International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Forecasting of Short-Term Weather Parameters Using Attention-Based Recurrent Neural Network Model

Ni Ketut Intan Rahayu*[1], Gede Putra Kusuma[1]

Submitted: 05/04/2026 **Revised:** 15/05/2026 **Accepted:** 28/05/2026

Abstract: Short-term weather forecasting refers to the prediction of meteorological conditions over a relatively short period. The development of short-term weather forecasting is assessed as having the potential to facilitate the development of dynamic models or methods within the local weather forecasting system. In this research, we propose the incorporation of an attention mechanism into the encoder-decoder architecture of Recurrent Neural Network (RNN)-based models. The purpose of adding this attention mechanism is to enable the model to acquire knowledge regarding the interdependencies among weather parameters, thereby facilitating the capture of abrupt weather fluctuations. This research specifically focuses on the prediction of several weather parameters, including temperature, relative humidity, and wind speed. In this study, a performance comparison was conducted among several types of RNN-based models, namely Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Bidirectional LSTM (Bi-LSTM), Bidirectional GRU (Bi-GRU), and the combination of each RNN model with an attention mechanism. The research results indicate that the model with the best performance is the Attention GRU for temperature prediction, with an RMSE value of 0.02681. For relative humidity prediction, the Attention Bi-LSTM performs the best with an RMSE value of 0.18343, and the Attention Bi-GRU achieves the highest performance for wind speed prediction with an RMSE value of 0.00395. The outcomes of this investigation demonstrate the efficacy of the attention mechanism in enhancing the accuracy of several encoder-decoder RNN models.

Keywords: *Attention mechanism, Encoder-decoder architecture, Recurrent Neural Network, Short-term weather forecasting, Weather parameters interdependencies*

1. Introduction

Weather parameter forecasting is the prediction of atmospheric conditions within a specific temporal and spatial domain through the utilization of scientific methodologies and technological advancements [1]. Weather parameter forecasting is one type of time series forecasting that predicts the future values of a target $y_{i,t}$ for a given i at time t [2]. The forecast result can be a map of a particular weather variable, a time series of one or more weather variables at a specific location or aggregated over a region, some aggregate statistics of a particular variable over a certain time range, or an index of categorical warnings [3]. Methods for weather parameter forecasting that utilize numerical models are designed to operate at specific spatial scales depending on the forecasted weather timeframe. These methods can be categorized based on the forecasted weather timeframes as follows [4]:

- Short-term weather parameter forecasting predicts weather conditions up to a maximum of 3 days in advance.
- Medium-term weather parameter forecasting predicts weather conditions up to 10 days in advance.
- Long-term weather parameter forecasting predicts weather conditions from 10 to 30 days in advance.

Under the circumstances of extreme weather, the likelihood of natural disasters escalates. Storm related disasters are among the most frequent occurrences in Indonesia [6]. Consequently, wind speed assumes paramount importance as a weather parameter to forecast, aiming to alleviate the detrimental impacts of storms. Temperature, another influential weather parameter, exerts a profound influence on various aspects of human life, including sustainable development, environmental protection, agriculture, water resource management, and early weather warnings [7]. Additionally, air humidity, as a weather parameter, plays a pivotal role in determining air quality levels [8]. Accurate information concerning air quality becomes particularly significant in regions susceptible to forest fires, such as Riau. Based on these explanations, this research focuses on the prediction of weather parameters, specifically temperature, wind speed, and air humidity.

Weather forecasting globally predominantly relies on the Numerical Weather Prediction (NWP) model. However, NWP is considered to require high computational resources. Deep learning can serve as an alternative method for weather forecasting processes. Deep learning relies on artificial neural networks that mimic the functioning of the human brain through interconnected artificial neurons, enabling communication between neurons [10]. Deep learning presents an alternative approach for weather forecasting tasks, exhibiting success in handling extensive datasets such as long-term weather parameters. One of the deep learning models, the Recurrent Neural Network (RNN), is specifically designed for learning time-dependent data features [3]. P. Hewage

1 Computer Science Department, BINUS Graduate Program – Master of Computer Science, Bina Nusantara University, Jakarta – 11480, Indonesia.

ORCID ID: 0009-0004-7793-628

** Corresponding Author Email: ni.rahayu002@binus.ac.id*

compared the accuracy of short- term weather forecasts derived from the Weather Research and Forecasting (WRF) model, an NWP model, with those obtained using the Long Short- Term Memory (LSTM) model, an RNN variant. The results demonstrated that LSTM-based short-term weather forecasting exhibited improved computational efficiency and lower Mean Squared Error (MSE) compared to WRF model forecasts [11]. Masooma A. R. S (2022) compared several RNN architectures, including Transductive LSTM, Bi-LSTM, ConvLSTM, and Spatial Feature Attention based LSTM. The research findings indicated that weather forecasts were less accurate when sudden changes occurred in weather parameters [12]. Meteorological studies have shown that changes in weather conditions are caused by variations in interrelated weather parameters. Accordingly, this research proposes the use of an attention mechanism to capture sudden weather changes by facilitating the learning of interactions among weather parameters. The attention mechanism assigns varying weights to input weather parameter data based on the model's evaluation of which weather parameters necessitate greater focus [12]. In this research, the addition of an attention mechanism is proposed to enhance the performance of encoder-decoder models in RNN. The RNN models utilized in this research include Long Short-Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), Gated Recurrent Unit, and Bidirectional GRU (Bi-GRU). The objective of incorporating the attention mechanism is to enable the model to capture sudden weather changes, considering the dynamic nature of weather data.

2. Literature Review

2.1. Related Works

In the advancement of modern weather forecasting methods, numerous studies have been conducted concerning the implementation of deep learning in weather prediction systems, aiming to establish more accurate weather forecasting systems. Several researchers have demonstrated that deep learning methods exhibit superior performance in weather parameter prediction compared to statistical models [13], [14]. LeeCun and Hinton also proved that in weather forecasting models, deep learning methods yield models with superior performance compared to machine learning methods [15]. In their research, deep neural networks are seen to outperform the classical Support Vector Regression Model. The LSTM model has been deemed effective for weather parameter forecasting. However, it has limitations in accurately predicting weather parameters when sudden changes occur. A.R Suleman conducted a study on short-term weather parameter forecasting using the Spatial Feature Attention based LSTM model [12]. The research dataset was obtained from the Saskatoon John G. Diefenbaker Intl. Airport weather observation station, covering the data period from January 2012 to December 2021. The dataset includes various weather parameters such as temperature, dew points, wind chill, relative humidity, air pressure, sea level pressure, and wind speed. The study captured accurate spatial and temporal relationships of multiple meteorological features to forecast air temperature by adding a spatial feature attention module in the decoder. This module aimed to capture spatial features from the input sequence and detect sudden changes in weather parameter values. The research results showed that the attention-based LSTM outperformed the LSTM model with MSE and MAE values of 0.0871 and 0.2317, respectively.

