Mathematical Journal of Modelling and Forecasting
Vol. 3, No. 1, June 2025, pp. 16-24

Strategy for Enhancing GRU-RNN Performance through Parameter Optimization

Hermansah[1,*]

[1] Department of Mathematics Education, University of Riau Kepulauan, Batam, Indonesia
[*] hermansah@fkip.unrika.ac.id

Abstract. This study examines the selection of optimal parameters in the Gated Recurrent Unit-Recurrent Neural Network (GRU-RNN) model for forecasting inflation in Indonesia. Accurate forecasting requires precise model parameter adjustments, especially for time-series data, which can be either linear or non-linear. The study evaluates several parameters, including learning rate, number of epochs, optimization methods (Stochastic Gradient Descent (SGD) and Adaptive Gradient (AdaGrad)), and activation functions (Logistic, Gompertz, and Tanh). The results show that the best combination consists of the SGD optimization method, logistic activation function, a learning rate of 0.05, and 450 epochs, which delivers the best performance by minimizing errors and achieving high prediction accuracy. When compared to other forecasting models such as Exponential Smoothing (ETS), Autoregressive Integrated Moving Average (ARIMA), Feedforward Neural Network (FFNN), and Recurrent Neural Network (RNN), the GRU-RNN model shows significant superiority. Additionally, the Logistic activation function proves to be more effective in maintaining stability and prediction accuracy, while the use of the Adaptive Gradient (AdaGrad) method results in lower performance. These findings underscore the GRU-RNN model's ability to handle non-linear time-series data and provide insights for developing more accurate and efficient forecasting models in the future.

Keywords: Forecasting, Gated Recurrent Unit (GRU), Recurrent Neural Network (RNN), Activation Functions, Stochastic Gradient Descent (SGD).

1 Introduction

Forecasting is an effort to predict future events by relying on data from the past. The main goal of forecasting is to produce predictions as accurately as possible. A good forecast must have high accuracy and the ability to reflect the behavioral patterns of time series data recorded previously [1]. There are two main categories in time series forecasting methods: statistical and machine learning or computational intelligence methods [2]. The commonly used statistical methods are autoregressive moving average and exponential smoothing, which generally require linear data. However, machine learning methods (deep learning) perform better when dealing with nonlinear data. Therefore, the appropriate forecasting method's choice depends significantly on the characteristics to be analyzed to improve the accuracy of the resulting predictions.

One of the algorithms in the deep learning category is the Recurrent Neural Network (RNN), which is part of the Artificial Neural Network (ANN). In RNN, the output from the hidden layer is reused as input for the following process [3]. However, RNN is weak in storing long-term memory, making it difficult for the model to remember previous information, a problem known as the vanishing gradient. The Gated Recurrent Unit (GRU) is used to overcome this challenge, an RNN architecture. GRU is specifically designed to address the vanishing gradient problem, allowing it to learn from longer sequence data. With its gating mechanism, GRU has two main gates: the reset gate, which controls how much past information will be forgotten, and the update gate, which determines how much new information is added to the memory. Additionally, the GRU structure is simpler than the Long Short-Term Memory (LSTM), making it faster to train and more efficient in computational resource usage. Therefore, GRU has proven effective in various applications, such as time-series forecasting, natural language processing, and speech recognition [4].

Several studies have applied the GRU-based RNN method for various purposes. In a survey by Aldi et al. [5], GRU was used to predict Bitcoin prices, achieving an excellent average accuracy of 95.36% on training data and 93.5% on test data. Meanwhile, Wiranda and Sadikin [6] applied GRU to predict product sales at PT Metiska Farma and found that a 90% training and 10% testing data split yielded the most optimal results, based on RMSE and MAPE values on the test data. Another study by Putra and Hendry [2] compared the performance of RNN with LSTM and GRU in predicting retail sales, with LSTM showing the best performance. Wong et al. [7] also conducted inflation rate prediction analysis in Samarinda City, East Kalimantan, using the Backpropagation Neural Network method. Hauriza et al. [8] also predicted monthly inflation rates in Indonesia using Artificial Neural Networks, obtaining a low MSE value. Finally, Hermansah et al. [9] used a nonlinear autoregressive neural network model with exogenous input to predict monthly inflation in Indonesia and found that the proposed method outperformed other models.

