ENTHUSIASTIC
INTERNATIONAL JOURNAL OF APPLIED STATISTICS AND DATA SCIENCE
Volume 4, Issue 2, October 2024, pp. 132-142

Indonesian Inflation Forecasting with Recurrent Neural Network Long Short-Term Memory (RNN-LSTM)

Hermansah [a,1,*], Muhammad Muhajir [b,2], Paulo Canas Rodrigues [b,c,3]

[a] Department of Mathematics Education, University of Riau Kepulauan, Batam, Indonesia
[b] Department of Statistics, Universitas Islam Indonesia, Yogyakarta, Indonesia
[c] Department of Statistics, Federal University of Bahia, Salvador, Brazil
[1] hermansah@fkip.unrika.ac.id*; [2] mmuhajir@uii.ac.id; [3] paulocanas@gmail.com
* Corresponding author

ARTICLE INFO

Keywords
forecasting
inflation
frequency approach
SGD
RNN-LSTM

ABSTRACT

This study forecasted inflation in Indonesia using the Recurrent Neural Network Long Short-Term Memory (RNN-LSTM) model, ideal for nonlinear, complex time series data. It evaluated the effects of different activation functions, such as Logistic, Gompertz, and Hyperbolic Tangent (tanh); and weight update methods, such as Stochastic Gradient Descent (SGD) and Adaptive Gradient (AdaGrad) on RNN-LSTM performance. Monthly inflation data from January 2005 to December 2023 underwent preprocessing, including normalization and autoregressive lag-based input selection. Model accuracy was assessed with Root Mean Squared Error (RMSE) and Symmetric Mean Absolute Percentage Error (SMAPE). The findings indicated that the RNN-LSTM model with the logistic activation function and SGD optimization achieved the highest accuracy, outperforming traditional models such as Exponential Smoothing (ETS), Autoregressive Integrated Moving Average (ARIMA), Feedforward Neural Network (FFNN), and Recurrent Neural Network (RNN). Additionally, optimal learning rate and epoch values were identified, enhancing model stability and precision. In conclusion, the study confirms that the RNN-LSTM model is effective for inflation forecasting when optimized with specific activation functions and optimization methods. It recommends further exploration of neuron configurations and alternative models, such as the Gated Recurrent Unit (GRU), to improve forecast accuracy.

1. Introduction

Forecasting is an attempt to predict future events based on past data. The goal is to produce accurate predictions. Good forecasting must have a high accuracy level and the ability to imitate the behavior of historical time series data [1]. There are two types of time series forecasting methods: statistical methods and machine learning or computational intelligence methods [2]. The statistical methods commonly used are Autoregressive Integrated Moving Average (ARIMA) and exponential smoothing. However, this method requires data that is linear. Meanwhile, for nonlinear data, the deep learning method has better capabilities.

p-ISSN 2798-253X
e-ISSN 2798-3153

One algorithm that is included in the deep learning family is Recurrent Neural Network (RNN). RNN is part of the Artificial Neural Network (ANN), where the output from the hidden layer is reused as input in the next process [3]. The weakness of the RNN method is that it cannot accommodate long-term memory, so it is challenging to remember previous information or what is called a vanishing gradient. The vanishing gradient problem is solved using the Long-Short Term Memory (LSTM) method. LSTM is an algorithm based on time series and is effective in making predictions and can store information for a long time using three types of gates, namely input gate, forget gate, and output gate [4].

Several studies have used LSTM-based RNN methods for various purposes. A study used LSTM to predict bitcoin prices and got the best results with an average accuracy of 95.36% on training data and 93.50% on testing data [5]. Meanwhile, another study used LSTM to predict PM_{10} concentrations in Lima, Peru, with accurate results for moderate pollution, but less precise for high pollution [6]. The LSTM model proved to be better than MLP, especially in predicting critical pollution episodes. A Recurrent Neural Network was also used with LSTM and Gated Recurrent Unit (GRU) to predict retail sales and found that the LSTM algorithm had the best performance [2]. Another research that has been carried out using inflation data is the analysis of inflation rate predictions in Samarinda City, East Kalimantan, using the backpropagation neural network method [7]. ANN was used to predict the monthly inflation rate in Indonesia method and obtained a low MSE value [8]. The monthly inflation rate in Indonesia was also predicted using a non-linear autoregressive neural network method with exogenous input and found that the proposed method outperformed other models [9].

