

Comparative Analysis of LSTM and GRU for River Water Level Prediction

Fakhri Al Faris¹, Ahmad Taqwa², Ade Silvia Handayani^{*3}, Nyayu Latifah Husni⁴, Wahyu Caesarendra⁵, Asriyadi⁶, Leni Novianti⁷, M. Arief Rahman⁸

^{1,2,3,4}Department of Electrical Engineering, State Polytechnic of Sriwijaya, Indonesia

⁵Department of Mechanical and Mechatronics Engineering, Curtin University Malaysia

⁶Department of Electronics And Communication Engineering, King Abdul Aziz University, Saudi Arabia

^{7,8}Department of Information Management, State Polytechnic of Sriwijaya, Indonesia

Email: ³ade_silvia@polsri.ac.id

Received : Jul 8, 2025; Revised : Jul 11, 2025; Accepted : Jul 12, 2025; Published : Oct 16, 2025

Abstract

Accurate river water level prediction is essential for flood management, especially in tropical areas like Palembang. This study systematically analyzes the performance of two deep learning models, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), for real-time water level forecasting using hourly rainfall and water level data collected from automatic sensors. A series of experiments were conducted by varying window sizes (10, 20, 30) and the number of layers (1, 2, 3) for both models, with model performance assessed using RMSE, MAE, MAPE, and NSE. The results demonstrate that both window size and network depth significantly influence prediction accuracy and computational efficiency. The LSTM model achieved its highest accuracy with a window size of 30 and a single layer, while the GRU model performed best with a window size of 20 and two layers. This work contributes by systematically analyzing hyperparameter configurations of LSTM and GRU models on hourly rainfall and water level time series for flood-prone regions, offering empirical insight into parameter tuning in recurrent neural architectures for hydrological forecasting. These findings highlight the importance of careful parameter selection in developing reliable early warning systems for flood risk management.

Keywords : Deep Learning, GRU, LSTM, Water Level Prediction.

This work is an open access article and licensed under a Creative Commons Attribution-Non Commercial 4.0 International License



1. INTRODUCTION

Accurate river water level prediction remains a critical concern for hydrology and water resource management, as it supports various applications ranging from infrastructure planning and hydropower operations to navigation and ecological protection [1], [2], [3], [4], [5], [6]. Uncertainty in water level variation affects not only ecosystem stability but also decision-making processes in environmental and regional planning [7]. In recent decades, global climate change and the rising occurrence of extreme weather events have increased the complexity of river flow prediction, highlighting the need for more precise and adaptive forecasting approaches. Nevertheless, water level prediction remains challenging due to the complexity of contributing factors, which range from meteorological variables to hydrological data processes within river basins [7], [8]. These challenges have encouraged researchers to seek new approaches that are more adaptive and effective.

The evolution of water level forecasting has paralleled advances in computational intelligence, moving beyond traditional hydrological simulation to embrace data-centric approaches. Machine learning and deep learning, in particular, have enabled the discovery of complex nonlinear relationships in river systems without dependence on detailed physical parameterization [9]. Numerous investigations have confirmed that these data-driven methods can outperform classical models, especially for short-

term prediction and in operational settings with limited physical data [10]. Despite this progress, comparative analysis of deep learning architectures and their parameter sensitivity in hydrological applications remains an ongoing area of research.

Numerous machine learning algorithms, including M5P, Random Forest, Regression Tree, and REPTree, have been successfully utilized for river water level prediction in various geographic settings, frequently achieving better performance than traditional hydrological models [8], [11]. However, the application of deep learning for water level prediction remains relatively unexplored, despite its proven success in many other fields. Among deep learning architectures, the Recurrent Neural Network (RNN) stands out in time series modeling because its network structure is specifically designed to process sequential data, such as periodically recorded river water levels [12]. Among the various types of recurrent neural networks, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models are extensively utilized because of their effectiveness in capturing long-term dependencies within time series data [10].

Several studies have evaluated the performance of LSTM and GRU models in the context of water level prediction. For instance, the application of LSTM at four stations along the Nam Ngum River in Laos PDR demonstrated NSE values ranging from 0.5 to 0.7 during most of the testing periods [13]. Meanwhile, GRU-based approaches have also been shown to improve the accuracy of water level predictions under extreme flood conditions when compared to conventional univariate methods [14]. These findings indicate that both LSTM and GRU, as RNN-based models, are highly effective for time series data processing and river water level prediction.

Although LSTM and GRU models have proven effective for water level prediction, most previous studies have only examined one model at a time, often using limited types of data and specific parameter settings. In many cases, researchers did not investigate how combining multiple input features, such as rainfall and water level together could improve prediction results. There is also a lack of studies that systematically test how different model parameters impact the accuracy of predictions [15], [16], [17]. As a result, direct comparisons between LSTM and GRU models for hourly water level forecasting, especially using real-time data and various input combinations, remain very limited.

To address these gaps, this study directly compares the performance of LSTM and GRU models by testing a range of parameter settings and input scenarios using real-time sensor data from Palembang. We investigate not only which model works best, but also how different configurations affect prediction accuracy in a tropical river environment. The findings from this research are expected to help improve the practical use of deep learning models for flood prediction and water resource management.

In this research, two deep learning models, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), were implemented to predict river water levels using historical rainfall and water level data collected from automatic sensors, with an hourly recording interval during the period from March 1, 2025, to May 1, 2025. Experiments were conducted by comparing variations in the number of layers and window sizes for both models to identify the most optimal parameter configuration. Model performance was assessed based on four key indicators: root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and the Nash–Sutcliffe efficiency (NSE). The evaluation results showed that the LSTM model achieved its best performance with a window size of 30 and a single layer (RMSE = 0.0348 m; MAE = 0.0223 m; MAPE = 1.43%; NSE = 0.9090), while the GRU model demonstrated optimal performance with a window size of 20 and two layers (RMSE = 0.0277 m; MAE = 0.0180 m; MAPE = 1.15%; NSE = 0.9422). These findings confirm that model parameter tuning plays a crucial role in improving the real-time prediction accuracy of river water levels using deep learning approaches.

