 https://doi.org/10.24191/jeesr.v26i1.010

Predicting User Trajectories using Deep Learning Algorithms

Ahmad Zaki Aiman Abdul Rashid, Azita Laily Yusof*, and Norsuzila Ya'acob

***Abstract*—In order to produce seamless handover performance, a user's trajectory acts as a catalyst in determining the exact time and position of making the handover from one base station to another base station. Due to this, this paper predicts user's future trajectory from past trajectory utilizing deep learning (DL) algorithms which are Long-Short Term Memory (LSTM), Bi-Directional LSTM, and Gated Recurrent Unit (GRU). Next, the performance of the model will be evaluated using regression metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and the Coefficient of Determination (R^2). The simulation results displayed LSTM model surpasses other models (GRU, Bi-Directional LSTM) on the basis of accuracy achieved such as lowest MSE (0.084), MAE (0.254), MAPE (83.6%) with the highest R^2 score (-0.379). Our LSTM model was also compared to other researchers LSTM-based model for trajectory prediction and produce greater accuracy with ADE of 0.2359 and FDE of 0.1834. These conclude that LSTM model are the most suitable model for predicting user trajectories among DL algorithms. This work demonstrates the potential of the LSTM model for predicting user trajectories with high accuracy and improve handover performance through prediction.**

***Index Terms*—Deep Learning, LSTM, Bi-Directional LSTM, GRU, Trajectory Prediction**

I. INTRODUCTION

Utilization Unmanned Aerial Vehicle's or drones as a Base Station (UAV-BS) is seen as a promising potential for extending the 5G signals through multiple areas. The definition of UAV-BS as stated in [1] is the UAV mounted with cellular communication framework to supply both of terrestrial users and aerial users with communication services. In heterogeneous networks, UAV-BS might be deployed with terrestrial BS to assist the traffic requirement which is seen as an economical way of supporting the demands instead of building another terrestrial BS [2].

This manuscript is submitted on February 17, 2025, revised on March 4, 2025, and accepted on March 7, 2025. Ahmad Zaki Aiman Abdul Rashid and Azita Laily Yusof are with the School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, Shah Alam.

*Corresponding author
Email address: azita968@uitm.edu.my

However, the main problem with high density area created by the heterogeneous networks is the increasing rate of handover either from terrestrial BS to UAV-BS or from UAV-BS to another UAV-BS which might produce signaling overhead [3].

On the basis of handover between two UAV-BS, handover performance was optimized by considering user trajectory prediction [4]. This approach can be further optimized, in relation to handover performance as illustrated in Fig. 1, which shows the correlation between artificial intelligence (AI), deep learning (DL), machine learning (ML), neural networks (NN), and generative AI on left-hand side, while the right-hand side displays sub-sets of ML. AI capable of assisting in deciding when a handover is necessary, ML capable of predicting the optimal base station for handover based on historical data about UE's motion and network traffic, and DL capable of refining these predictions by incorporating real-time data such as UE sped, UE trajectory, optimizing handover decisions to guarantee seamless connectivity and good QoS is achieved.

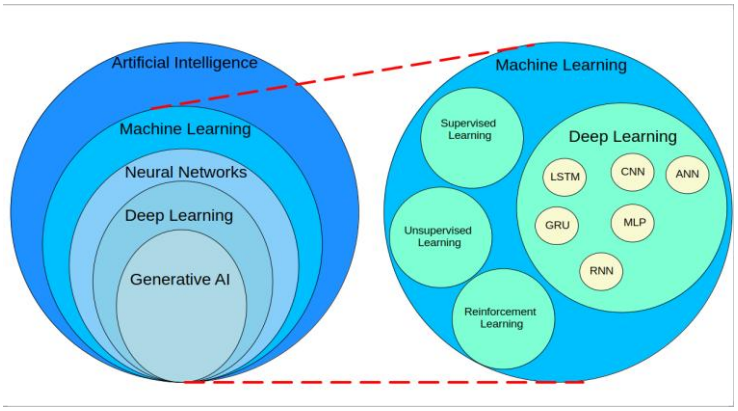

Fig. 1. Correlation between AI, ML, NN, DL and Generative AI

According to G. Guney et. al [5], a DL customized version of ML begin its usage in the year 2010s. R. K. Mishra et. al [6] stated that DL is in the group of methods utilized in ML and uses multiple layers to classify various factors relevant to the input data in order to extract characteristics from the raw data. L. Alzubaidi et. al in [7] concluded that DL algorithm is better than ML algorithm due to DL has the capability of eliminating the ML algorithm long process and capable of following a manner that is highly automated. As depicted in Fig. 1, DL technique consists of Artificial Neural Network (ANN), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Convolutional Neural Network (CNN), Long-Short Term Memory (LSTM), and Multilayer Perceptron (MLP).

According to [8], LSTM have been used widely for

analyzing trajectory due to its great predictive end-results. P. Stojkovic and P. Tadic in [9] utilized LSTM model to predict the object location coordinates which able to minimize the error and concluded that the model predicts the location of objects with a good degree of accuracy. The proposed model achieved a reduction in error by 76% when compared to the conventional Kalman filter method. J. B. Fernandez et. al in [10] utilized LSTM model to forecast the objects upcoming path which are moving in traffic. Besides, LSTM have great performance for time series data, they also able to prove that LSTM has great accuracy for trajectory prediction due to having minimum Average Displacement Error (ADE) for all the tested objects such as pedestrians (0.01m), cyclists (0.02m), and vehicles (0.02m).

Z. Zainuddin et. al in [11] also stated that GRU have better accuracy in predicting future state. In this paper, the RNN-GRU model proposed showed an accuracy of 87% for the prediction. P. Han et. al in [12] uses GRU model to predict short-term real-time trajectory coordinate point which showed lower Real Mean Squared Error (RMSE) when compared to other models such as ARIMA and Holt-Winters. In this paper, the Real-time GRU achieved a lower longitude error at 0.032 when compared to Normal-GRU (0.065) and Normal-LSTM (0.067). D. Guan et. al in [13] uses GRU with the additional GCN to forecast the trajectory of vehicles, and, showed higher prediction accuracy in contrast to alternative models like GCN, and Bi-directional GRU. The GCN-GRU proposed in this paper produced lower MAE (0.76) , RMSE (0.74), and capable of achieving 95% accuracy in trajectory prediction.