In addition, previous research results have shown that the choice of activation function and weight update in the RNN model has a significant impact on the prediction model's performance [10]. RNNs with activation functions such as logistic [11], Gompertz [12] and tanh [13] have proven to be more accurate than ReLU activation functions. In the case of prediction, improvising commonly used weight update strategies such as Stochastic Gradient Descent (SGD) [14] and Adaptive Gradient (AdaGrad) [15] can be used to improve the performance of RNN models. In 2021, a study discussed a comparison of various activation functions in artificial neural networks, including Sigmoid, tanh, ReLU, and others [16], guiding the choice of the most appropriate activation function for real-world applications. In 2022, a study tested 26 alternative activation functions in LSTM for classification. The finding showed that several alternative functions, such as modified Elliott and softsign, yielded higher accuracy than tanh [17]. Later in 2023, a study proposed the use of the log-sigmoid activation function in LSTM for time series data classification, showing improved accuracy with various optimization algorithms [18]. These studies highlight the importance of selecting activation functions in improving the performance of artificial neural networks. However, there has been no research that specifically investigates the impact of selecting activation functions and updating weights in the RNN-LSTM model on time series data applications, especially in the context of predicting the inflation rate in Indonesia.

Based on the research above, this research was intended to fill this knowledge gap by conducting a comparative study of weight update methods and activation functions in the RNN-LSTM model. This research considered two weight updates, namely Stochastic Gradient Descent (SGD) and Adaptive Gradient (AdaGrad), to compare the performance of their prediction models. The three activation functions are logistic, Gompertz, and tanh. This research aimed to apply the RNN-LSTM model to forecast inflation in Indonesia. Inflation is a condition where prices generally continue to increase on an ongoing basis, attributable to various factors. In other words, inflation can also be interpreted as a continuous process of decreasing the value of the currency [19]. In this research, neurons in the input layer for the RNN-LSTM model were proposed based on autoregressive lag with a frequency approach. The frequency attribute states the quantity of data in a certain period, usually defined per year, such as monthly data (frequency = 12), quarterly data (frequency = 4), quarterly data (frequency = 3), semiannual data (frequency = 2), and data annually (frequency = 1). If the frequency of the time series data is m, then m sequential autoregressive lags starting from

p-ISSN 2798-253X
e-ISSN 2798-3153

autoregressive lag one is used. At the same time, an evaluation was also carried out on the influence of learning rate, number of epoch iterations, optimization method, and activation function. Next, the RNN-LSTM model was compared with several models, namely the Exponential Smoothing (ETS) model described in [20], the ARIMA model described in [21], the Feed Forward Neural Network (FFNN) model described in [22], and the RNN model described in [23]. Meanwhile, the comparison of forecasting accuracy was measured using Symmetric Mean Absolute Percentage Error (SMAPE).

2. Method

2.1. Recurrent Neural Network (RNN) Model

RNN is a type of ANN that can see hidden relationships in data in applications, such as voice recognition, natural language processing, and time series prediction [3]. In modeling sequence problems, RNN is very good to use because it can combine input information as well as trace information obtained previously through repeated connections. RNN is said to be a recurrent neural network because the output from the previous hidden layer is reused as input data in subsequent processing [24]. RNN has a characteristic: in making predictions, it does not only use input at one time but requires input from previous input so that the input is interconnected and can provide information to the hidden layer [25].

RNN consists of an input layer, one or more hidden layers, and an output layer [26]. The RNN model has one-way information from the input layer to the hidden layer and one-way information synthesis from the temporarily hidden layer to the current hidden layer. The architectural model of RNN is as follows.

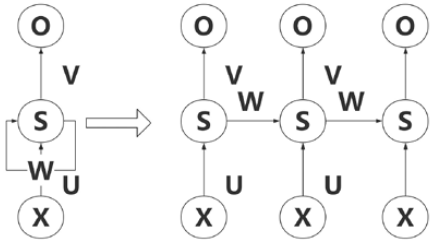

Fig. 1 Recurrent neural network architecture.

As shown in Fig. 1, the mapping of S_t and O_t can be represented as follows.

$$S_t = f(U \times X_t + W \times S_{t-1}) \quad (1)$$

$$O_t = g(V \times S_t) \quad (2)$$

where S_t is network memory at the time t; U, V, and W are the weight matrices in each layer; X_t and O_t are input and output at the time t; $f(\dots)$ and $g(\dots)$ are nonlinear functions.

2.2. Long-Short Term Memory (LSTM) Model

LSTM was first proposed by Sepp Hochreiter and Juergen Schmidhuber in 1997. It is a development of the RNN model to overcome the vanishing gradient problem when processing long-term sequential data [27]. The RNN architecture has limitations in handling long-term dependencies because it does not store previous information properly, the old, stored memory will be increasingly useless and overwritten by new memory, causing a vanishing gradient problem. LSTM can reduce this problem by using memory cells to store information over time intervals and using gate units to regulate the flow of information in and out of cells so that it can better overcome long-term dependencies.

