE LSTM models of various hyperparameters.

Epochs = 100			
Batch Size	**Model (Neuron Hidden)**	**RMSPE Testing**	**RMSE Testing**
16	LSTM 16 unit	0.117218	0.05981
	LSTM 32 unit	0.112257	0.045326
	LSTM 64 unit	0.111903	0.044433
32	LSTM 16 unit	0.116204	0.057198
	LSTM 32 unit	0.11368	0.045036
	LSTM 64 unit	0.114863	0.042206
64	LSTM 16 unit	0.115225	0.052383
	LSTM 32 unit	0.113061	0.043457
	LSTM 64 unit	0.112155	**0.041049**
Epochs = 200			
Batch Size	**Model (Neuron Hidden)**	**RMSPE Testing**	**RMSE Testing**
16	LSTM 16 unit	0.116252	0.057017
	LSTM 32 unit	0.112624	0.046369
	LSTM 64 unit	0.111746	0.044081
32	LSTM 16 unit	0.113567	0.047196
	LSTM 32 unit	0.112563	0.045633
	LSTM 64 unit	0.111696	0.042764
64	LSTM 16 unit	0.114321	0.051041
	LSTM 32 unit	0.112774	0.047073
	LSTM 64 unit	0.111572	0.042945
Epochs =500			
Batch Size	**Model (Neuron Hidden)**	**RMSPE Testing**	**RMSE Testing**
16	LSTM 16 unit	0.112865	0.046437
	LSTM 32 unit	0.112281	0.04578
	LSTM 64 unit	0.11211	0.044225
32	LSTM 16 unit	0.115828	0.056293
	LSTM 32 unit	0.112503	0.045414
	LSTM 64 unit	0.111777	0.043474
64	LSTM 16 unit	0.115952	0.056139
	LSTM 32 unit	0.11246	0.04575
	LSTM 64 unit	0.111642	0.043214
Epochs = 1000			
Batch Size	**Model (Neuron Hidden)**	**RMSPE Testing**	**RMSE Testing**
16	LSTM 16 unit	0.114435	0.049963
	LSTM 32 unit	0.1129	0.046128
	LSTM 64 unit	0.11195	0.043569
32	LSTM 16 unit	0.113284	0.048368
	LSTM 32 unit	0.112285	0.044674
	LSTM 64 unit	0.111746	0.043285
64	LSTM 16 unit	0.11456	0.052073
	LSTM 32 unit	0.11216	0.044984
	LSTM 64 unit	**0.11154**	0.042801

There are 2160 datasets trained in LSTM modeling, so it will be difficult if these datasets are processed all at once. To process the entire dataset, the process is divided into batch sizes whose values have been determined in Table 4. In the setting of batch size 16, each process has 16 data entered 1 by 1 as input into the LSTM architectures as shown in Figures 10 until the entire dataset is processed in the architecture. This study uses time series data so that the selected data is sequential (not random). If all the data has been processed, this is usually referred to as 1 epoch. The number of processes to complete 1 epoch is called iteration, in this case,

there are 2160: 16 = 135 iterations. Other hyperparameters specified in Table 4 are also processed in the same way.

- **LSTM Model**

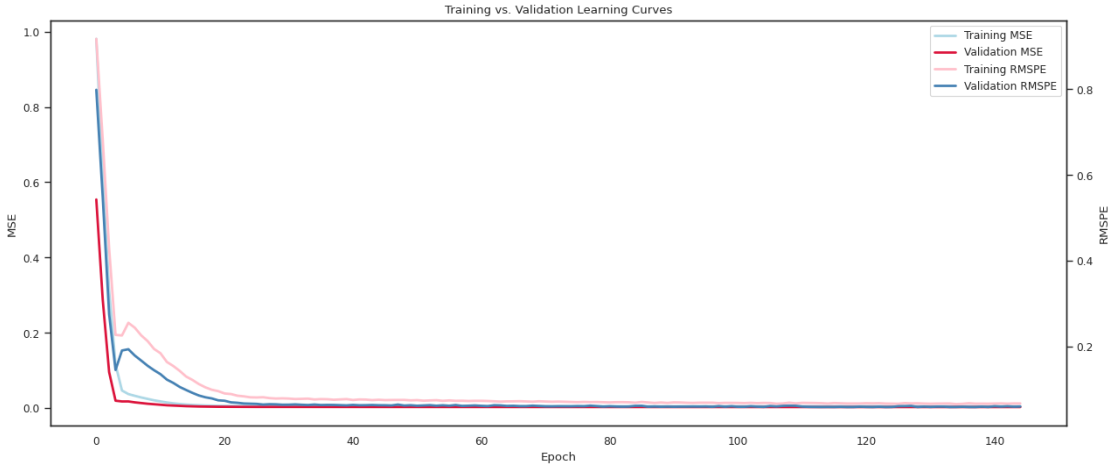

Fig. 11. Learning curve LSTM (64 units) batch size 64 epoch 1000.

