Evaluating the efficacy of univariate LSTM approach for COVID-19 data prediction in Indonesia

Tegar Arifin Prasetyo[1], Joshua Pratama Silitonga[1], Matthew Alfredo[1], Risky Saputra Siahaan[1], Roberd Saragih[2], Dewi Handayani[2], Rudy Chandra[1]
[1]Department of Information Technology, Faculty of Vocational Studies, Institut Teknologi Del, Toba, Indonesia
[2]Department of Mathematics, Faculty of Mathematics and Natural Sciences, Institut Teknologi Bandung, Bandung, Indonesia

Article Info

Article history:

Received Dec 4, 2023
Revised Jan 21, 2024
Accepted Feb 4, 2024

Keywords:

COVID-19
Deep learning
Long short-term memory
Multivariate model
Prediction
Univariate model

ABSTRACT

The coronavirus disease 2019 (COVID-19) pandemic, originating in 2020, has emerged as a critical global issue due to its rapid and widespread transmission. Indonesia, among the affected nations, has taken measures to address the situation, including the development of a deep learning model for predicting future COVID-19 infection and spread. This predictive tool serves as a valuable reference for the government and stakeholders, aiding them in making informed decisions and implementing appropriate measures to contain the virus. The deep learning model employs the long short-term memory (LSTM) algorithm, chosen for its ability to recognize temporal patterns in the country's COVID-19 data. The model creation process involves data collection, preprocessing, model architecture planning, modeling, training, and evaluation. Two LSTM models were developed: a univariate and a multivariate model. Following thorough training and evaluation, the univariate model emerged as the superior choice, boasting evaluation metrics of 16.72 for mean absolute percentage error (MAPE) and 66.36 for root mean squared error (RMSE). This model was then deployed on a publicly accessible website, presenting visualizations of past COVID-19 data and predictions of future cases through line graphs. This user-friendly platform enables the public to access and analyze the data easily.

Corresponding Author:

Tegar Arifin Prasetyo
Department of Information Technology, Faculty of Vocational Studies, Institut Teknologi Del
Sisingamangaraja Street, Sitoluama, Laguboti 22381, Toba, Indonesia
Email: tegar.prasetyo@del.ac.id

1. INTRODUCTION

The coronavirus, more commonly known as coronavirus disease 2019 (COVID-19), is a single-stranded ribonucleic acid (RNA) virus that causes a disease in humans like pneumonia, accompanied by various other symptoms [1]. It's called COVID-19 because this virus was first discovered in late 2019, specifically in December 2019 in Wuhan, Hubei, China [1], [2]. Due to its ability to spread quickly from one person to another in proximity, it rapidly spread, leading the World Health Organization (WHO) to declare a global COVID-19 pandemic on March 11, 2020 [3]. This rapid spread resulted in a high number of confirmed COVID-19 infections worldwide. From the first cases of COVID-19 until September 2022, more than 603 million people have tested positive for COVID-19, with over 6 million deaths attributed to the infection. In Indonesia alone, by September 2022, more than 6 million people had tested positive for COVID-19, with over 150,000 deaths due to the infection [3]. The significant number of COVID-19 infection cases has made the COVID-19 pandemic a global focus of attention. As of now, no specific treatment has been found for COVID-19, but various types of vaccines have been developed for preventing the virus's infection in the body. Governments worldwide continue to innovate and seek the best ways to address the impact of the COVID-19 pandemic in all affected sectors. The significant impact caused by the COVID-19 pandemic has made it a highly important global issue that needs to be addressed. One form of addressing the impacts of the COVID-19 pandemic is by creating a system or model to predict future infection and spread data of COVID-19. Having a system capable of predicting COVID-19 data with high accuracy is crucial for governments to facilitate decision-making and policy formulation in their efforts to handle COVID-19 cases in the future [4]–[6].

Research related to the development of COVID-19 data prediction models has been conducted previously by [6], [7]. They developed a machine learning model using the Alpha-Sutte approach for eight countries, namely Italy, New Zealand, the United States, Brazil, Pakistan, Spain, South Africa, and India. The developed model was trained using 90% of the available data, while the remaining 10% of the data was used to evaluate the model’s performance. As mentioned earlier, this research employed the Alpha-Sutte approach, which is a forecasting method in statistics that utilizes the concept of moving averages to predict using the average of previous data within a specific time range [7]. The research successfully predicted COVID-19 cases in these eight countries with a relatively small prediction error rate, indicated by an average mean absolute percentage error (MAPE) below 1% when the model was used to predict the 10% of data not used for training [7], [8]. This demonstrates that the model is quite accurate in predicting COVID-19 data in several countries. However, the Alpha-Sutte approach can only be used for short-term data predictions, with an average data time range of 8 months for each country. Therefore, a model trained with only an 8-month time range is less effective for predicting data over longer time periods, given that Alpha-Sutte relies on the moving average concept, where the next data point can only be predicted by finding the average of previous data within a specific time range.

Another study applied convolutional neural network (CNN) and long short-term memory (LSTM) to forecast COVID-19 cases in Saudi Arabia, achieving exceptional predictive accuracy, achieving error rates below 5% [9]. Another related study on the development of COVID-19 data prediction models was conducted by [10], and it focused on Saudi Arabia. In this research, the multiple linear regression (MLR) approach was employed [10]. The data used in developing this model were divided into a training dataset (80%) and a testing dataset (20%). MLR analysis was used to predict the value of one variable based on the values of other variables. The variable to be predicted was the dependent variable, which is the number of COVID-19 cases expected in Saudi Arabia. The independent variables used as references to predict the dependent variable included the region's name, previous COVID-19 infection rate, mortality rate, and recovery rate [10], [11]. This research successfully predicted COVID-19 statistical data in cities in Saudi Arabia with a high level of accuracy, as indicated by an R-squared value (a statistical measure representing the proportion of variance in the dependent variable explained by the independent variables or variables in the regression model) of 96% [11]. However, this model has limitations. In time-series COVID-19 data, there is the possibility of changes in independent variables over time influenced by climate, weather, lifestyle, and local policies regulations that affect the results of the dependent variable. Therefore, the independent variables are not guaranteed to remain constant without change [11]. Furthermore, research on developing COVID-19 data prediction models using the MLR approach has not been conducted in Indonesia, so there is no solid evidence to determine whether this approach is suitable for predicting COVID-19 data in Indonesia.

