Design of a Deep Learning-based Flight Delay Prediction Model for Flight Disruptions Using an Optimized Integrated Recommendation System

Jacob Seth Barcelona
Technological Institute of the Philippines
Quezon City, Philippines
qjssbarcelona@tip.edu.ph

Maria Antonette Castillo
Technological Institute of the Philippines
Quezon City, Philippines
qmaocastillo@tip.edu.ph

Nil Jedidiah San Gabriel
Technological Institute of the Philippines
Quezon City, Philippines
qnjgsangabriel@tip.edu.ph

Roman M. Richard
Technological Institute of the Philippines
Quezon City, Philippines
rrichard.cpe@tip.edu.ph

***Abstract*— Empirical data points flight delay as the dominating factor that contributes to the passenger's disrupted flight experience. Until late, there is no direct solution to address flight delays— which render the passengers clueless, not informed whether their flight would be delayed. Through deep learning and an optimization algorithm, this paper aims to provide a user-based recommendation system that integrates both concepts to have the capability in mitigating the impact of flight delays. The dataset that was used for this study is "Flight Status Prediction", a dataset from Kaggle. The time series model that was integrated is Long Term Short Memory (LSTM)— obtaining a total of 9.22 RMSE and 6.78 MAE on unseen data, the lowest possible delay among other time series models (Simple RNN and GRU). Utilizing Pareto Optimality, the Genetic algorithm was the chosen Metaheuristic algorithm since it was capable of computing the optimized alternative route, achieving a 9.96 score. The recommendation system was tested to have an 86% precision in recommending non-delay flight dates and a total of 612.1952 average difference in recommending alternative optimized routes; ergo, the system is capable of providing reliable options based on the passenger's preference of flight schedule. Results obtained from this research may be used to improve and test alternative approaches to further address the impacts of flight delays.**

Keywords*— *Flight Disruption; Flight Experience; Aviation; Optimized Routes; Metaheuristic Algorithm, Booking Efficiency

I. INTRODUCTION

With the exponential increase of the industry of aviation, flight delays— until present, persists as a dilemma to travelers. Statistics suggest that economic loss is evident when experiencing flight delays, and it is stated that billions of money worldwide have been expended for both passengers and airlines in respect to recuperating for the delays [1]. Following that, it is frustrating for the passengers without prior knowledge whether their respective flight would be delayed or not. According to a passenger flight survey in 2023 from Civil Aviation Authority [2], flight delays prevails as one of the dissatisfying issues for passengers, ultimately leading the survey.

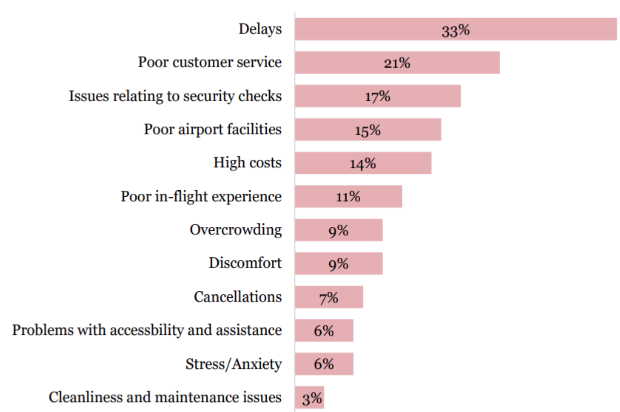

FIGURE 1. Passenger Travel Experience Survey [2]

The bar graph in Figure 1 shows the findings of the survey where it highlights the overall flight experience asking the passengers about the primary factor of their dissatisfaction. Notably, it could be inferred that the factor that influenced the experience of the passengers was flight delay— garnering a percentage of 33% of the responses. Majority of the passengers who participated in the survey expressed their dissatisfaction towards this factor, which ultimately indicates that flight delays have a significant impact on the passengers' experience.

From the official website of the United States government— Bureau of Transportation Statistics [3], states that from January 2020 to February 2024, among 26,761,631 flight operations, a total of 3,239,900 flight operations were affected by delays. The total delay operations came from two categories; specifically, Air Carrier Delay— a delay that is caused by the airline itself, mainly because of maintenance, scheduling problems, or any technical or operational issues that are within the jurisdiction of the airline amounted to 1,673,034 flights and Aircraft Arriving Late— a delay cause of the late arrival of the aircraft itself from the previous flight, a delay that is beyond the airline's jurisdiction amounted to 1,566,866 flights affected. A visual representation of the flight delays would be then shown below, where the statistics that are mentioned would be presented through a bar graph, indicating the total number of delays from 2020 to 2024.