2. METHOD

In this study, river water level prediction was carried out using two deep learning models, namely Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). The overall research methodology is illustrated in Figure 1. In general, the research steps include input data collection, preprocessing, experimentation and model training, as well as accuracy evaluation to ensure optimal performance.

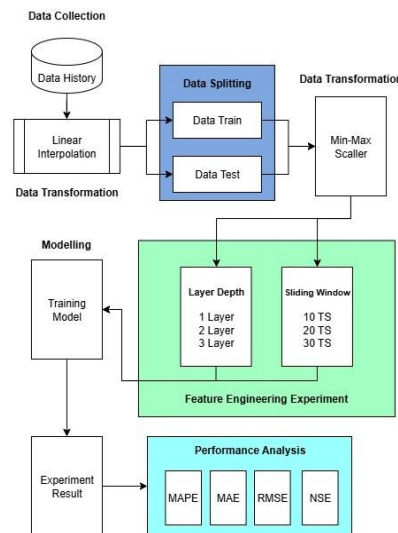


Figure 1. Workflow Diagram

2.1. Data Collection

The dataset utilized in this analysis comprises hourly rainfall and water level measurements, which were automatically recorded at the Radial Pump House, Palembang, between March and May 2025, resulting in a total of 1,488 paired observations. Each timestamp contains synchronized values for both rainfall intensity and river stage height, providing a comprehensive time series for model training and evaluation.

Table 1. History Data of Rainfall and Water Level

Timestamp	Rainfall (mm/h)	Water Level (m)
3/1/2025 00:00	0.1	1626
3/1/2025 01:00	0	1626
3/1/2025 02:00	0	1625
3/1/2025 03:00	0	1624
3/1/2025 04:00	0	1623
3/1/2025 05:00	0	1622
3/1/2025 06:00	0	1621
3/1/2025 07:00	0	1620
3/1/2025 08:00	0	1619
...	...	...
5/1/2025 22:00	0	1484
5/1/2025 23:00	0	1485

The implementation of automatic recording at fixed intervals ensured the collection of continuous and representative data that accurately captured river flow dynamics throughout the study period.

Subsequently, the complete dataset served as the foundation for testing and evaluating the performance of the river water level prediction models.

2.2. Data Preprocessing

Data preprocessing in this study commenced with the conversion of water level measurements from millimeters to meters to ensure all variables were standardized to a consistent scale, thereby facilitating more straightforward analysis [18]. To simulate real-world forecasting conditions and preserve temporal integrity, the time series was divided chronologically, with 80% assigned to the training set and the final 20% reserved exclusively for out-of-sample model evaluation.

All features and target variables underwent Min-Max normalization, with scaling parameters derived solely from the training subset, in order to prevent data leakage and promote faster convergence during the training phase. The normalization was performed using the following formula:

$$X_{Scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

This process scaled all variables to the [0, 1] range, ensuring that each feature contributed proportionally during model training, which helped prevent certain features from dominating and facilitated faster convergence [18], [19]. Consistent and systematic preprocessing steps are essential for building robust predictive models and for maintaining data quality and integrity throughout the experiments.

2.3. Model Development

All experiments were performed in Python 3.9 using TensorFlow 2.8 and the Keras API. Both LSTM and GRU models had a single recurrent layer (50 units), dropout (0.2), and a dense output. Input sequences used sliding windows of 10, 20, and 30 time steps and various layer depth 1, 2, and 3 layers. Models were trained with the Adam optimizer (learning rate 0.0001), batch size 16, and early stopping. Evaluation on the test set employed RMSE, MAE, MAPE, and NSE. The specific architecture and computational mechanism of each model are described in the following subsections.

2.3.1. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a variant of recurrent neural networks (RNNs) specifically introduced to overcome the inherent difficulties of traditional RNNs, such as vanishing or exploding gradients, which often hinder effective learning over long temporal sequences [20]. This architecture relies on memory cells that use three principal gates, input, forget, and output to modulate information flow at each time step. Two internal states are managed within each cell: the cell state, retaining long-term patterns, and the hidden state, holding short-term representations [21]. These gates dynamically control which signals are updated, preserved, or output, thus allowing the network to maintain relevant sequence information over extended periods [22]. The mathematical operations for the LSTM cell are outlined as follows, where each gate operation is governed by dedicated weights and activation functions [23]:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (2)$$

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

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (4)$$

$$\tilde{c}_t = \tanh(w_{\tilde{c}}[h_{t-1}, x_t] + b_{\tilde{c}}) \quad (5)$$

$$C_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (6)$$

$$h_t = o_t * \tanh(c^t) \quad (7)$$

The symbol σ represents the sigmoid activation function, W and b stands for the weight matrix the bias term for each respective gate. x_t and h_{t-1} represent the current input and previous hidden state, respectively. The input (i_t) forget (f_t), and output (o_t) gates, defined in Eq. (2), (3), and (4), control how information is added, retained, or passed on at each step. The candidate cell state ($\tilde{c}_t$) given in Eq. (5), proposes new content for the memory cell. The actual cell state update combines the previous cell state and candidate state weighted by their respective gates, as described in Eq. (6). Finally, the hidden state (h_t) is produced by modulating the cell state with the output gate, as shown in Eq. (7). This gating mechanism enables LSTM to adaptively filter and propagate relevant temporal information.

