ISSN: 2339-2541
JURNAL GAUSSIAN, Volume 12, Nomor 2, Tahun 2023, Halaman 287 - 295
Online di: https://ejournal3.undip.ac.id/index.php/gaussian/

PREDIKSI HARGA EMAS DUNIA MENGGUNAKAN METODE *LONG-SHORT TERM MEMORY*

Tania Giovani Lasijan[1], Rukun Santoso[2], Arief Rachman Hakim[3]
[1,2,3] Departemen Statistika, Fakultas Sains dan Matematika, Universitas Diponegoro
***e-mail**: giovanitania93@gmail.com

DOI: 10.14710/j.gauss.12.2.287-295

Article Info:
Received: 2022-10-14
Accepted: 2023-05-14
Available Online: 2023-07-28

Keywords:
Gold; Long-Short Term Memory; Recurrent Neural Network

Abstract: Gold investment is one of the investments that is quite lot of interest by the public and also is considered safer because it has relatively low risk and tends to be stable compared to other investment instruments, especially amid the uncertainty of global economic conditions caused by the COVID-19 pandemic. Awareness about gold price predictions can provide information to people who want to invest in gold so they have higher opportunity to earn profits and minimize the risks obtained. The gold prices prediction method used in this study is Long-Short Term Memory (LSTM) using RStudio. LSTM is one of the method that is widely used to predict time series data. LSTM is a variation of the Recurrent Neural Network (RNN) that is used as a solution to overcome the occurrence of exploding gradient or vanishing gradient in RNN when processing long sequential data. The best LSTM model in this study for predicting gold prices is the model with MAPE value 2,70601, which is a model with a training data and testing data comparison 70% : 30% and hyperparameters batch size 1, units 1, AdaGrad optimizer, and learning rate 0,1 with 500 epochs.

1. PENDAHULUAN

Investasi telah mengalami perkembangan yang signifikan dari integrasi regional di negara-negara Asia dalam beberapa dekade terakhir. Begitu pula di Indonesia, saat ini pertumbuhan ekonomi di Indonesia dapat dibilang tumbuh cukup pesat yang diikuti dengan berubahnya pola pikir masyarakat Indonesia di bidang ekonomi dan investasi. Data dari Badan Pusat Statistik menunjukkan bahwa ekonomi Indonesia pada triwulan III 2020 terhadap triwulan II tahun 2020 tumbuh membaik dengan mengalami peningkatan sebesar 5,05% sedangkan terhadap triwulan III tahun 2019 mengalami kontraksi pertumbuhan sebesar 3,49%. Pertumbuhan ekonomi Indonesia tetap dapat tumbuh dengan positif dikarenakan tetap terjaganya konsumsi pemerintah, permintaan domestik, serta investasi (BPS, 2020).

Investasi emas menjadi salah satu investasi yang cukup banyak diminati oleh masyarakat dan dianggap lebih aman karena memiliki risiko yang relatif rendah, bersifat likuid dan cenderung stabil dibandingkan dengan instrumen investasi lainnya, terutama di tengah ketidakpastian kondisi perekonomian global yang disebabkan oleh adanya pandemi COVID-19 seperti saat ini (Napompech et al., 2010). Di akhir Oktober 2020, secara *year to date* harga emas mengalami kenaikan sebesar 23%, berbanding terbalik dengan kinerja IHSG yang mengalami penurunan sebesar 19%. Salah satu penyebab pergerakan harga emas yang melonjak tinggi akhir-akhir ini yaitu adanya persaingan dagang antara Tiongkok-Amerika Serikat. Di sisi lain, penjualan harga saham tetap stagnan bahkan mencapai angka negatif disebabkan oleh persaingan dagang tersebut. Hal ini menyebabkan emas menjadi pilihan yang menarik untuk dijadikan aset investasi dikarenakan nilainya yang dapat bertahan bahkan di saat perkembangan ekonomi sedang melemah.