2.2. Recurrent Neural Network (RNN)

RNN is one type of Artificial Neural Network (ANN) whose neurons send feedback signals to each other [18]. RNN demonstrates the capability to learn features and long-term dependencies from sequential and time-series data [19]. A simple RNN consists of an input layer, a recurrent hidden layer, and an output layer. The input layer is composed of N input units, where the input in this layer is a sequential vector series through time $t\{\dots, x_{t-1}, x_t, x_{t+1}\}$ where $x_t = (x_1, x_2, \dots, x_N)$. The connection between the input unit and the hidden unit is defined by the weight matrix W_{IH}. The hidden layer comprises M hidden units $h_t = (h_1, h_2, \dots, h_M)$, with each hidden unit being interconnected. The hidden layer defines the 'memory' of the system as follows:

$$h_t = f_H(o_t) \tag{1}$$

where:

$$o_t = W_{IHx_t} + W_{HHh_{t-1}} + b_h \tag{2}$$

$f_H(.)$ represents the activation function of the hidden layer, and b_h is the bias vector value of the hidden units. The hidden units are connected to the output layer through weight connections denoted by W_{HO}. The output layer consists of P units $y_t = y_1, y_2, \dots, y_P$, which are calculated using the following equations:

$$y_t = f_O(W_{HO}h_t + b_O) \tag{3}$$

$f_O(.)$ represent the activation function of the output layer and b_O is the bias vector value of the output layer.

There are several types of RNN architectures, including: Deep RNN, recurrent memory networks, bidirectional RNN, structurally constrained RNN, unitary RNN, gated orthogonal recurrent unit, multidimensional RNN, Long Short-Term Memory (LSTM), Gated Recurrent Unit, and hierarchical subsampling RNN [19]. LSTM is a type of Recurrent Neural Network that possesses the capability to learn long-term dependencies, thereby mitigating the gradient vanishing issue encountered in traditional RNNs [20]. LSTM retains information over long periods of time due to the presence of inner cells, which enable the preservation and transportation of information without altering its content [21]. The bidirectional architecture of LSTM, known as Bi-LSTM, represents one of the RNN models that allows for additional training processes by traversing the input data [20]. The Bi-LSTM architecture employs two LSTM models applied to the input data. In the first stage, an LSTM is applied to the input sequence (feedback layer), and then, in the second stage, the reverse form of the input data is fed into another LSTM (backward layer). Another architecture of Recurrent Neural Networks (RNNs) is the Gated Recurrent Unit (GRU), which consists of gating units, without distinct memory cells, that influence the flow of information within the unit [19]. These gating units consist of an update gate z_t and a reset gate r_t. The main distinction between GRU and LSTM lies in the fact that GRU has only two gates and lacks a cell state, which is present in LSTM [22]. The bidirectional architecture of Gated Recurrent Unit (GRU), known as Bi-GRU, represents another type of Recurrent Neural Network (RNN) model consisting of two GRU models, each processing data in both forward and backward directions. The structure of Bi-GRU is determined by the interplay of two unidirectional states with opposing directions [23]. One GRU moving forward starts the process from the initial data, while the other GRU moving backward initiates the process from the

final data. This process enables information from both the future and the past to influence the current state [24].

2.3. Attention Based Encoder-Decoder RNN

The attention mechanism was originally introduced in neural machine translation systems to address challenges associated with the encoder-decoder architecture, where the encoder compresses a sentence into a fixed-length vector [25]. This limitation affects the performance of neural network models when dealing with input sequences of varying lengths, as the translation output needs to be condensed into a fixed-length vector, regardless of the input sequence length. To overcome this issue, an extension component was introduced in the encoder-decoder architecture, which enables the prediction of words based on a context vector that considers the source data position and previously generated target words. The attention mechanism differs from traditional encoder-decoder architectures in that it does not translate the entire input into a fixed-length vector. Instead, it translates the input data into a sequence of vectors and selectively focuses on a subset of these vectors while decoding the translation output [25]. The encoder produces a context vector, which serves as input to the decoder. The context vector enables the decoder to concentrate on the most relevant information during the translation process. Each context vector is a weighted sum of the encoder inputs, reflecting their respective importance in generating the output. This research utilizes a dot-product attention mechanism, which is known for its computational efficiency achieved through highly optimized matrix multiplication code [26]. The attention mechanism comprises three key components: Queries (Q), Keys (K), and Values (V). The dot product is employed to measure the similarity between the query and key vectors. The mathematical equations for computing these three components are as follows:

- Each query vector $q = s_{(t-1)}$ is compared with the key vectors in the database to compute the score value. This comparison is accomplished through the dot product operation between each query and key vector. Mathematically, it can be expressed as follows:

$$x_{(q,k_i)} = q \cdot k_i \quad (4)$$

- The attention weight is obtained as the result of the score value $e(q, k_i)$ and is then passed through the Softmax function to obtain a normalized score value.

$$\alpha_{(a,k_i)} = softmax(e_{a,k_i}) \quad (5)$$

- The context vector is then computed as the weighted sum of the value vectors $V_{(k_i)}$, where each value vector is paired with its corresponding key.

$$attention(q, K, V) = \alpha_{(q,k_i)} V_{(k_i)} \quad (6)$$

3. Data and Methods

3.1. Datasets

In this research, the historical daily weather data in Pekanbaru-Riau, positioned at longitude 101.4190 E and latitude 0.4920 N was utilized. The data, which consists of actual weather data recorded from January 1, 2012, to January 1, 2022, comprises 3,654 rows. The data was sourced from the online data repository of the Indonesia Meteorology, Climatology, and Geophysics Agency (BMKG). Historical weather data comprises various weather parameters, which served as inputs for different recurrent neural network (RNN) models. The data includes the following information: date, month, year; temperature (^{0}C); relative humidity (%); wind speed ($^m/_s$); and sunshine duration (hour).