The LSTM model consists of a series of unique memory cells and replaces neurons in the hidden layer of the RNN [28]. The LSTM model filters information through a gate structure to maintain and update the state of the memory cell. One memory cell consists of three gates: forget gate, output gate, and input gate. The LSTM structure is presented in Fig. 2.

p-ISSN 2798-253X
e-ISSN 2798-3153

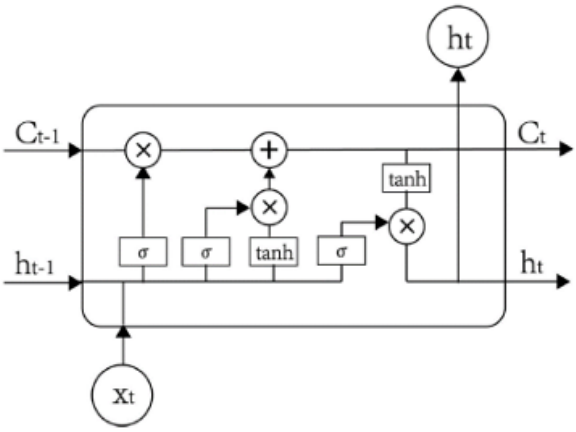

Fig. 2 Long-short term memory architecture.

2.2.1. Forget Gate

Forget gate is a gate in the LSTM model that determines whether or not the information in the cell state will be deleted. This calculation uses the previous output data h_{t-1} and x_t as input data. The forget gate equation is formulated in (3).

$$f_t = \sigma\left(W_f * [h_{t-1}, x_t] + b_f\right) \quad (3)$$

where f_t is a forget gate; σ is a sigmoid function; W_f is the weight value for the forget gate; h_{t-1} is the output value before the tth order; x_t is the input value at the time t; and b_f is the bias value at the forget gate.

The weight value is formulated in (4).

$$W = \left(-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}\right) \quad (4)$$

where W is the weight and d is the amount of data.

2.2.2. Input Gate

The input gate is a gate that uses two types of activation functions (sigmoid and tanh) and aims to select the part that will be updated. The gate input is formulated in (5).

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \quad (5)$$

where i_t input gate; σ sigmoid function; W_i weight value for input gate; h_{t-1} output value before tth order; x_t input value at tth order, and b_i the bias value at the gate input.

The new candidate equation is formulated in (6).

$$\tilde{C}_t = tanh(W_C * [h_{t-1}, x_t] + b_C) \quad (6)$$

where $\tilde{C}_t$ is the new value added to the cell state; $tanh$ is a hyperbolic tangent function; W_C is the weight value for the cell state; h_{t-1} is the output value before the tth order; x_t is the input value at order t; and b_C is the bias value in the cell state.

After that, the old cell state will be updated to become a new cell state by multiplying the old state by the forget gate (f_t) to delete the information specified in the forget gate, then the value will be added with $i_t * \tilde{C}_t$ which is the new value to update state, thus producing the cell state equation as follows.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (7)$$

where C_t cell state; f_t forget gate; C_{t-1} cell state before tth order; i_t input gate, and $\tilde{C}_t$ new values that can be added to the cell state.

2.2.3. Output Gate

The output gate functions to determine what information will be generated based on the input and memory blocks. The output from the cell state is entered into the tanh layer and then multiplied by the sigmoid gate so that the resulting output matches what has been previously decided. The gate output equation is formulated as in (8).

$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o) \quad (8)$$

where o_t output gate; σ sigmoid function; W_o weight value for output gate; h_{t-1} output value before tth order; x_t input value at tth order; and b_o bias value at the gate output.

After getting the value from the gate output, the cell state is placed through tanh. Then, it is multiplied by the gate output and sigmoid layer. The tth order output equation is formulated as follows.

$$h_t = o_t * tanh(C_t) \quad (9)$$

where h_t is the tth order output value; o_t output gate; $tanh$ hyperbolic tangent function, and C_t cell state.

2.3. LSTM Modeling Algorithm

The analysis in this paper displays the best RNN-LSTM model for predicting inflation in Indonesia. The following are the analysis stages in RNN-LSTM modeling.

a. Determining neurons in the input layer (autoregressive lag) based on data frequency x, assuming that x was the time series data to be predicted. If the data frequency x is equal to m, then successive autoregressive lags from 1 to m are considered autoregressive lags. For example, for monthly data it is considered $1: 12$ as an autoregressive lag.
b. Preprocessing data, namely by scaling data or normalizing data. Data normalization was carried out using (10):

$$y = \frac{x - min(x)}{max(x) - min(x)} \quad (10)$$

where y is the value of data normalization; x is the data to be predicted; $min(x)$ is the minimum predicted data value, and $max(x)$ is the maximum predicted data value.

c. Splitting data by dividing data into training data and testing data. The scenario for splitting data in this research was that the data was divided into training data (from January 2005 to December 2022) and testing data (from January 2023 to December 2023).
d. Building the RNN-LSTM model by initializing the required parameters, namely neurons, learning rate, epoch, optimization method, activation function, and hypertuning other parameters as in [29].
e. Perform data predictions.
f. Carrying out the data denormalization process. Data denormalization was carried out using the (11):

$$x^* = y * [max(x) - min(x)] + min(x) \quad (11)$$

where x^* is the value of data denormalization; y is the value of data normalization; $min(x)$ is the minimum predicted data value, and $max(x)$ is the maximum predicted data value.