Pengetahuan mengenai prediksi harga emas ini dapat memberikan informasi kepada masyarakat yang ingin berinvestasi emas sehingga dapat memiliki kesempatan yang lebih tinggi untuk memperoleh keuntungan dan dapat meminimalisir risiko yang diperoleh. Salah satu metode yang cukup banyak digunakan untuk melakukan prediksi terhadap data runtun waktu adalah *Neural Network* (NN). Berbagai macam metode pengembangan dari NN, *Recurrent Neural Network* (RNN) merupakan salah satu bentuk arsitektur NN yang dirancang khusus untuk mengolah data berurutan atau sekuensial (*sequential data*). Namun, RNN memiliki masalah dalam melakukan update gradiennya yang menyebabkan terjadinya *vanishing gradient* atau *exploding gradient.* Permasalahan gradien ini menyebabkan RNN gagal dalam mendeteksi ketergantungan jangka panjang (*long-term dependencies*) yang mengakibatkan berkurangnya akurasi hasil prediksi (Zhao et al., 2017).

Long-Short Term Memory (LSTM) adalah salah satu variasi dari RNN yang melakukan modifikasi pada RNN dengan cara menambahkan *memory cell* yang berguna untuk menyimpan informasi jangka panjang (Manaswi & John, 2018). LSTM digunakan sebagai solusi untuk mengatasi terjadinya *exploding gradient* atau *vanishing gradient* pada RNN ketika mengolah data sekuensial yang panjang. LSTM menghasilkan hasil prediksi yang jauh lebih baik dan dapat mengatasi permasalahan ketergantungan jangka panjang (*long-term dependencies*) yang kompleks (Hochreiter & Schmidhuber, 1997b). Oleh karena itu, digunakan metode *Long-Short Term Memory* ini untuk melakukan prediksi terhadap harga emas dunia. Metode *Long-Short Term Memory* ini dapat memberikan hasil prediksi yang cukup akurat karena mampu mengatasi ketergantungan jangka panjang dari data runtun waktu.

2. TINJAUAN PUSTAKA

Long-Short Term Memory (LSTM) merupakan varian dari *Recurrent Neural Network* (RNN) yang digunakan dalam bidang *Deep Learning* yang mampu untuk mengatasi keterbatasan memori jangka pendek (Hochreiter & Schmidhuber, 1997b). Varian ini dinamakan dengan *Long-Short Term Memory* karena arsitektur LSTM menggunakan struktur yang didasarkan pada proses memori jangka pendek untuk membuat memori jangka panjang. Memori jangka panjang dalam arsitektur LSTM merujuk pada bobot yang dipelajari dan memori jangka pendek merujuk pada nilai *cell state* dalam gates yang berubah setiap langkah melalui waktu t.

LSTM diadaptasi untuk mempelajari ketergantungan jangka panjang (*long-term dependencies*) pada data sekuensial yang sulit dilakukan oleh RNN biasa. Semakin panjang suatu data sekuensial, semakin sulit pula RNN biasa dalam melakukan update gradien yang menyebabkan suatu gradien mengalami *vanish* atau *explode* pada saat proses training. Ketidakmampuan RNN biasa inilah yang umumnya dapat diatasi oleh jaringan LSTM. Jaringan LSTM sangat cocok untuk mengklasifikasikan, memproses, dan membuat prediksi berdasarkan data runtun waktu. Saat melakukan *training* pada RNN menggunakan *backpropagation* (propagasi balik), gradien yang dipropagasi balik menjadi "*vanish*" atau menghilang (cenderung bernilai nol) atau dapat juga menjadi bernilai tak hingga atau "*explode*". RNN yang menggunakan LSTM mampu memecahkan masalah *vanishing gradient* atau *exploding gradient*, karena unit dalam LSTM memungkinkan gradien mengalir dengan tetap dan tidak berubah-ubah (Goodfellow et al., 2014).