In addition, there is also research on the development of another COVID-19 data prediction model conducted by [12] in Brazil, using the LSTM approach. In this study, they used 70% of the data he collected to train the model, and the remaining 30% of the data was used to evaluate the performance of the trained model. A total of 4,200 models were developed using the LSTM algorithm, with a relatively small amount of data, specifically only 30 to 40 days of new COVID-19 case data in Brazil. Out of these 4,200 models, one best-performing model was found with a reasonably good accuracy level, indicated by an R-squared (R2) value of 0.665. A higher R-squared value indicates better model performance. The final results of his research showed that various predictions are needed for COVID-19 cases in Brazil related to the dataset used to gain a clearer understanding of the situation and support future research. However, the various models developed by Hawas have not yet been able to predict over longer time periods due to the limited amount of data, as previously mentioned. Taking into account what has happened in Indonesia up to 2023, there is a possibility of daily COVID-19 case surges in Brazil, similar to the surges that have occurred in Indonesia. The very limited data available may not be representative of what will happen in the future when Hawas conducted his research, as the models have not been able to fully understand or learn the patterns of daily COVID-19 cases in Brazil. Therefore, machine learning models have not yet been able to identify growth patterns accurately, resulting in predictions that may not provide a very clear picture of COVID-19 cases that will occur in Brazil over longer time periods.

Previous studies mentioned earlier have fallen short in fully addressing the challenge of predicting COVID-19 data in Indonesia. This limitation arises due to disparities in COVID-19 cases between Indonesia and other countries, influenced by distinct lifestyles and demographic factors. Furthermore, there is a notable absence of similar studies using the LSTM algorithm on Indonesian COVID-19 data, a gap that warrants exploration. In light of these gaps, this research endeavors to contribute to the field by implementing a COVID-19 data prediction model based specifically on Indonesian data spanning from March 2020 to June 2023. The chosen LSTM algorithm is expected to reveal patterns within the dataset, enabling accurate predictions for time-dependent data, commonly referred to as time series data. The model's accuracy will be measured by the MAPE value, targeted to be below 20 [13], [14]. In this study, two LSTM models were created with different types: a univariate model and a multivariate model. Both models underwent the same training and evaluation stages to determine and obtain the best model among them after comparing the existing models. The selected model will be used to predict COVID-19 data in Indonesia. Based on the findings of this research, the best-performing model was found to be the univariate type, with MAPE evaluation metric result of 16.72% and root mean squared error (RMSE) metric result of 66.36. This optimal model is then deployed onto a website to visualize both historical COVID-19 data and predictions regarding potential new COVID-19 cases. The website aims to provide a clear representation of the data and insights from the model's predictions for the public.

2. METHOD

The COVID-19 pandemic continues to be a significant concern in Indonesia and globally at the time of conducting this research. Developing a predictive model for COVID-19 cases is one way to minimize and address the potential impact of the virus's spread [15]–[17]. However, to date, there is no proven and highly accurate prediction model that can accurately forecast COVID-19 data across all countries worldwide. Therefore, this research aims to implement a deep learning model for predicting COVID-19 cases in Indonesia. The chosen model will utilize the LSTM algorithm, as COVID-19 infection data is of a time-series nature. The developed model will be thoroughly evaluated for its accuracy and subsequently deployed in a web-based application accessible to the general public. The methodology design is elaborated in the following description, illustrated in Figure 1.

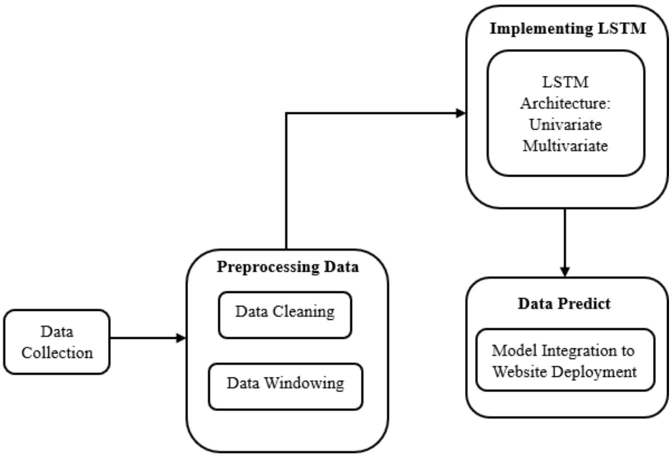


Figure 1. Methodology design

2.1. Data collection

The data to be used for implementing the model is a dataset containing daily COVID-19 cases in many countries, including Indonesia. The dataset is sourced from our world in data platform. It provides information on the daily number of COVID-19 cases, along with other relevant data that can be utilized to develop a predictive model for COVID-19 cases. The data spans from March 2020 to the latest available data. For Indonesia, the dataset consists of 1,040 rows, representing daily COVID-19 cases in Indonesia for over 2 years. The data will be divided into training, validation, and testing sets. A total of 940 rows will be used for training, 79 rows for validation, and 7 rows for testing. The dataset comprises 67 columns or features, indicating that it includes not only daily COVID-19 case numbers but also other relevant information. These additional features can be leveraged to improve the performance of the model if they are found to be relevant or correlated with the daily case numbers. The collected data should ideally go through the data preprocessing stage, considering that real-life data can sometimes be incomplete, inconsistent, redundant, and contain noise that can diminish the overall data quality [18]–[20]. Therefore, data preprocessing becomes crucial to ensure that the data can be effectively used in the implemented model, enabling the resolution of existing issues. Several steps need to be taken on the collected data for this research, including data cleaning and feature selection.

2.2. Data cleaning

Data cleaning is the process of detecting and handling damaged, inaccurate, inconsistent, and irrelevant data within a dataset. As the name suggests, the data is cleaned using various techniques to modify or remove such data that tends to degrade the dataset's quality [18]. Firstly, checking for duplicate or repetitive rows is performed, and those rows are removed from the dataset as they only reduce its quality. In the case of the data collected from our world in data, a check will be conducted on all rows of data for the location of Indonesia since only COVID-19 data specific to Indonesia will be used. The dataset has already been checked for duplicate data, and no duplicate rows have been found, eliminating the need for data deletion. Next, for each row in a column that contains missing values and where the number of missing values is less than 70% of the total number of rows, the missing values will be filled with the mean, median, or mode attribute of the data in that column [21]. However, if more than 70% of the rows in a column have missing values, that column will not be used and will be removed.

As mentioned in 2.1, the dataset to be used has 67 columns. All these columns need to be checked for each row of data within them to determine if the data is complete or if there are any missing values. Out of the 67 columns, it was found that 14 columns had more than 70% of their rows with missing values. Out of these 14 columns, 12 columns had no data at all, meaning all rows in those columns were empty. Therefore, all 14 columns will be deleted from the dataset, leaving 53 columns. There are 29 columns with less than 70% missing values, requiring the filling of missing values for the rows in those columns. The rows in each column will be filled using the mean value or the most frequent number in that column, except for columns related to death data. This is because the missing data in those columns indicates that no deaths occurred at the early stage of COVID-19 entering Indonesia, as indicated by the date column in the same row as the missing death data. Subsequently, all the empty rows in columns related to death numbers will be filled with the value 0, indicating no deaths occurred. After filling in the missing data, a final check is performed to ensure that no columns contain missing values. If this is the case, the next step of data preprocessing, which is feature selection, will be pursued. In this study, the method used for feature selection is bivariate analysis, which involves analyzing the correlation between each feature and the target variable. The target variable in this case is the new COVID-19 cases.