The research by Cihan [10] shows that the selection of activation functions and weight update methods in RNN models significantly impacts prediction performance. RNNs using activation functions such as Logistic [11], Gompertz [12], and Tanh [13] proved to be more accurate compared to the ReLU activation function. To improve the performance of RNN models, common weight update strategies like Stochastic Gradient Descent (SGD) [14] and Adaptive Gradient (AdaGrad) [15] can be implemented. Tomasz Szandała [16] compares various activation functions in neural networks, including Sigmoid, Tanh, and ReLU, and guides on selecting the appropriate activation function for real-world applications. In 2022, Mohamed H. Essai Ali et al. [17] tested 26 alternative activation functions on GRU for classification and found that some functions, such as Modified Elliott and Softsign, showed higher accuracy than Tanh. Furthermore, Sajal Das [18] proposed using the Log-Sigmoid activation function in GRU for time-series data classification, which showed improved accuracy with various optimization algorithms. These studies emphasize the importance of selecting activation functions to enhance neural network performance. However, no research has explored the impact of activation function selection and weight updates in the GRU-RNN model for forecasting inflation rates in Indonesia.

This study aims to fill the existing knowledge gap by conducting a comparison between two weight update methods in the GRU-RNN model, namely Stochastic Gradient Descent (SGD) and Adaptive Gradient (AdaGrad), as well as three different activation functions: Logistic, Gompertz, and Tanh. The study applies the GRU-RNN model to forecast inflation in Indonesia, a phenomenon of sustained price increases linked to the currency's depreciation [19]. In developing this model, the GRU-RNN input layer is proposed based on autoregressive lag with a frequency approach, where the frequency represents the amount of data in a specific time unit, such as monthly, quarterly, or yearly. The evaluation also includes the influence of the learning rate and the number of epoch iterations. Additionally, the GRU-RNN model will be compared with other models, such as Exponential Smoothing (ETS), ARIMA, Feedforward Neural Network (FFNN), and Recurrent Neural Network (RNN), with forecasting accuracy measured using the Symmetric Mean Absolute Percentage Error (SMAPE).

2 Research Methods

2.1 RNN Model

Recurrent Neural Network (RNN) is a type of Artificial Neural Network (ANN) that is effective in identifying hidden patterns in data, particularly in applications such as speech recognition, natural language processing, and time series forecasting [3]. The main advantage of RNN lies in its ability to handle sequential data, where information from the current input is not only processed directly but is also influenced by the traces of information obtained previously. This process allows RNN to leverage a broader context in predicting outcomes. A distinctive feature of RNN is reusing the output from the hidden layer generated in the previous step, which is then reintroduced as input for the following process [20]. This makes RNNs highly useful in models that require dependencies between inputs over time, allowing the inputs to interact and provide continuous information to generate more accurate predictions [21].

According to Yin et al. [22], the Recurrent Neural Network (RNN) has three main layers: the input, hidden, and output layers. The RNN model operates with a unidirectional flow of information, starting from the input layer and flowing to the hidden layer. Then, the information is synthesized unidirectionally from the previous hidden layer to the current hidden layer being processed. This structure allows the RNN to link information received at a specific time with previously processed data, creating a continuous flow of information throughout the network's computational process. The architecture model of the RNN is as follows.

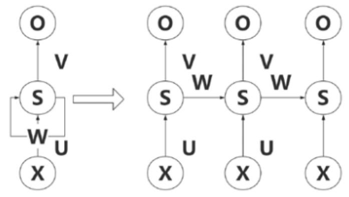

Fig 1. Architecture of the Recurrent Neural Network.

As shown in Figure 1, the mapping between S_t and O_t can be depicted in the following form.

