LSTM-RNN Automotive Stock Price Prediction

Kevin Johan, Julio C. Young, Seng Hansun

Abstract: Stock is a securities or paper sheet as proof of ownership of a company. In terms of buying and selling a stock, stock price information is very important for the investors, since the purchase of stock usually will be made when the stock at the lowest price and the sale of stock will be made at the highest price. The focus of this study is to use the Long Short Term Memory algorithm to predict stocks' price in automotive companies. The Long Short Term Memory algorithm is often used for prediction application, for example, in the analysis and implementation of Long Short Term Memory Neural Network in bitcoin prices prediction. This research was conducted using five automotive stock data that were taken from Yahoo Finance! The research experiments were conducted to get the effect of the number of hidden layer and epochs on the accuracy of stock predictions. From the results of the experiments, we found that the more usage of hidden layers and epochs will make the accuracy results better.

Index Terms: ADAM, automotive, LSTM, prediction, RNN, stock, accuracy.

1 Introduction

STOCK is a securities or paper sheet as proof of ownership of a company. In general, stocks have two general groups, i.e., common stock and preferred stock [1]. The shareowner usually will profit from the company. The process of sharing profit to shareholders is done with dividends which are in accordance with capital included or part of shares [2]. As an investor, it is very important to know how to analyze stocks to understand the situation and condition of the upcoming shares [3], [4]. In general, an investor can predict the price of a stock based on trends from the pass stock movement data [5]. Traders, brokers, and investors need a tool to help them predict the stock price. With the existence of a tool in predicting a stock, it can help trader, broker, and investor to take decision when to buy a share or to sell a share to get high profit [1]. One of the techniques that can be used for prediction is the Long Short Term Memory (LSTM) algorithm. Long Short Term Memory is a type of Recurrent Neural Network (RNN) [6]. It has been used in several pieces of research, such as bitcoin price prediction [7], IDR-USD currencies pair prediction [8], and even in human trajectory prediction in crowded spaces [9]. In this research, we are trying to use the LSTM algorithm as a tool in predicting stock prices. We focus on five different automotive companies, i.e., Honda, Mitsubishi, Toyota, Suzuki, and Daihatsu. Moreover, R-Squared method will be used to calculate the accuracy level of this research.

- *Kevin Johan is a graduate student from the Informatics department in Universitas Multimedia Nusantara, Indonesia, 15131. E-mail: kevin.johan@student.umn.ac.id*
- *Julio C. Young is currently a lecturer in Informatics department in Universitas Multimedia Nusantara, Indonesia, 15131. E-mail: julio.christian@umn.ac.id*
- *Seng Hansun is currently a lecturer in Informatics*
- *department in Universitas Multimedia Nusantara, Indonesia, 15131. E-mail: hansun@umn.ac.id*

R-Squared is a technique that can be used to measure how much percentage in a dependent variable is. It is also used to measure the accuracy of the prediction result [10]. In the next section, several methods being used in this research will be described briefly, such as the basic concept of RNN, LSTM, ADAM, and R-Squared. In Section 3, the research's results and analysis of the data will be given. Section 4, as the last section, will emphasize the research's findings as the conclusion part of this paper.

2 Research Methodology

Several methods had been incorporated in this study, such as Recurrent Neural Network and Long Short Term Memory. Adaptive Moment Estimation Optimization as optimizer and R-Squared as accuracy measurement also being described here.

2.1 Recurrent Neural Network

Recurrent neural network (RNN) was developed in 1996 by Jeff Elman. RNN is a variant of Artificial Neural Network (ANN); however, there are major differences between RNN and ANN. In RNN, the output data is forwarded as input for itself. RNN will store information related to the data at a certain time. The output that is forwarded as the input for the next cell is called the context layer on RNN. Based on the explanation, RNN has a memory that contains a result of recording information from the previous result [11].

2.2 Long Short Term Memory

Long Short Term Memory (LSTM) is a type of RNN. It is created by Hochreiter and Schmidhuber in 1997. LSTM will store information based on the pattern found in data. The neuron in LSTM consist of forget gates, input gates, and output gates that can regulate the memory of each neuron itself, and LSTM can process many time-series data [6]. Forget gates process is carried out to select what data affects the calculation at the next stage and what data is not. The output is between 0 and 1. 0 means the data is not used anymore, and 1 means the data is still being used [6]. Forget gates can be formulized as Eq. (1). $f_t = \sigma(W_f[s_{t-1}, x_t] + b_t)$ (1) where f_t is the forget gate, σ is the sigmoid function, W_f is the weight of each neuron at forget gate, s_{t-1} is the result of the previous process, x_t is the input, and b_t is the bias at forget gate. Input gates have two processes, first is the sigmoid function activation, which is responsible for deciding which values will be updated. In the second process, it is followed by the tanh function to create a new vector which will be stored in