This LSTM architecture enables the model to effectively retain essential information over long sequences of data and mitigates information loss caused by gradient issues. A schematic illustration of the LSTM architecture, depicting the relationships between the cell state, hidden state, and main gates, is presented in Figure 2:

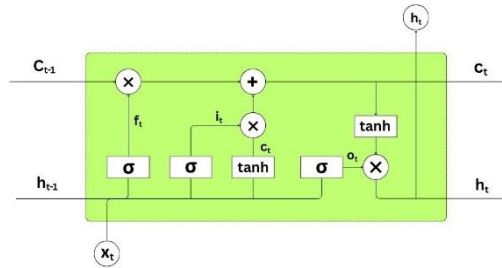


Figure 2. LSTM Architecture

2.3.2. Gated Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU) is a model developed based on the LSTM architecture, aiming to optimize its network structure while maintaining comparable performance [24]. In LSTM networks, overfitting can occur when available data are insufficient, as the model requires a larger number of parameters than conventional RNNs to address long-term dependency issues. GRU addresses these shortcomings by modifying the LSTM structure. Unlike LSTM, the GRU architecture consists of only two gates namely the update gate and the reset gate which effectively handle time series prediction problems with long intervals and significant delays. The update gate regulates the amount of information from previous time steps that is retained and transferred to the current time step, whereas the reset gate determines the degree to which information from the preceding time step is disregarded [10], [24].

The architecture of the GRU is depicted in Figure 3 illustrated the GRU employs two primary gates, the update gate and the reset gate, to regulate information flow within the network. To further clarify the GRU's computational process, the definitions of the key variables and the mathematical formulations are presented with eq (8) - (12).

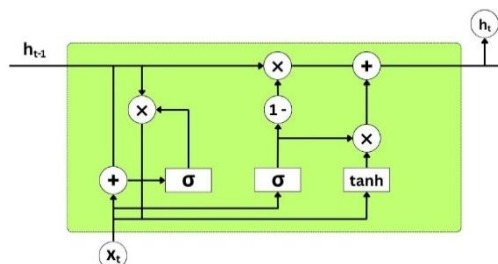


Figure 3. GRU Architecture

r_t is the output of the reset gate at time step - t, z_t is the output of the update gate at time step- t, h_t and h_{t-1} represent the outputs at time steps - t and t-1, respectively x_t denotes the input at time step - t. The symbol σ refers to the sigmoid activation function. The computation process for the memory unit is expressed in Equations (8) - (12):

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (8)$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \quad (9)$$

$$\hat{h}_t = \tanh(w_{\hat{h}} \cdot [r_t * h_{t-1}, x_t]) \quad (10)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \hat{h}_t \quad (11)$$

$$\hat{y}_t = \sigma(W_o \cdot h_t) \quad (12)$$

Under comparable parameter settings, the GRU has demonstrated notable strengths in comparison to the LSTM. On various datasets, GRU models often converge faster during training on CPUs, which leads to a significant reduction in computational time. The GRU's simpler parameter update mechanism also results in a more stable training process [24]. As a result, GRU is considered an efficient and competitive option, particularly for applications with limited computational resources or moderate data complexity.

2.4. Performance Indicators

2.4.1. MAE

Mean Absolute Error (MAE) quantifies the average magnitude of the absolute differences between observed and predicted values. As it is measured in the same units as the original data, MAE offers an intuitive measure of model accuracy. Lower MAE values reflect superior predictive performance in capturing the true data patterns. [25]. The calculation of MAE is presented in Equation (13).

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (13)$$

2.4.2. MAPE

Mean facilitating interpretation and enabling model performance comparisons across different datasets with varying scales. By calculating the average absolute percentage difference between actual and predicted values, MAPE serves as one of the most popular indicators for evaluating model performance, particularly in cases where percentage values are more relevant for analysis [26]. The calculation of MAPE is presented in Equation (14).

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\% \quad (14)$$

2.4.3. RMSE

Root Mean Squared Error (RMSE) measures the square root of the mean of the squared differences between observed and predicted values. Due to its sensitivity to large errors, a high RMSE value signifies substantial discrepancies between the model's predictions and the actual observations [27]. The calculation of RMSE is presented in Equation (15).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (15)$$

2.4.4. NSE

Nash Sutcliffe Efficiency (NSE) is a commonly employed statistical metric for assessing the performance of hydrological models. An NSE value near 1 signifies that the model effectively represents the observed data, while values near zero or negative indicate inadequate predictive capability [28]. The calculation of NSE is presented in Equation (16).

$$NSE = 1 - \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2} \quad (16)$$

3. RESULT

3.1. Analysis of Window Size Selection on Model Accuracy

Window size is a critical parameter in LSTM and GRU models, as it determines the amount of historical data used as input to predict the next value. Selecting an appropriate window size is essential for enabling the model to capture temporal patterns and long-term dependencies in time series data. A window size that is too small may lead to the omission of relevant historical information, whereas an excessively large window size can introduce noise and increase computational burden [29], [30].

In this study, experiments were conducted using window sizes of 10, 20, and 30 for both models. The evaluation results for the LSTM model indicate that increasing the window size generally improves prediction accuracy. A window size of 10 resulted in an RMSE of 0.0396 m, MAE of 0.0254 m, MAPE of 1.64%, and NSE of 0.8870. When the window size was increased to 20, the error values decreased, with an RMSE of 0.0356 m, MAE of 0.0236 m, MAPE of 1.50%, and NSE of 0.9038. The window size of 30 produced the best results, with an RMSE of 0.0349 m, MAE of 0.0224 m, MAPE of 1.43%, and NSE of 0.9090, although it required a longer training time.