Fungsi aktivasi adalah suatu fungsi yang digunakan pada *neural network* untuk mengaktifkan dan menonaktifkan suatu neuron yang berguna untuk membantu *neural network* dalam mempelajari pola kompleks dalam data serta membantu menormalkan output yang dihasilkan oleh neuron (Julpan et al., 2015). Fungsi aktivasi yang digunakan dalam

arsitektur LSTM adalah fungsi aktivasi tanh dan fungsi aktivasi logistik sigmoid. Fungsi aktivasi tanh dan fungsi aktivasi logistik sigmoid digunakan karena sifat terdiferensiasinya kompatibel dengan algoritma *backpropagation* (Mary et al., 2014). Fungsi aktivasi tanh (*hyperbolic tangent*) digunakan untuk membantu dalam mengatur nilai-nilai yang mengalir dalam jaringan. Fungsi aktivasi tanh memadatkan nilai output menjadi selalu berada di antara -1 dan 1 (Chung & Shin, 2018). Persamaan dari fungsi aktivasi tanh dapat dilihat pada Persamaan (1).

$$tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{1}$$

Turunan pertama dari fungsi aktivasi tanh dijabarkan pada Persamaan (2).

$$tanh'(x) = sech^2(x) \tag{2}$$

Fungsi aktivasi logistik sigmoid merupakan salah satu fungsi aktivasi non linier yang terdapat di dalam *gates* yang digunakan untuk meningkatan kemampuan dari model. Fungsi aktivasi logistik sigmoid bekerja dengan memadatkan nilai menjadi berada di antara 0 dan 1. Persamaan dari fungsi aktivasi logistik sigmoid dapat dilihat pada Persamaan (3).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

Turunan pertama dari fungsi aktivasi sigmoid dijabarkan pada Persamaan (4).

$$\sigma'(x) = \sigma(x)\big(1 - \sigma(x)\big) \tag{4}$$

Jaringan LSTM diadaptasi untuk mempelajari ketergantungan jangka panjang dengan menambahkan sel memori yang terdiri dari tiga elemen utama, yaitu *input gate*, *output gate*, dan *forget gate*. Tiga *gates* ini memungkinkan sel untuk mengumpulkan dan memanfaatkan informasi untuk waktu yang lama (Hochreiter & Schmidhuber, 1997b). Algoritma dalam unit LSTM dijelaskan sebagai berikut:

1. Input saat ini (x_t) dan *hidden state* dari *timesteps* sebelumnya (h_{t-1}) bergerak secara bersama-sama menuju *forget gate* yang berisi fungsi aktivasi sigmoid. Nilai yang dihasilkan oleh fungsi aktivasi sigmoid dalam *forget gate* ini yaitu berada di antara 0 dan 1. Output dari *forget gate* (f_t) dapat dilihat pada Persamaan (5).

$$f_t = \sigma\,(U_f x_t + W_f h_{t-1} + b_f) \tag{5}$$

   dengan f_t adalah *output* dari *forget gate*, σ adalah fungsi aktivasi logistik sigmoid, U_f adalah bobot *input node* dalam *forget gate*, x_t adalah *input node*, W_f adalah bobot *hidden state node* dalam *forget gate*, h_{t-1}adalah *hidden state node* dari *timestep* sebelumnya, dan b_f adalah bobot bias dalam *forget gate*.
2. Input saat ini (x_t) dan *hidden state* dari *timesteps* sebelumnya (h_{t-1}) kemudian bergabung menuju *input gate* melewati fungsi aktivasi sigmoid. *Input gate* berfungsi untuk memperbarui *cell state*. Langkah ini untuk memutuskan nilai yang akan diperbarui dengan mengubah nilai menjadi berada di antara 0 dan 1. Output dari *input gate* (i_t) dapat dilihat pada Persamaan (6).

$$i_t = \sigma(U_i x_t + W_i h_{t-1} + b_i) \tag{6}$$

dengan i_t adalah *output* dari *input gate*, σ adalah fungsi aktivasi sigmoid, U_i adalah bobot *input node* dalam *input gate*, x_t adalah *input node*, W_i adalah bobot *hidden state node* dalam *input gate*, h_{t-1}adalah *hidden state node* dari *timestep* sebelumnya, dan b_i adalah bobot bias dalam *input gate.*