the memory cell [6]. Eq. (2) and (3) show the operations that will be done in a cell's input gates. $I_t = \sigma(W_i . [s_{t-1}, x_t] + b_i) \check{C}_t = tanh(W_c . [s_{t-1}, x_t] + b_c$ (3)where I_t is the input gate, $\check{C}_t$ is C tilde, W_i is the weight of each neuron at the input gate, W_c is the weight of each neuron at C tilde, b_i is the bias at the input gate, and b_c is the bias at C tilde. Output gates have three processes, the first process will decide which part of the memory cell will be used by using a sigmoid function. In the second process, the value of the memory cell will be placed in tanh function. In the last process, the two values will be multiplied, and the result will become output for next process [6]. Eq. (4) and (5) show the processes in the output gates. $O_t = \sigma(W_o . [s_{t-1}, x_t] + b_o)$(4) $S_t = O_t \tanh(C_t)$ (5) where O_t is the output gate, S_t is the final LSTM result, W_o is the weight of each neuron at the output gate, b_o is the bias at the output gate, and C_t is the new memory cell. Memory cell process to update previous memory cell to new memory cell [6] can be done by using Eq. (6). $C_t = f_t * C_{t-1} + I_t * \check{C}_t$ (6) where f_t is the result of forget gate, C_{t-1} is the previous result of the memory cell, I_t is the result of the input gate, and $\check{C}_t$ is the new memory cell candidate.

2.3 Adaptive Moment Estimation Optimization

ADAM is an optimizer algorithm developed using the benefit of Adaptive Gradient (AdaGrand) and Root Mean Square Propagation (RMSProp). ADAM only requires a first-order gradient and a little memory. ADAM calculates a level of learning from each individual in different parameters from an estimate of the first and second moments of gradient [12].

2.4 R-Squared

R-Squared is used to calculate the accuracy of prediction. Usually, the value range of R-Squared is between 0 to 1, if the value is getting closer to 1, then it means the result is good [13]. The formula of R-Squared is shown in Eq. (7). $R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$ (7) where SS_{res} and SS_{tot} can be found using Eq. (8) and (9) respectively. $SS_{res} = \sum_{i=0}^{n} (y_i - f_i)^2$ (7) $SS_{tot} = \sum_{i=0}^{n} (y_i - \bar{y})^2$ (7) Here, y_i is the actual data, f_i is the predicted data, $\bar{y}$ is the mean of data, and n is the total number of data

3 Results and Discussion

```
In [9]: batch_size =400
        epoch = 800
        N = len(X_train)
        T = X_train.shape[1]

        learning_curve = []
        test_learning_curve = []
        optimizer = Adam(lr=0.001) #adam digunakan untuk optimaizer lstm
        for i in range(epoch):
            loss = 0
            test_loss = 0
            perm = np.random.permutation(N)
            for j in range(N//batch_size):
                train_batch = X_train[perm[j*batch_size : (j+1)*batch_size]]
                response_batch = y_train[perm[j*batch_size : (j+1)*batch_size]]
                l = 0
                with sequential.train():
                    for t in range(T):
                        z = sequential(train_batch[:, t, :])
                        l = rm.mse(z, response_batch)
                    sequential.truncate()
                l.grad().update(optimizer)
                loss += l.as_ndarray()
            l_test = 0
            for t in range(T):
                z = sequential(X_test[:, t, :])
                l_test = rm.mse(z, y_test)
            sequential.truncate()
            test_loss += l_test.as_ndarray()
            if i % 50 == 0: #diprint per 50 epotch
                print("epoch:{} {} {}".format(i, loss[0], test_loss[0]))
            learning_curve.append(loss)
            test_learning_curve.append(test_loss)
```

Fig. 1. *Training data snipped code*

Fig. 1 shows the training data phase code. The learning process on the data using LSTM will be done in this phase. The looping process will be done by considering the epoch number and batch size to get the train_loss and test_loss values as shown in Fig. 2.

```
epoch:0 0.9409400820732117 0.14314718544483185
epoch:50 0.0038681288715451956 0.027484603226184845
epoch:100 0.003664719872176647 0.022118497639894485
epoch:150 0.0032104854471981525 0.01099689956754446
epoch:200 0.0025967697147279978 0.0008317159372381866
epoch:250 0.0023385274689644575 0.0002613566175568849
epoch:300 0.001764159183949232 0.0019272237550467253
epoch:350 0.0015709255822002888 0.0004976153140887618
epoch:400 0.0013427699450403452 0.00010102565283887088
epoch:450 0.0013657831586897373 0.0001940316433319822
epoch:500 0.0013520071515813 47 0.00010609248420223594
epoch:550 0.0013672548811882734 0.00010294254025211558
epoch:600 0.00160217157099396 0.00016385719936806709
epoch:650 0.0013521037762984633 0.00022349819482769817
epoch:700 0.001429026946425438 0.00013836671132594347
epoch:750 0.0013997489586472511 0.00019348463683854789
```