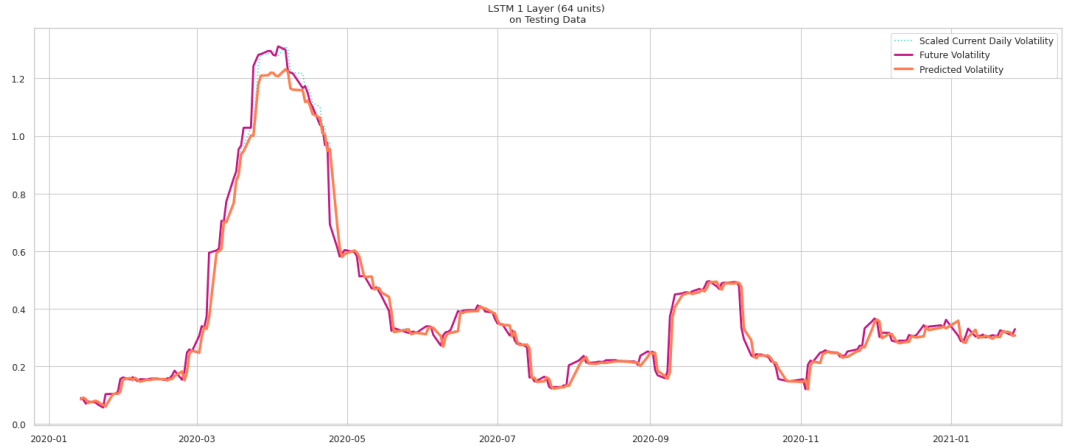

Fig. 12. LSTM volatility (64 units) batch size 64 epoch 1000.

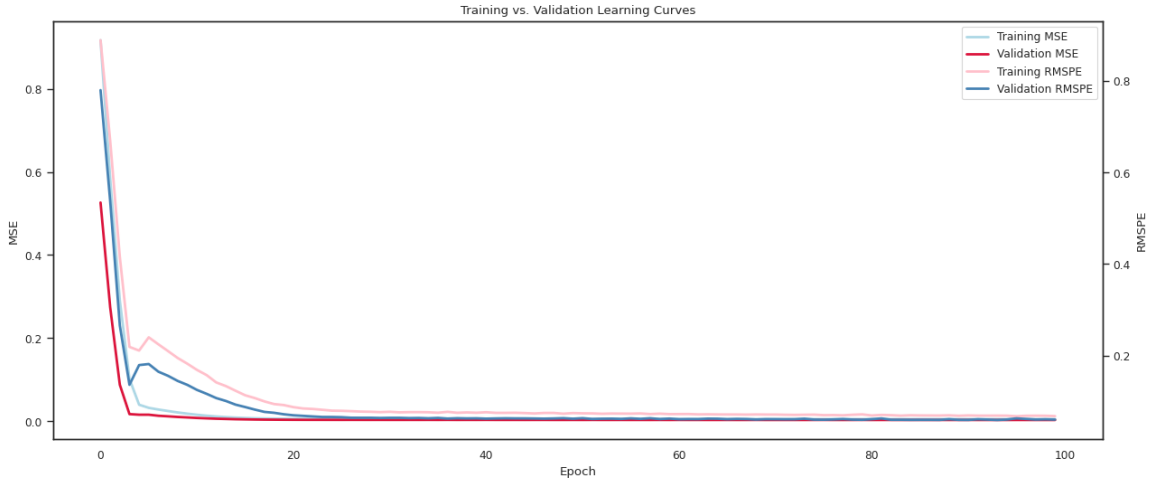

Fig. 13. Learning curve LSTM (64 units) batch size 64 epoch 100.

Determination of the best model for the prediction can be seen from the smallest RMSPE and RMSE values in the testing data. The RMSPE and RMSE LSTM models of various hyperparameters on the testing data are given in Table 5.

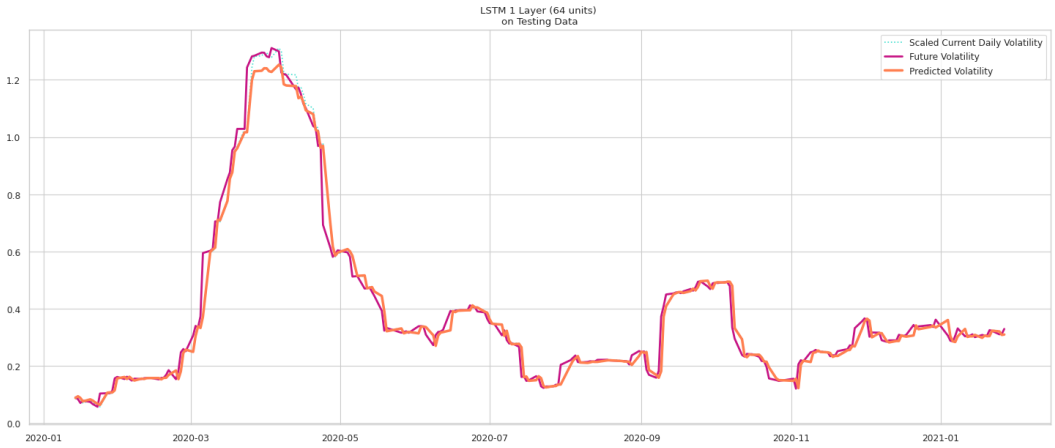

Fig. 14. Volatilitas LSTM (64 units) batch size 64 epoch 100.