A similar trend was also observed in the GRU model. With a window size of 10, the GRU achieved an RMSE of 0.0328 m, MAE of 0.0206 m, MAPE of 1.33%, and NSE of 0.9225. The model's performance further improved at a window size of 20, yielding an RMSE of 0.0295 m, MAE of 0.0188 m, MAPE of 1.20%, and NSE of 0.9340, which represents the optimal results. However, at a window size of 30, model performance tended to decline, with RMSE increasing to 0.0336 m, MAE to 0.0207 m, MAPE to 1.33%, and NSE decreasing to 0.9153. In addition, training time increased with larger window sizes for both models.

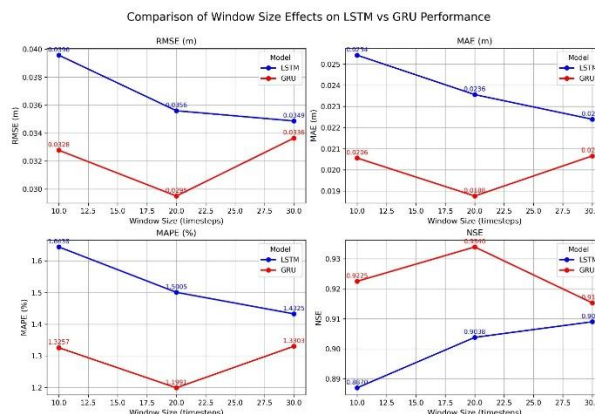


Figure 4. Comparison of Window Size Effects on LSTM vs GRU Performance

A comprehensive illustration of model performance is provided in Figure 4, which presents a comparison of RMSE, MAE, MAPE, and NSE values for the LSTM and GRU models. In addition, Table 2 summarizes the key evaluation results for each model and window size combination.

Table 2. Summary of LSTM and GRU model performance across different window sizes.

Model	Window Size	RMSE (m)	MAE (m)	MAPE (%)	NSE
LSTM	10	0.0395	0.0254	1.64	0.8870
LSTM	20	0.0356	0.0235	1.50	0.9038
LSTM	30	0.0348	0.0223	1.43	0.9090
GRU	10	0.0327	0.0205	1.32	0.9225
GRU	20	0.0294	0.0187	1.19	0.9339
GRU	30	0.0336	0.0206	1.33	0.9153

Overall, the experimental results indicate that selecting an optimal window size can significantly enhance model accuracy. For the LSTM model, a window size of 30 yielded the best results, while for the GRU model, a window size of 20 was the most optimal configuration. These findings underscore the importance of tuning the window size parameter in the development of time series prediction models, while also considering the balance between accuracy and computational efficiency.

3.2. Analysis of Layer Number Selection on Model Accuracy

The number of layers (depth) is a key architectural parameter in LSTM and GRU models that plays a crucial role in determining the model's capacity and ability to learn complex patterns in time series data. Models with a greater number of layers generally have an enhanced capability to capture deeper features and nonlinear relationships; however, they also carry a higher risk of overfitting and increased computational burden [31], [32].

The experimental results demonstrate that variations in the number of layers in the LSTM and GRU models have a significant impact on the accuracy of water level predictions. For the LSTM model, using a single layer yielded the best performance, with an RMSE of 0.0363 m, MAE of 0.0220 m, MAPE of 1.41%, and NSE of 0.9012. When the number of layers was increased to two, model performance declined, with the RMSE rising to 0.0397 m, MAE to 0.0264 m, MAPE to 1.69%, and NSE dropping to 0.8820. Adding a third layer further deteriorated performance, with the RMSE increasing to 0.0540 m, MAE to 0.0357 m, MAPE to 2.31%, and NSE decreasing to 0.7816. This decline in accuracy and the rise in error metrics observed when using three layers in the LSTM model indicate possible overfitting. As the network depth increases, the model becomes more complex and may begin to memorize the training data, which impairs its ability to generalize to new, unseen data.

A slightly different trend was observed in the GRU model. A single GRU layer yielded an RMSE of 0.0300 m, MAE of 0.0191 m, MAPE of 1.22%, and NSE of 0.9328. When the number of layers was increased to two, model performance improved, achieving the lowest RMSE of 0.0278 m, MAE of 0.0181 m, MAPE of 1.15%, and the highest NSE of 0.9422. However, adding a third layer did not provide further improvements; in fact, there was a slight decline in some metrics, with an RMSE of 0.0280 m, MAE of 0.0171 m, MAPE of 1.10%, and NSE of 0.9412, along with an increase in training time. Although the GRU model maintained high accuracy with two layers, further increases in depth (three layers) did not result in significant improvement and, in some cases, led to marginally worse performance. This suggests a diminishing return and the onset of overfitting or excess model capacity.

Table 3 offers a detailed comparison of RMSE, MAE, MAPE, and NSE metrics for each layer configuration in the LSTM and GRU models, providing insight into the impact of layer number on overall performance.

Table 3. Summary of LSTM and GRU model performance across different layers

Model	Layers	RMSE (m)	MAE (m)	MAPE (%)	NSE
LSTM	1	0.0363	0.0219	1.40	0.9011
LSTM	2	0.0397	0.0264	1.68	0.8819
LSTM	3	0.0540	0.0357	2.30	0.7815
GRU	1	0.0299	0.0190	1.21	0.9328
GRU	2	0.0277	0.0180	1.15	0.9422
GRU	3	0.0280	0.0170	1.09	0.9411

A comprehensive illustration of the training process dynamics and prediction accuracy for different layer configurations is provided in Figure 5, which presents the changes in training and validation loss. Figure 6 offers a comparative overview of the RMSE, MAE, MAPE, and NSE metrics for both models. In this figure, each color represents a different model-layer configuration as detailed in the legend, facilitating direct performance comparison across settings.