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

$$O_t = g(V \times S_t) \tag{2}$$

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

2.2 GRU Model

The Gated Recurrent Unit (GRU) is a type of architecture in artificial neural networks that belongs to the Recurrent Neural Networks (RNN) category. It is designed to address issues faced by traditional RNNs, such as vanishing gradients, which can hinder the model's ability to learn from long sequence data. GRU uses a gating mechanism that enables the model to retain important information longer [23, 24].

Characteristics of GRU:

[1] Gating Mechanism
GRU has two main gates: the reset gate and the update gate. The reset gate controls how much information from the past is forgotten, while the update gate determines how much new information is added to the memory.

[2] Simplicity
Compared to Long-Short-Term Memory (LSTM), GRU has a simpler structure, making it faster to train and more efficient in using computational resources.

[3] Good Performance
GRU has been proven effective in various applications, including time series forecasting, natural language processing, and speech recognition [25].

This study analyzes the best GRU-RNN model for forecasting inflation in Indonesia. The following are the steps taken in the GRU-RNN modeling process for forecasting, which includes various stages that will be outlined to achieve optimal prediction results.

[1] Data Collection
Collect relevant data for the forecasting problem to be solved. This data could be historical data that includes variables affecting the outcome to be predicted.

[2] Data Preprocessing
- Perform data cleaning to remove missing values or outliers.
- Determine the neurons in the input layer (autoregressive lag) based on the frequency of data x, assuming that x is the time series data to be predicted. If the frequency of data x is m, then the autoregressive lags from 1 to m are considered as autoregressive lags. For example, for monthly data, the autoregressive lag is considered 1:12.
- Normalize or standardize the data to ensure all features are on the same scale. Data normalization is done by scaling the data using the following equation:
$$y = \frac{x - min(x)}{max(x) - min(x)} \tag{3}$$
where y is the normalized data value; x is the data to be predicted; $min(x)$ is the minimum value of the predicted data; and $max(x)$ is the maximum value of the predicted data.
- Splitting data into training and testing sets.

[3] Model Development
- Determine the GRU architecture, including the number of layers, neurons in each layer, learning rate, epochs, and other hyperparameters, as discussed by Keskar and Socher [26].

- ➢ Choose an activation function (such as Logistic, Gompertz, or Tanh) and an optimizer (such as Stochastic Gradient Descent (SGD) or Adaptive Gradient (AdaGrad)).

[4] Model Training

- ➢ Train the model using the training data. During training, the model will learn to identify patterns in the data and optimize parameters to minimize prediction error.
- ➢ Use cross-validation techniques to avoid overfitting.

[5] Model Evaluation

Test the model using the testing data to evaluate its performance. Use metrics such as Symmetric Mean Absolute Percentage Error (SMAPE) to measure prediction accuracy. SMAPE measures accuracy in forecasting models by comparing predicted values with actual values. SMAPE addresses some of the weaknesses of MAPE by avoiding issues when actual values approach zero. SMAPE gives results in percentage form, and a lower SMAPE value indicates a more accurate model. The SMAPE value is formulated as follows:

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n} \frac{|A_i - F_i|}{\frac{|A_i| + |F_i|}{2}} \times 100\% \quad (4)$$

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

[6] Prediction

- ➢ After the model has been trained and evaluated, use the model to make predictions on new data, namely forecasting the inflation rate in Indonesia for January 2024 - December 2024.
- ➢ Perform the denormalization of the data. Data denormalization is done using the following equation:

  $$x = y * [max(x) - min(x)] + min(x) \quad (5)$$

  where x is the denormalized data value; y is the normalized data value; $min(x)$ is the minimum value of the predicted data; and $max(x)$ is the maximum value of the predicted data.
- ➢ Compare the predicted and actual data using the SMAPE value.

[7] Monitoring and Maintenance

Continuously monitor the model's performance over time and update it as necessary. This includes retraining the model with the latest data to ensure sustained accuracy [27, 28].