3.2. Methodology

Fig. 1 illustrates the detailed workflow of the models. The model design process begins with data collection from the online data repository of the Indonesia Meteorology, Climatology, and Geophysics Agency (BMKG). The subsequent steps involved imputing missing values using the linear interpolation technique and normalizing the data. In this research, multiple Recurrent Neural Network (RNN)-based models were used, including Long Short-Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), Gated Recurrent Unit (GRU), and Bidirectional GRU (Bi-GRU), both with and without the incorporation of an attention mechanism. The final step of this study was to evaluate the performance of each RNN model with an attention mechanism and compare it with RNN models without an attention mechanism. The objective of this research is to assess whether the attention mechanism can improve the performance of the RNN models.

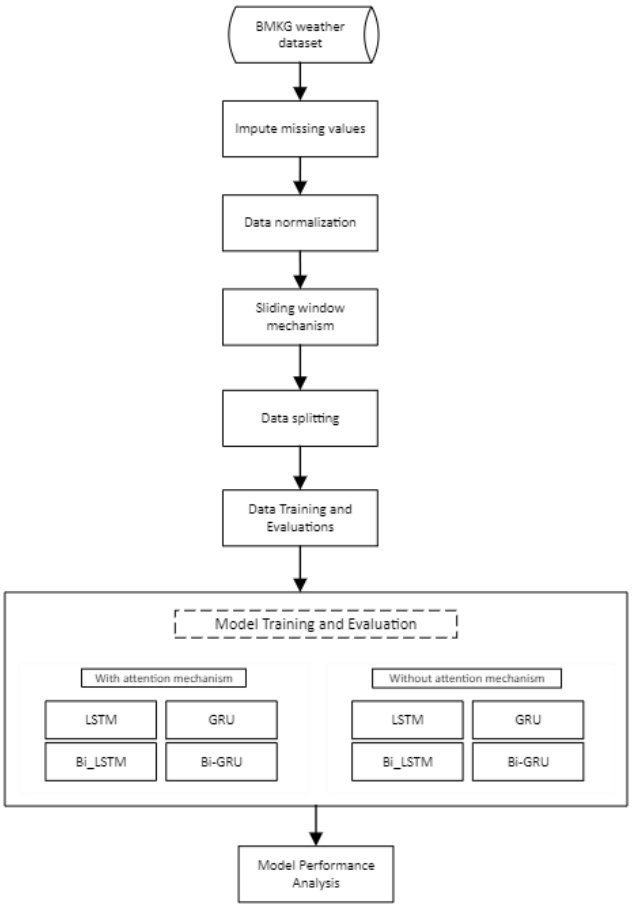

Fig. 1. Model Workflow

3.2.1. Data Preprocessing

The missing values in the data were imputed using the linear interpolation method in the forward direction. In this technique, missing values were filled in by using the values that were available in the dataset prior to the missing value. The missing value was interpolated based on the trend or pattern observed in the preceding values. This approach assumes a linear relationship between consecutive data points and extends the trend to estimate the missing value.

Weather parameter data can exhibit varying value ranges. For instance, temperature data may span from 25 to 30, while relative humidity percentages may range from 51 to 98. Data normalization is employed to standardize the data within a consistent value range, commonly between 0 and 1. In this research, the Min-Max normalization technique was utilized for data normalization. The mathematical equation utilized for Min-Max normalization is expressed as follows:

$$\tilde{x} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{7}$$

Where $\tilde{x}$ represents the normalized training data value, x_{min} represents the lowest value of the training data, and x_{max} represents the highest value of the training data.

In this research, a sliding window size of 30 was employed to forecast the data for the subsequent time step. Weather patterns often exhibit short-term variations and periodicities. By using a window size equal to the approximate number of days in a month, it ensures the model has enough historical data to learn from. Fig. 2 provides an illustration of the sliding windows, where a sliding window size of 30 was utilized to predict the weather parameter value at the next 1-day time step. As demonstrated in Fig. 2, the input features consist of a vector x_1 through x_{30} representing the weather parameter data. These features were employed to predict the label data at the next 1-day time step, denoted as x_{31}, which corresponds to the actual unnormalized weather parameter value.

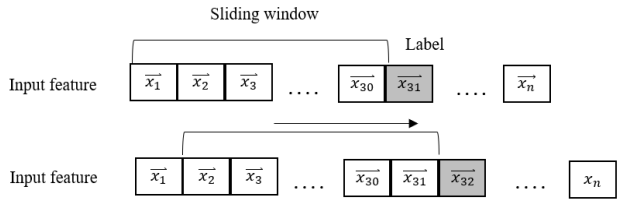

Fig. 2. Sliding Window with the size of 30 to predict the next 1-day time step.

In this study, four types of weather parameters, namely temperature, relative humidity, wind speed, and sunshine duration, were utilized. These four weather parameters were subsequently treated as input sequences for the model. The size of a single input sequence is represented as a vector with dimensions of $S \times F$, where S corresponds to the sliding window size of 30, and F signifies the number of input features, which totals 4. The partitioning of time series data into training, validation, and test sets was not conducted randomly but rather based on chronological or sequential order according to time. The dataset was divided into proportions of 60% training data (covering the period from 2012 to early 2018), 20% validation data (spanning from 2018 to early 2020), and 20% test data (encompassing the period from 2020 to early 2022). The training data was utilized to train multiple RNN models with their corresponding hyperparameter values. The performance of these RNN models was subsequently assessed using validation data to detect any signs of overfitting. Once it was ensured that overfitting was not present, the RNN models were subjected to further testing using the test data. The test data was employed to evaluate the forecasting model for weather parameters.

3.2.2. Proposed Models

In this research, the addition of an attention mechanism to the encoder-decoder architecture of multiple RNN models is proposed. The objective is to enhance the model's ability to focus on the most relevant weather parameter information, beyond solely relying on the last hidden state. This addition aims to improve the accuracy of weather parameter forecasting models. The attention module was positioned between the encoder and decoder modules of the RNN. Its output is a context vector, enabling the decoder to concentrate on specific segments of the input sequence during the prediction process.