S. Zhang et. al in [14] utilized Bi-Directional LSTM to predict the ship's trajectory and able to produce effective trajectory prediction accuracy with MSE of 0.00047, greater than LSTM. D. Sahadevan et. al in [15] utilized Bi-directional LSTM model to forecast the trajectory of the aircraft using dataset from ADS-B, and, showed that Bi-directional LSTM has greater accuracy when compared to other models such as Back Propagation Neural Network, LSTM and CNN-LSTM. P. Casabianca et. al in [16] used Bi-directional LSTM to forecast the vehicle's upcoming destination of a vehicle by taking consideration the journey's history. Results from the simulation showed that Bi-directional LSTM with an attention mechanism provides better accuracy prediction than other models by obtaining 96% accuracy.

Due to this, an analysis is required to determine which DL algorithms have the highest accuracy in predicting users trajectories. In this paper, DL algorithms which are LSTM, Bi-Directional LSTM, and GRU will be employed to predict user trajectories. The rest of this paper is constructed into four sections. Section II outlines the research's methodology, Section III displays the results and analysis, and Section IV concludes the article.

II. METHODOLOGY

This section presents the methodologies used to carry out the research is explained, starting with the utilities such as hardware and software used to run the DL models. Next, the creation of dataset is explained including the splitting of datasets for testing and training. Process of building the DL model follows, detailing each of the DL models (LSTM, GRU, and Bi-Directional LSTM) and their structures. The methodology then continues with hyperparameter tuning for each models to determine the best hyperparameters for optimizing performance. Lastly, the evaluation of the DL models is carried out using regression metrics such as R^2 , MAE, MAPE, and, MSE.

A. Hardware and Software

Hardware used for running the ML algorithms was Lenovo ThinkPad T420 with Processor of Intel Core i7-2620M x 4 running Ubuntu 24.02.1 LTS. The software used was Visual Studio Code (VSCode) to run the programming language, Python and its libraries such as Numpy, Matplotlib, Pandas Scikit-Learn, Keras, and Tensorflow. Pandas was utilized to read the CSV file containing past coordinates (x, y), Numpy was used to convert data into sequences for time-series forecasting, Matplotlib was used to plot the graph, Keras was used to build the DL models and for hyperparameter tuning, Scikit-Learn was utilized for coordinates normalization, for dividing the dataset into testing/training dataset, and to evaluate the DL models using regression metrics (e.g MSE, MAPE, MAE, R^2). Tensorflow was used for importing the Keras libraries. Fig. 2, Fig. 3 and Fig. 4 which depicts each of the DL model structure was plotted using Keras Utils called "plot_model".

B. Dataset Creation

The dataset containing past coordinates (x, y) was created using Python programming language with the Random library and Pandas library. Total of 100 random coordinates were generated and was saved as a comma-delimited file (e.g. CSV). The dataset was then converted to NumPy array and goes through data normalization using the MinMaxScaler. The data is then converted into sequences with the sequence length, SEQ_LENGTH equals to 10. After creating sequences, data was separated into testing (20) and training (80) dataset to produce X_train shape of (72,10,2) and Y_train shape of (72, 2) for all the DL models. The reason for using this dataset, specifically is to standardize the data training and for preventing from bias occurred during the training. The reason for splitting the dataset into training (80) and testing (20) is to guarantee that 20 percentage of the data is utilized for testing, avoiding from any points in the testing dataset to appear in the training dataset, thus preventing leakage of data [17]. The reason for testing (20) dataset was utilized was to guarantee the model's performance can be evaluated on data that it has no knowledge of, thus capable of avoiding from overfitting to be occur [18]. The X_train, which is the input training data, offers an organized depiction of the information, thus enabling the model to find correlations and patterns between the features [19], consists of 72 samples, 10 time-steps, and each time step contains 2 features at each time-step. The Y_train, which is the output data, contains the target variable that the model aims to predict [20], consists of 72 samples, and 2 output variables.

C. DL Model Building

Three of these models were developed by utilizing the Keras's Sequential class for grouping multiple stacks of layers and transforms it into a DL model.

1) Long-Short Term Memory (LSTM)

LSTM model as depicted in Fig. 2 was created with five (5) layers, with the first LSTM layer using rectified linear unit (ReLU) activation. Second layer consists of the first dropout layer with min_value of 0.1, max_value of 0.5 and step of 0.1. This indicates the lower bound starting from 0.1 with increment of 0.1 up till 0.5 for the optimization of dropout layer. Third layer consists of another LSTM layer made similar to the first one with min_value of 32, max_value of 128 and step of 16. This indicates that the number of LSTM units is being optimized through hyperparameter tuning, ranging from 32 to 128, with increment of 16. Fourth layer consists of another dropout layer similar to the first layer of dropout. Fifth layer consists of the Dense layer with activation type linear. Compilation of the model is made with Adam optimizers and three different values of learning rate (0.001, 0.0005 and, 0.0001).

Fig. 2. LSTM Model

2) Bi-Directional LSTM

The Bi-Directional LSTM model shown in Fig. 3 was created with five(5) layers consisting the first Bi-Directional LSTM layer made with min_value of 32, max_value of 128, and step of 16. The second layer is the first dropout layer with min_value of 0.1, max_value of 0.5 and step of 0.1. Third layer consists of another Bi-Directional LSTM layer made similar to the first one. Fourth layer consists of another dropout layer similar to the first dropout layer. Fifth layer consists of Dense layer with activation type linear. Adam optimizers and three different values of learning rate (0.001, 0.0005, and 0.0001) was used for compiling the model.

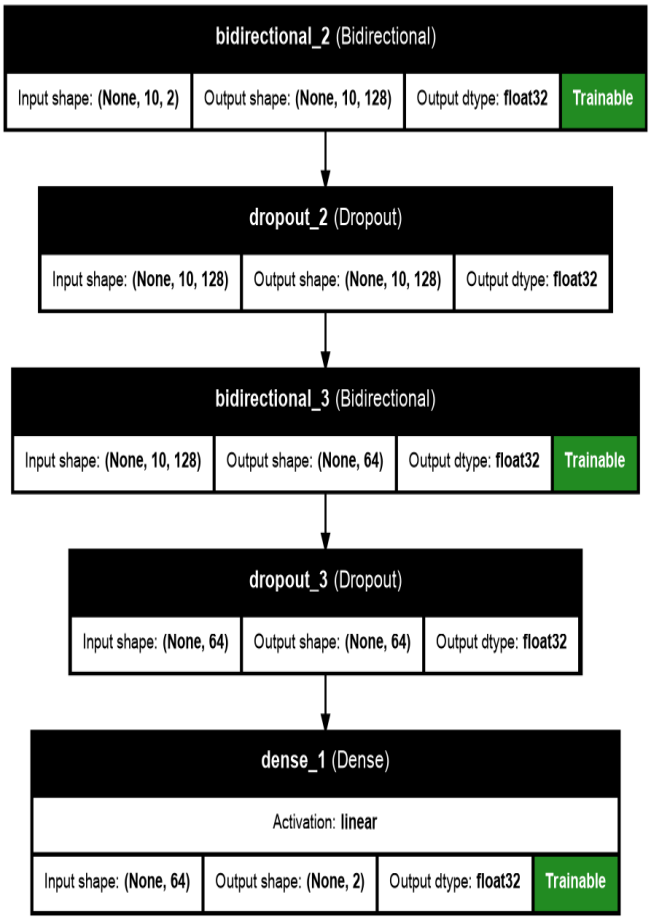

Fig. 3. Bi-Directional LSTM Model

3) Gated Recurrent Unit (GRU)