2.3. Data windowing

Data windowing is the process of dividing a time-series dataset into "windows" or larger sequences that can be used as input for deep learning models such as recurrent neural networks (RNNs) and LSTMs [22]. The goal is to enable the model to learn from the temporal relationships between time points in the data. By providing the model with a sequence of time-series data, the model can learn how the values of the data change over time and use this information to make predictions for future data points.

The size of the window or sequence is chosen depending on the requirements, the model's accuracy when it is built and evaluated, and other factors. The size of the window also affects the performance of the model and computational efficiency, such as the resources required and runtime. In this study, the dataset will be divided into smaller windows containing 14 data points. Since this dataset represents daily COVID-19 cases in Indonesia, the window size refers to a 14-day period of COVID-19 case data. Through each window containing 14 days of COVID-19 case data, the model will learn patterns in the data and make predictions for the next day's COVID-19 cases.

2.4. LSTM

LSTM is a variation of RNN that addresses the issues faced by RNN. RNN is a component of deep learning commonly used for processing sequential data, including time-series data, where the chronological order of the data is crucial [23]. RNN shares similarities with deep learning and machine learning as it is inspired by how the human brain learns. It considers the received information, compares it with past data, and makes decisions accordingly. RNN can remember past time-series data to make predictions about future data [24]. However, RNN faces challenges such as vanishing gradient and exploding gradient, especially in learning long-term sequential or time-series data, with vanishing gradient being more common. The vanishing gradient is a problem in deep learning where the training of models using gradient-based methods, such as gradient descent, aims to optimize the loss function by finding its global minimum.

In deep learning models, including RNNs, information flows from the input layer to the output layer sequentially based on the time order of the data. The output of one-time step becomes the input for the next time step, and backpropagation is performed after processing all the data. This process leads to the occurrence of the vanishing gradient. As the length of the time sequence increases, the gradients obtained during backpropagation using the optimizer algorithm tend to decrease and approach zero [25]. These gradients are used to update the weights of the model. When the gradients approach zero, the weight updates become minimal, and the model fails to effectively learn new information since the weights hardly change. Conversely, the exploding gradient occurs when the gradients obtained during backpropagation become extremely large, causing explosive and unstable weight updates. Both issues prevent the model from reaching an optimal performance characterized by the global minimum of the loss function. Therefore, RNNs are not suitable for handling long-term time-series data. These challenges ultimately result in RNNs having suboptimal performance in predicting future data [24].

To address these problems, the LSTM algorithm was introduced by Sepp Hochreiter and Jürgen Schmidhuber in 1997. LSTM is capable of handling long-term time-series data, overcoming the issues of vanishing and exploding gradients encountered in RNNs [26]. Figure 2 illustrates the architecture of LSTM. In this LSTM stage, we design the model architecture that will be created. The model will consist of multiple layers that enable it to learn from the preprocessed data. These layers include LSTM and fully connected layers that follow the LSTM layer. Two architectures will be developed, which have minor differences in the input data format used during training. The first architecture is the LSTM Univariate, which utilizes only one feature, namely new cases. The second architecture is the LSTM Multivariate, which incorporates more than one feature, specifically new cases and positive rate.

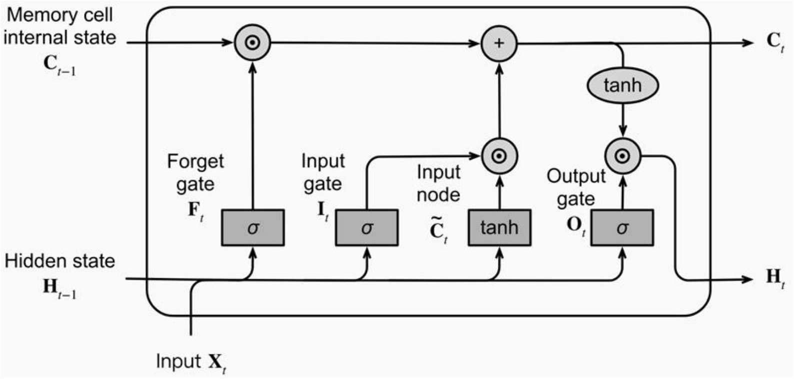

Memory cell internal state $\mathbf{C}_{t-1}$
$\odot$
+
$\mathbf{C}_t$
tanh
$\odot$
$\odot$
Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Input node $\tilde{\mathbf{C}}_t$
Output gate $\mathbf{O}_t$
σ
σ
tanh
σ
Hidden state $\mathbf{H}_{t-1}$
$\mathbf{H}_t$
Input $\mathbf{X}_t$

Figure 2. LSTM architecture [27]

2.5. Model integration to website deployment

The implemented model utilizes the TensorFlow library in the Python programming language. TensorFlow allows models developed using its library to be exported and used in other programming languages. A model can be exported in a format called TensorFlow Lite, which can be used in programming languages such as Java, Kotlin, C++, and Swift. Additionally, a model can be exported or converted into a format compatible with JavaScript and TypeScript. The conversion process results in a model saved in a JavaScript Object Notation (JSON) file format, and each weight of the model is stored as a binary file (.bin), which may consist of multiple files. The converted files can be used in JavaScript by utilizing the TensorFlow.js library. In JavaScript, the model needs to be loaded by calling the JSON file. Once the model is successfully loaded, it is ready to make predictions for COVID-19 data in Indonesia. Furthermore, the model in JSON and binary file format can also be retrained using the TensorFlow.js library with updated datasets. This enables the predictions made by the model to remain relevant to the latest occurrences of COVID-19 cases. By leveraging TensorFlow.js, the model can be used for prediction in JavaScript and can be updated with new datasets, ensuring that the predictions align with the most recent developments in COVID-19 cases.

In Figure 3, the workflow of the system in the web interface can be observed. First, when the user accesses the web interface and selects a date range for viewing the predicted data, the web interface will send a request to TensorFlow to perform the prediction. Then, the response, which includes the weights from the model.json, will be loaded. Subsequently, TensorFlow will provide the prediction results, which will be displayed as a graph on the web interface. Simultaneously, the prediction results will be stored in the database. Therefore, if there is a subsequent prediction request from the user with the same date range input, the web interface can retrieve the historical prediction data directly from the database for display purposes.