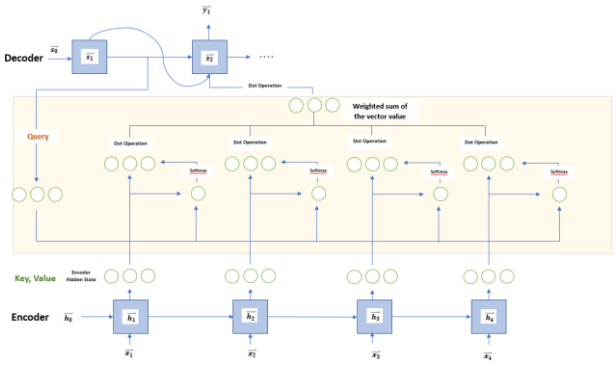

Fig. 3. Attention Mechanism

The initial step involved calculating the score or weight values, which were scalar values. These scores were computed by taking the dot product between each key value (representing the hidden states of the encoder) and the query value (representing the hidden states of the decoder). The resulting scores were then passed through the Softmax function. In Fig. 3, the key and value values corresponded to the encoder's hidden states, while the query value corresponded to the decoder's hidden state. Subsequently, each score value was multiplied by its corresponding value vector to generate the context vector. This context vector combined all the hidden states from the input, representing a weighted sum of the value vectors.

The architecture of the attention based RNN is depicted in Fig. 4. The input layer represents the input sequence data, which has a length of 30 in accordance with the sliding window size. The number of neurons in the input layer is equal to the number of input features, which is 4 in this research. The input data was processed through the attention-based encoder and decoder. Specifically, a single hidden layer RNN was utilized in the encoder, while a single hidden layer RNN was employed in the decoder. The output of the attention-based RNN encoder-decoder is a vector with a size corresponding to the number of hidden states. Dropout layers were incorporated to mitigate overfitting. The output from the dense layer was then forwarded to the output layer of the model, which consists of a single node responsible for predicting a specific type of weather parameter at the target time step.

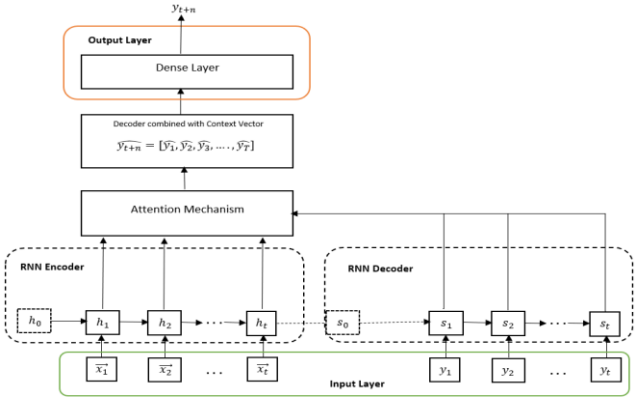

Fig. 4. Model Architecture

3.2.3. Model Training

We utilized the processed dataset, which included four weather variables as described earlier, to make predictions for temperature, humidity, and wind speed. The attention-based encoder-decoder recurrent neural network (RNN) received the four weather variable sequences as input variables. Subsequently, three distinct models were employed to predict the output variables, namely temperature, humidity, and wind speed.

Prior to the training phase, a hyperparameter optimization process was performed to determine the optimal combination of hyperparameters for the RNN models. Hyperparameters play a crucial role in determining the model's performance. Manual hyperparameter search is deemed inefficient due to its computational time requirements [27]. To address this issue, this research employed automated Hyperparameter Optimization using the Bayesian Optimization algorithm. In particular, Bayesian Optimization first presupposes a functional link between the values of the hyperparameters and the loss function that needs to be optimized [28], formulated in the following equation:

$$p^* = argminloss(p);\ p \in P \tag{8}$$

Where P is the set of all hyperparameters, and p is the set of hyperparameter combinations within P. p^* represents the most optimal hyperparameter combination, and $loss\ (.)$ is the objective function that needs to be optimized. The loss function used in this research is Mean Square Error (MSE). Bayesian optimization involves constructing a probability model, or Surrogate Model $P\ (y|x)$ of the objective function based on previous observations. This model is utilized to identify the most optimal hyperparameter values using the information obtained from previous observations. Bayesian optimization builds upon Random Search optimization by considering the evaluation of hyperparameter configurations from the previous observations. Evaluating hyperparameter configurations from past observations aids the search by focusing on areas with the most promising hyperparameters value.

The hyperparameters targeted for optimization in this research include the dimension of the RNN hidden units, dropout rate, batch size, learning rate, and the type of model optimizer. Keras Tuner was employed to facilitate the looping process for the exploration of the most optimal hyperparameter values. The looping process iterated over the predefined range of hyperparameter values until it identified the combination that yielded the highest performance for the RNN model. Table 1 contains the initial value and domain space of the Hyperparameters.

The combination of hyperparameter values from each model that yielded the smallest MSE was chosen for application in the final model during the model training process. During the model training process, the early stopping method was utilized with an epoch limit set to 100 and a patience value of 5. The epoch limit defines the maximum number of epochs allowed in the training procedure. Additionally, the patience value represents the maximum number of epochs permitted after observing no improvement in the validation error.

Table 1 Domain Range of Hyperparameters

Hyperparameter	Domain Space
RNN Hidden Unit Dimension	Min Value: 32; Max Value: 64
Dropout Rate	Min Value: 0.0; Max Value: 0.3
Batch Size	Min Value: 32; Max Value: 64
Optimizers	Adam, RMSProp

3.2.4. Model Evaluation

In this study, RNN models were developed both with and without an attention mechanism. Additionally, several RNN models incorporating various combinations of attention mechanisms were compared against RNN models without attention mechanisms. This comparative analysis aims to assess whether the inclusion of an attention mechanism enhances the performance of weather parameter forecasting models. The performance metric used to evaluate the model's performance is Root Mean Square Error (RMSE). RMSE has been utilized as a statistical metric to assess model performance in the field of meteorology and climate research [29]. RMSE is a performance measure that quantifies the standard deviation of residuals, which represent the discrepancies between predicted and observed values. By measuring the spread of residuals around the regression line, RMSE provides insights into the extent of model fit. The RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_{pred} - y_{test})^2}{n}} \tag{9}$$

y_{pred} represents the weather parameter forecast values, which are the outputs of the model, while y_{test} corresponds to the test data values. The weather forecasting model predicts weather parameters, including temperature, air humidity, and wind speed, with time steps of 1, 2, and 3 days ahead. The evaluation of weather parameter forecast outputs was performed with varying timestamps to analyze potential performance degradation in predicting weather parameters over extended time horizons.