3 Results and Discussion

In this study, a case study was conducted using monthly inflation data in Indonesia, covering the period from January 2005 to December 2024. The training data was taken from the first 228 months, from January 2005 to December 2023, while the last 12 months, from January 2024 to December 2024, were used as test data. This data can be accessed through the official Bank Indonesia website. Furthermore, the tests conducted include an analysis of the learning rate, the number of epoch iterations, and comparing two model optimization methods. Additionally, testing was carried out on the activation function by comparing three different types of functions.

3.1 Testing the Learning Rate

In this study, the neurons in the input layer are proposed based on autoregressive lags using a frequency approach. In the context of time series data, there are two essential attributes: time and frequency. The time attribute indicates the time unit for each observation point, while the frequency attribute represents the number of data points collected within a specific period, typically expressed per year. For example, monthly data has a frequency of 12, quarterly data has a frequency of 4, triannual data has a frequency of 3, semiannual data has a frequency of 2, and annual data has a frequency of 1. Therefore, in this study, the number of neurons used in the input layer is 12, representing the 12 months as autoregressive lags.

Subsequently, experiments were conducted with various learning rate values, namely 0.001, 0.002, 0.050, 0.100, 0.200, 0.500, and 0.900. The optimization method applied in this study is Stochastic Gradient Descent (SGD), with a total of 200 epochs. The activation function used is the logistic function. Additionally, other hyperparameter tuning values were determined using the same approach as described by Keskar and Socher [26]. This experiment aims to evaluate the effect of the learning rate size on the learning process and data testing. The experiment's results regarding the learning rate can be seen in Table 1.

Table 1. Learning Rate Testing Results

Learning Rate	Training Data		Testing Data	
	SMAPE	**Accuracy**	**SMAPE**	**Accuracy**
0.001	0.37539	0.82438	0.80315	0.58739
0.002	0.40281	0.62709	0.78724	0.31390
0.050	0.17738	0.95233	0.36343	0.82258
0.100	0.19031	0.95207	0.40683	0.68363
0.200	0.18334	0.95732	0.36520	0.61712
0.500	0.24320	0.84631	0.37202	0.78444
0.900	1.99996	0.41033	1.35091	0.17351

Table 1 shows the accuracy results obtained based on different learning rate variations. The highest accuracy for the training data reached 95.73% with a learning rate of 0.200, while the highest accuracy for the testing data was recorded at 82.26% with a learning rate of 0.050. At a learning rate of 0.050, the difference in accuracy between the training and testing data was not very significant. However, at learning rates of 0.100 and 0.200, there was a noticeable difference between the training and testing accuracy. This is because if the learning rate is too large, the machine learning process can cause an increase in error rather than a reduction during training. Thus, a higher learning rate can result in mistakes during weight updates, ultimately affecting training accuracy. The optimization model becomes unstable if the learning rate is too large, hindering the achievement of the desired target error. Conversely, if the learning rate is too small, too many iterations are needed to reach the desired target.

3.2 Testing the Epoch Iterations

This experiment was conducted using a learning rate of 0.05, the Stochastic Gradient Descent (SGD) optimization method, and the logistic activation function, with other hyperparameter tuning values determined following the method used by Keskar and Socher [26]. The autoregressive lag was also set using the same approach in the learning rate testing. This experiment aims to evaluate the effect of epoch iteration size on the learning process and data testing. The experiment results regarding the epoch values can be seen in Table 2.

Table 2. Epoch Value Testing Results

Epoch	Training Data		Testing Data	
	SMAPE	**Accuracy**	**SMAPE**	**Accuracy**
50	0.26559	0.91299	0.56181	0.60488
100	0.23017	0.93624	0.35897	0.56154
150	0.20140	0.94121	0.43106	0.60306
200	0.19879	0.94840	0.38185	0.62180
250	0.18647	0.95366	0.37663	0.49993
300	0.18391	0.95179	0.36894	0.58385
350	0.17647	0.95585	0.39771	0.63800
400	0.18710	0.95930	0.47650	0.48050
450	0.16662	0.96342	0.34517	0.67827
500	0.16756	0.95983	0.43689	0.44859

Table 2 shows the difference in accuracy results obtained based on varying epoch values. The best epoch value was recorded at 450, with an accuracy of 96.34% for the training data and 67.83% for the testing data. Within the epoch range of 50 to 500, there was no significant difference in accuracy levels. The higher the epoch value, the better the accuracy achieved. However, if the epoch value is too substantial, the stability of the optimization model may be disrupted, which could hinder achieving the desired target error.