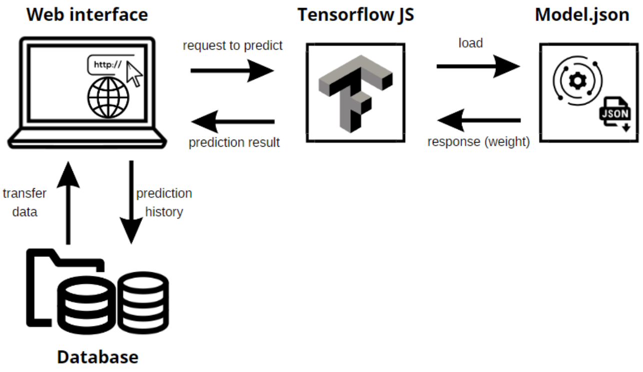


Figure 3. Overview of the deploy in website system

3. RESULTS AND DISCUSSION

In this chapter, we delve into a meticulous examination of the outcomes obtained at the conclusion of our research implementation. Through a comprehensive exploration, we aim to dissect and analyze the findings in detail, offering insights into their significance and broader implications. Our objective is to provide a nuanced understanding of the results, shedding light on the intricate details that emerged during the research process.

3.1. Result of data preprocessing

Throughout the meticulous data cleaning process, we will thoroughly evaluate the pertinence of the current dataset to the intended model. Data that is unused and lacks relevance will be diligently removed and will not be incorporated into the subsequent stages. Table 1 presents the column names of the dataset that have less than 70% empty data rows.

Table 1. The following are columns with missing values with total less than 70% of the data

Column name	Empty number	Column name	Empty number
Total_cases	59	Positive_rate	795
New_cases	1	Tests_per_case	795
New_cases_smoothed	6	Total_vaccinations	704
Total_cases_per_million	59	People_vaccinated	660
New_cases_per_million	1	People_fully_vaccinated	658
New_cases_smoothed_per_million	6	New_vaccinations	751
Total_deaths_per_million	68	New_vaccinations_smoothed	425
New_deaths_smoothed	5	Total_vaccinations_per_hundred	704
Reproduction_rate	137	People_vaccinated_per_hundred	660
Total_tests	789	People_fully_vaccinated_per_hundred	658
New_tests	791	New_vaccinations_smoothed_per_million	425
Total_tests_per_thousand	789	New_people_vaccinated_smoothed	425
New_tests_per_thousand	791	New_people_vaccinated_smoothed_per_hundred	425
New_tests_smoothed	795	Stringency_index	85
New_tests_smoothed_per_thousand	795	Total_deaths	68

Based Table 1, we can show that all columns are contain relevant data for further selection in the feature selection phase. Columns with empty data exceeding 70% of the total data in that column are not used because they are less relevant and may potentially disrupt the results of the feature selection stage. In the feature selection process, the dataset that has been processed in the data cleaning stage will undergo column selection to determine the features to be used in the modeling phase for a multivariate model. This selection is done by finding the correlation between all features and the target variable (new cases). Features with a positive correlation above 0.5 will be chosen for use in the modeling phase. Based on the correlation between these features, in Table 2 we can see the correlation between the features and the target. Table 2 show that the selected features are those with a correlation value greater than or equal to 0.5, namely new death and positive rate. However, due to the high correlation with the new death data, which is not caused by the significant influence of new death on new cases, but rather the other way around, where new cases influences new death, new death is not selected as a feature to be used. Therefore, the final selected feature is positive rate.

Table 2. Correlation between features and target

Feature	Correlation	Feature	Correlation
Total_cases	0.008	Positive_rate	0.678
New_cases	1.000	Tests_per_case	-0.359
New_cases_smoothed	1.000	Total_vaccinations	-0.094
Total_cases_per_million	0.008	People_vaccinated	-0.082
New_cases_per_million	1.000	People_fully_vaccinated	-0.118
New_cases_smoothed_per_million	0.969	New_vaccinations	0.032
Total_deaths_per_million	0.064	New_vaccinations_smoothed	0.138
New_deaths_smoothed	0.638	Total_vaccinations_per_hundred	-0.094
Reproduction_rate	0.118	People_vaccinated_per_hundred	-0.082
Total_tests	0.119	People_fully_vaccinated_per_hundred	-0.118
New_tests	0.309	New_vaccinations_smoothed_per_million	0.138
Total_tests_per_thousand	0.119	New_people_vaccinated_smoothed	0.080
New_tests_per_thousand	0.309	New_people_vaccinated_smoothed_per_hundred	0.080
New_tests_smoothed	0.298	Stringency_index	0.260
New_tests_smoothed_per_thousand	0.298	Total_deaths	0.063
New_deaths	0.677	New_deaths_per_million	0.677
New_deaths_smoothed_per_million	0.638		

3.2. Univariate model training result

Table 3 provides a concise overview of the architecture for the univariate model, outlining the key details essential for the subsequent model training process. The presented information encompasses the structural framework that the model will adopt during training. This architectural summary serves as a foundational reference for understanding the upcoming phases of the model implementation.

Table 3. Architectural summary of the univariate model

Layer (type)	Output shape	Param #
Lstm_33 (LSTM)	(None, 14, 300)	362,400
Lstm_34 (LSTM)	(None, 14, 250)	551,000
Lstm_35 (LSTM)	(None, 14, 200)	360,800
Lstm_36 (LSTM)	(None, 150)	210,600
Dense_18 (Dense)	(None, 150)	22,650
Dense_19 (Dense)	(None, 1)	151
Total params: 1,507,601		
Trainable params: 1,507,601		
Non-trainable params: 0		

In Table 3, we can see that this univariate model has 1,507,601 parameters to be trained. The univariate model will be trained using three different learning rates (0.001, 0.0005, 0.0001). The first experiment involves training the model with a learning rate of 0.001 for the optimizer. Figure 4 shows the visualization of the loss value’s progression throughout the iterations during the first training.

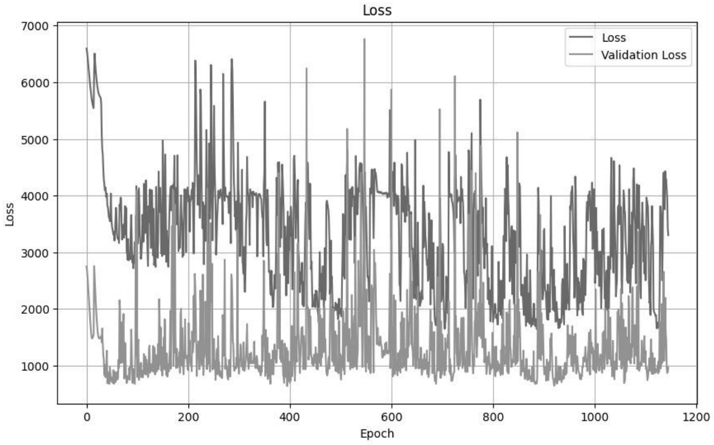


Figure 4. Loss visualization and validation loss model univariate with learning rate 0.001

Based on the results of the first experiment, as seen in Figure 4, both the loss and validation loss values fluctuate and tend to stabilize, indicating that there is no decreasing trend in the loss values. One of the reasons for this is that the learning rate is too high, causing significant changes in the weights of each layer in the model. Consequently, the optimizer algorithm fails to reach the optimal point for the actual loss value. Therefore, in the second experiment, training is conducted with a lower learning rate of 0.0005. Figure 5 illustrates the visualization of the loss value's progression throughout the iterations during the second training.