g. Comparing predicted and actual data using RMSE and SMAPE values. RMSE is a measure used to assess how well a predictive model is at predicting value. RMSE measures the difference between the value predicted by the model and the actual value, giving more weight to larger errors. A lower RMSE value indicates a model that is better at making predictions, because it shows that the average prediction error is smaller. A higher RMSE value indicates that the model is less accurate in its predictions, as higher values indicate larger errors. RMSE also has the same units as the measured data, making interpretation easier. However, RMSE is sensitive to outliers,

p-ISSN 2798-253X
e-ISSN 2798-3153

so it should be considered along with other metrics for a more comprehensive model evaluation. RMSE is calculated using (12):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(A_i - F_i)^2} \quad (12)$$

SMAPE is used to measure accuracy in forecasting models by comparing predicted values with actual values. SMAPE overcomes some of the disadvantages of MAPE by avoiding problems when the actual value approaches zero. SMAPE provides results in percentages, and a lower SMAPE value indicates a more accurate model. The SMAPE value is formulated as in (13).

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|A_i - F_i|}{\frac{|A_i|+|F_i|}{2}} \times 100\% \quad (13)$$

where A_i is the actual value in the period i; F_i is the predicted value in period i; n is the number of periods (data) used in the calculation.

h. Forecasting inflation values in Indonesia in January 2023 - December 2023.

3. Results and Discussion

This research conducted a case study using inflation rate data in Indonesia. The data observed was monthly data from January 2005 to December 2023, as shown in Fig. 3. Training data was carried out on the first 216 data (from January 2005 to December 2022), and the last 12 data was used as testing data (from January 2023 to December 2023). This data can be found and accessed on the Bank Indonesia website.

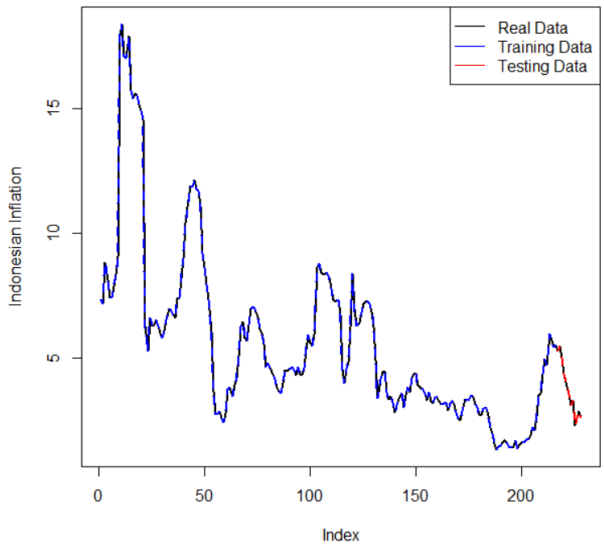

Fig. 3 Time series plot of Indonesian inflation.

Next, testing consisted of testing the learning rate, the number of epoch iterations, the optimization model by comparing two methods, and the activation function by comparing three functions.

3.1. Learning Rate Testing

The neurons in the input layer in this research was proposed to be based on an autoregressive lag with a frequency approach, where time series data has time and frequency attributes. The time attribute states the unit of time from each observation point, while the frequency attribute states the quantity of data in a certain period, usually defined per year, such as monthly data (frequency = 12), quarterly data (frequency = 4), quarterly data (frequency = 3), semiannual data (frequency = 2), and annual data (frequency = 1). Therefore, the number of neurons in the input layer in this study was 12 neurons (12 months as an autoregressive lag).

Next, experiments were carried out with learning rates of 0.001, 0.002, 0.050, 0.100, 0.200, 0.500, and 0.900. Apart from that, the optimization method used was SGD and 500 epochs. The activation function used was the logistic function. Meanwhile, other hypertuning parameters were determined in the same way as in [29]. This experiment was conducted to determine the effect of learning rate on the learning process and testing data. The results obtained from the learning rate experiment can be seen in Table 1.

Table 1. Impact of Different Learning Rates on SMAPE and RMSE for Training and Testing Data

Learning Rate	Training Data		Testing Data	
	SMAPE	***RMSE***	***SMAPE***	***RMSE***
0.001	0.32809	0.16419	0.27956	0.12744
0.002	0.27832	0.09724	0.26729	0.10531
0.050	0.17027	0.06290	0.16393	0.10982
0.100	0.15518	0.06183	0.16120	0.11569
0.200	0.15708	0.05983	0.22301	0.14623
0.500	0.19121	0.08220	0.26478	0.16909
0.900	0.24173	0.51087	0.15048	0.38976

Table 1 shows the performance of the RNN-LSTM model with varying learning rates, evaluated using SMAPE and RMSE for both training and testing data. The optimal learning rate was 0.050, which provided the best balance between accuracy and error, achieving a SMAPE of 16.39% and RMSE of 0.10982 for testing data. At higher learning rates (0.200 and 0.500), the model exhibited overfitting, showing higher accuracy on training data but poorer performance on testing data. Conversely, very small learning rates (0.001 and 0.002) resulted in slower learning and less efficient convergence. Therefore, the results highlight that selecting the appropriate learning rate, such as 0.050, is critical for ensuring both accuracy and stability in model performance.