4. Result and Discussion

This research proposes the addition of an attention mechanism to several encoder-decoder RNN-based models mentioned above. The performance of these models was evaluated using the RMSE performance metric. LSTM, Bi-LSTM, GRU, and Bi-GRU models were developed with and without the inclusion of the attention mechanism. Each model was trained to predict temperature, humidity, and wind speed. Furthermore, the models were utilized to forecast output variables at various time steps.

4.1. Result of Hyperparameter Tuning

The values of the hyperparameters utilized in the implemented models and the RMSE score of the validation data are presented in Table 2. From the results of the training process, the models that yielded the lowest RMSE values in the validation data for temperature forecasting, relative humidity, and wind speed, respectively, are Bi-GRU, Attention Bi-LSTM, and Bi-LSTM. The training and validation data curve during the training process of each model is illustrated in Fig. 5, 6, 7. The figures represent the performance of a model during the training process. The x-axis represents the number of epochs, while the y-axis represents the evaluation metric used during the training process, which is Mean Square Error (MSE). The figures show good-fit learning curves, as identified by both the training and validation curves exhibited a decreasing trend over the course of training iterations or epochs. The plot of validation loss decreased to a point of stability and had a small gap with the training loss. These conditions indicated that the model was effectively learning and improving its performance on the training data.

Table 2 Hyperparameters Value and Data Validation RMSE Score

Output Variable	Models	Hyperparameters	Data Validation RMSE		
			1-day ahead	2-days ahead	3-days ahead
Temperature	Attention LSTM	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.03887	0.03743	0.03754
	LSTM	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.07514	0.07233	0.0723
	Attention GRU	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.02845	0.02578	0.02585
	GRU	Hidden Unit: 32; Dropout Rate: 0.2; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.39564	0.51208	0.51222
	Attention Bi-LSTM	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.02609	0.02763	0.02769
	Bi-LSTM	Hidden Unit: 32; Dropout Rate: 0.1; Batch Size: 64; Learning Rate: 0.001; Optimizer: rmsprop;	0.51221	0.4017	0.4058
	Attention Bi-GRU	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.01; Optimizer: adam;	0.02266	0.02107	0.02134
	Bi-GRU	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.02041	0.01932	0.0193
Relative Humidity	Attention LSTM	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.40784	0.40783	0.40776
	LSTM	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 64; Learning Rate: 0.01; Optimizer: adam;	0.53356	0.53357	0.53363
	Attention GRU	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.28877	0.28887	0.28894
	GRU	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 64; Learning Rate: 0.01; Optimizer: adam;	1.5209	1.5209	1.52082
	Attention Bi-LSTM	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.16592	0.16594	0.16593
	Bi-LSTM	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.01; Optimizer: rmsprop;	2.95698	3.01198	3.03783
	Attention Bi-GRU	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.23107	0.23105	0.23119
	Bi-GRU	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 64; Learning Rate: 0.01; Optimizer: adam;	0.48874	0.48873	0.48873
Windspeed	Attention LSTM	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.001; Optimizer: adam;	0.00848	0.0086	0.00856
	LSTM	Hidden Unit: 64; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.01; Optimizer: adam;	0.01069	0.01075	0.0107
	Attention GRU	Hidden Unit: 64; Dropout Rate: 0; Batch Size: 48; Learning Rate: 0.001; Optimizer: adam;	0.0237	0.02369	0.02369
	GRU	Hidden Unit: 32; Dropout Rate: 0.2; Batch Size: 32; Learning Rate: 0.01; Optimizer: rmsprop;	0.0592	0.05923	0.05924
	Attention Bi-LSTM	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 16; Learning Rate: 0.01; Optimizer: adam;	0.00442	0.0044	0.00444
	Bi-LSTM	Hidden Unit: 48; Dropout Rate: 0; Batch Size: 64; Learning Rate: 0.01; Optimizer: adam;	0.00328	0.00336	0.00346
	Attention Bi-GRU	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 32; Learning Rate: 0.01; Optimizer: adam;	0.00471	0.00472	0.0048
	Bi-GRU	Hidden Unit: 32; Dropout Rate: 0; Batch Size: 64; Learning Rate: 0.001; Optimizer: adam;	0.01873	0.01885	0.01893

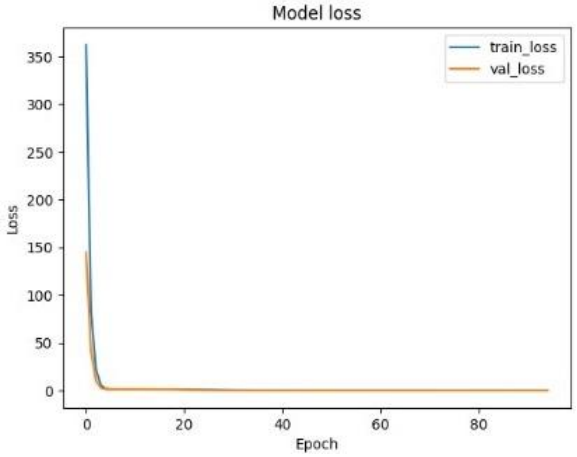

Fig. 5. Training & Validation Loss Curve of Temperature Forecasting using Bi-GRU Model

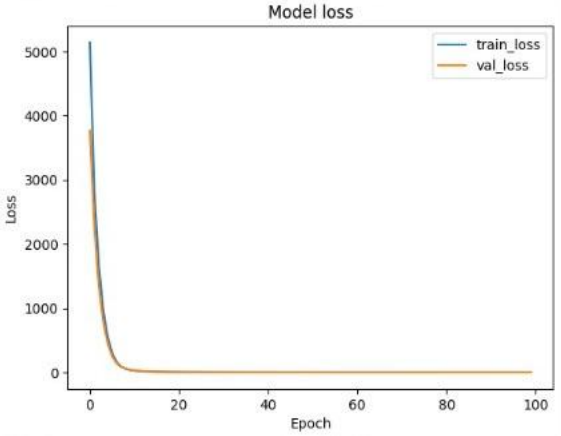

Fig. 6. Training & Validation Loss Curve of Relative Humidity Forecasting using Attention Bi-LSTM Model