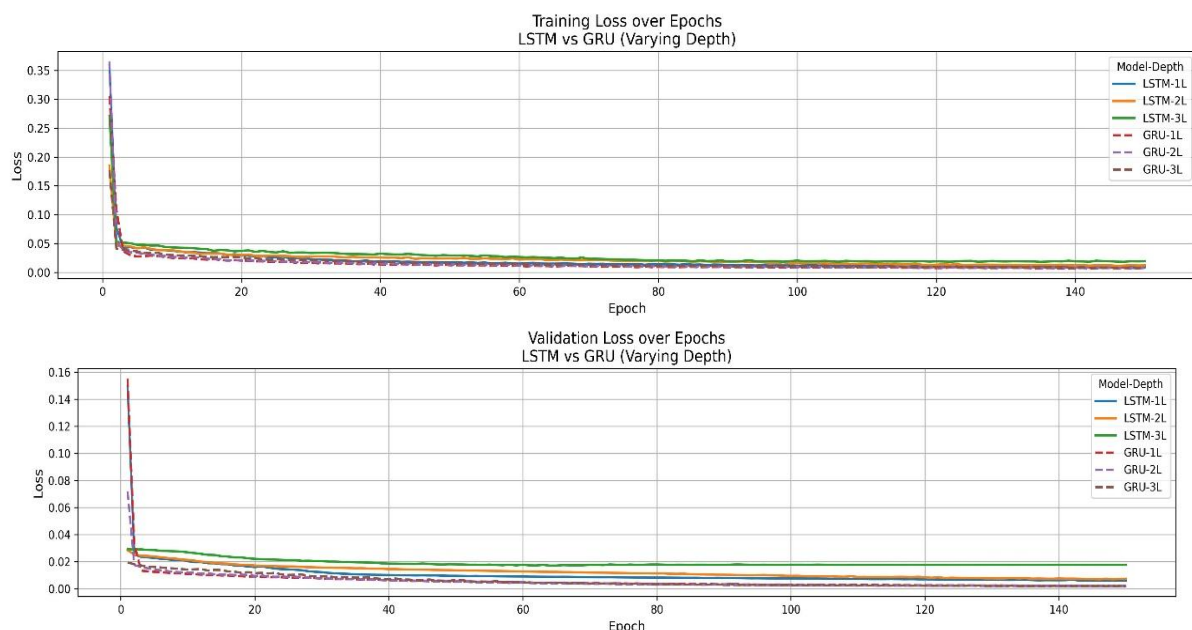


Figure 5. Comparison Training and Validation loss of Depth Layer Effects on LSTM vs GRU

Figure 5 presents the training and validation loss curves for each LSTM and GRU configuration with varying depths, as indicated in the legend. Solid and dashed lines distinguish LSTM and GRU models, respectively. These plots allow readers to compare convergence speed and final loss values across model architectures, as well as to observe signs of overfitting, such as a widening gap between training and validation loss in deeper networks.

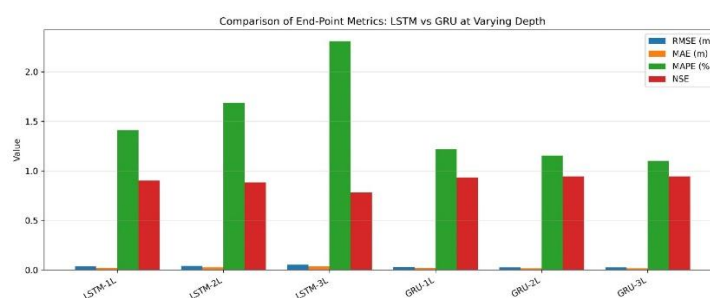


Figure 6. Metrics Performance of LSTM vs GRU

In Figure 6, each color in the bar chart corresponds to a specific number of layers for the LSTM and GRU models, as detailed in the legend. This color coding enables an easy visual comparison of predictive accuracy across all model configurations, helping to highlight which setup delivers the lowest error and the highest NSE for water level prediction.

Overall, the experimental results confirm that increasing the number of layers does not always correspond to improved model accuracy. In this case, the LSTM model achieved its best performance with a single layer, while the GRU model reached optimal results with two layers. Excessive addition of layers can, in fact, lead to reduced accuracy, increased risk of overfitting, and greater computational burden, as demonstrated in the evaluation graphs and tables. These findings highlight the importance of selecting an appropriate architectural configuration in developing time series prediction models to achieve a balance between accuracy, computational efficiency, and model complexity. The best configuration for each model, as determined from the sequential hyperparameter optimization experiments, is summarized in Table 4 and Table 5. This table highlights the optimal window size and number of layers, along with the corresponding evaluation metrics, for both LSTM and GRU models

Table 4. Summary of optimal window size for both LSTM and GRU models

Model	Window Size	RMSE (m)	MAE (m)	MAPE (%)	NSE
LSTM	30	0.0348	0.0223	1.43	0.9090
GRU	20	0.0294	0.0187	1.19	0.9339

Table 5. Summary of optimal layer depth for both LSTM and GRU models

Model	Layer	RMSE (m)	MAE (m)	MAPE (%)	NSE
LSTM	1	0.0363	0.0219	1.40	0.9011
GRU	2	0.0277	0.0180	1.15	0.9422

In this study, hyperparameter optimization was performed in a sequential manner. The window size parameter was first optimized by varying its value (10, 20, 30, etc.) while keeping the number of layers fixed at one. Once the optimal window size was identified, the effect of increasing model depth was investigated by varying the number of layers (1, 2, 3) using the previously determined optimal window size. As a result, the best configurations for each model are reported based on the optimal settings found in these independent experiments, rather than from an exhaustive grid search of all possible parameter combinations.

4. DISCUSSIONS

In this study, a comprehensive analysis was conducted to examine the effects of window size and number of layers on the performance of LSTM and GRU models for river water level prediction based on historical data in Palembang. The experimental results demonstrate that both the selection of window size and the number of layers have a significant impact on model prediction accuracy and computational efficiency.