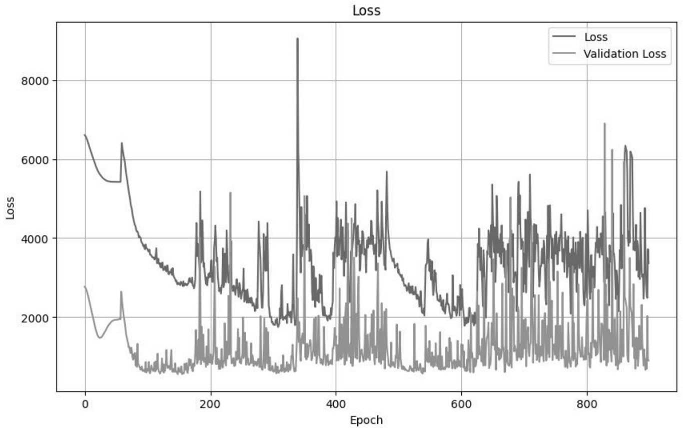


Figure 5. Loss visualization and validation loss model univariate with learning rate 0.0005

Based on Figure 5, both loss values show a decreasing trend in the first 100 iterations. However, after that, the loss values start to fluctuate, and eventually, the EarlyStopping callback stops the training process due to the lack of improvement in the validation loss. Therefore, in the third experiment, training is conducted again with an even lower learning rate of 0.0001. Figure 6 illustrates the visualization of the loss value's progression throughout the iterations during the third training.

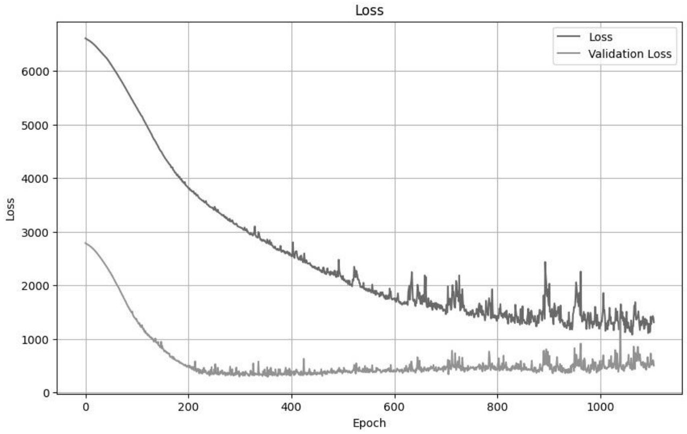


Figure 6. Loss visualization and validation loss model univariate with learning rate 0.0001

Based on Figure 6, the loss value shows a relatively long decreasing trend, extending up to the first 800 iterations. On the other hand, the validation loss value exhibits an increasing trend starting from around the 300th iteration and onwards. Both values eventually cease to decrease after the 800th iteration, prompting the EarlyStopping callback to halt the training process. As mentioned, during the iteration range of 200 to 300, the validation loss value reaches its lowest point. Therefore, EarlyStopping has saved the weights of all layers in the model when the validation loss value reached its lowest point, and these weights are then set back to the model. This model will be further evaluated in the subsequent stage.

3.3. Multivariate model training result

Table 4 presents an overview of the architecture of the multivariate model to be implemented during the model training process. This table details important aspects of the model structure that will be integral to the subsequent training phase. This architectural summary serves as a fundamental reference, providing crucial insights into the configuration and design of the multivariate model for efficient training.

Table 4. Architectural summary of the multivariate model

Layer (type)	Output shape	Param #
Lstm_8 (LSTM)	(None, 14, 300)	363600
Lstm_9 (LSTM)	(None, 14, 250)	551000
Lstm_10 (LSTM)	(None, 14, 200)	360800
Lstm_11 (LSTM)	(None, 150)	210600
Dense_4 (Dense)	(None, 150)	22650
Dense_5 (Dense)	(None, 1)	151
Total params: 1508801		
Trainable params: 1508801		
Non-trainable params: 0		

Based on Table 4, we can observe that the multivariate model has 1,508,801 parameters to be trained. In the multivariate model, the same training process is conducted, but with data consisting of two features as previously mentioned: new cases and positive rate. Considering the good results obtained by the univariate model, the same learning rate of 0.0001 is used for the multivariate model. Figure 7 visualizes the progress of the loss in the multivariate model with a learning rate of 0.0001. Based on Figure 7, both loss values show a very brief decreasing trend, occurring only in the first 100 iterations. Afterwards, the loss values fluctuate and eventually the training process is stopped by the callback.