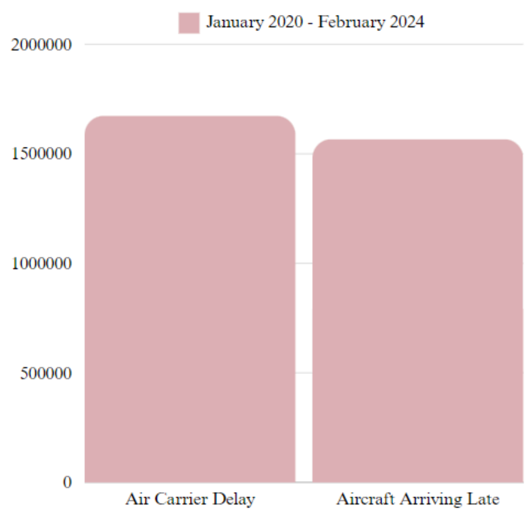

FIGURE 2. Delayed Flight Operations from BTS [3]

This research objectively aims to develop three things that will encourage and minimize flight disruptions according to the passengers' experience— a deep learning model that would be able to predict the number of delays (based on the chosen flight schedule of the passenger) expressed in minutes. Utilize the Metaheuristic algorithm— an optimization algorithm that is used to address complicated issues that cannot be solved using standard approaches [4], which will be specifically used for finding an alternative optimized route (it will be based from the passenger's origin to destination route). Lastly, a user-based recommendation system that will integrate the deep learning model and algorithm for the passenger's smooth experience in booking (since the recommendation system will be deployed into a web-application).

This paper is structured as follows: Section 2 outlines the details of methodology. Section 3 shows the results of the comparison of deep learning models, the performance evaluation of different types of Metaheuristic algorithms [5], and the _____ of the recommendation system. Lastly, Section 4 presents a conclusion with recommended actions for future research.

II. METHODOLOGY

A. Data Gathering

The dataset that was used for training and testing the deep learning model(s) is from Kaggle [6], where the dataset is equipped with 61 attributes along with 512,201,319 total elements— which will be minimized once pre-processed, to only choose relevant attributes to predict the target feature "DepDelay" using a time series model. In the Metaheuristic algorithm, the "Origin" and "Dest" feature from the dataset was applied to determine the alternative optimized route. Additionally, a library called airportsdata [7], an extensive database which was used for the longitude and latitude for the target routes, Haversine [8] which is used to calculate the distance between two points, and MEALPY [9] a cutting-edge meta-heuristic algorithms to in the fields of approximate optimization. Ideally, a dataset that is equipped with a weather attribute would have been preferred; however, due to the lack of resources, where premium subscriptions were required, this was the only decent dataset which was available that catered the objectives of this research. Although web scraping was used as an additional resort, it was not able to exactly accomplish anything due to past limited data. With all alternatives exhausted, and Freedom of Information (FOI) not being responsive to our queries, thus the reason for using this data.

B. Data Pre-processing

In this section, the first pre-processing step was to execute a correlation matrix and to observe the relevant features that would help in predicting the target variable, departure delay. Scrutinizing the correlation matrix, from originally 61 attributes— it was reduced to 4 attributes, which is correlated to the weak relevance of the other features. Through careful analysis, the items listed in table 1 would be the final features that would be used to train and test for the deep learning models, with the goal of predicting the delay expressed in minutes.

TABLE I. LIST OF FEATURES CHOSEN FOR TIME SERIES MODEL

Flight Date	Day
Month	Delay per Day

TABLE II. LIST OF DATA SET FEATURES USED FOR METAHEURISTIC ALGORITHM

Origin	Destination

In table 2, there are exactly two features that are listed, where testing for different Metaheuristic algorithms— an algorithm which is distance and coordination driven, other attributes of the dataset did not have any importance when executing such an optimization algorithm. The two features that are listed bear the IATA codes [10] which are variables that are important when computing for the optimized alternative route based either from the point-to-point or origin-to-destination of the passenger.