4.1. Evaluation of Window Size and Its Effect on Accuracy

The experimental results indicate that window size is a key parameter in the development of time series prediction models. For the LSTM model, a gradual increase in window size (10, 20, 30) resulted in improved accuracy, with the best performance achieved at a window size of 30 (RMSE 0.0349 m; MAE 0.0224 m; MAPE 1.43%; NSE 0.9090). In contrast, the GRU model attained optimal accuracy at

a window size of 20 (RMSE 0.0294 m; MAE 0.0187 m; MAPE 1.19%; NSE 0.9339), while further increasing the window size to 30 actually led to decreased performance.

The visualization of the relationship between observed and predicted values for the best configurations of both models is presented in Figure 7. The scatter plot shows that the GRU model (window size 20) exhibits predictions that are more closely distributed along the identity line ($y = x$) compared to the LSTM model (window size 30), indicating higher prediction accuracy for the GRU. This finding is further supported by the evaluation metrics, with the GRU consistently demonstrating lower RMSE and MAE values as well as higher NSE values compared to the LSTM.

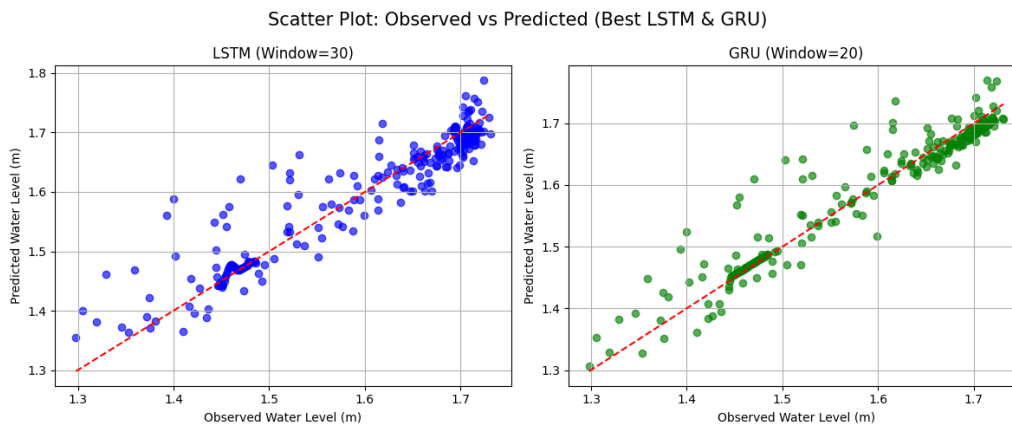


Figure 7. Scatter Plot of Best Window Size

Overall, these results confirm that selecting an appropriate window size is crucial for optimizing model accuracy. This finding is consistent with previous studies that have highlighted the importance of tuning the window size parameter in deep learning models for time series prediction [29], [33]. Furthermore, the trade-off between accuracy and computational efficiency must also be considered, as excessively large window sizes can increase training time without yielding significant performance gains.

4.2. Evaluation of Model Layer Depth and Its Effect on Accuracy

The number of layers (depth) is also an important architectural parameter that affects the capacity and generalization ability of both LSTM and GRU models. The experimental results show that increasing the number of layers does not always correspond to improved model accuracy. For the LSTM model, the single-layer configuration delivered the best performance, while adding two or three layers led to decreased accuracy, higher error values, and an increased risk of overfitting. In the GRU model, optimal performance was achieved with two layers, whereas using three layers did not yield further improvements and even tended to decrease some evaluation metrics.

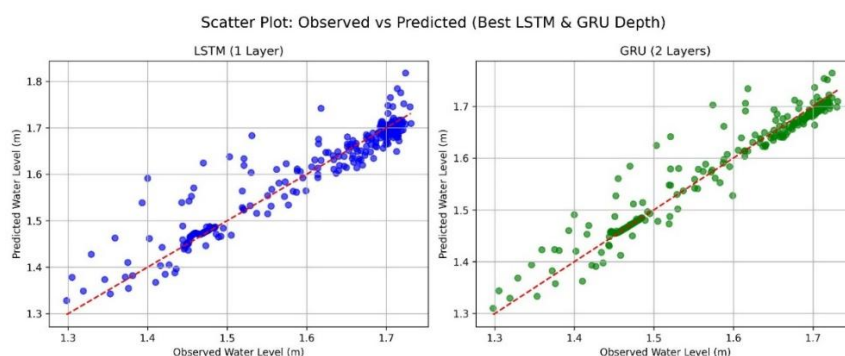


Figure 8. Scatter Plot of Best Layer Depth

A visual representation of the models' generalization ability for each layer configuration is provided in Figure 8, which displays scatter plots of observed versus predicted values for the LSTM model (1 layer) and the GRU model (2 layers), the best-performing combinations based on the evaluation results. These scatter plots illustrate the proximity of the predicted values to the actual data; the closer the points are to the identity line ($y = x$), the better the model's ability to represent the data.

The data distribution in these scatter plots confirms that the single-layer LSTM and two-layer GRU models accurately captured the relationship between input and target variables in the river water level dataset. Nevertheless, some deviations are still present in the predictions of the LSTM (with two and three layers) and GRU (with three layers), indicating the potential for overfitting or model complexity that exceeds the available data volume. This finding is consistent with previous studies, which recommend using relatively simple architectures for time series prediction models especially those based on LSTM and GRU to maintain a balance between accuracy, computational efficiency, and generalization capability [31], [32]. Therefore, the selection of the number of layers should take into account data complexity and the risk of overfitting, rather than simply increasing architectural depth.