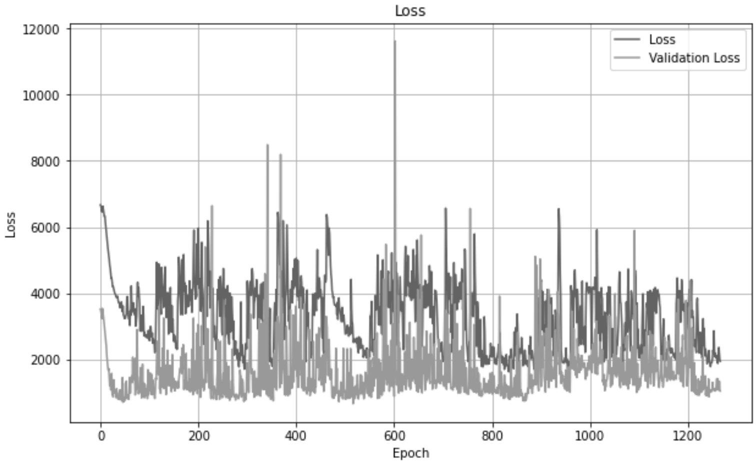
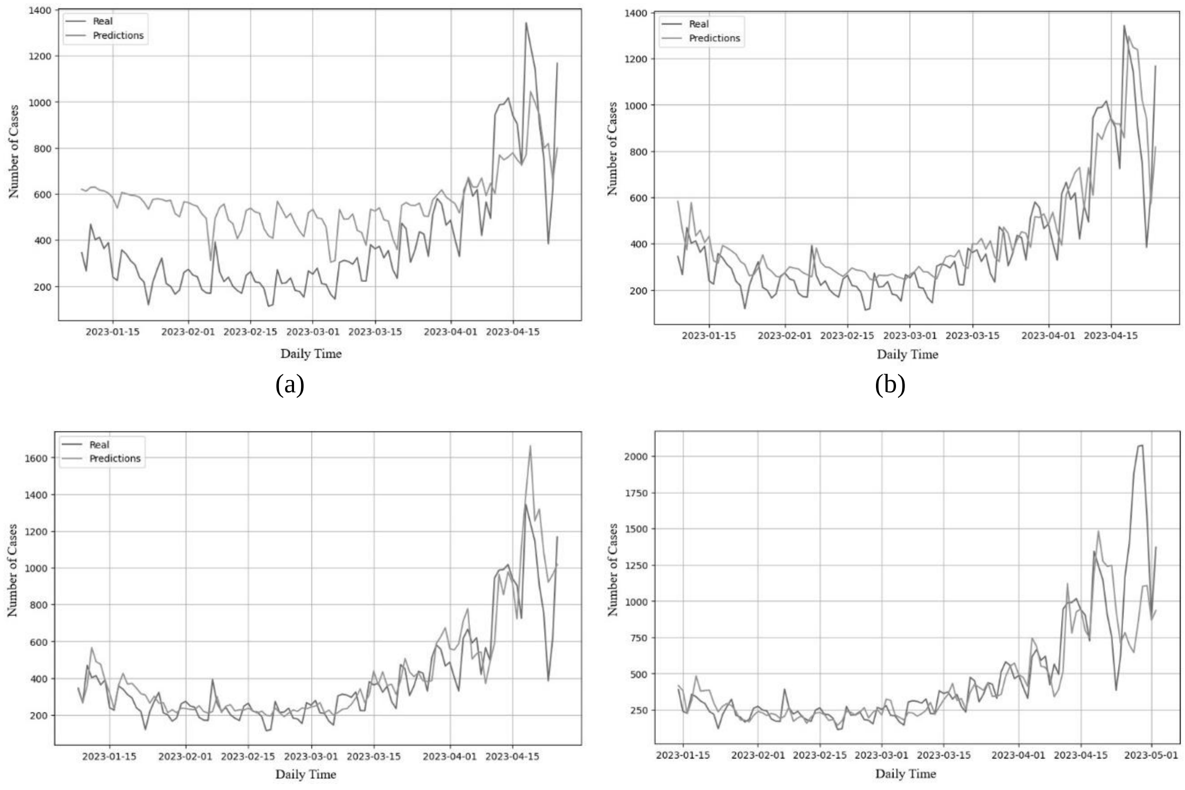


Figure 7. Loss visualization and validation loss model multivariate with learning rate 0.0001

3.4. Univariate model evaluation result

In the evaluation stage, the model that has been trained will be evaluated to assess its performance and determine how accurate and effective it is in making predictions. This evaluation process involves using validation or testing data that is separate from the data used for training the model. In this deep learning model evaluation process, two evaluation metrics are used: MAPE and RMSE.

Figure 8 displays predictions from both the univariate and multivariate models, highlighting variations in learning rate values. In Figure 8(a), we can see the comparison between the predictions of the univariate model with a learning rate of 0.001 and the actual testing data. The blue line represents the actual data, while the orange line represents the model’s predictions. Based on these predictions, we found a MAPE value of 38.03% and an RMSE value of 127.36. Figure 8(b) shows the predictions of the univariate model with a learning rate of 0.0005. Based on these predictions, the MAPE value obtained is 22.27% and the RMSE value is 126.91. Figure 8(c), based on the available prediction results, the MAPE and RMSE values can be determined, as mentioned earlier. Based on these prediction results, the MAPE value of the model for the testing data is 16.72% and the RMSE value is 66.36. Figure 8(d) is the result of the prediction made by the multivariate model using a learning rate of 0.0001. Based on the prediction results obtained from the multivariate model, the MAPE value is 23.86% and the RMSE value is 253.33.



(a)

(b)

(c)

(d)

Figure 8. Univariate and multivariate model prediction with learning rate; (a) model univariate with learning rate 0.001, (b) model univariate with learning rate 0.0005, (c) model univariate with learning rate 0.0001, and (d) model multivariate with learning rate 0.0001

3.5. Discussion

Table 5 encapsulates the conclusions drawn from the evaluation results of each model discussed earlier. The table serves as a consolidated summary, presenting the overall assessment and findings derived from the analysis of the models. This provides a succinct reference for comprehending the key takeaways from the evaluation process.

Table 5. Summary of the evaluation of all models

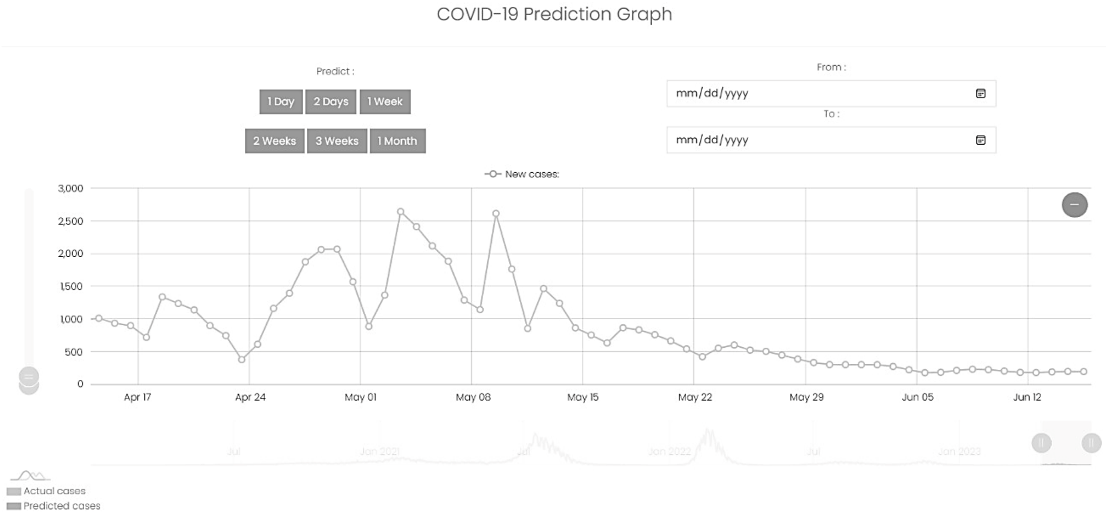
Model type	Learning rate	MAPE value	RMSE value
Univariate LSTM	0.001	38.03%	127.36
Univariate LSTM	0.0005	22.27%	126.91
Univariate LSTM	0.0001	16.72%	66.36
Multivariate LSTM	0.0001	23.86%	253.33

Based on the results obtained from the entire modeling process, both univariate and multivariate, there are several differences. Table 6 shows the comparison between the univariate and multivariate models. Table 6 show that it can be seen that the evaluation metrics obtained from the univariate model are better than those of the multivariate model. Therefore, in the deployment phase, the model that will be deployed is the univariate model.