3.3 Comparing the Two Optimization Methods

This study applies the Stochastic Gradient Descent (SGD) and Adaptive Gradient (AdaGrad) optimization methods. SGD is an iterative learning algorithm, where the training data updates the model. Each step in this algorithm aims to improve the model's parameters gradually. In each iteration, the model with the existing parameters is used to make predictions on a training dataset, and then the prediction results are compared with the expected outcomes. Next, the error is calculated to update the model's parameters [29]. On the other hand, AdaGrad is an algorithm derived from SGD, which adjusts the learning rate based on the parameters, allowing for more minor model updates. Both methods commonly develop Recurrent Neural Network (RNN) models based on Gated Recurrent Units (GRU).

Subsequently, both optimization methods were applied using a learning rate of 0.05 and 450 epochs to obtain more detailed training for both methods. The activation function used is the logistic function, and other hyperparameter tuning values were determined following the same approach described by Keskar and Socher [26]. The autoregressive lag was also set in the same manner as in the previous learning rate testing. The results of the testing conducted with both optimization methods can be seen in Table 3.

Table 3. Optimization Method Testing Results

Optimization	**Training Data**		**Testing Data**	
	SMAPE	**Accuracy**	**SMAPE**	**Accuracy**
SGD	0.21108	0.94258	0.49882	0.42058
AdaGrad	0.42971	0.65846	0.99751	0.10991

Based on the results shown in Table 3, there is a significant difference in accuracy between the SGD optimization model and the AdaGrad model, with the SGD optimization model showing much higher accuracy. The advantage of the SGD optimization method over AdaGrad is evident from the lower error values, measured by SMAPE. SGD achieved a SMAPE value of 21.11% for the training data and 49.88% for the testing data, with an accuracy of 94.26% for the training data and 42.06% for the testing data. Meanwhile, the AdaGrad optimization model showed higher SMAPE values, with 42.97% for the training data and 99.75% for the testing data, along with lower accuracy values, namely 65.85% for the training data and 10.99% for the testing data.

3.4 Comparing the Three Activation Functions

This study applies three activation functions: logistic, Gompertz, and hyperbolic tangent (tanh), which are commonly used in forecasting model learning. All three activation functions use a learning rate of 0.05 and 450 epochs to obtain more detailed training from the Stochastic Gradient Descent (SGD) optimization method. The autoregressive lag and hyperparameter tuning are also determined using the same approach as in the previous optimization method testing. The results of the testing conducted with these three activation functions can be seen in Table 4.

Table 4. Activation Function Testing Results

Activation Function	**Training Data**		**Testing Data**	
	SMAPE	**Accuracy**	**SMAPE**	**Accuracy**
Logistic	0.18758	0.95456	0.41689	0.68355
Gompertz	0.22105	0.93996	0.34706	0.40774
Tanh	0.50923	0.63862	0.82660	0.22203

Based on Table 4, the testing results show that the logistic and Gompertz activation functions yield similar accuracy levels, while the hyperbolic tangent (tanh) activation function shows the lowest accuracy. Although both activation functions have their respective advantages and disadvantages, this study found that the logistic activation function outperformed Gompertz, which recorded a higher accuracy. The logistic activation function achieved an accuracy of 95.46% for the training data and 68.36% for the testing data, while the Gompertz activation function recorded an accuracy of 94.00% for the training data and 40.77% for the testing data. On the other hand, the Gompertz activation function had a lower error value (SMAPE).