This study provides empirical guidance for selecting LSTM and GRU model architectures in hydrological forecasting applications, particularly for near real-time river water level prediction in tropical regions. The findings emphasize the necessity of hyperparameter tuning that considers both model accuracy and computational resources, ensuring that the developed models are not only effective but also practical for operational use.

5. CONCLUSION

This study successfully demonstrates that both window size and the number of layers are critical factors in constructing effective deep learning models for river water level prediction. The experimental findings reveal that the optimal parameter configurations vary between the two architectures: the LSTM model achieves its highest accuracy with a window size of 30 and a single-layer structure, whereas the GRU model performs best with a window size of 20 and two layers. These hyperparameter choices not only enhance prediction accuracy but also significantly affect computational efficiency and the risk of overfitting, emphasizing the need for careful tuning during model development.

This comparative evaluation offers practical guidelines for selecting deep learning configurations in hydrological contexts, enabling researchers and practitioners to balance model complexity, accuracy, and computational resources more effectively. The results strengthen the existing body of literature underscoring the importance of hyperparameter optimization in hydrological time series forecasting. Furthermore, this study provides empirical guidance for selecting model architectures in hydrological applications, particularly for near real-time forecasting scenarios in tropical regions.

To further advance this research, future work is recommended to explore additional hyperparameter settings, incorporate external datasets to improve generalizability, and implement real-time prediction scenarios to validate model adaptability and operational feasibility.

ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to Sriwijaya State Polytechnic for providing the research facilities, with special appreciation to the Telecommunications Laboratory, supported by a dedicated team of academic researchers and technicians. We also wish to convey our heartfelt thanks to the Dinas PUPR Kota Palembang and Balai Besar Wilayah Sungai Sumatera VIII (BBWSS VIII) for their generous provision of the essential data for our research. Furthermore, we extend our deepest appreciation to the Ministry of Higher Education, Science, and Technology for their financial support through the Matching Fund program in 2024. Additionally, we would like to

acknowledge Opole University of Technology, Poland, for their valuable collaboration in our research and publication efforts.

REFERENCES

- [1] B. Wang, B. Wang, W. Wu, C. Xi, and J. Wang, "Sea-water-level prediction via combined wavelet decomposition, neuro-fuzzy and neural networks using SLA and wind information," *Acta Oceanologica Sinica*, vol. 39, no. 5, pp. 157–167, May 2020, doi: 10.1007/s13131-020-1569-1.
- [2] M. Gan, S. Pan, Y. Chen, C. Cheng, H. Pan, and X. Zhu, "Application of the machine learning lightgbm model to the prediction of the water levels of the lower columbia river," *J Mar Sci Eng*, vol. 9, no. 5, May 2021, doi: 10.3390/jmse9050496.
- [3] R. Tur, E. Tas, A. T. Haghighi, and A. D. Mehr, "Sea level prediction using machine learning," *Water (Switzerland)*, vol. 13, no. 24, Dec. 2021, doi: 10.3390/w13243566.
- [4] A. M. Mihel, J. Lerga, and N. Krvavica, "Estimating water levels and discharges in tidal rivers and estuaries: Review of machine learning approaches," May 01, 2024, *Elsevier Ltd.* doi: 10.1016/j.envsoft.2024.106033.
- [5] H. Li, L. Zhang, Y. Zhang, Y. Yao, R. Wang, and Y. Dai, "Water-Level Prediction Analysis for the Three Gorges Reservoir Area Based on a Hybrid Model of LSTM and Its Variants," *Water (Switzerland)*, vol. 16, no. 9, May 2024, doi: 10.3390/w16091227.
- [6] M. Li *et al.*, "Prediction of water level at Huayuankou station based on rating curve," *Sci Rep*, vol. 14, no. 1, Dec. 2024, doi: 10.1038/s41598-024-71510-3.
- [7] Q. Wang and S. Wang, "Machine learning-based water level prediction in lake erie," *Water (Switzerland)*, vol. 12, no. 10, pp. 1–14, Oct. 2020, doi: 10.3390/w12102654.
- [8] V. H. Nhu *et al.*, "Daily water level prediction of zrebar lake (Iran): A comparison between m5p, random forest, random tree and reduced error pruning trees algorithms," *ISPRS Int J Geoinf*, vol. 9, no. 8, Aug. 2020, doi: 10.3390/ijgi9080479.
- [9] Y. Ding, Y. Zhu, J. Feng, P. Zhang, and Z. Cheng, "Interpretable spatio-temporal attention LSTM model for flood forecasting," *Neurocomputing*, vol. 403, pp. 348–359, Aug. 2020, doi: 10.1016/j.neucom.2020.04.110.
- [10] M. Cho, C. Kim, K. Jung, and H. Jung, "Water Level Prediction Model Applying a Long Short-Term Memory (LSTM)–Gated Recurrent Unit (GRU) Method for Flood Prediction," *Water (Switzerland)*, vol. 14, no. 14, Jul. 2022, doi: 10.3390/w14142221.
- [11] B. Li, G. Yang, R. Wan, X. Dai, and Y. Zhang, "Comparison of random forests and other statistical methods for the prediction of lake water level: A case study of the Poyang Lake in China," *Hydrology Research*, vol. 47, pp. 69–83, Dec. 2016, doi: 10.2166/nh.2016.264.
- [12] M. Pan *et al.*, "Water Level Prediction Model Based on GRU and CNN," *IEEE Access*, vol. 8, pp. 60090–60100, 2020, doi: 10.1109/ACCESS.2020.2982433.
- [13] C. S. Kim, C. R. Kim, K. H. Kok, and J. M. Lee, "Water Level Prediction and Forecasting Using a Long Short-Term Memory Model for Nam Ngum River Basin in Lao PDR," *Water (Switzerland)*, vol. 16, no. 13, Jul. 2024, doi: 10.3390/w16131777.
- [14] K. Park, Y. Seong, Y. Jung, I. Youn, and C. K. Choi, "Development of Water Level Prediction Improvement Method Using Multivariate Time Series Data by GRU Model," *Water (Switzerland)*, vol. 15, no. 3, Feb. 2023, doi: 10.3390/w15030587.
- [15] V. Atashi, H. T. Gorji, S. M. Shahabi, R. Kardan, and Y. H. Lim, "Water Level Forecasting Using Deep Learning Time-Series Analysis: A Case Study of Red River of the North," *Water (Switzerland)*, vol. 14, no. 12, Jun. 2022, doi: 10.3390/w14121971.
- [16] Y. Liu, H. Wang, W. Feng, and H. Huang, "Short term real-time rolling forecast of urban river water levels based on lstm: A case study in Fuzhou city, China," *Int J Environ Res Public Health*, vol. 18, no. 17, Sep. 2021, doi: 10.3390/ijerph18179287.
- [17] V. Gude, S. Corns, and S. Long, "Flood Prediction and Uncertainty Estimation Using Deep Learning," *Water (Switzerland)*, vol. 12, no. 3, Mar. 2020, doi: 10.3390/w12030884.