3. Langkah selanjutnya yaitu meneruskan gabungan input saat ini (x_t) dan *hidden state* dari *timesteps* sebelumnya (h_{t-1}) menuju fungsi aktivasi tanh yang memadatkan nilai menjadi berada di antara -1 dan 1 agar dapat membantu mengatur pergerakan dalam jaringan. Output dari fungsi aktivasi tanh ini disebut dengan *cell state candidate* ($\tilde{C}_t$). Output dari proses ini dapat dilihat pada Persamaan (7).

$$\tilde{C}_t = \tanh\left(U_C x_t + W_C h_{t-1} + b_C\right) \quad (7)$$

dengan $\tilde{C}_t$ adalah *cell state candidate*, tanh adalah fungsi aktivasi tanh, U_C adalah bobot *input node* dalam *candidate memory*, x_t adalah *input node*, W_C adalah bobot *hidden state node* dalam *candidate memory*, h_{t-1}adalah *hidden state node* dari *timestep* sebelumnya, dan b_c adalah bobot bias dalam *candidate memory*.

4. Informasi-informasi tersebut telah melewati *forget gate* dan *input gate*, maka langkah selanjutnya yaitu melakukan operasi matematika dalam *cell state*. *Cell state* berperan sebagai penghubung yang menyalurkan informasi ke seluruh sel. Pertama, *cell state* sebelumnya (C_{t-1}) dikalikan dengan output dari *forget gate* (f_t). Proses perkalian ini memberikan kemungkinan untuk menghapus informasi-informasi dalam *cell state* sebelumnya (C_{t-1}) apabila memiliki nilai yang mendekati 0. Selanjutnya yaitu menjumlahkan hasil tersebut dengan perkalian antara output dari *input gate* (i_t) dengan *cell state candidate* ($\tilde{C}_t$) sehingga menghasilkan nilai *cell state* baru (C_t). Output dari proses yang menghasilkan *cell state* baru (C_t) ini dapat dilihat pada Persamaan (8).

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t \quad (8)$$

dengan C_t adalah *cell state* baru, f_t adalah *output* dari *forget gate*, C_{t-1} adalah *cell state* dari *timestep* sebelumnya, i_t adalah *output* dari *input gate*, dan $\tilde{C}_t$ adalah *cell state candidate*.

5. Langkah selanjutnya yaitu meneruskan gabungan input saat ini (x_t) dan *hidden state* dari *timesteps* sebelumnya (h_{t-1}) menuju *output gate* yang di dalamnya terdapat suatu fungsi aktivasi sigmoid. *Output gate* bekerja dengan cara memutuskan keadaan *hidden state* saat ini (h_t). *Hidden state* dapat digunakan untuk membuat prediksi mengingat bahwa *hidden state* juga berisi informasi mengenai *input* sebelumnya (x_{t-1}). *Output gate* memiliki suatu fungsi aktivasi sigmoid. Output dari *output gate* (o_t) dapat dilihat pada Persamaan (9).

$$o_t = \sigma(U_o x_t + W_o h_{t-1} + b_o) \quad (9)$$

dengan o_t adalah *output* dari *output gate*, σ adalah fungsi aktivasi sigmoid, U_o adalah bobot *input node* dalam *output gate*, x_t adalah *input node*, W_o adalah bobot *hidden state node* dalam *output gate*, h_{t-1}adalah *hidden state node* dari *timestep* sebelumnya, dan b_o adalah bobot bias dalam *output gate.*