Table 6. Comparison results between the univariate and multivariate model

No	Univariate	Multivariate
1	It has 1507601 parameters.	It has 1508801 parameters.
2	The only feature used is new_cases.	The features used are new_cases and positive_rate.
3	At a learning rate of 0.0001, the evaluation results obtained are an MAPE value of 16.72% and an RMSE value of 66.36.	At a learning rate of 0.0001, the evaluation results obtained are an MAPE value of 23.86% and an RMSE value of 253.33.

Based on the results of the deployment phase, both in the implementation of the frontend and backend web, a web-based application has been created that is capable of providing visualizations of daily COVID-19 infection cases, including both actual data and predicted data from the model. The data is visualized in the form of a line graph, where the blue line represents the actual data and the orange line represents the predicted data. Additionally, the created graph also has a scrollbar that allows users to scroll through the available data. There are also buttons to display the predicted data on the graph for specific time ranges, such as 1 day ahead, 2 days ahead, 1 week ahead, 2 weeks ahead, 3 weeks ahead, and 1 month ahead. Furthermore, there is a datepicker input field that allows users to display data on the graph based on their desired date range. Figure 9 shows the result of the deployment in the form of a website to predict COVID-19 cases in Indonesia.


COVID-19 Prediction Graph

Predict :

From : mm/dd/yyyy

To : mm/dd/yyyy

Figure 9. The deployment website results for predicting COVID-19 case data in Indonesia

The conducted research yields insightful findings that pave the way for critical discussions, comparisons, and interpretations. Delving into the implications of these findings, a set of recommendations emerges for shaping future work in this domain. Primarily, the model developed in this research showcases its proficiency in predicting new COVID-19 infection cases in Indonesia. To bolster its utility, it is advisable to broaden its scope, incorporating the prediction of new recovery cases and fatalities linked to the COVID-19 virus. This expansion would not only enhance the model's comprehensiveness but also contribute to a more holistic understanding of the pandemic's dynamics. Furthermore, while the current model excels in forecasting new cases within a 30-day timeframe, it is prudent for future studies to advance beyond this temporal constraint. Developing a model capable of predicting new cases over an extended duration is crucial for anticipating the evolving patterns of the virus and informing more effective public health measures. Lastly, the best-performing model in this research achieved a commendable MAPE evaluation metric of 16.72%. However, aspiring for greater precision remains imperative. Future studies should aim for models surpassing this benchmark, striving for a MAPE evaluation metric below 16.72%. Such enhancements would not only signify improved accuracy but also contribute to the reliability of predictive models in the context of infectious disease dynamics. In contemplating the ramifications of these recommendations, it becomes evident that refining predictive models for infectious diseases holds significant potential for proactive public health interventions. As we navigate the uncertainties of the future, the implementation of these suggestions can prove invaluable in mitigating the impact of emerging health crises and facilitating a more robust response to infectious diseases.

4. CONCLUSION

Based on the experiments conducted to obtain a highly accurate deep learning model, this research draws the following conclusions. Firstly, the application of the LSTM algorithm enabled the successful prediction of new COVID-19 infection cases in Indonesia. Secondly, among the models tested, the univariate model type was found to be the most effective when implementing the LSTM algorithm. Lastly, the choice of learning rate significantly impacts the model's quality. This research identified that a learning rate of 0.0001 yielded the highest accuracy results, as indicated by the lowest MAPE evaluation metric value of 16.72% achieved during model training.

ACKNOWLEDGEMENTS

This work was supported in part by Del Institute of Technology and Bandung Institute of Technology, Indonesia.

REFERENCES

[1] Y. C. Wu, C. S. Chen, and Y. J. Chan, "The outbreak of COVID-19: an overview," *Journal of the Chinese Medical Association*, vol. 83, no. 3, pp. 217–220, 2020, doi: 10.1097/JCMA.0000000000000270.

[2] T. Singhal, "A review of coronavirus disease-2019 (COVID-19)," *Indian Journal of Pediatrics*, vol. 87, no. 4, p. 281, Apr. 2020, doi: 10.1007/S12098-020-03263-6.

[3] A. N. Poudel *et al.*, "Impact of COVID-19 on health-related quality of life of patients: a structured review," *PLoS One*, vol. 16, no. 10, p. e0259164, Oct. 2021, doi: 10.1371/journal.pone.0259164.

[4] W. Akram and R. Kumar, "A study on positive and negative effects of social media on society," *International Journal of Computer Sciences and Engineering*, vol. 5, no. 10, pp. 351–354, Oct. 2017, doi: 10.26438/ijcse/v5i10.351354.

[5] E. Campillo-Funollet *et al.*, "Predicting and forecasting the impact of local outbreaks of COVID-19: use of SEIR-D quantitative epidemiological modelling for healthcare demand and capacity," *International Journal of Epidemiology*, vol. 50, no. 4, pp. 1103–1113, Aug. 2021, doi: 10.1093/ije/dyab106.

[6] L. Xu, R. Magar, and A. B. Farimani, "Forecasting COVID-19 new cases using deep learning methods," *Computers in Biology and Medicine*, vol. 144, p. 105342, May 2022, doi: 10.1016/j.compbiomed.2022.105342.

[7] A. M. C. H. Attanayake and S. S. N. Perera, "Forecasting COVID-19 cases using alpha-sutte indicator: a comparison with autoregressive integrated moving average (ARIMA) method," *BioMed Research International*, vol. 2020, 2020, doi: 10.1155/2020/8850199.

[8] A. S. Ahmar, A. Rahman, and U. Mulbar, "α-Sutte indicator: a new method for time series forecasting," *Journal of Physics: Conference Series*, vol. 1040, no. 1, p. 012018, Jun. 2018, doi: 10.1088/1742-6596/1040/1/012018.

[9] A. Al-Rashedi and M. A. Al-Hagery, "Deep learning algorithms for forecasting COVID-19 cases in Saudi Arabia," Applied Sciences 2023, vol. 13, no. 3, p. 1816, Jan. 2023, doi: 10.3390/app13031816.

[10] A. Jadi, "COVID-19 prediction model using machine learning," *IJCSNS International Journal of Computer Science and Network Security*, vol. 21, no. 8, 2021, doi: 10.22937/IJCSNS.2021.21.8.33.

[11] D. H. Maulud and A. M. Abdulazeez, "A review on linear regression comprehensive in machine learning," *Journal of Applied Science and Technology Trends*, vol. 1, no. 4, pp. 140–147, Dec. 2020, doi: 10.38094/jastt1457.

[12] M. Hawas, "Generated time-series prediction data of COVID-19′s daily infections in Brazil by using recurrent neural networks," *Data Brief*, vol. 32, p. 106175, Oct. 2020, doi: 10.1016/j.dib.2020.106175.