The GRU model in Fig. 4 was created with five (5) layers consisting the first GRU layer made with min_value of 32, max_value of 128, and step of 16. The second layer is the first dropout layer with min_value of 0.1, max_value of 0.5 and step of 0.1. Third layer consists of GRU layer made similar to the first one. Fourth layer consists of another dropout layer similar to the first dropout layer. The last layer, fifth layer is layered with Dense and linear activation. Similar to the two models, GRU model is compiled with Adam optimizers and similar learning rate as the two models previously.

Three of the models were standardized to have two layers of each model (e.g. LSTM L1, LSTM L2, GRU L1, GRU L2, Bi-directional LSTM L1, Bi-directional LSTM L2) and two dropout layer for each model (Dropout L1, Dropout L2) and one Dense layer for each model (Dense L1). The reason for this is

to standardize the experiment in order to prevent from having bias that might influence the experiment. The minimum (min_value) and maximum (max_value) value for Dropout layer and all the DL models (LSTM, Bi-LSTM, GRU) is also being standardized at the start of optimization in order to get even results, however, after hyperparameter optimization, the min_value and max_value will be different due to it has gone through optimization.

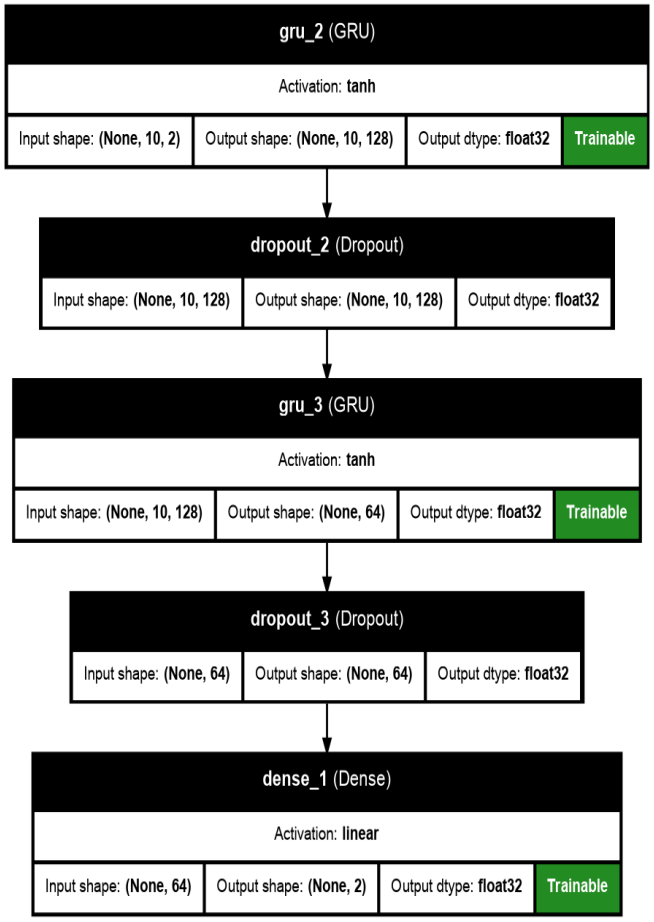

Fig. 4. GRU Model

Three of these models goes through hyperparameter optimization by tuning them using Keras Tuner with optimization algorithms type Random Search. All of the models have similar objectives which are the validation loss (val_loss), maximum trials (max_trials) of 5 and number of executions per trial was set to 1, for the purpose of standardization. Study made by J. Bergstra and Y. Bengio in [21] concluded that Random Search is better at hyperparameter tuning than Manual Search and Purely Grid Search with lesser computational time, and, highly effective in searching better models in search spaces that has greater dimensions. This study is supported by K. E. S. Pilario et. al in [22] where Random Search is concluded to be highly efficient and practical than other methods based on population such as PSO and GA due to having lower computational costs than the two method. Fig. 5 displays the Random Search used for Hyperparameter Tuning where the points were selected randomly across the search space and increases the chances of locating optimal hyperparameters at a fast rate due to its sampling from diversified regions.

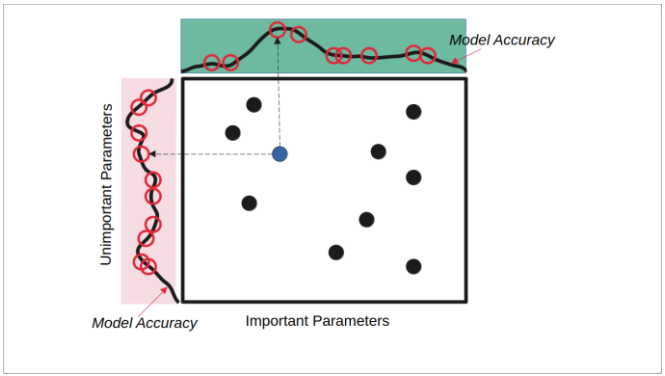

Fig. 5. Random Search for Hyperparameter Tuning

E. Evaluation Metrics

Primary purpose of evaluating the DL model through performance metrics such as MSE, MAE, MAPE, and R^2 is to compare predictions made by the DL model that has been trained with the original dataset from the testing dataset [23].