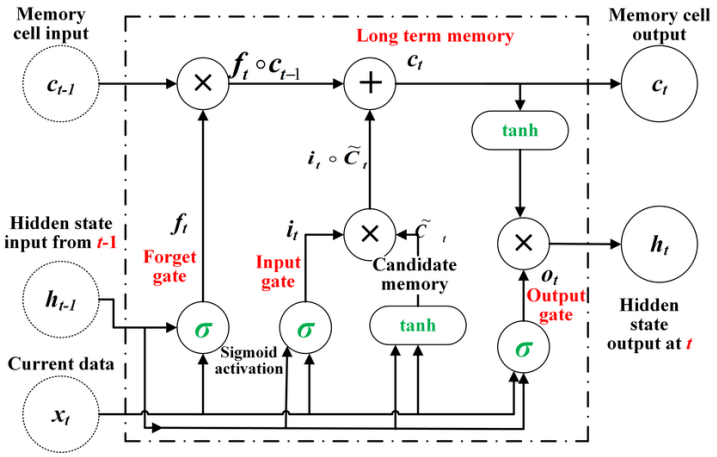
6. Langkah selanjutnya yaitu meneruskan *cell state* baru (C_t) menuju fungsi aktivasi tanh yang kemudian dilakukan operasi perkalian dengan output dari *output gate* (o_t) yang

menghasilkan *hidden state* baru (h_t). *Hidden state* baru (h_t) dan *cell state* baru (C_t) kemudian dibawa menuju *timesteps* selanjutnya ($t+1$). Persamaan (10) menunjukkan output hasil operasi yang mengasilkan *hidden state* baru (h_t):

$$h_t = o_t \tanh(C_t) \tag{10}$$

dengan h_t adalah *hidden state* baru, o_t adalah *output* dari *output gate*, tanh adalah fungsi aktivasi tanh, dan C_t adalah *cell state* baru.

Arsitektur LSTM dapat dilihat pada Gambar 1.



Gambar 1. Arsitektur LSTM

Feature Scaling atau normalisasi data merupakan salah satu metode yang digunakan untuk menormalisasikan range dari variabel independen atau fitur dari data. Penerapan *Feature Scaling* menyebabkan penurunan gradien mengalami konvergensi yang lebih cepat dibandingkan tanpa menerapkan *Feature Scaling* (Ioffe & Szegedy, 2015). *Min-Max Normalization* adalah salah satu metode *Feature Scaling* paling sederhana yang umum digunakan yang bekerja dengan cara mengubah skala data menjadi berada di antara interval tertentu. Penelitian ini, menggunakan interval (-1,1) dikarenakan fungsi aktivasi pada *output gate* yaitu fungsi aktivasi tanh yang memadatkan nilai output menjadi selalu berada di antara -1 dan 1. persamaan untuk *min-max* (-1,1) dapat dilihat pada Persamaan (11).

$$x_i' = \frac{2(x_i - min(\boldsymbol{x}))}{max(\boldsymbol{x}) - min(\boldsymbol{x})} - 1 \tag{11}$$

dengan x_i' adalah data hasil normalisasi, x adalah data asli, min($\boldsymbol{x}$) adalah nilai minimum dari data asli, dan max($\boldsymbol{x}$) adalah nilai maksimum dari data asli. Tahap untuk mengubah kembali data *output* menjadi bentuk asli disebut dengan denormalisasi data. Persamaan denormalisasi dapat dilihat pada Persamaan (12).

$$x_i'' = \frac{(x_i'+1)(max(\boldsymbol{x}) - min(\boldsymbol{x}))}{2} + min(\boldsymbol{x}) \tag{12}$$

dengan x_i'' merupakan data hasil denormalisasi.

Mean Absolute Percentage Error (MAPE) merupakan suatu ukuran evaluasi yang umum digunakan untuk mengetahui ketepatan model hasil prediksi. MAPE adalah ukuran ketetapan relatif yang digunakan untuk mengetahui persentase penyimpangan hasil prediksi. Rumus umum dari MAPE terdapat pada Persamaan (13) berikut (Makridakis et al., 1999):

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{x_i - \hat{x}_i}{x_i}\right| \times 100\% \quad (13)$$

dengan n adalah banyaknya data, x_i adalah data asli, dan $\hat{x}_i$ adalah data hasil prediksi.