[13] J. J. M. Moreno, A. P. Pol, A. S. Abad, and B. C. Blasco, "Using the R-MAPE index as a resistant measure of forecast accuracy," *Psicothema*, vol. 25, no. 4, pp. 500–506, 2013, doi: 10.7334/psicothema2013.23.

[14] A. Yadav, C. K. Jha, and A. Sharan, "Optimizing LSTM for time series prediction in Indian stock market," *Procedia Computer Science*, vol. 167, pp. 2091–2100, Jan. 2020, doi: 10.1016/j.procs.2020.03.257.

[15] R. S. Hirschprung and C. Hajaj, "Prediction model for the spread of the COVID-19 outbreak in the global environment," *Heliyon*, vol. 7, no. 7, p. e07416, Jul. 2021, doi: 10.1016/j.heliyon.2021.e07416.

[16] A. L. Bertozzi, E. Franco, G. Mohler, M. B. Short, and D. Sledge, "The challenges of modeling and forecasting the spread of COVID-19," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 117, no. 29, pp. 16732–16738, Jul. 2020, doi: 10.1073/pnas.2006520117.

[17] M. M. Alam, A. M. Fawzi, M. M. Islam, and J. Said, "Impacts of COVID-19 pandemic on national security issues: Indonesia as a case study," *Security Journal*, vol. 35, no. 4, pp. 1067–1086, Dec. 2022, doi: 10.1057/s41284-021-00314-1.

[18] V. Agarwal, "Research on data preprocessing and categorization technique for smartphone review analysis," *International Journal of Computer Applications*, vol. 131, no. 4, p. 30, 2015.

[19] T. A. Prasetyo *et al.*, "Sales forecasting of marketing using adaptive response rate single exponential smoothing algorithm," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 31, no. 1, pp. 423–432, Jul. 2023, doi: 10.11591/ijeecs.v31.i1.pp423-432.

[20] T. A. Prasetyo, V. L. Desrony, H. F. Panjaitan, R. Sianipar, and Y. Pratama, "Corn plant disease classification based on leaf using residual networks-9 architecture," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, no. 3, pp. 2908–2920, Jun. 2023, doi: 10.11591/ijece.v13i3.pp2908-2920.

[21] S. I. Khan and A. S. M. L. Hoque, “SICE: an improved missing data imputation technique,” *Journal of Big Data*, vol. 7, no. 1, pp. 1–21, Dec. 2020, doi: 10.1186/s40537-020-00313-w.

[22] M. Guven and F. Uysal, “Time series forecasting performance of the novel deep learning algorithms on stack overflow website data,” *Applied Sciences 2023*, Vol. 13, Page 4781, vol. 13, no. 8, p. 4781, Apr. 2023, doi: 10.3390/app13084781.

[23] R. DiPietro and G. D. Hager, “Deep learning: RNNs and LSTM,” *Handbook of Medical Image Computing and Computer Assisted Intervention*, pp. 503–519, Jan. 2020, doi: 10.1016/B978-0-12-816176-0.00026-0.

[24] R. Chandra, A. Jain, and D. S. Chauhan, “Deep learning via LSTM models for COVID-19 infection forecasting in India,” *PLoS One*, vol. 17, no. 1, Jan. 2022, doi: 10.1371/JOURNAL.PONE.0262708.

[25] S. H. Noh, “Analysis of gradient vanishing of RNNs and performance comparison,” information 2021, Vol. 12, Page 442, vol. 12, no. 11, p. 442, Oct. 2021, doi: 10.3390/info12110442.

[26] S. Althubiti, W. Nick, J. Mason, X. Yuan, and A. Esterline, “applying long short-term memory recurrent neural network for intrusion detection,” *Conference Proceedings - IEEE SOUTHEASTCON*, vol. 2018-April, Oct. 2018, doi: 10.1109/SECON.2018.8478898.

[27] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, “Dive into deep learning,” *Journal of the American College of Radiology*, vol. 17, no. 5, pp. 637–638, Jun. 2021, doi: 10.1016/j.jacr.2020.02.005.

BIOGRAPHIES OF AUTHORS

Tegar Arifin Prasetyo is current lecturer and researcher member in Information Technology Department at Institut Teknologi Del since 2020. Have experience specializing in building mathematical models, machine learning, analytical android tools development, control system, and computer programming. He dedicates himself to university teaching and conducting research. His research interests include artificial intelligence, machine learning, computational algorithms, optimal control, mathematical model in epidemiology and bioinformatics. He can be contacted at email: tegar.prasetyo@del.ac.id or arifintegar12@gmail.com.

Joshua Pratama Silitonga is a fresh graduate with a degree in Information Technology from the Del Institute of Technology, currently employed as a Software Engineer at PLN Icon Plus. He has a keen interest in software engineering, particularly in web platforms, project management, and data analysis. He can be contacted at email: joshua.silitonga@iconpln.co.id.

Matthew Alfredo is a fresh graduate in Information Technology from the Del Institute of Technology, currently seeking new opportunities in various places, including participating in a bootcamp program at Sea Labs Indonesia. Interested in the field of software development and engineering, particularly in back-end development and other technologies related to back-end engineering. He can be contacted at email: matthew.alfredo@shopee.com or matthewwalfredoo@gmail.com.

Risky Saputra Siahaan is a fresh graduate with a degree in Information Technology from the Del Institute of Technology, currently employed as a Software Engineer at Jobseeker Company. He has a keen interest in backend developer (Java Spring, Mongo, Kafka, Microservice Architecture) dedicated to crafting optimal APIs. He can be contacted at email: riskysaputrasiahaan@gmail.com.

Prof. Dr. Roberd Saragih received his B.S. degree in mathematics and a Magister's degree in instrumentation and control from Institut Teknologi Bandung, Indonesia, in 1986, and 1993, respectively. He received a Ph.D. degree in mechanical engineering from Keio University, Japan, in 1998. From 1989, he joined the Department of Mathematics, Institut Teknologi Bandung, where he is currently a professor of mathematics. His general area of interest is robust control, system theory, and stochastic control. He can be contacted at email: roberd@math.ac.id or roberdsaragih58@gmail.com.

Dewi Handayani received the B.S. degree, magister degree, and doctor degree in mathematics from Bandung Institute of Technology. She is a current Lecturer and Researcher member in Department of Mathematics at Institut Teknologi Bandung. Her general area of interest is biomathematical model, robust control, stochastic control and system theory. She can be contacted at email: dewi.handayani@math.itb.ac.id.

Rudy Chandra current Lecturer and Researcher member in Information Technology Department at Institut Teknologi Del since 2022. Have experience at specializing in building machine learning model, artificial neural network, decision support system, distributed system, software testing, and computer programming. He dedicates himself to university teaching and conducting research. His research interests include artificial intelligence, neural network, machine learning and algorithm computational, decision support system, distributed syste, and software testing. He can be contacted at email: rudychandra@del.ac.id or rudychandra0@gmail.com.