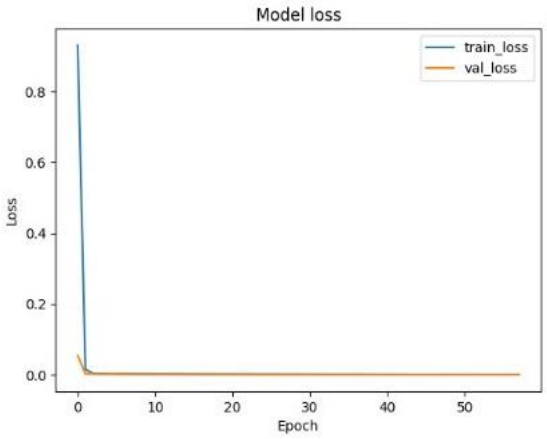

Fig. 7. Training & Validation Loss Curve of Windspeed Forecasting using Bi-LSTM Model

4.2. Testing Result of Temperature Forecasting

The performance evaluation results of the temperature forecasting model are depicted in Fig. 8. The Attention GRU had the lowest RMSE value in the testing phase, with an RMSE value of 0.0268. During the training and validation processes, the Bi-GRU model without an attention mechanism achieved the lowest RMSE value. However, in the testing phase, the GRU model with an attention mechanism demonstrated the smallest RMSE. These results provide evidence that attention mechanisms enable the model to selectively focus on important features or patterns in the data, thereby improving its ability to generalize effectively to unseen test data. The comparison between actual values and forecasted values in the Attention GRU model for temperature forecasting for a 1-day ahead timestamp is shown in Fig. 9.

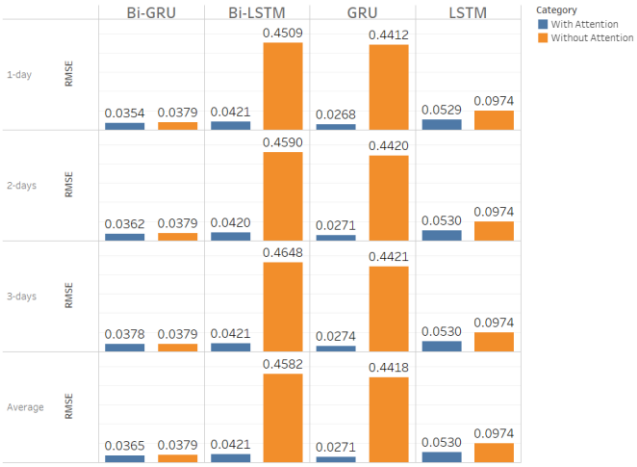

Fig. 8. Testing Results of Temperature Forecasting

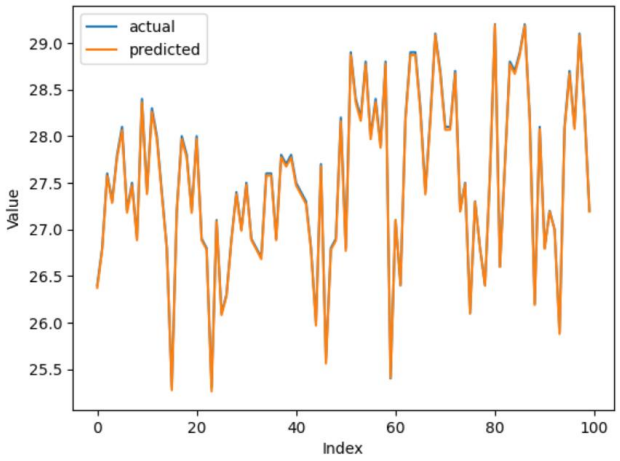

Fig. 9. Forecasted Temperature vs Actual Temperature by Attention GRU for 1-day ahead Timestamp

4.3. Testing Result of Relative Humidity Forecasting

The performance evaluation results of the relative humidity forecasting model are illustrated in Fig. 10. The model with the best performance is the Attention Bi-LSTM model, with an RMSE value of 0.1834. The Attention-based Bi-LSTM model exhibited the lowest RMSE value during both the validation and testing data processes, which can be attributed to the effectiveness of the attention mechanism in capturing important pat- terns and features within the data. The comparison between actual values and forecasted values in the Attention Bi-LSTM model for relative humidity prediction is shown in Fig. 11.