1) Mean Squared Error (MSE)

Mean Squared Error (MSE) can be measured using (1). Predicted coordinates is depicted as x_i^{pred}, y_i^{pred} , actual coordinates is depicted as x_i^{true}, y_i^{true} and the number of data points is depicted as n. According to [24], the value of MSE is between 0 to ∞, and, the greatest MSE value for a model is the value nearest to zero.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\left(x_i^{pred} - x_i^{true}\right)^2 + \left(y_i^{pred} - y_i^{true}\right)^2\right) \quad (1)$$

2) Mean Absolute Error (MAE)

Mean Absolute Error (MAE) can be described as the average of the absolute differences between values that are actual and values that are predicted. MAE can be calculated using (2). According to [25], MAE is commonly utilized when the ML model performance is calculated on data variables that are continuous.

$$MAE = \frac{1}{n}\sum\left(\left|x_i^{pred} - x_i^{true}\right|\right) + \left(\left|y_i^{pred} - y_i^{true}\right|\right) \quad (2)$$

3) Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) calculates the average differences between values that are actual and values that are predicted, in percentage. MAPE can be calculated using (3). Sungil. K and Heeyoung. K in [26] indicates that MAPE is among the most prominent calculations for determining the accuracy of the forecast.

$$\text{MAPE}=\frac{1}{n}\sum_{i=1}^{n}\left(\left(\frac{\left|x_i^{pred}-x_i^{true}\right|}{\left|x_i^{true}\right|}\right)+\frac{\left|y_i^{pred}-y_i^{true}\right|}{\left|y_i^{true}\right|}\right)x100 \quad (3)$$

4) Coefficient of Determination (R^2)

The definition of coefficient of determination R-squared (R^2) as stated by D. Zhang in [27] calculates the variation's proportions in the responses explained by the predictors that is available. R^2 can be calculated using (4). RSS is the sum of squares residual which consists of actual coordinates (x, y) and predicted coordinates (x^{pred}, y^{pred}) and TSS is the total sum of squares consisting actual coordinates (x, y) and mean values of the actual coordinates ($\bar{x}, \bar{y}$).

$$R^2 = \frac{(TSS - RSS)}{TSS} = 1 - \frac{RSS}{TSS} \quad (4)$$

III. Results and Discussion

This section presents the results and discussion of the research. Fig. 6, Fig. 7, and Fig. 8 displays the learning curve plots of Epoch vs Validation Loss (val_loss) and Training Loss (loss) at Epochs of 50 for each model. Table I presents the best hyperparameters that can be used for each models after going through hyperparameter tuning.

TABLE I. Best Hyperparameters For Each Models

DL Model	Best 1st LSTM Unit	Best 2nd LSTM Unit	Best Dropout (Layer 1)	Best Dropout (Layer 2)	Best Learning Rate
LSTM	112	112	0.5	0.300000000 00000004	0.001
	Best 1st Bi-LSTM Unit	Best 2nd Bi-LSTM Unit	Best Dropout (Layer 1)	Best Dropout (Layer 2)	Best Learning Rate
Bi-Directional LSTM	64	32	0.1	0.4	0.001
	Best 1st GRU Unit	Best 2nd GRU Unit	Best Dropout (Layer 1)	Best Dropout (Layer 2)	Best Learning Rate
GRU	128	64	0.5	0.300000000 00000004	0.001

After building the best model for each of the model through hyperparameter tuning, the following step is model training through **fit** method by Keras, such as below;

```
trainingTheModel = best_model.fit(X_train, y_train,
epochs=50, validation_split=0.2, batch_size=32)
```

The **fit** method will then produces "history" object containing a combination list of Training Loss (loss) which calculates the error available on training dataset for every iteration during the training of the model, and, Validation Loss (val_loss) indicates that the error on a different dataset, which is the validation dataset which the model did not have knowledge during the training. A total of 50 values for loss and 50 values for val_loss was created in the history object after training of the data using the **fit** method. 50 values for each losses (loss, val_loss) indicates that for 50 epochs, 50 values will be created, indicating 1 value for each epoch (e.g. 1 value = 1 epoch).

Fig. 6 displays the LSTM model with hyperparameter tuning learning curve plot. The graph shows that the learning curves is overfitting at the first 10 epochs and proceeds to show good fitting for the next epochs with smaller generalization gaps between training loss and validation loss.

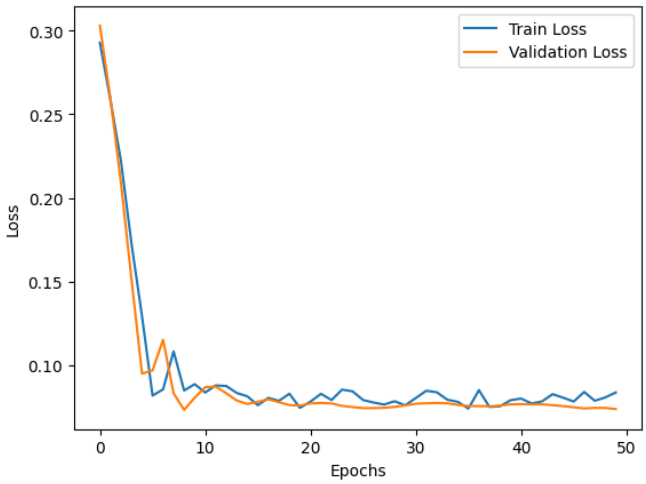

Fig. 6. LSTM + Hyperparameter Tuning (Epochs vs Loss)

Fig. 7 displays the Bi-Directional LSTM model with hyperparameter tuning learning curve graphs containing training loss and validation loss for epochs of 50. The graph shows that it has higher generalization gap for the first 10 epochs and the gap decreases at a lower rate as the epochs increases. Higher gap is seen between both losses for this model when compared to the LSTM model.

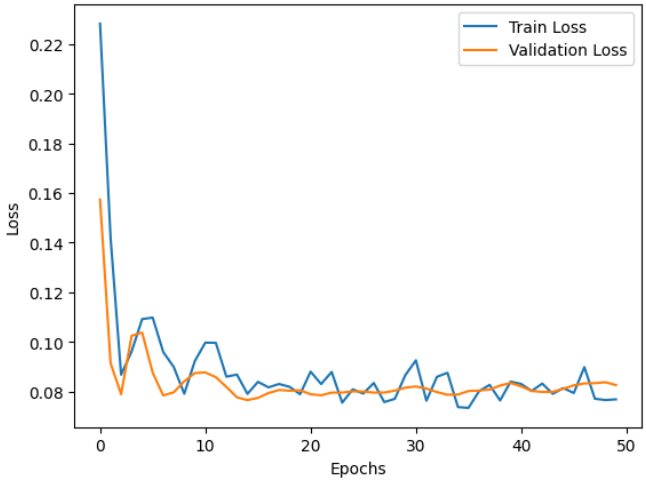

Fig. 7. Bi-Directional LSTM + Hyperparameter Tuning (Epochs vs Loss)

Fig. 8 presents the GRU model with hyperparameter tuning learning curve graph containing training loss and validation loss for 50 epochs. The graph shows that the highest gap between

both losses occurs at the first 10 epochs and the gap decreases as the epochs increases. The generalization gap between the two losses is the highest among the two other models (LSTM, Bi-Directional LSTM).