Fig. 2*. Train loss and test loss on the training data*

Moreover, Fig. 3 shows the snipped code in the testing phase. There is only one looping in testing, and the results will be saved as test_predict variable.

```
for t in range(7):
    test_predict = sequential(X_test[:, t, :])
sequential.truncate()
test_predict = np.array([max(val,0) for val in test_predict])
test_predict.shape = (test_predict.shape[0],1)
# test_predict = np.array([min(val,1) for val in test_predict])

# honda_scaled = (honda_close - min_price)/(max_price - min_price)
# scaled * (max_price - min price) + min_price = honda_close
#print("Root mean squared error:{}".format(np.sqrt(mean_squared_error(y_test, test_predict))))

plt.figure(figsize=(16,8))
plt.title("predictions")
plt.grid(True)
plt.plot(y_test, marker="^", label ="original")
plt.plot(test_predict, marker=".", label="predict")

plt.show()
plt.clf()
```

***Fig. 3**. Testing data snipped code*

Fig. 4 displays a prediction graph results, which were built by using matplotlib library. The blue line is the original data and the orange line is the prediction result.

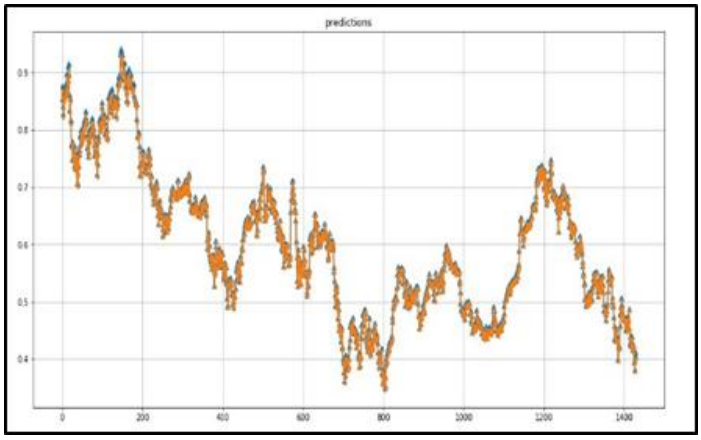

***Fig. 4**. Prediction results graph*

In the trial phase, we used five data of automotive stocks, i.e., Honda, Mitsubishi, Toyota, Suzuki, and Daihatsu. The trial consisted of 100 up to 800 epoch and one up to four hidden layers. From the trial results, it can be concluded that by increasing the epoch and hidden layers number, it will make the accuracy of prediction better, but when using small epoch and large hidden layers results in lesser accuracy level of the prediction result. By looking at the average value of the prediction accuracy results by using LSTM algorithm, in general, the LSTM algorithm has a good level of accuracy to make the prediction. Table 1 until Table 5 depicted the result of accuracy in each data stock being observed in this research.

TABLE 1
HONDA DATA RESULTS

Honda	Batch size 400			
epoch/HL	1	2	3	4
100	0,76	0,34	-0,58	-1,98
200	0,64	0,87	0,82	0,87
300	0,84	0,95	0,85	0,92
400	0,93	0,99	0,98	0,96
500	0,99	0,99	0,99	0,99
600	0,98	0,98	0,99	0,97
700	0,99	0,98	0,98	0,99
800	0,99	0,98	0,98	0,99

TABLE 2
MITSUBISHI DATA RESULTS

Mitsubishi	Batch size 400			
epoch/HL	1	2	3	4
100	0,26	-0,12	-1,94	-3,92
200	0,96	0,97	0,45	-1,65
300	0,9	0,91	0,91	0,93
400	0,9	0,98	0,95	0,95
500	0,96	0,98	0,98	0,98
600	0,98	0,97	0,97	0,98
700	0,97	0,96	0,98	0,98
800	0,97	0,96	0,94	0,97

TABLE 3
TOYOTA DATA RESULTS

Toyota	Batch size 400			
epoch/HL	1	2	3	4
100	0,01	0,51	-3,44	-7,57
200	0,97	0,62	0,85	-1,34
300	0,97	0,61	0,09	0,35
400	0,72	0,97	0,95	0,97
500	0,96	0,97	0,93	0,94
600	0,95	0,97	0,94	0,95
700	0,97	0,97	0,97	0,95
800	0,97	0,97	0,97	0,97