C. Deep Learning Models for Time Series Prediction

This part focuses on steps to prepare the data for training and testing. After the selection of the features for the time series model, a variable named delay data was created. This variable will store all of the flight operations under the year of 2021. Afterwards, the flight date is first converted into a date-time data type since its original data type was object. Calculating the average departure delay per day is also executed in this part, along with extracting the month and day from the 'FlightDate' column. Consequently, merging of the daily average delay data with the 'DayOfWeek' column from the delay data frame was applied, for the day-specific delay information to be uniform, and at the same time duplicate entries were dropped. Lastly, the resulting data frame— daily average delay— had its columns rearranged specifically to 'FlightDate', 'Month', 'Day', 'DayOfWeek', and 'DelayPerDay'. To visually represent the newly created data frame, figure 3 shows the summary of the average minutes that is usually experienced by passengers and depicts that December had the highest peak of average delays in 2021.

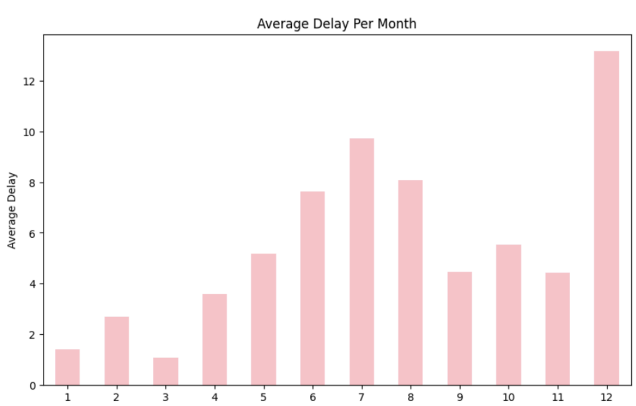

FIGURE 3. Average Delay per Month in 2021

To proceed, the flight date was label encoded since feeding a model typically requires a numerical input. In this case, the data type of flight date is datetime as mentioned at the beginning. After ensuring that the data types of the daily average delay data frame are all either integers or floats, splitting the data in an 80:20 ratio, for training and testing respectively, will now be allowed. Scaling the data using a standard scaler [11] would be the first step, to achieve normally distributed data. The independent variables— which is X, are the columns FlightDate, Month, Day, and DayOfWeek. The target variable— y, is DelayPerDay. The three time series models used to predict the number of delayed minutes are Long Short-Term Memory (LSTM) [12], Simple RNN [13], and Gated Recurrent Unit (GRU) [14].

D. Metaheuristic Algorithm Selection

This algorithm focuses on finding the optimized alternative route based on the user-input. To develop the Metaheuristic algorithm, an algorithm is first created to find all the possible combinations of routes to take— developing such an algorithm requires the mentioned libraries, which are airportsdata, Haversine, and MEALPY. Through the utilization of the libraries, a code is created where it loads the IATA airport data, defines a function to get coordinates for given airport codes, and maps these codes to their coordinates. The code then identifies and lists all of the airport codes whichever returns invalid coordinates (0,0). Another code is created to define the Metaheuristic algorithm to find the shortest path between two points, which considers the potential flight delays, using the MEALPY library. A 'RouteFinder' class is then constructed that extends to the 'Problem' variable, to evaluate the fitness of a route based on the total travel distance. A 'Meta' class is then initialized with the waypoints, origin, destination, and model parameters, which will be ultimately used for generating a distance graph from waypoint coordinates (with the help of the library of Haversine), and updates the graph dynamically. A parameter where setting the optimization model is also available to find the shortest path. 'run_meta' method is also formed, where it iterates the continuous validating and updating of the graph, while minimizing the travel distance. It will continue to iterate until an optimal path is found or the number of iterations is reached, the final output then includes the optimal route, its distance, and a benchmark distance from utilizing the Haversine formula [15].

Additionally, the kind of Metaheuristic algorithm that is specifically used to find the optimized alternative route are Ant Colony Optimization [16], Genetic Algorithm [17], and Particle Swarm Optimization [18].

E. Deployment of Recommendation System

This section focuses on the functionalities of a recommendation system, where it is integrated with a deep learning model designed to predict flight delays. First, a user-input is required, where they are problem to enter their flight date and route (origin and destination IATA codes). Afterwards, the model then predicts whether the specified flight of the passenger will experience a delay. If a delay is predicted, the system will have the capability of recommending alternative dates for the same route where flights are predicted to not have any delays. Moreover, a Metaheuristic algorithm is incorporated with the recommendation system, where it is responsible for finding the optimized alternative route according to the user's specified origin and destination. The route recommended is the best path, but does not guarantee that there will be no delays. The recommendation system would be deployed using Streamlit [19] for an interactive web-application, and for user-input to be enabled. The overall recommendation system would be able to easily interact with passengers, in which they will be informed of delays of their selected flight, get recommendations about alternative dates and best routes to take, which overall leaves an impact on flight disruptions.