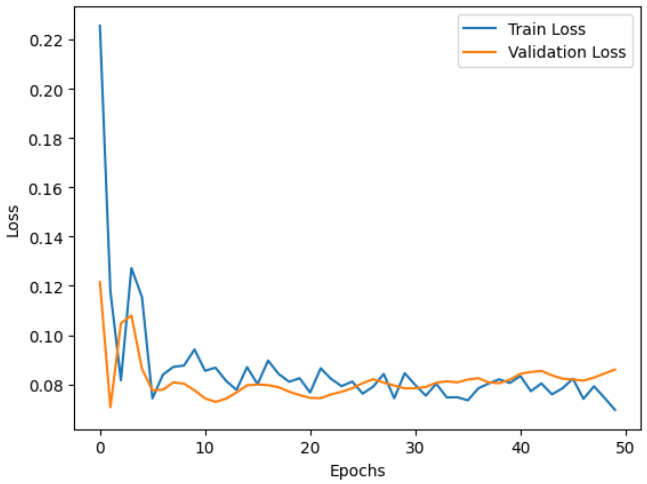

Fig. 8. GRU + Hyperparameter Tuning (Epochs vs Loss)

Fig. 9, Fig. 10, and Fig. 11 presents the actual path (x_{true}, y_{true}) and predicted path (x_{pred}, y_{pred}) for each of the DL models used, respectively. The actual path is in blue color and the predicted path is in red. Fig. 9 displays the Actual Path vs Predicted Path for the LSTM model. The figure shows that the predicted path is nearer to the actual path indicating highest accuracy in predicting the future path.

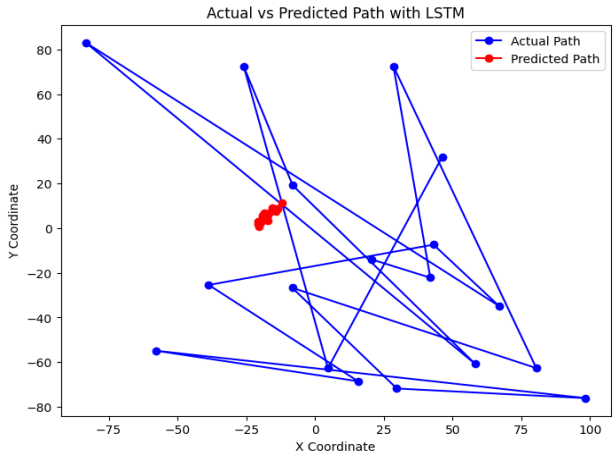

Fig. 9. LSTM for Actual vs Predicted Path

Fig. 10 shows the Actual Path vs Predicted Path for the Bi-Directional LSTM model. The predicted path shows good accuracy, however, there's a distance and gap between the predicted path and the actual path showing that is has lower accuracy rate than the LSTM model previously.

Fig. 11 presents the graph of Actual Path vs Predicted Path for the GRU model. Several predicted path have closer distance and minimal gap to the actual path, however, the accuracy of the predicted path is shown to have lower rate than the other two models through the scattered coordinates displayed.

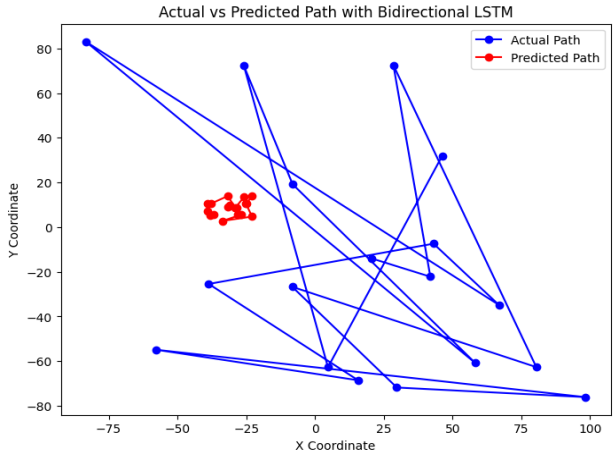

Fig. 10. Bi-Directional LSTM for Actual vs Predicted Path

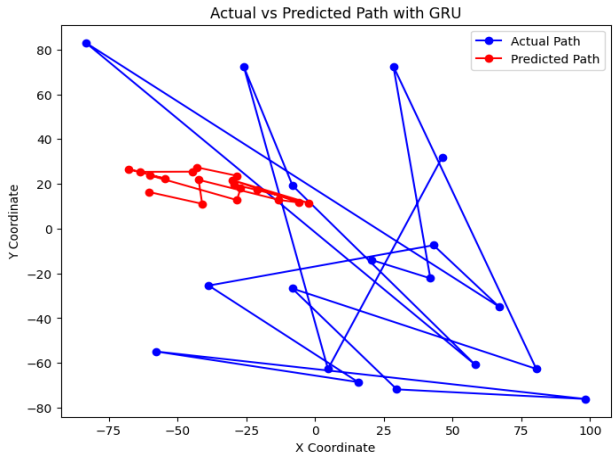

Fig. 11 GRU for Actual vs Predicted Path

Table II displays DL models evaluation using metrics which as Mean Squared Error (MSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and Coefficient of Determination/R-Squared (R^2). In the terms of MSE, LSTM has the nearest value to zero while GRU is the farthest from zero value. This indicates that LSTM has better predictive accuracy when compared to the other two DL models, and, GRU performs the worst in terms of accuracy. In terms of MAE, LSTM also showed that it has smaller MAE value when compared to Bi-Directional LSTM and GRU. This shows that LSTM has smaller errors and has greater prediction accuracy. In the context of MAPE, LSTM shows smaller percentages than the other two models displaying greater accuracy than the other two models. Lastly, in the context of coefficient of determination/R-squared (R^2), even though all the R^2 has negative values indicating underperforming models, however, LSTM model displayed greater R^2 values among the three of the models due to being closer to the value zero.