Subsequently, the Gated Recurrent Unit-Recurrent Neural Network (GRU-RNN) model is compared with several other models, including the Exponential Smoothing (ETS) model described by Hyndman et al. [30], the ARIMA model outlined by Hyndman and Khandakar [31], the Feedforward Neural Network (FFNN) model discussed by Hermansah et al. [32], and the RNN model explained by Hermansah et al. [33, 34]. The comparison results from the empirical study can be seen in Table 5, which shows the performance of the GRU-RNN model compared to these models. Empirical data indicate that the GRU-RNN model yields the best results, as reflected in the lowest SMAPE value on the testing data. Respectively, the SMAPE values for the testing data for the GRU-RNN, ETS, ARIMA, FFNN, and RNN models are 16.47%, 22.21%, 21.54%, 43.78%, and 27.13%.

Table 5. Comparison Results between the GRU-RNN Model and Several Models

Month	Actual Data	GRU-RNN Model	ETS Model	ARIMA Model	FFNN Model	RNN Model
Jan 2024	2.57	2.86	2.61	2.54	2.69	9.01
Feb 2024	2.75	2.59	2.61	2.54	2.64	2.17
Mar 2024	3.05	2.78	2.61	2.54	2.85	2.71
Apr 2024	3.00	3.07	2.61	2.54	3.15	3.22
May 2024	2.84	3.27	2.61	2.54	3.36	3.33
Jun 2024	2.51	3.06	2.61	2.54	3.61	3.07
Jul 2024	2.13	2.82	2.61	2.54	3.85	2.82
Aug 2024	2.12	2.55	2.61	2.54	3.82	2.56
Sep 2024	1.84	2.42	2.61	2.54	4.16	2.42
Oct 2024	1.71	2.15	2.61	2.54	4.12	2.15
Nov 2024	1.55	1.96	2.61	2.54	3.99	1.96
Dec 2024	1.57	1.84	2.61	2.54	4.11	1.84
SMAPE		0.16465	0.22211	0.21543	0.43776	0.27128

4 Conclusion

The selection of appropriate parameters in the Gated Recurrent Unit-Recurrent Neural Network (GRU-RNN) model significantly impacts the accuracy of inflation predictions in Indonesia. This study tested parameter combinations, including learning rate, number of epochs, optimization methods (SGD and AdaGrad), and activation functions (Logistic, Gompertz, and Tanh). The results show that the best combination consists of the SGD optimization method, logistic activation function, a learning rate of 0.05, and 450 epochs, which provided optimal performance with the lowest error (SMAPE) and high accuracy. Compared to other forecasting models such as Exponential Smoothing (ETS), Autoregressive Integrated Moving Average (ARIMA), Feedforward Neural Network (FFNN), and Recurrent Neural Network (RNN), this model shows significant superiority. Additionally, the logistic activation function proved to be more effective in maintaining stability and prediction accuracy, while using the AdaGrad method resulted in much lower performance. These findings confirm that GRU-RNN, with the correct parameters, excels in handling non-linear time series data, such as inflation, and provides essential insights for developing more accurate and efficient predictive models in the future.

References

[1] Ruhiat, D., & Suwanda, C. (2019). Peramalan data deret waktu berpola musiman menggunakan metode regresi spektral (Studi Kasus: Debit Sungai Citarum-Nanjung). TEOREMA: Teori dan Ris. Mat., 4(1), 1. https://doi.org/10.25157/teorema.v4i1.1887

[2] Radite Putra, R. B., & Hendry, H. (2022). Multivariate time series forecasting pada penjualan barang retail dengan Recurrent Neural Network. INOVTEK Polbeng - Seri Inform., 7(1), 71. https://doi.org/10.35314/isi.v7i1.2398

[3] Tian, C., Ma, J., Zhang, C., & Zhan, P. (2018). A deep neural network model for short-term load forecast based on Long Short-Term Memory network and Convolutional Neural Network. Energies, 11(12), 3493. https://doi.org/10.3390/en11123493