3.2. Epoch Iteration Testing

The experiment was carried out with a learning rate of 0.05, SGD optimization method, logistic activation function, and hypertuning other parameters, as in [29]. Meanwhile, autoregressive lag was determined in the same way as learning rate testing. This was done to assess the effect of iteration size in epochs on the learning process and data testing. The results obtained from the epoch value experiment can be seen in Table 2.

Table 2. Influence of Epoch Values on Training and Testing Data Performance Using Learning Rate

Epoch	Training Data		Testing Data	
	SMAPE	***RMSE***	***SMAPE***	***RMSE***
50	0.30334	0.13552	0.40858	0.19479
100	0.22936	0.09501	0.30254	0.14728
150	0.19520	0.07952	0.20424	0.10063
200	0.20143	0.08010	0.27102	0.13865
250	0.18728	0.07276	0.24238	0.13177
300	0.17536	0.06903	0.20006	0.11993
350	0.18293	0.07343	0.22039	0.12858
400	0.16622	0.06631	0.13611	0.08055
450	0.17058	0.06629	0.16799	0.12047
500	0.17020	0.06657	0.19185	0.10862

Table 2 shows the effect of the number of epochs on the performance of the RNN-LSTM model using the SMAPE and RMSE metrics. The best results were obtained at the 400th epoch with SMAPE 0.13611 and RMSE 0.08055 on the testing data. At smaller number of epochs, such as 50 and 100, the SMAPE and RMSE values showed worse performance, while increasing the number of epochs up to 400 resulted in a significant increase in accuracy. However, after 400 epochs, further increases tended to give more volatile results and sometimes decreased the stability of the model, as

p-ISSN 2798-253X
e-ISSN 2798-3153

seen at epochs 450 and 500. Therefore, 400 epochs was the optimal number to achieve a balance between accuracy and model stability.

3.3. Comparing Between Two Optimization Methods

This research used the SGD and AdaGrad optimization methods. SGD is an iterative learning algorithm that uses training data to update the model. This algorithm is iterative, meaning that each step will try to improve the model parameters slightly. Each iteration involves using the model with the current parameters to make predictions on some training data, comparing the predictions with the expected results, calculating the error, and using the error to update the model parameters [30]. AdaGrad is a derivative algorithm of SGD that adapts to learning rates with smaller parameters and model updates. These two methods are methods that are often used for LSTM-based RNN models.

Furthermore, both optimization methods used a learning rate of 0.05 and epochs of 200 to get more detailed training from the two optimization methods. The activation function used was a logistic function and hypertuning other parameters, as in [29]. Meanwhile, autoregressive lag was determined similarly to learning rate testing. Test results with the two optimization methods can be seen in Table 3.

Table 3. Performance Comparison of SGD and AdaGrad Optimization Methods

Optimization	Training Data		Testing Data	
	SMAPE	***RMSE***	***SMAPE***	***RMSE***
SGD	0.15968	0.06149	0.15505	0.11254
AdaGrad	0.79070	0.57503	0.92814	0.63431

Table 3 shows that SGD outperforms AdaGrad in terms of performance on both training and testing data. SGD had lower SMAPE and RMSE (SMAPE: 0.15968; RMSE: 0.06149 for training and SMAPE: 0.15505; RMSE: 0.11254 for testing) compared to AdaGrad (SMAPE: 0.79070; RMSE: 0.57503 for training and SMAPE: 0.92814; RMSE: 0.63431 for testing). This shows that SGD produces more accurate and stable predictions than AdaGrad.

3.4. Comparing Between Three Activation Functions

This research used the logistic activation function, Gompertz, and hyperbolic tangent (tanh). These three activation functions are activation functions that are often used for the forecasting model learning process. The three activation functions used a learning rate of 0.05 and an epoch of 200 to obtain more detailed training from the SGD optimization method. Meanwhile, the autoregressive lag and hypertuning parameters were determined similarly to testing optimization methods. The test results with the three activation functions can be seen in Table 4.