TABLE II. EVALUATION OF DL MODELS

DL Model	MSE	MAE	MAPE	R^2 Score
LSTM	0.084	0.254	0.836 ≈ (83.6 %)	-0.379
Bi-Directional LSTM	0.116	0.3000	0.950 ≈ 95.01 %	-0.924
GRU	0.122	0.305	0.968 ≈ 96.771%	-1.036

For comparison with previous works, we have chosen the research paper in [28] that utilized IA-LSTM for predicting pedestrian trajectory prediction. The dataset used for the trajectory prediction is based on the UCY-ZARA02 from [29, 30] for standardized purpose. The comparison with other researchers paper based on the the metrics utilized for evaluating trajectory predictions models as stated in [10, 31, 32] such as Average Displacement Error (ADE) [33] as depicted in Formula (5) which is defined as average Euclidean distance between the real value and predicted value over all time steps, and , Final Displacement Error (FDE) [34] as depicted in Formula (6) which is defined as the Euclidean distance between the final real value and final predicted value.

$$ADE = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\left(\left(x_{true}-x_{predicted}\right)^2+\left(y_{true}-y_{predicted}\right)^2\right)} \quad (5)$$

$$FDE = \sqrt{\left(x_{true}-x_{predicted}\right)^2-\left(y_{true}-y_{predicted}\right)} \quad (6)$$

Table III displays the comparison of configurations between our LSTM model and the IA-LSTM model in [28]. It consists of the dataset used, the number LSTM layer used, the dropout layer used, the activation, sequence length, test size, number of units/hidden neurons, optimizer, epochs, learning rate, batch size, the ADE and the FDE.

TABLE III. COMPARISON OF CONFIGURATIONS BETWEEN OUR MODEL AND MODEL [28].

	Our LSTM Model (After Hyperparameter Tuning)	[28]
Dataset	UCY-ZARA02	UCY-ZARA02
LSTM Layer	2	1
Dropout Layer	2	0
Activation	Tanh	Relu
Sequence Length	10	N/A
Test Size	0.2	N/A
Number of Units/Hidden Neurons	LSTM (Layer 1) : 112 LSTM (Layer 2): 80 Dropout (Layer 1): 0.30000000000000004 Dropout (Layer 2: 0.30000000000000004	128
Optimizer	SGD	Adam
Epochs	50	150
Batch Size	32	8
ADE	0.2359	0.4827
FDE	0.1834	0.7099
Learning Rate	0.005	0.001

Fig. 11 displays the graph of ADE and FDE for our model and the IA-LSTM model in [28]. Our LSTM model capable of achieving ADE of 0.2359 and FDE of 0.1834, which was which have differences of 73.5058% and 133.685%, respectively with the LSTM model in [28]. This indicates that our model has greater prediction accuracy when compared to other models.

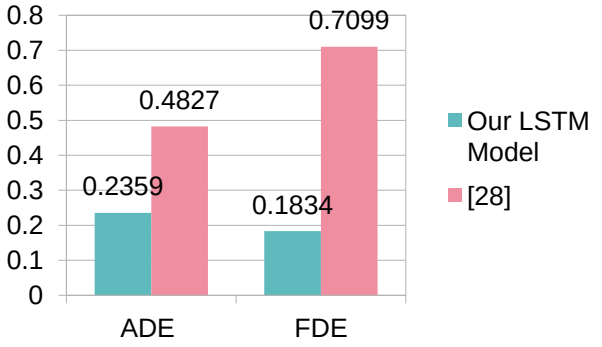

Fig 11. Average Displacement Error (ADE) and Final Displacement Error (FDE) for Our LSTM Model and IA-LSTM Model [28]

As shown in Table III, the differences between our model and model [28] is the optimizer, the number of epochs, the batch size, learning rate and the number of layers used by the models. The model [28] uses the Adaptive Moment Estimation (Adam) optimizer while our model utilizes the Stochastic Gradient Descent (SGD) optimizer. In this context, the types of optimizer might affect the value of ADE and FDE. This statement is supported by the researchers in [35] where they stated the the Adam optimizer commonly leads to degrading performance than SGD for training deep neural networks. Despite their advantages such as high speed convergence, the Adam optimizer was stated in [36 , 37 , 38] to have poorer generalization performance than SGD. The rate of the epochs might also played a role in the ADE and FDE values. Based on [39], increasing the epochs at a higher rate does not improve the learning and might produce overfitting. Another reasons that affect the ADE and FDE values is the batch size, where our model has a batch size of 32 while model [28] has batch size of 8. Even though smaller batch size has the advantages of

minimizing the training time and less memory usage, larger batch size are known to produce minimum amount of prediction error [40], therefore, it will affect the model's ADE and FDE. Another reason that plays a role in the differences between ADE and FDE is the sequence length used. The model [28] does not utilized sequence length and our model utilized sequence length of 10. This is supported by a research made by [41] where sequence length might affects the model's prediction performance. Longer sequence length makes the model hard to remember information that is relevant and shorter sequence length makes the model to have insufficient context for making predictions accurately. Last reason why our models have greater ADE and FDE than model [28] is due to the usage of multiple LSTM layers and Dropout layers with the additional hyperparameter tuning. Multi-LSTM layers produces better prediction performance due to its capability of capturing temporal dependencies that are much more longer than single LSTM layers and produces lower losses [42]. Adding dropout layer to the LSTM is one of the techniques to overcome overfitting and enhances the model's performances [43]. Optimizing the model's through hyperparameter tuning affect the model's prediction performance positively [44], and, with the Random Search tuner utilized with this model, it provides the most suitable hyperparameter's for predicting the coordinates accurately. Random Search was stated to have greater advantages than other tree-based algorithms for hyperparameter tuning [45].

As a summary, the reason for differences of ADE and FDE between our model and model [28] is due to:

i. Type of Optimizer [36 , 37 , 38]
ii. Rate of Epochs [39]
iii. Batch Size [40]
iv. Sequence Length [41]
v. Multiple LSTM Layers [42] and Dropout Layers [43] with Hyperparameter Tuning [44 , 45]

IV. CONCLUSION

In this paper, three DL models for predicting trajectory were analyzed. Each of the three models underwent hyperparameter tuning to optimize the performance. Learning curve graph for validation loss and training loss were plotted for each models. Next, the actual path was plotted against the predicted path for each model, displaying their accuracy and accuracy of prediction. Lastly, the three models were evaluated using regression metrics such as MSE, MAE, MAPE, and R^2. It can be concluded that the LSTM model has the greatest accuracy in trajectory prediction when compared to the other models. The LSTM model was also compared with other LSTM-based models and showed greater prediction accuracy by achieving ADE of 0.2359 and FDE of 0.1834, lower than the other LSTM-based models. The trajectory prediction can be applied in terms of handover performance where in a UAV-BS network, the UE trajectory is predicted to target the next base station for initiating handover. For future work, the Early Stopping method could be included when training the DL model to optimize the LSTM model. Additionally, it would be beneficial to analyze the accuracy of prediction of the LSTM with different types, such as Convolutional LSTM or LSTM with Attention Mechanism.