-
- [18] M. M. Ahsan, M. A. P. Mahmud, P. K. Saha, K. D. Gupta, and Z. Siddique, "Effect of Data Scaling Methods on Machine Learning Algorithms and Model Performance," *Technologies (Basel)*, vol. 9, no. 3, Sep. 2021, doi: 10.3390/technologies9030052.
- [19] A. Kumar Dubey, A. Kumar, V. García-Díaz, A. Kumar Sharma, and K. Kanhaiya, "Study and analysis of SARIMA and LSTM in forecasting time series data," *Sustainable Energy Technologies and Assessments*, vol. 47, Oct. 2021, doi: 10.1016/j.seta.2021.101474.
- [20] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
- [21] T. Y. Kim and S. B. Cho, "Predicting residential energy consumption using CNN-LSTM neural networks," *Energy*, vol. 182, pp. 72–81, Sep. 2019, doi: 10.1016/j.energy.2019.05.230.
- [22] K. Singh and J. Malhotra, "Two-layer LSTM network-based prediction of epileptic seizures using EEG spectral features," *Complex and Intelligent Systems*, vol. 8, no. 3, pp. 2405–2418, Jun. 2022, doi: 10.1007/s40747-021-00627-z.
- [23] P. Oliveira, B. Fernandes, C. Analide, and P. Novais, "Forecasting energy consumption of wastewater treatment plants with a transfer learning approach for sustainable cities," *Electronics (Switzerland)*, vol. 10, no. 10, May 2021, doi: 10.3390/electronics10101149.
- [24] W. Li, H. Wu, N. Zhu, Y. Jiang, J. Tan, and Y. Guo, "Prediction of dissolved oxygen in a fishery pond based on gated recurrent unit (GRU)," *Information Processing in Agriculture*, vol. 8, no. 1, pp. 185–193, Mar. 2021, doi: 10.1016/j.inpa.2020.02.002.
- [25] A. H. Kamel, H. A. Afan, M. Sherif, A. N. Ahmed, and A. El-Shafie, "RBFNN versus GRNN modeling approach for sub-surface evaporation rate prediction in arid region," *Sustainable Computing: Informatics and Systems*, vol. 30, Jun. 2021, doi: 10.1016/j.suscom.2021.100514.
- [26] V. Someetheram, M. F. Marsani, M. S. M. Kasihmuddin, S. Z. M. Jamaludin, M. A. Mansor, and N. E. Zamri, "Hybrid double ensemble empirical mode decomposition and K-Nearest Neighbors model with improved particle swarm optimization for water level forecasting," *Alexandria Engineering Journal*, vol. 115, pp. 423–433, Mar. 2025, doi: 10.1016/j.aej.2024.12.035.
- [27] A. I. Pathan *et al.*, "Comparative assessment of rainfall-based water level prediction using machine learning (ML) techniques," *Ain Shams Engineering Journal*, vol. 15, no. 7, Jul. 2024, doi: 10.1016/j.asej.2024.102854.
- [28] J. F. Ruma, M. S. G. Adnan, A. Dewan, and R. M. Rahman, "Particle swarm optimization based LSTM networks for water level forecasting: A case study on Bangladesh river network," *Results in Engineering*, vol. 17, Mar. 2023, doi: 10.1016/j.rineng.2023.100951.
- [29] L. Dong *et al.*, "Prediction of streamflow based on dynamic sliding window lstm," *Water (Switzerland)*, vol. 12, no. 11, pp. 1–11, Nov. 2020, doi: 10.3390/w12113032.
- [30] D. Tomar, P. Tomar, A. Bhardwaj, and G. R. Sinha, "Deep Learning Neural Network Prediction System Enhanced with Best Window Size in Sliding Window Algorithm for Predicting Domestic Power Consumption in a Residential Building," *Comput Intell Neurosci*, vol. 2022, 2022, doi: 10.1155/2022/7216959.
- [31] A. G. Salman, Y. Heryadi, E. Abdurahman, and W. Suparta, "Single Layer & Multi-layer Long Short-Term Memory (LSTM) Model with Intermediate Variables for Weather Forecasting," in *Procedia Computer Science*, Elsevier B.V., 2018, pp. 89–98. doi: 10.1016/j.procs.2018.08.153.
- [32] B. S. Kwon, R. J. Park, and K. Bin Song, "Short-Term Load Forecasting Based on Deep Neural Networks Using LSTM Layer," *Journal of Electrical Engineering and Technology*, vol. 15, no. 4, pp. 1501–1509, Jul. 2020, doi: 10.1007/s42835-020-00424-7.
-