The test shows that the smallest RMSPE value, namely 0.11154 obtained from the LSTM model (64 units) with hyperparameter batch size 64 and epochs 1000, while the smallest RMSE value, namely 0.041049 obtained from the LSTM model (64 units) with hyperparameter batch size 64 and epochs 100. Learning curve visualization and volatility visualization of the LSTM model (64 units) with hyperparameter batch size 64 and epochs 1000 are given in Figures 11 and 12. Learning curve visualization and volatility visualization of the LSTM model (64 units) with hyperparameter batch size 64 and epochs 100 are given in Figures 13 and 14.

IV. CONCLUSION

Based on the results of the study, the conclusion obtained is that volatility prediction is carried out by calculating the log returns which are used to determine realized volatility as LSTM input. The log returns are obtained from the close prices of the JCI. Volatility prediction performance on LSTM is measured by the value of RMSPE and RMSE, the best model is the model that has the smallest RMSPE and RMSE of all models. The LSTM model testing was carried out with several different scenarios using various hyperparameters. Based on the test results, it was found that the LSTM can predict the volatility of the JCI with good accuracy seen from the RMSPE and RMSE. All models used have a small value with the smallest value at the smallest RMSPE, which is 0.11154 which is obtained from the LSTM model (64 units) batch size 64 epochs 1000, while the smallest RMSE value is 0.041049 which is obtained from the LSTM model (64 units) batch size 64 epochs 100.

The results obtained and the various limitations that exist in this study, the suggestions that can be given for further research are that it would be better if we could use a method that can determine the optimal hyperparameter automatically, and hybrid LSTM with other methods can also be used to make better predictions on time series data, such as stocks, weather, gold prices, and so on.

REFERENCES

[1]. IDX. 2018. Daftar Istilah. https://www.idx.co.id/footer-menu/tautan-langsung/daftar-istilah/. [accessed on December 8, 2020].
[2]. _____. 2021. IDX Stock Index Handbook v1.2. https://www.idx.co.id/media/9816/idx-stock-index-handbook-v12-_-januari-2021.pdf. [accessed on April 5, 2021].
[3]. BPS. 2021. Kajian Perkembangan Pasar Saham dan Keuangan Emiten selama Pandemi Covid-19. https://www.bps.go.id/publication/2021/12/29/2375b1f0fbd84c1a20139e6c/kajian-perkembangan-pasar-saham-dan-keuangan-emiten-selama-pandemi-html. [accessed on January 27, 2022].
[4]. Chai, T., dan Draxler, R. R. 2014. Root Mean Square Error (RMSE) or Mean Absolute Error (MAE)? –Arguments Against Avoiding RMSE in the Literature. *Geoscientific Model Development*. 7, 1247–1250.
[5]. Doyog, N. D., Lin, C., Lee, Y. J., Lumbres, R. I. C., Daipan, B. P. O., Bayer, D. C., dan Parian, C. P. 2021. Diagnosing Pristine Pine Forest Development Through Pansharpened-Surface-Reflectance Landsat Image Derived Aboveground Biomass Productivity. *Forest Ecology and Management*. 487.
[6]. Karim, R. 2018. Animated RNN, LSTM and GRU. https://towardsdatascience.com/animated-rnn-lstm-and-gru-ef124d06cf45. [accessed on January 27, 2022].
[7]. Karmiani, D., Kazi, R., Nambisan, A., Shah, A., dan Kamble, V. 2019. Comparison of Predictive Algorithms: Backpropagation, SVM, LSTM and Kalman Filter for Stock Market. Amity International Conference on Artificial Intelligence (AICAI). 228-234.
[8]. Nurhaliza, S. 2020. IHSG Trading Halt, Anjlok 5 Persen Kena Sentimen PSBB. https://www.idxchannel.com/market-news/ihsg-trading-halt-anjlok-5-persen-kena-sentimen-psbb. [accessed on December 12, 2021].
[9]. Putra, J. W. G. 2020. Pengenalan Pembelajaran Mesin dan Deep Learning. https://wiragotama.github.io/resources/ebook/parts/JWGP-intro-to-ml-front-secured.pdf. [accessed on December 13, 2020].
[10]. TensorFlow. 2021. tf.keras.layers.Lambda. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Lambda. [accessed on November 24, 2021].
[11]. Yang, C., Zhai, J., dan Tao, G. 2020. Deep Learning for Price Movement Prediction Using Convolutional Neural Network and Long Short-Term Memory. Mathematical Problems in Engineering.