F. Metrics and Evaluation

To evaluate the time series models in deep learning, MAE and RMSE [20] will be used to properly showcase the performance of each model. Since the predicted variable will not be categorical, but expressed in minutes— the loss outcome by using the formulas are the difference of delayed minutes that are predicted by the model compared to the ground truth. In RMSE, where y_i ; $\hat{y}_i$ is the predicted value during training, and the calculation result is in minutes after the square and square root. Likewise with MAE, while the calculation result is also in minutes, it instead averages the absolute differences.

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

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

On the other hand, in evaluating different approaches in the Metaheuristic algorithm, Pareto Optimality [21] is used to determine which algorithm is the optimal one, where the highest score is the base reference of it. In this case, there are three approaches, and whichever is computed the highest score would be tallied as the approach when obtaining the optimal route.

$$PC_{norm} = 9\frac{Max_{raw} - PC_{raw}}{Max_{raw} - Min_{raw}} + 1$$

Lastly, there would be two evaluation metrics that would be tested on the recommendation system [22], precision [23] and average difference. To be specific, precision will be tested in computing for the date recommendations generated by the recommendation system. The formula for computing the precision of the developed recommendation system is as follows:

$$Precision = \frac{\#\ of\ correct\ recommendations}{\#\ of\ recommendations}$$

Utilizing this formula, through the preferred data of the user, the deep learning model is in charge of predicting whether it has a delay. Then, a threshold of 15 [24] is the reference of the Boolean value. Corresponding to 0 (which indicates that the flight has no delay), and 1 (that the flight of the passenger has a delay). Whereas if the predicted delayed minutes is exactly > 15, it returns 1. Moreover, this metric is used to observe the accuracy between the correct recommendations and total recommendation. The former refers to the ground truth, while the latter pertains to the number of recommended dates that has no delay. The ground truth is the reference of the correct recommendations, to check if it is accurate.

On the other hand, testing for the generated recommended distance formula is as follows:

$$Difference = AD - OD$$

After computing the difference, it is then averaged. Where AD stands for Alternative Distance, and OD stands for Original Distance. Alternative distance is the pertained distance generated using the Genetic algorithm. While original distance computed distance from the passenger's exact origin to destination.

III. RESULTS AND DISCUSSIONS

To begin, the main objective of this paper is for passengers to know beforehand if their respective flight of choice would be delayed. Results that would be first presented would be the three models utilized to predict the delay in minutes. Shown in table 3 are the summary which indicates its corresponding RMSE and MAE. The training data contains the flight delays of the year of 2021, while the unseen is the year of 2022.

TABLE III. SUMMARY OF RESULTS ON VALIDATION DATA

Time Series Model	Loss (in minutes)	
	RMSE	MAE
LSTM	6.16	4.72
GRU	7.18	5.61
Simple RNN	7.68	6.05

First off, observing the results from table 3, it is evident that the lowest values in both RMSE and MAE category belonged to LSTM, which is inferred that the model outperformed the two time series models during prediction. The three time series models are ranked in the table, by which model managed to obtain the lowest possible predicted value. With LSTM, only being at the 6 threshold— 6.16 to be specific in RMSE, and the only model that managed to attain the lowest value among the three, garnering an MAE value of 4.72. It evidently means that LSTM was able to capture the data patterns more effectively compared to the other two models. In standard, lower values of RMSE and MAE indicate that the model is performing good [25][26][27]. To be precise, a lower RMSE means that its predictions are close to the ground truth— in LSTM's case at least. Similarly, a lower MAE indicates that the LSTM's average prediction error is smaller. Consistent lower error metrics, in both RMSE and MAE in this case, points to the inference that the model is able to handle the complexities of the data. It could then be concluded that among the three models, LSTM is proven to be more effective for this time series prediction task, given how it showed a great performance in learning and generalizing from the fed training data.

To reiterate, lower values of MAE and RMSE of a model points to the fact that it performs well, and learns from the underlying pattern from the training data. Both of the mentioned metrics' values range from 0 to ∞ and are negatively-oriented [27]; therefore, in the sense of "lower values", when values are predicted to be close to 0, it indicates better performance.

For further elaboration in regards to the performance of the time series models, 3 graphs will be shown - where each graph visually represents the model's predicted values against the ground truth. The figure will be shown from LSTM, Simple RNN, and GRU respectively.