[4] Hastomo, W., Karno, A. S. B., Kalbuana, N., Nisfiani, E., & ETP, L. (2021). Optimasi deep learning untuk prediksi saham di masa pandemi Covid-19. J. Edukasi dan Penelit. Inform., 7(2), 133. https://doi.org/10.26418/jp.v7i2.47411

[5] Aldi, M. W. P., Jondri, & Aditsania, A. (2018). Analisis dan implementasi Long Short Term Memory Neural Network untuk prediksi harga Bitcoin. e-Proceeding Eng., 5(2), 3548–3555.

[6] Wiranda, L., & Sadikin, M. (2019). Penerapan Long Short Term Memory pada data time series untuk memprediksi penjualan produk PT. Metiska Farma. JANAPATI, 8(3), 184–196.

[7] Wong, K., Wibawa, A. P., Pakpahan, H. S., Prafanto, A., & Setyadi, H. J. (2019). Prediksi tingkat inflasi dengan menggunakan metode backpropagation neural network. Sains, Apl. Komputasi dan Teknol. Inf., 1(2), 8. https://doi.org/10.30872/jsakti.v1i2.2600

[8] Hauriza, B., Muladi, M., & Wirawan, I. M. (2021). Prediksi tingkat inflasi bulanan Indonesia menggunakan metode jaringan saraf tiruan. J. Teknol. dan Inf., 11(2), 152–167. https://doi.org/10.34010/jati.v11i2.4924

[9] Hermansah, H., Rosadi, D., Abdurakhman, A., & Utami, H. (2021). Automatic time series forecasting using nonlinear autoregressive neural network model with exogenous input. Bull. Electr. Eng. Informatics, 10(5), 2836–2844. https://doi.org/10.11591/eei.v10i5.2862

[10] Cihan, P. (2023). Effect of parameter selection on heart attack risk prediction in an RNN model. Int. Conf. Appl. Eng. Nat. Sci., 1(1), 56–60. https://doi.org/10.59287/icaens.964

[11] Xu, B., Huang, R., & Li, M. (2016). Revise saturated activation functions. ArXiv. https://arxiv.org/abs/1602.0

[12] Vilca-Huayta, O. A., & Tito, U. Y. (2022). Efficient function integration and a case study with Gompertz functions for Covid-19 waves. Int. J. Adv. Comput. Sci. Appl., 13(8), 545–551.

[13] De Ryck, T., Lanthaler, S., & Mishra, S. (2021). On the approximation of functions by tanh neural networks. Neural Networks, 143, 732–750. https://doi.org/10.1016/j.neunet.2021.08.015

[14] Banerjee, K., et al. (2019). Optimizing deep learning RNN topologies on Intel architecture. Supercomput. Front. Innov., 6(3), 64–85. https://doi.org/10.14529/jsfi190304

[15] Xia, M., Shao, H., Ma, X., & de Silva, C. W. (2021). A stacked GRU-RNN-based approach for predicting renewable energy and electricity load for smart grid operation. IEEE Trans. Ind. Informatics, 17(10), 7050–7059. https://doi.org/10.1109/TII.2021.3056867

[16] Szandała, T. (2021). Review and comparison of commonly used activation functions for deep neural networks. In Bio-inspired Neurocomputing (pp. 203–224). Springer Singapore. https://doi.org/10.1007/978-981-15-5495-7_11

[17] Essai Ali, M. H., Abdel-Raman, A. B., & Badry, E. A. (2022). Developing novel activation functions based deep learning LSTM for classification. IEEE Access, 10, 97259–97275. https://doi.org/10.1109/ACCESS.2022.3205774

[18] Ranjan, P., Khan, P., Kumar, S., & Das, S. K. (2024). $\log$-Sigmoid activation-based Long Short-Term Memory for time-series data classification. IEEE Trans. Artif. Intell., 5(2), 672–683. https://doi.org/10.1109/TAI.2023.3265641

[19] BPS, B. P. S. (2009). Data Strategis BPS. Jakarta: Badan Pusat Statistik.