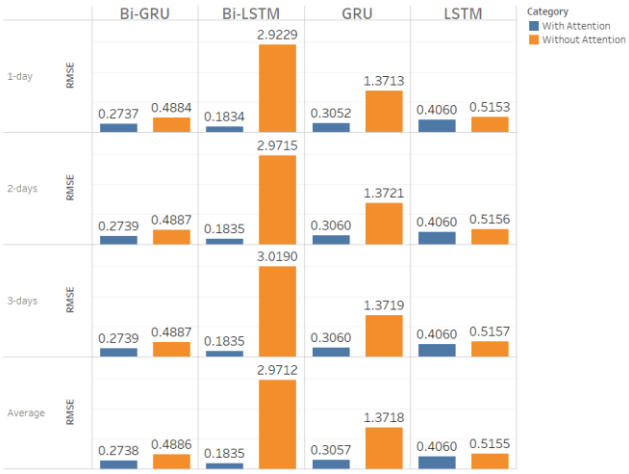

Fig. 10. Testing Results of Relative Humidity Forecasting

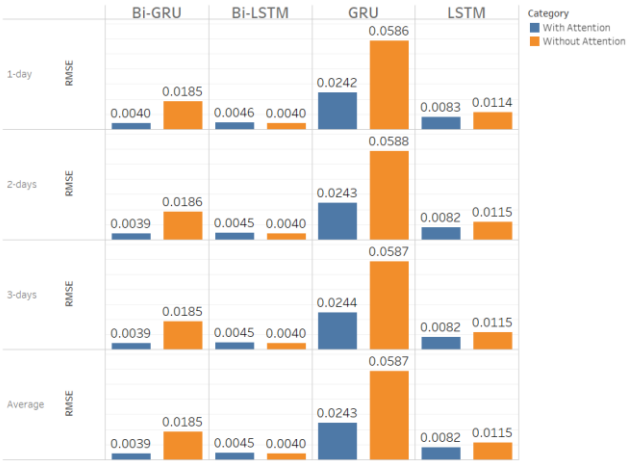

Fig. 12. Testing Results of Windspeed Forecasting

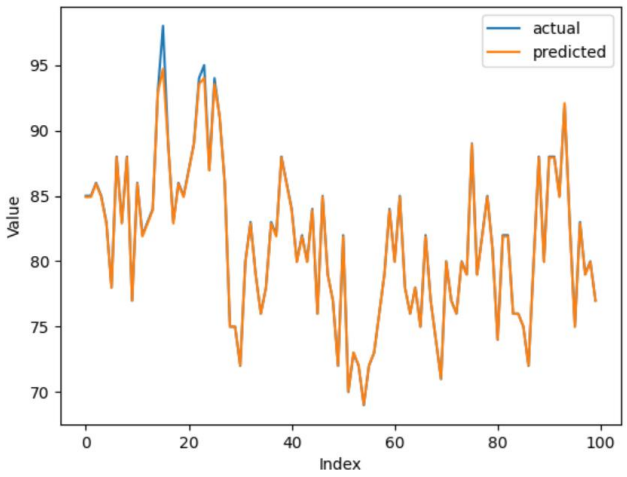

Fig. 11. Forecasted Relative Humidity vs Actual Relative Humidity by Attention Bi-LSTM for 1-day ahead Timestamp

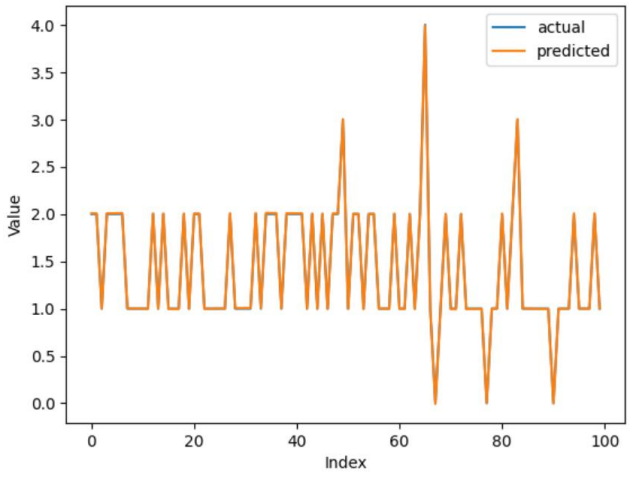

Fig. 13. Forecasted Windspeed vs Actual Windspeed by Attention Bi-GRU for 1-day ahead Timestamp

4.4. Testing Result of Windspeed Forecasting

The performance evaluation results of the windspeed prediction model are depicted in Fig. 12. The model with the best performance is the Attention Bi-GRU model, with an RMSE value of 0.0039. The performance of the attention-based Bi-GRU model was found to be better during the testing process compared to the validation process. The comparison between actual values and fore- casted values in the Attention Bi-GRU model for windspeed prediction is depicted in Fig. 13.

4.5. Summary of Testing Result

Based on the results of the model evaluation, it can be demonstrated that the incorporation of the attention mechanism leads to improved performance of the models for all output parameters investigated in this study. Accurate weather parameter forecasting is facilitated by the interplay of different interrelated meteorological input features, particularly when there is a shift in weather patterns throughout the sequence. In the models employed, multivariate weather variables were utilized to predict a singular target weather variable. This enables the determination of the reciprocal influence and attention weight of multiple weather variables in relation to the target variable. The incorporation of an attention mechanism facilitates the assignment of distinct weights to input variables, thereby identifying the specific aspects of the input data that require emphasis in the model.

A comparison was conducted among the top three models, namely Attention GRU, Attention Bi-LSTM, and Attention Bi-GRU. This comparison evaluated the average RMSE values specifically for the 1-day ahead timestamp, which has been observed to have the lowest RMSE among the other timestamps. The comparison of the average RMSE values for each output variable of the models is presented in Fig. 14. From the comparison results, it was found that

the Attention Bi-LSTM model has the lowest average RMSE value of 0.0767. This outcome demonstrated that the model with the best performance is the Attention Bi-LSTM model.

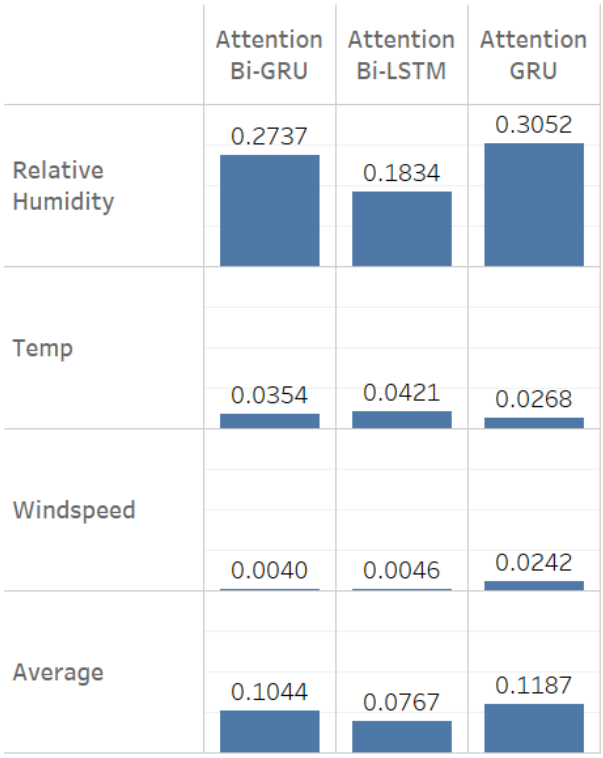

Fig. 14. Average RMSE Comparison of Attention GRU, Attention Bi-LSTM, and Attention Bi-GRU Models for Each Output Variable

5. Conclusion

This study demonstrated that the addition of the attention mechanism to the encoder-decoder architecture of the RNN-based model can effectively enhance the performance of the RNN-based encoder-decoder model in forecasting weather parameters such as temperature, relative humidity, and wind speed. The incorporation of the attention mechanism provides a solution for the encoder-decoder model by allowing it to assign weights to input sequences without being limited to a fixed input vector length. This enables the model to capture crucial information, particularly in cases of sudden weather pattern changes.