3. METODE PENELITIAN

Jenis data yang digunakan dalam penelitian ini adalah data sekunder. Data yang digunakan dalam penelitian ini yaitu data kuantitatif berupa data harga emas dunia yang bersumber dari website *www.lbma.org.uk.* Data emas yang digunakan merupakan data harga emas harian per satuan *ounce* yang dicatat pada waktu 10.30 AM BST. Periode data yang digunakan yaitu sejak 1 Januari 2018 hingga 30 Juni 2021 sebanyak 884 data.

Analisis data pada penelitian ini dilakukan dengan menggunakan bantuan *software* RStudio. Adapun tahap-tahapan analisis yang digunakan dalam penelitian ini dijabarkan sebagai berikut:

1. Melakukan input data
2. Melakukan uji statistik deskriptif terhadap data
3. Membuat *lagged dataset*
4. Membagi data menjadi data *training* dan data *testing*
5. Melakukan normalisasi data atau *feature scaling*
6. Membuat model LSTM
7. Melakukan *training model* LSTM
8. Evaluasi model LSTM
9. Pemilihan model LSTM terbaik
10. Melakukan prediksi
11. Melakukan denormalisasi data hasil prediksi
12. Evaluasi hasil prediksi

4. HASIL DAN PEMBAHASAN

Tahap data *pre-processing* terdiri dari 3 tahap, yaitu membuat *lagged dataset*, membagi data menjadi data *training* dan data *testing*, dan melakukan normalisasi data atau *feature scaling*. *Lagged dataset* merupakan suatu dataset yang terdiri dari 2 kolom data yang dibuat dengan menempatkan variabel data periode sebelumnya (x-1) pada kolom pertama dan variabel data asli (*x*) pada kolom kedua. *Lagged dataset* dibuat bertujuan untuk membuat prediksi, karena untuk membuat suatu prediksi pada periode yang akan datang dibutuhkan data pada periode sebelumnya.

Pembagian data *training* dan data *testing* yang optimal diperoleh dengan cara *trial and error.* Analisis data ini menggunakan 3 macam pembagian data *training* dan data *testing*, yaitu 70% data *training* : 30% data *testing,* 80% data *training* : 20% data *testing*, dan 90% data *training* : 10% data *testing*. Ketiga macam pembagian data *training* dan data *testing* tersebut kemudian dicari proporsi yang optimal yang menghasilkan hasil prediksi terbaik.

Feature Scaling atau normalisasi data pada penelitian ini menggunakan metode *Min-Max Normalization. Feature Scaling* ini bekerja dengan cara mengubah skala data menjadi berada di antara interval (-1,1) yang bertujuan untuk mempercepat konvergensi penurunan gradien. Hasil dari tahap sebelumnya menunjukkan bahwa masing-masing data *training* dan data *testing* memiliki 2 kolom, dengan kolom pertama yaitu variabel data periode sebelumnya (*x-1*) dan kolom kedua yaitu variabel data asli (*x*). Masing-masing data *training* dan data *testing* pada kolom pertama diberi label *x* dan pada kolom kedua diberi label *y*.

Tahap selanjutnya yaitu membuat model LSTM. Tahap ini dilakukan dengan cara *trial and error* pada setiap *hyperparameter* yang mungkin untuk mendapatkan model terbaik.

Untuk *epochs*, ditentukan nilai sebesar 500 *epochs* dan untuk *batch size* ditentukan sebesar 1 untuk semua model LSTM. Plan untuk *trial and error* tersebut dijabarkan pada Tabel 1.

Tabel 1. Plan Model LSTM

Trial	Training	Testing	Units	Optimizer	Learning Rate
1	70%	30%	1	Adam	0,0001
2	70%	30%	1	AdaGrad	0,1
3	70%	30%	1	SGD	0,01
4	70%	30%	100	Adam	0,0001
5	70%	30%	100	AdaGrad	0,1
6	70%	30%	100	SGD	0,01
7	80%	20%	1	Adam	0,0001
8	80%	20%	1	AdaGrad	0,1
9	80%	20%	1	SGD	0,01
10	80%	20%	100	Adam	0,0001
11	80%	20%	100	AdaGrad	0,1
12	80%	20%	100	SGD	0,01
13	90%	10%	1	Adam	0,0001
14	90%	10%	1	AdaGrad	0,1
15	90%	10%	1	SGD	0,01
16	90%	10%	100	Adam	0,0001
17	90%	10%	100	AdaGrad	0,1
18	90%	10%	100	SGD	0,01