Table 4. Impact of Different Activation Functions on Training and Testing Data Performance

Activation Function	Training Data		Testing Data	
	SMAPE	***RMSE***	***SMAPE***	***RMSE***
Logistic	0.15968	0.06149	0.15505	0.11254
Gompertz	0.16029	0.06564	0.18196	0.11010
Tanh	0.60584	0.39279	0.85418	0.54439

Based on Table 4, the logistic activation function performed best, with the lowest SMAPE and RMSE values in both training (SMAPE: 0.15968, RMSE: 0.06149) and testing (SMAPE: 0.15505, RMSE: 0.11254) data, indicating more accurate predictions. Gompertz was slightly behind with a higher SMAPE on testing (0.18196) but had a lower testing RMSE (0.11010). In contrast, tanh showed the worst performance with a much higher SMAPE and RMSE, making it a less suitable function for this dataset.

Next, the best RNN-LSTM model obtained was compared with several models, namely the Exponential Smoothing (ETS) model described in [20], the ARIMA model described in [21], the FFNN model described in [22], and the RNN model described in [23]. The comparison results of empirical studies can be seen in Table 5 for the best RNN-LSTM model with several models. The

empirical research showed that RNN-LSTM performed best with the lowest SMAPE (0.14570) and lowest RMSE (1.24531), signaling the most accurate and stable prediction. RNN followed with a fairly good performance (SMAPE 0.16282 and RMSE 2.14246), while traditional models such as ETS and ARIMA had higher SMAPE and RMSE, indicating less accurate predictions. FFNN performed the worst with a SMAPE of 0.38357 and RMSE of 1.63286, indicating the highest error rate among all models compared. Overall, the RNN-LSTM outperformed the other models in predicting inflation. A comparison of graphic illustrations in the form of plots of actual data and testing data (out-sample forecasting) is shown in Fig. 4.

Table 5. Comparative Performance of RNN-LSTM, ETS, ARIMA, FFNN, and RNN Models on Monthly Forecasting

Month	Actual Data	RNN-LSTM Model	ETS Model	ARIMA Model	FFNN Model	RNN Model
Jan 2023	5.28	9.35	2.61	2.54	2.53	12.5
Feb 2023	5.47	6.10	2.61	2.54	2.39	4.03
Mar 2023	4.97	5.59	2.61	2.54	2.41	4.51
Apr 2023	4.33	4.82	2.61	2.54	2.55	4.41
May 2023	4.00	4.37	2.61	2.54	2.42	4.08
Jun 2023	3.52	3.75	2.61	2.54	2.49	3.45
Jul 2023	3.08	3.45	2.61	2.54	2.54	3.13
Aug 2023	3.27	3.06	2.61	2.54	2.31	2.82
Sep 2023	2.28	2.84	2.61	2.54	2.48	2.79
Oct 2023	2.56	2.54	2.61	2.54	2.42	2.28
Nov 2023	2.86	2.33	2.61	2.54	2.23	2.36
Dec 2023	2.61	2.37	2.61	2.54	2.22	2.58
SMAPE		0.14570	0.32133	0.33857	0.38357	0.16282
RMSE		1.24531	1.51180	1.55994	1.63286	2.14246

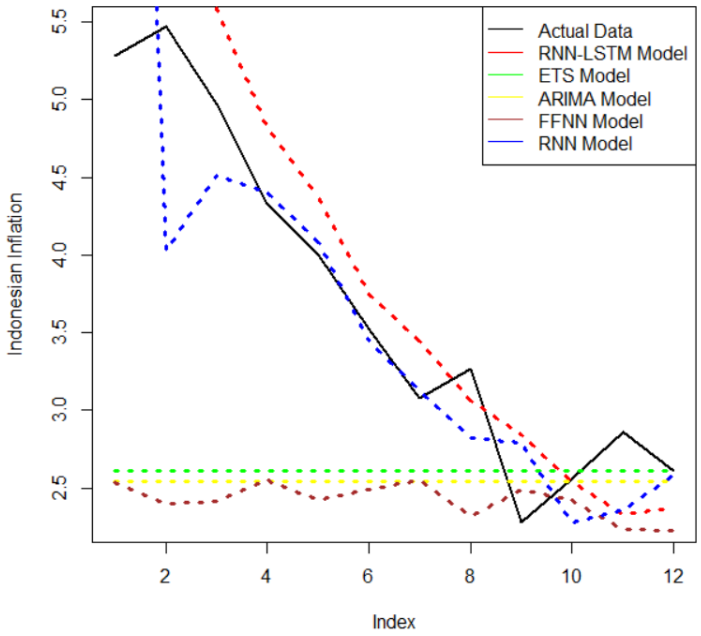

Fig. 4 Plot actual data and out-sample forecasting using the RNN-LSTM model and several models for inflation rate data in Indonesia.