In this study, the findings reveal that the RNN-based encoder-decoder model with the attention mechanism consistently outperforms the RNN based encoder-decoder models without the attention mechanism. The best-performing model for temperature forecasting is the Attention GRU, while the Attention Bi-LSTM proves to be the most suitable model for predicting relative humidity, and the Attention Bi-GRU is the top performing model for wind speed prediction. These results indicate that the attention mechanism can be integrated into various types of RNN-based models and is not limited to a specific RNN-based architecture. As a future research direction, integrating weather image data could be considered to further enhance the model's performance. The utilization of topographic images from a specific region can provide valuable insights into the correlation between the geographical conditions of an area and the local weather changes. Additionally, using longer term data can serve as an alternative approach to improving the model's performance.

References

[1] Balasubramanian A. Weather Forecasting. Mysore: Department of Studies in Earth Science University of Mysore. 2017.

[2] Lim B, Zohren S. Time-series forecasting with deep learning: a survey. Philosophical Transactions of the Royal Society A. 2021;379(2194):20200209.

[3] Schultz MG, Betancourt C, Gong B, Kleinert F, Langguth M, Leufen LH, et al. Can deep learning beat numerical weather prediction? Philosophical Transactions of the Royal Society A. 2021;379(2194):20200097.

[4] Iseh A, Woma T. Weather forecasting models, methods and applications. Int J Eng Res Technol. 2013;2(12):1945–1957.

[5] Jaseena K, Kovoor BC. Deterministic weather forecasting models based on intelligent predictors: A survey. Journal of King Saud University-Computer and Information Sciences. 2022;34(6):3393–3412.

[6] Abdillah MR, Sarli PW, Firmansyah HR, Sakti AD, Fajary FR, Muharsyah R, et al. Extreme Wind Variability and Wind Map Development in Western Java, Indonesia. International Journal of Disaster Risk Science. 2022;13(3):465–480.

[7] Hou J, Wang Y, Zhou J, Tian Q. Prediction of hourly air temperature based on CNN– LSTM. Geomatics, Natural Hazards and Risk. 2022;13(1):1962–1986.

[8] Hanoon MS, Ahmed AN, Zaini N, Razzaq A, Kumar P, Sherif M, et al. Developing machine learning algorithms for meteorological temperature and humidity forecasting at Terengganu state in Malaysia. Scientific Reports. 2021;11(1):18935.

[9] Sahasrabuddhe DV, Jamsandekar P. Data structure for representation of big data of weather forecasting: a review. International Journal of Computer Science Trends and Technology (IJCST). 2015;3(6):48–56.

[10] Perry GL, Seidl R, Bellv´e AM, Rammer W. An outlook for deep learning in ecosystem science. Ecosystems. 2022;25(8):1700–1718.

[11] Hewage P, Trovati M, Pereira E, Behera A. Deep learning-based effective fine-grained weather forecasting model. Pattern Analysis and Applications. 2021;24(1):343–366.

[12] Suleman MAR, Shridevi S. Short-term weather forecasting using spatial feature attention based LSTM model. IEEE Access. 2022;10:82456–82468.

[13] Kreuzer D, Munz M, Schlu¨ter S. Short-term temperature forecasts using a convolutional neural network—An application to different weather stations in Germany. Machine Learning with Applications. 2020;2:100007.

[14] De Saa E, Ranathunga L. Comparison between arima and deep learning models for temperature forecasting. arXiv preprint arXiv:201104452. 2020;.

[15] LeCun Y, Bengio Y, Hinton G. Deep learning. nature. 2015;521(7553):436–444.

[16] Salman AG, Heryadi Y, Abdurahman E, Suparta W. Single layer & multi-layer long short-term memory (LSTM) model with inter- mediate variables for weather forecasting. Procedia Computer Science. 2018;135:89–98.

[17] Li Y, Zhu Z, Kong D, Han H, Zhao Y. EA- LSTM: Evolutionary attention-based LSTM for time series prediction. Knowledge-Based Systems. 2019;181:104785.

[18] Grossberg S. Recurrent neural networks. Scholarpedia. 2013;8(2):1888.

[19] Salehinejad H, Sankar S, Barfett J, Colak E, Valaee S. Recent advances in recurrent neural networks. arXiv preprint arXiv:180101078. 2017;.
[20] Siami-Namini S, Tavakoli N, Namin AS. The performance of LSTM and BiLSTM in fore- casting time series. In: 2019 IEEE Inter- national conference on big data (Big Data). IEEE; 2019. p. 3285–3292.
[21] Zaytar MA, El Amrani C. Sequence to sequence weather forecasting with long short- term memory recurrent neural networks. Inter- national Journal of Computer Applications. 2016;143(11):7–11.
[22] Lin Y, Koprinska I, Rana M. Temporal convolutional neural networks for solar power forecasting. In: 2020 International Joint Conference on Neural Networks (IJCNN). IEEE; 2020. p. 1–8.
[23] Xiong C, Merity S, Socher R. Dynamic memory networks for visual and textual question answering. In: International conference on machine learning. PMLR; 2016. p. 2397–2406.
[24] Liu X, Wang Y, Wang X, Xu H, Li C, Xin
X. Bi-directional gated recurrent unit neural network based nonlinear equalizer for coherent optical communication system. Optics Express. 2021;29(4):5923–5933.
[25] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:14090473. 2014;.
[26] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. Advances in neural information processing systems. 2017;30.
[27] Shahriari B, Swersky K, Wang Z, Adams RP, De Freitas N. Taking the human out of the loop: A review of Bayesian optimization. Proceedings of the IEEE. 2015;104(1):148–175.
[28] Jin XB, Zheng WZ, Kong JL, Wang XY, Bai YT, Su TL, et al. Deep-learning fore- casting method for electric power load via attention-based encoder-decoder with bayesian optimization. Energies. 2021;14(6):1596.
[29] Chai T, Draxler RR. Root mean square error (RMSE) or mean absolute error (MAE)?– Arguments against avoiding RMSE in the literature. Geoscientific model development. 2014;7(3):1247–1250.