Evaluasi model LSTM dilakukan untuk mengetahui seberapa baik model yang telah dilakukan proses *training* dengan *hyperparameter* tertentu. Evaluasi model LSTM dilakukan pada data *training* dan data *testing* yang menghasilkan nilai *loss* dan nilai MAPE pada masing-masing trial. Perbandingan nilai *loss* dan nilai MAPE pada masing-masing trial dijabarkan pada Tabel 2.

Tabel 2. Perbandingan Nilai *Loss* dan Nilai MAPE pada 18 Trial

Trial	*Loss Training*	MAPE *Training*	*Loss Testing*	MAPE *Testing*
1	0,0003916921	2,3477556705	0,0021466320	3,1832411290
2	0,0002638499	2,1282858849	0,0008290517	1,9876978397
3	0,0003580689	2,3444144726	0,0019342270	3,0840797420
4	0,0003580689	3,7426896095	0,0008471418	2,0732665062
5	0,0011013410	5,9485054020	0,0006715939	1,7836974859
6	0,0006769093	3,9863119125	0,0010061510	2,1757371430
7	0,0002766496	3,0481185913	0,0004430370	2,1226236820
8	0,0001971267	2,6878662109	0,0002837275	1,6546676159
9	0,0003023072	3,0943608284	0,0004615560	2,1809103490
10	0,0002787669	3,2529296875	0,0003962143	2,1152575016
11	0,0003496896	4,1730318069	0,0003885107	1,9828039408
12	0,0004079836	4,4448876380	0,0004896906	2,2459070683
13	0,0003012501	2,9750924110	0,0003351777	1,9877322912
14	0,0002150694	2,5241103172	0,0002046868	1,5335800648
15	0,0003171033	2,9204001427	0,0003284727	1,9111307620
16	0,0003565372	3,8498718739	0,0003334862	2,1283261776
17	0,0004740906	4,8556990623	0,0003299902	2,0304276943
18	0,0005034564	4,9514861107	0,0004487787	2,3420693874

Tabel 2 menunjukkan bahwa model yang memiliki nilai *loss* terkecil pada data *training* yaitu model Trial 8 dengan nilai *loss* sebesar 0,0001971267 serta nilai MAPE terkecil pada data *training* yaitu model Trial 2 dengan nilai MAPE sebesar 2,1282858849. Model yang memiliki nilai *loss* terkecil pada data *testing* yaitu model Trial 14 dengan nilai *loss* sebesar

0,0002046868 serta nilai MAPE terkecil yaitu model Trial 14 dengan nilai MAPE sebesar 1,5335800648. Kesimpulan yang didapat yaitu model yang digunakan untuk melakukan prediksi adalah model Trial 2, Trial 8, dan Trial 14.

Tahap selanjutnya yaitu melakukan prediksi pada model Trial 2, Trial 8, dan Trial 14 selama 30 hari ke depan menggunakan 30 data *testing* terakhir. Hasil prediksi yang diperoleh yaitu masih dalam bentuk data yang ternormalisasi (-1,1). Oleh karena itu, data hasil prediksi tersebut diubah terlebih dahulu menjadi ke bentuk data aslinya menggunakan rumus denormalisasi data seperti pada Persamaan (12). Hasil prediksi yang telah diperoleh tersebut kemudian dievaluasi menggunakan ukuran evaluasi. Ukuran evaluasi yang digunakan dalam mengevaluasi hasil prediksi yaitu MAPE. Semakin kecil nilai MAPE hasil prediksi suatu model, maka semakin baik model tersebut. Tabel 3 menunjukkan perbandingan nilai *accuracy* yang dihasilkan oleh ketiga model LSTM terbaik.