4. Conclusion

Based on the results of the analysis, the frequency approach in the RNN model based on LSTM can provide the best results for monthly data on the inflation rate in Indonesia. These results indicate that the frequency approach is an effective way to increase forecasting accuracy. On the other hand, learning rate testing showed that the optimal value for testing data accuracy was 0.05, with a significant difference at a larger learning rate. Higher epochs improve accuracy, but too large epochs can make the model less stable. In addition, the SGD optimization method was proven to be superior to AdaGrad in terms of error and accuracy, with much lower SMAPE values and higher accuracy on training and testing data. Meanwhile, the logistic activation function showed slightly better accuracy than the Gompertz function, even though the Gompertz function had a lower error value. Finally, the best RNN-LSTM model tested showed superior results compared to other models (ETS, ARIMA, FFNN, and RNN), with the lowest SMAPE and RMSE value on the testing data. Further research can be focused on the influence of the number of neurons in the input layer using a different approach to the RNN-LSTM model and comparing it with other models, such as Gated Recurrent Unit (GRU) based RNN.

References

[1] D. Ruhiat and C. Suwanda, "Peramalan Data Deret Waktu Berpola Musiman Menggunakan Metode Regresi Spektral (Studi Kasus: Debit Sungai Citarum-Nanjung)," *Teorema Teori dan Riset Matematika*, Vol. 4, No. 1, pp. 1–12, Mar. 2019, doi: 10.25157/teorema.v4i1.1887.

[2] R.B.R. Putra and H. Hendry, "Multivariate Time Series Forecasting pada Penjualan Barang Retail dengan Recurrent Neural Network," *Jurnal Inovtek Polbeng Seri Informatika*, Vol. 7, No. 1, pp. 71–82, Jun. 2022, doi: 10.35314/isi.v7i1.2398.

[3] C. Tian, J. Ma, C. Zhang, and P. Zhan, "A Deep Neural Network Model for Short-Term Load Forecast Based on Long Short-Term Memory Network and Convolutional Neural Network," *Energies*, Vol. 11, No. 12, Dec. 2018, Art. no. 3493, doi: 10.3390/en11123493.

[4] W. Hastomo, A.S.B. Karno, N. Kalbuana, E. Nisfiani, and L. ETP, "Optimasi Deep Learning untuk Prediksi Saham di Masa Pandemi Covid-19," *Jurnal Edukasi dan Penelitian Informatika*, Vol. 7, No. 2, pp.140–133, Aug. 2021, doi: 10.26418/jp.v7i2.47411.

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

[6] C.H. Cordova, M.N.L. Portocarrero, R. Salas, R. Torres, P.C. Rodrigues, and J.L. López-Gonzales, "Air Quality Assessment and Pollution Forecasting Using Artificial Neural Networks in Metropolitan Lima-Peru," *Scientific Reports*, Vol. 11, No. 1, Dec. 2021, Art. no. 24232 (2021), doi: 10.1038/s41598-021-03650-9.

[7] K. Wong, A.P. Wibawa, H.S. Pakpahan, A. Prafanto, and H.J. Setyadi, "Prediksi Tingkat Inflasi Dengan Menggunakan Metode Backpropagation Neural Network," *Sains, Aplikasi, Komputasi dan Teknologi Informasi*, Vol. 1, No. 2, pp. 8–13, Aug. 2019, doi: 10.30872/jsakti.v1i2.2600.

[8] B. Hauriza, M. Muladi, and I.M. Wirawan, "Prediksi Tingkat Inflasi Bulanan Indonesia Menggunakan Metode Jaringan Saraf Tiruan," *Jurnal Teknologi dan Informasi*, Vol. 11, No. 2, pp. 152–167, Sep. 2021, doi: 10.34010/jati.v11i2.4924.

[9] H. Hermansah, D. Rosadi, A. Abdurakhman, and H. Utami, "Automatic Time Series Forecasting Using Nonlinear Autoregressive Neural Network Model With Exogenous Input," *Bulletin of Electrical Engineering and Informatics*, Vol. 10, No. 5, pp. 2836–2844, Oct. 2021, doi: 10.11591/eei.v10i5.2862.

[10] P. Cihan, "Effect of Parameter Selection on Heart Attack Risk Prediction in an RNN Model," in *5th International Conference on Applied Engineering and Natural Sciences ICAENS 2023*, 2023, pp. 56–60, doi: 10.59287/icaens.964.

[11] B. Xu, R. Huang, and M. Li, "Revise Saturated Activation Functions," 2016, *arXiv:1602.05980.*

[12] O.A. Vilca-Huayta and U. Y. Tito, "Efficient Function Integration and a Case Study with Gompertz Functions for Covid-19 Waves," *International Journal of Advanced Computer Science and Applications (IJACSA)*, Vol. 13, No. 8, pp. 545–551, 2022, doi: 10.14569/IJACSA.2022.0130863.

[13] T. De Ryck, S. Lanthaler, and S. Mishra, "On the Approximation of Functions by Tanh Neural Networks," *Neural Networks*, Vol. 143, pp. 732–750, Nov. 2021, doi: 10.1016/j.neunet.2021.08.015.

[14] K. Banerjee et al., "Optimizing Deep Learning RNN Topologies on Intel Architecture," *Supercomputer Frontiers and Innovations*, Vol. 6, No. 3, pp. 64–85, Sep. 2019, doi: 10.14529/jsfi190304.