ACKNOWLEDGMENT

This paper is part of research work supported by FRGS Grant file no.: FRGS/1/2024/TK07/UITM/02/6 and School of Electrical Engineering College of Engineering, Universiti Teknologi MARA Shah Alam.

REFERENCES

[1] D. Mishra, H. Gupta and E. Natalizio, "SKY5G: Prototyping 5G Aerial Base Station (UAV-BS) for On-Demand Connectivity from Sky," 2024 IEEE Wireless Communications and Networking Conference (WCNC), Dubai, United Arab Emirates, 2024, pp. 1-6, doi: 10.1109/WCNC57260.2024.10570546.

[2] T. Hirai, K. Doi and N. Wakamiya, "Optimal Deployment of an Aerial Base Station in Heterogeneous Cellular Networks for Heterogeneous User Traffic Demands," 2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring), Florence, Italy, 2023, pp. 1-6, doi: 10.1109/VTC2023-Spring57618.2023.10200246.

[3] J. Zhong, L. Zhang, M. Alhabo, J. Serugunda and S. N. Mugala, "A Hybrid Scheme Using TOPSIS and Q-Learning for Handover Decision Making in UAV Assisted Heterogeneous Network," in IEEE Access, vol. 12, pp. 31422-31430, 2024, doi: 10.1109/ACCESS.2024.3368916.

[4] B. Hu, H. Yang, L. Wang and S. Chen, "A trajectory prediction based intelligent handover control method in UAV cellular networks," in China Communications, vol. 16, no. 1, pp. 1-14, Jan. 2019, doi: 10.12676/j.cc.2019.01.001.

[5] G. Guney et al., "An Overview of Deep Learning Algorithms and Their Applications in Neuropsychiatry," Clinical Psychopharmacology and Neuroscience, vol. 19, no. 2, pp. 206–219, May 2021, doi: https://doi.org/10.9758/cpn.2021.19.2.206.

[6] R. K. Mishra, G. Y. S. Reddy, and H. Pathak, "The Understanding of Deep Learning: A Comprehensive Review," Mathematical Problems in Engineering, vol. 2021, pp. 1–15, Apr. 2021, doi: https://doi.org/10.1155/2021/5548884.

[7] Laith Alzubaidi et al., "A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications," Journal of Big Data, vol. 10, no. 1, Apr. 2023, doi: https://doi.org/10.1186/s40537-023-00727-2.

[8] J. Violos, Stylianos Tsanakas, Maro Androutsopoulou, Georgios Palaiokrassas, and T. Varvarigou, "Next Position Prediction using LSTM Neural Networks," 11th Hellenic Conference on Artificial Intelligence, pp. 232–240, Sep. 2020, doi: https://doi.org/10.1145/3411408.3411426.

[9] P. Stojković and P. Tadić, "Object Location Prediction in Real-time using LSTM Neural Network and Polynomial Regression," arXiv, Jan. 2023, doi: https://doi.org/10.48550/arxiv.2311.13950.

[10] J. B. Fernandez, S. Little and N. E. O'Connor, "A Single-Shot Approach Using an LSTM for Moving Object Path Prediction," 2019 Ninth International Conference on Image Processing Theory, Tools and Applications (IPTA), Istanbul, Turkey, 2019, pp. 1-6, doi: 10.1109/IPTA.2019.8936126.

[11] Z. Z., P. A. E. A., and H. M. H., "Predicting machine failure using recurrent neural network-gated recurrent unit (RNN-GRU) through time series data," Bulletin of Electrical Engineering and Informatics, vol. 10, no. 2, pp. 870–878, Apr. 2021, doi: https://doi.org/10.11591/eei.v10i2.2036.

[12] P. Han, W. Wang, Q. Shi and J. Yang, "Real-time Short- Term Trajectory Prediction Based on GRU Neural Network," 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC), San Diego, CA, USA, 2019, pp. 1-8, doi: 10.1109/DASC43569.2019.9081618.

[13] D. Guan, N. Ren, K. Wang, Q. Wang, and H. Zhang, "Checkpoint data-driven GCN-GRU vehicle trajectory and traffic flow prediction," Scientific Reports, vol. 14, no. 1, Dec. 2024, doi: https://doi.org/10.1038/s41598-024-80563-3.

[14] S. Zhang, L. Wang, M. Zhu, S. Chen, H. Zhang and Z. Zeng, "A Bi-directional LSTM Ship Trajectory Prediction Method based on Attention Mechanism," 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 2021, pp. 1987-1993, doi: 10.1109/IAEAC50856.2021.9391059.

[15] D. Sahadevan, H. P. M, P. Ponnusamy, V. P. Gopi, and M. K. Nelli, "Ground-based 4d trajectory prediction using bi-directional LSTM networks," Applied Intelligence, Mar. 2022, doi: https://doi.org/10.1007/s10489-022-03309-6.

[16] P. Casabianca, Y. Zhang, M. Martínez-García, and J. Wan, "Vehicle Destination Prediction Using Bidirectional LSTM with Attention Mechanism," Sensors, vol. 21, no. 24, p. 8443, Dec. 2021, doi: https://doi.org/10.3390/s21248443.

[17] J. Stourac et al., "Training and test datasets for the PredictONCO tool," Zenodo (CERN European Organization for Nuclear Research), Dec. 2023, doi: https://doi.org/10.5281/zenodo.10374835.

[18] M. Sivakumar, S. Parthasarathy, and Thiyagarajan Padmapriya, "Trade-off between training and testing ratio in machine learning for medical image processing," PeerJ Computer Science, vol. 10, pp. e2245–e2245, Sep. 2024, doi: https://doi.org/10.7717/peerj-cs.2245.

[19] C. Lingenfelder, M. Wurst, and P. Pompey, "Method and system for predictive modeling," Nov. 03, 2011 [Online]. Available: https://patents.google.com/patent/WO2012084320A3/en

[20] Khashayar Ghadirinejad et al., "Supervised machine learning for the prediction of post-operative clinical outcomes of hip and knee replacements: a review," ANZ Journal of Surgery, vol. 94, no. 7–8, pp. 1228–1233, Apr. 2024, doi: https://doi.org/10.1111/ans.19003.