[20] Rizal, A. A., & Soraya, S. (2018). Multi time steps prediction dengan Recurrent Neural Network Long Short Term Memory. MATRIK J. Manajemen, Tek. Inform. dan Rekayasa Komput., 18(1), 115–124. https://doi.org/10.30812/matrik.v18i1.344

[21] Sen, S., Sugiarto, D., & Rochman, A. (2020). Prediksi harga beras menggunakan metode Multilayer Perceptron (MLP) dan Long Short Term Memory (LSTM). Ultim. J. Tek. Inform., 12(1), 35–41. https://doi.org/10.31937/ti.v12i1.1572

[22] Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5, 21954–21961. https://doi.org/10.1109/ACCESS.2017.2762418

[23] Jiang, X., Zhang, Y., & Liu, Y. (2023). Short-term power load forecasting based on PSO-GRU. In 2023 IEEE 2nd International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA) (pp. 764–769). https://doi.org/10.1109/EEBDA56825.2023.10090721

[24] Huynh, A., & Nguyen, T. (2024). The comparison of GRU and LSTM in solar power generation forecasting application. International Journal of Science and Research Archive. https://doi.org/10.30574/ijsra.2024.13.1.1831

[25] Flores, A., Tito-Chura, H., & Yana-Mamani, V. (2021). An ensemble GRU approach for wind speed forecasting with data augmentation. International Journal of Advanced Computer Science and Applications (IJACSA), 12(6), 2021. http://dx.doi.org/10.14569/IJACSA.2021.0120666

[26] Keskar, N. S., & Socher, R. (2017). Improving generalization performance by switching from adam to sgd. arXiv Prepr. arXiv1712.07628. https://doi.org/10.48550/arXiv.1712.07628

[27] Sukanda, A., & Adytia, D. (2022). Wave forecast using bidirectional GRU and GRU method case study in Pangandaran, Indonesia. In 2022 International Conference on Data Science and Its Applications (ICoDSA) (pp. 278–282). https://doi.org/10.1109/ICoDSA55874.2022.9862832

[28] Diqi, M., Wakhid, A., Ordiyasa, W., Wijaya, N., Hiswati, M., & Info, A. (2023). Harnessing the power of stacked GRU for accurate weather predictions. Indonesian Journal of Artificial Intelligence and Data Mining. https://doi.org/10.24014/ijaidm.v6i2.24769

[29] Tyagi, A. K., & Abraham, A. (2022). Recurrent Neural Networks. Boca Raton: CRC Press. https://doi.org/10.1201/9781003307822

[30] Hyndman, R. J., Koehler, A. B., Snyder, R. D., & Grose, S. (2002). A state space framework for automatic forecasting using exponential smoothing methods. International Journal of Forecasting, 18(3), 439–454. https://doi.org/10.1016/S0169-2070(01)00110-8

[31] Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for R. Journal of Statistical Software, 27(3), 1–22. https://doi.org/10.18637/jss.v027.i03

[32] Hermansah, H., Rosadi, D., Abdurakhman, A., & Utami, H. (2020). Selection of input variables of nonlinear autoregressive neural network model for time series data forecasting. MEDIA Stat., 13(2), 116–124. https://doi.org/10.14710/medstat.13.2.116-124

[33] Hermansah, H., Muhajir, M., & Rodrigues, P. C. (2024). Indonesian inflation forecasting with recurrent neural network long short-term memory (RNN-LSTM). Enthusiastic: International Journal of Applied Statistics and Data Science, 4(2), 132–142. https://doi.org/10.20885/enthusiastic.vol4.iss2.art5

[34] Yotenka, R., Muhajir, M., Hermansah, H., & Rodrigues, P. C. (2025). Comparative analysis of activation functions in recurrent neural network: An application to Indonesian inflation forecasting. Mathematical Modelling of Engineering Problems, 12(3), 754–762. https://doi.org/10.18280/mmep.120302