[15] M. Xia, H. Shao, X. Ma, and C.W. de Silva, "A Stacked GRU-RNN-Based Approach for Predicting Renewable Energy and Electricity Load for Smart Grid Operation," *IEEE Transactions on Industrial Informatics*, Vol. 17, No. 10, pp. 7050–7059, Oct. 2021, doi: 10.1109/TII.2021.3056867.

[16] T. Szandała, "Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks," in *Bio-inspired Neurocomputing*, Vol. 903, A.K. Bhoi, P.K. Mallick, C.-M. Liu, and V. E. Balas, Eds., Singapore, Singapore: Springer, 2021, pp. 203–224, doi: 10.1007/978-981-15-5495-7_11.

[17] M.H.E. Ali, A.B. Abdel-Raman, and E.A. Badry, "Developing Novel Activation Functions Based Deep Learning LSTM for Classification," *IEEE Access*, Vol. 10, pp. 97259–97275, 2022, doi: 10.1109/ACCESS.2022.3205774.

[18] P. Ranjan, P. Khan, S. Kumar, and S.K. Das, "log-Sigmoid Activation-Based Long Short-Term Memory for Time-Series Data Classification," *IEEE Transactions on Artificial Intelligence*, Vol. 5, No. 2, pp. 672–683, Feb. 2024, doi: 10.1109/TAI.2023.3265641.

[19] *Data Strategis BPS*. Badan Pusat Statistik, 2009. [Online]. Available: https://www.bps.go.id/id/publication/2009/08/15/70be6e12dc0e2d87204a6d58/data-strategis-bps-2009.html. Accessed: September 16, 2024.

[20] R.J. Hyndman, A.B. Koehler, R.D. Snyder, and S. Grose, "A State Space Framework for Automatic Forecasting Using Exponential Smoothing Methods," *International Journal of Forecasting*, Vol. 18, No. 3, pp. 439–454, Jul. 2002, doi: 10.1016/S0169-2070(01)00110-8.

[21] R.J. Hyndman and Y. Khandakar, "Automatic Time Series Forecasting: The forecast Package for R," *Journal of Statistical Software*, Vol. 27, No. 3, pp. 1–22, 2008, doi: 10.18637/jss.v027.i03.

[22] H. Hermansah, D. Rosadi, A. Abdurakhman, and H. Utami, "Selection of Input Variables of Nonlinear Autoregressive Neural Network Model for Time Series Data Forecasting," *Media Statistika*, Vol. 13, No. 2, pp. 116–124, Dec. 2020, doi: 10.14710/medstat.13.2.116-124.

[23] F. Martínez, F. Charte, M.P. Frías, and A.M. Martínez-Rodríguez, "Strategies for Time Series Forecasting with Generalized Regression Neural Networks," *Neurocomputing*, Vol. 491, pp. 509–521, Jun. 2022, doi: 10.1016/j.neucom.2021.12.028.

[24] A.A. Rizal and S. Soraya, "Multi Time Steps Prediction dengan Recurrent Neural Network Long Short Term Memory," *MATRIK Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, Vol. 18, No. 1, pp. 115–124, Nov. 2018, doi: 10.30812/matrik.v18i1.344.

[25] S. Sen, D. Sugiarto, and A. Rochman, "Komparasi Metode Multilayer Perceptron (MLP) dan Long Short Term Memory (LSTM) dalam Peramalan Harga Beras," *Ultimatics Jurnal Teknik Informatika*, Vol. 12, No. 1, pp. 35–41, Jul. 2020, doi: 10.31937/ti.v12i1.1572.

[26] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," *IEEE Access*, Vol. 5, pp. 21954–21961, 2017, doi: 10.1109/ACCESS.2017.2762418.

[27] I.N. Husada and H. Toba, "Pengaruh Metode Penyeimbangan Kelas Terhadap Tingkat Akurasi Analisis Sentimen pada Tweets Berbahasa Indonesia*," Jurnal Teknik Informatika dan Sistem Informasi*, Vol. 6, No. 2, pp. 400–413, Aug. 2020, doi: 10.28932/jutisi.v6i2.2743.

[28] J. Qiu, B. Wang, and C. Zhou, "Forecasting Stock Prices with Long-Short Term Memory Neural Network Based on Attention Mechanism," *PLoS One*, Vol. 15, No. 1, Jan. 2020, Art. no. e0227222, doi: 10.1371/journal.pone.0227222.

[29] N.S. Keskar and R. Socher, "Improving Generalization Performance by Switching from Adam to SGD," 2017, *arXiv: 1712.07628*.

[30] A.K. Tyagi and A. Abraham, *Recurrent Neural Networks*. Boca Raton, FL, USA: CRC Press, 2022.

p-ISSN 2798-253X
e-ISSN 2798-3153