TABLE 4
SUZUKI DATA RESULTS

Suzuki	Batch size 400			
epoch/HL	1	2	3	4
100	0,78	0,68	0,36	-0,03
200	0,83	0,72	0,48	0,07
300	0,88	0,81	0,53	0,16
400	0,96	0,94	0,67	0,36
500	0,98	0,96	0,83	0,55
600	0,99	0,99	0,93	0,75
700	0,99	0,99	0,99	0,91
800	0,99	0,99	0,99	0,98

TABLE 5
DAIHATSU DATA RESULTS

Daihatsu	Batch size 400			
epoch/HL	1	2	3	4
100	0,83	0,78	-0,45	-2,54
200	0,84	0,93	0,83	0,85
300	0,93	0,97	0,96	0,97
400	0,97	0,97	0,97	0,98
500	0,98	0,97	0,98	0,94
600	0,97	0,97	0,97	0,98
700	0,98	0,98	0,96	0,97
800	0,97	0,98	0,98	0,98

4 Conclusion

Based on the research that has been done, it can be concluded that the design and development of prediction application for automotive stocks using Long Short Term Memory have been successfully done. Based on the result of trials on Honda data, the highest prediction accuracy is 99% by using 800 epoch and four hidden layers. For Mitsubishi data, the highest prediction accuracy is 98% by using 700 epoch and four hidden layers. For Toyota data, the highest prediction accuracy is 97% by using 800 epoch and four hidden layers. For Suzuki data, the highest prediction accuracy is 99% by using 800 epoch and three hidden layers, and for Daihatsu data, the highest prediction accuracy is 98% by using 800 epoch and four hidden layers. In general, LSTM algorithm is good to use for predicting automotive stock prices. Furthermore, there are several suggestions for future research, i.e., (1) make additional testing on batch size parameter in training process, (2) use the Z-Score method for evaluation, and (3) use the Decimal scaling for normalization technique.

References

[1] Muchlas and T. Sutikno, "Prediksi Harga Saham Berbasis Web dengan Sistem Inferensi Fuzi Tsukamoto," Proc. Seminar Nasional Aplikasi Teknologi Informasi (SNATI 2007), pp. D-27-D-31, June 2007.

[2] Mudjiyono, "Investasi dalam Saham & Obligasi dan Meminimalisasi Risiko Sekuritas pada Pasar Modal Indonesia," Jurnal STIE Semarang, vol. 4, no. 2, pp. 1-18, 2012.

[3] S. Hansun, "Jakarta Stock Exchange (JKSE) Forecasting using Fuzzy Time Series," Proc. 2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems (ROBIONETICS), 2013.

[4] D. Widodo and S. Hansun, "Implementasi Simple Moving Average dan Exponential Moving Average dalam Menentukan Tren Harga Saham Perusahaan," ULTIMATICS, vol. 7, no. 2, pp. 113-124, 2015.

[5] T.A. Widi, Perbandingan Model Chen dan Lee pada Metode Fuzzy Time Series untuk Prediksi Harga Saham Bank BRI. Thesis. Universitas Islam Indonesia, Yogyakarta, 2018.

[6] M.W.P. Aldi, Jondri, and A. Aditsania, "Analisa dan Implementasi Long Short Term Memory Neural Network untuk Prediksi Harga Bitcoin," e-Proceeding of Engineering, vol. 5, no. 2, pp. 3548-3555, 2018.

[7] C-H. Wu, C-C. Lu, Y-F. Ma, and R-S. Lu, "A New Forecasting Framework for Bitcoin Price with LSTM," Proc. 2018 IEEE International Conference on Data Mining Workshops (ICDMW), 2018.

[8] H. Prasetyanwar and Jondri, "Peramalan Nilai Tukar IDR-USD Menggunakan Long Short Term Memory," e-Proceeding of Engineering, vol. 5, no. 2, pp. 3820-3826, 2018.

[9] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 961-971, 2016.

[10] H. Halin, H. Wijaya, and R. Yusilpi, "Pengaruh Jual Harga Kaca Patri Jenis Silver Terhadap Nilai Penjualan pada CV. Karunia Kaca Palembang Tahun 2004-2015," Jurnal Ecoment Global, vol. 2, no. 2, pp. 49-56, 2017.

[11] R.A. Juanda, Jondri, and A.A. Rohmawati, "Prediksi Harga Bitcoin dengan Menggunakan Recurrent Neural Network," e-Proceeding of Engineering, vol. 5, no. 2, pp. 3682-3690, 2018.

[12] D.P. Kingma and J.L. Ba, "Adam: A Method For Stochastic Optimization," Proc. 3rd International Conference for Learning Representations (ICLR), 2015.

[13] S. Glantz, B. Slinker, and T. Neilands, Primer of Applied Regression and Analysis of Variance. USA: McGraw-Hill, 2016.