[21] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization Yoshua Bengio," Journal of Machine Learning Research, vol. 13, pp. 281–305, 2012. ISSN:1532-4435.

[22] K. E. S. Pilario, Y. Cao, and M. Shafiee, "A Kernel Design Approach to Improve Kernel Subspace Identification," IEEE Transactions on Industrial Electronics, vol. 68, no. 7, pp. 6171–6180, Jul. 2021, doi: https://doi.org/10.1109/tie.2020.2996142.

[23] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM networks," Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., Montreal, QC, Canada, 2005, pp. 2047-2052 vol. 4, doi: 10.1109/IJCNN.2005.1556215.

[24] A. Jadon, A. Patil, and S. Jadon, "A Comprehensive Survey of Regression Based Loss Functions for Time Series Forecasting," arXiv.org, Nov. 05, 2022. https://doi.org/10.48550/arXiv.2211.02989

[25] H. Wahid, N. I. Abdul Razak, and S. A. Che Abdullah, "Machine Learning Model for Performance Prediction in Mobile Network Management," Journal of Electrical & Electronic Systems Research, vol. 21, no. OCT2022, pp. 101–107, Nov. 2022, doi: https://doi.org/10.24191/jeesr.v21i1.013.

[26] S. Kim and H. Kim, "A new metric of absolute percentage error for intermittent demand forecasts," International Journal of Forecasting, vol. 32, no. 3, pp. 669–679, Jul. 2016, doi: https://doi.org/10.1016/j.ijforecast.2015.12.003.

[27] D. Zhang, "A Coefficient of Determination for Generalized Linear Models," The American Statistician, vol. 71, no. 4, pp. 310–316, Dec. 2016, doi: https://doi.org/10.1080/00031305.2016.1256839.

[28] J. Yang, Y. Chen, S. Du, B. Chen and J. C. Principe, "IA-LSTM: Interaction-Aware LSTM for Pedestrian Trajectory Prediction," in IEEE Transactions on Cybernetics, vol. 54, no. 7, pp. 3904-3917, July 2024, doi: 10.1109/TCYB.2024.3359237.

[29] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by Example," Computer Graphics Forum, vol. 26, no. 3, pp. 655–664, Sep. 2007, doi: https://doi.org/10.1111/j.1467-8659.2007.01089.x.

[30] L. Matteo, C. Pasquale, and B. Lamberto, "Social and Scene-Aware Trajectory Prediction in Crowded Spaces," arXiv (Cornell University), Jan. 2019, doi: https://doi.org/10.48550/arxiv.1909.08840.

[31] A. Mohamed, D. Zhu, W. Vu, M. Elhoseiny, and C. Claudel, "Social-Implicit: Rethinking Trajectory Prediction Evaluation and The Effectiveness of Implicit Maximum Likelihood Estimation," arXiv (Cornell University), Jan. 2022, doi: https://doi.org/10.48550/arxiv.2203.03057.

[32] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in Proc. Int. Conf. Robot. Autom., 2018, pp. 1–7 , doi: https://doi.org/10.48550/arxiv.1710.04689.

[33] S. Pellegrini, A. Ess, K. Schindler and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 2009, pp. 261-268, doi: 10.1109/ICCV.2009.5459260.

[34] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 961-971, doi: 10.1109/CVPR.2016.110.

[35] A. Gupta, R. Ramanath, J. Shi, and S. Keerthi, "Adam vs. SGD: Closing the generalization gap on image classification." , OPT2021: 13th Annual Workshop on Optimization for Machine Learning , 2021. [Online]. Available: https://www.opt-ml.org/papers/2021/paper53.pdf .

[36] P. Zhou, J. Feng, C. Ma, C. Xiong, Steven, and Weinan E, "Towards Theoretically Understanding Why SGD Generalizes Better Than ADAM in Deep Learning," Oct. 2020, doi: https://doi.org/10.48550/arxiv.2010.05627.

[37] L. Luo, Y. Xiong, Y. Liu, and X. Sun. "Adaptive gradient methods with dynamic bound of learning rate". In Int'l Conf. Learning Representations, 2019, doi: https://arxiv.org/abs/1902.09843v1.

[38] Nitish Shirish Keskar and R. Socher, "Improving Generalization Performance by Switching from Adam to SGD," arXiv, Jan. 2017, doi: https://doi.org/10.48550/arxiv.1712.07628.

[39] Napoleão Verardi Galegale and C. I. Shimabukuro, "Deep Learning Applied to Stock Prices: Epoch Adjustment in Training an LSTM Neural Network," International Journal of Business and Management, vol. 19, no. 4, pp. 80–80, Jun. 2024, doi: https://doi.org/10.5539/ijbm.v19n4p80.

[40] J.-S. Hwang, S.-S. Lee, J.-W. Gil, and C.-K. Lee, "Determination of Optimal Batch Size of Deep Learning Models with Time Series Data," Sustainability, vol. 16, no. 14, pp. 5936–5936, Jul. 2024, doi: https://doi.org/10.3390/su16145936.

[41] Safwan Mahmood Al-Selwi, Mohd Fadzil Hassan, Said Jadid Abdulkadir, and Amgad Muneer, "LSTM Inefficiency in Long-Term Dependencies Regression Problems," Journal of Advanced Research in Applied Sciences and Engineering Technology, vol. 30, no. 3, pp. 16–31, May 2023, doi: https://doi.org/10.37934/araset.30.3.1631.

[42] F. Xiao, "Time Series Forecasting with Stacked Long Short-Term Memory Networks," arXiv , Jan. 2020, doi: https://doi.org/10.48550/arxiv.2011.00697.

[43] Y. Li et al., "A Survey on Dropout Methods and Experimental Verification in Recommendation," arXiv , Jan. 2022, doi: https://doi.org/10.48550/arxiv.2204.02027.

[44] R. Hossain and D. Timmer, "Machine Learning Model Optimization with Hyper Parameter Tuning Approach ," Global Journal of Computer Science and Technology: D Neural & Artificial Intelligence, vol. 21, 2021. [Online]. Available: https://globaljournals.org/GJCST_Volume21/2-Machine-Learning-Model-Optimization.pdf

[45] Muhammad Hevny Rizky, M. R. Faisal, I. Budiman, Dwi Kartini, and F. Abadi, "Effect of Hyperparameter Tuning Using Random Search on Tree-Based Classification Algorithm for Software Defect Prediction," IJCCS (Indonesian Journal of Computing and Cybernetics Systems), vol. 18, no. 1, pp. 95–95, Jan. 2024, doi: https://doi.org/10.22146/ijccs.90437.