Tabel 3. Perbandingan Nilai *Accuracy* pada 3 Model LSTM Terbaik

Trial	ME	RMSE	MAE	MPE	MAPE
2	38,73432	52,58527	49,99022	2,074345	2,70601
8	55,48158	68,4305	63,00506	2,948469	3,368498
14	51,14533	64,56249	59,95109	2,722673	3,215041

Hasil evaluasi terhadap 3 model LSTM terbaik, dapat disimpulkan bahwa model Trial 2 memiliki nilai MAPE terkecil dibandingkan dengan model lain yaitu sebesar 2,70601. Oleh karena itu, dapat disimpulkan bahwa model LSTM terbaik yaitu model Trial 2 yang merupakan model dengan perbandingan data *training* 70% : data *testing* 30%, serta memiliki *hyperparameter* yaitu *batch size* sebanyak 1, *units* sebanyak 1, jenis *optimizer* AdaGrad, dan *learning rate* sebesar 0,1 dengan proses *training* yang dilakukan sebanyak 500 *epochs*. Hasil prediksi dari model Trial 2 ini yang nantinya digunakan sebagai acuan dalam mengamati harga emas dunia di masa yang akan datang.

5. KESIMPULAN

Kesimpulan yang diperoleh dari hasil analisis dan pembahasan mengenai prediksi harga emas menggunakan metode *Long-Short Term Memory*, yaitu analisis data ini menggunakan *trial and error* untuk menentukan proporsi data *training* dan data *testing* serta untuk menentukan *hyperparameter*, menghasilkan 18 model Trial LSTM yang mungkin. Pemilihan model terbaik setelah dilakukan proses *training* pada 18 model LSTM, diperoleh kesimpulan model LSTM terbaik yaitu model Trial 2 dengan nilai MAPE sebesar 2,70601, yang merupakan yang merupakan model dengan perbandingan data *training* 70% : data *testing* 30%, serta memiliki *hyperparameter* yaitu *batch size* sebanyak 1, *units* sebanyak 1, jenis *optimizer* AdaGrad, dan *learning rate* sebesar 0,1 dengan proses *training* yang dilakukan sebanyak 500 *epochs*.

DAFTAR PUSTAKA

BPS. (2020). Pertumbuhan Ekonomi Indonesia Triwulan III-2020. *Berita Resmi Statistik*, *No. 15/02/*(15), 1–12.

Chung, H., & Shin, K. S. (2018). Genetic algorithm-optimized long short-term memory network for stock market prediction. *Sustainability (Switzerland)*, *10*(10), 1–18. https://doi.org/10.3390/su10103765.

Goodfellow, I., Bengio, Y., & Courville, A. (2014). Deep Learning. In *MIT Press*. MIT Press. http://files.sig2d.org/sig2d14.pdf#page=5.

Hochreiter, S., & Schmidhuber, J. (1997b). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.173.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International Conference on Machine Learning*, 448–456.

Julpan, Nababan, E. B., & Zarlis, M. (2015). Analisis Fungsi Aktivasi Sigmoid Biner Dan Sigmoid Bipolar Dalam Algoritma Backpropagation Pada Prediksi Kemampuan Siswa. *Jurnal Teknovasi*, *02*(1), 103–116.

Makridakis, S., Wheelwright, S. C., & McGee, V. E. (1999). Metode dan aplikasi peramalan. *Jakarta: Erlangga.*

Manaswi, N. K., & John, S. (2018). *Deep Learning with Applications Using Python*. Springer.

Mary, P. M., Pushpa, R., & Manimala, K. (2014). *Implementation of Hyperbolic Tangent Activation Function in VLSI*. *2*(March), 225–22.

Napompech, K., Tanpipat, A., & Nidpa, U. (2010). Factors Influencing Gold Consumption for Savings and Investments by People in the Bangkok Metropolitan Area. *International Journal of Arts and Sciences*, *3*(7), 508–520. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.301.2007&rep=rep1&type=pdf

Zhao, Z., Chen, W., Wu, X., Chen, P. C. Y., & Liu, J. (2017). LSTM network: a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems*, *11*(2), 68–75.