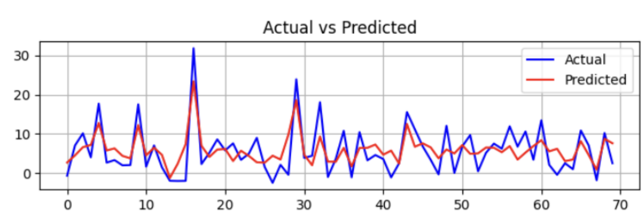

Figure 4. LSTM Performance on Validation Data

Based on the performance of LSTM, the visualized graph is accurate based on the summarized values in the table. It was mentioned that the lower the predicted values are, the closer it is to the ground truth; therefore, this graph supports that claim. It can be seen that the model is able to capture similar patterns to the actual data, and is not having great difficulties in capturing the underlying pattern of the data. Which again supports the values in table 3, where LSTM had the lowest error value. To add, the numbers in the X-axis pertain to the dates. There are minimal periods where the predicted values contain higher and lower delays than the ground truth values, but overall, the model is robust during training, since the predicted minutes of delay are tied closely with the ground truth.

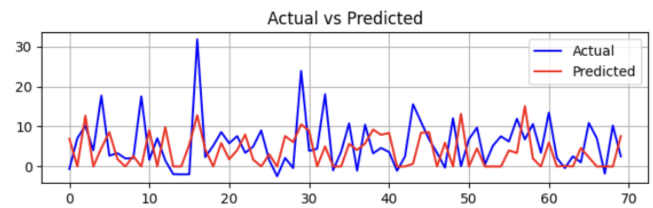

Figure 4.1. Simple RNN Performance on Validation Data

Simple RNN is ranked the lowest among the three, where it had the highest loss values, in both RMSE and MAE metric. To reiterate, in RMSE and MAE, lower values are preferred. Based on the graph, the values in table 3 under Simple RNN confirms it. Looking at it, there are periods where the model's predicted values are far from the ground truth values - which means that the model does not quite capture the pattern unlike the performance of LSTM. Generally, it is the highest loss value, but a 7.68 RMSE and 6.05 MAE is not too bad, but in this study's case, among the three models, it is the one that performed poorly.

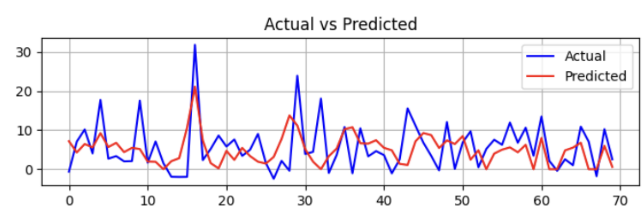

Figure 4.2. GRU Performance on Validation Data

GRU is the next model that performed well in generalizing the underlying pattern. This model obtained a 7.18 RMSE and a 5.61 MAE; it performed quite similarly with Simple RNN, given how its RMSE is in the 7 threshold. However, scrutinizing the graph, GRU indeed performed better than Simple RNN since it managed to capture most of the pattern, considering how close the predicted values are to the ground truth compared to the last ranked model. In table 3, LSTM is more than a point lower in both metrics, which ultimately infers that LSTM performed exceptionally well among the three models. Also, GRU on the other hand did not perform quite poorly, but its performance is at least decent.

TABLE IV. FINAL RESULT ON UNSEEN DATA

LSTM Model	Loss (in minutes)	
	RMSE	MAE
	9.22	6.78

LSTM was chosen as the model that will be tested on unseen data since it was the model that performed well, as to further verify if the model is truly able to capture the pattern properly. To add, the reason why the best performing model was only selected to be tested on unseen data was because of hardware limitations, which will be thoroughly discussed in the conclusion and recommendations section. Going back with table 4, it is shown that the RMSE and MAE metric stated that the model performed quite poorly compared to the validation set from the training data. Where the model has an RMSE of 9.22, and a MAE of 6.78. While RMSE is generally considered to be always greater than MAE [23], due to its squaring operation, an almost 3-point difference—specifically 2.44, between the two metrics is massive, comparing it to the training data, where it merely had a 1.44 difference. When exposed to unseen data, the model did not perform as well as it did during training, considering how the RMSE and MAE went higher instead of the predicted delayed minutes being further minimized.

To further observe the performance of the LSTM model on unseen data, figure 5 indicates the performance of the model within the next 200 dates.

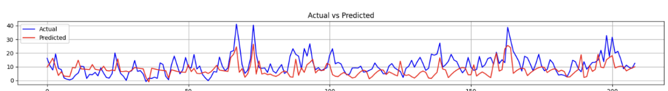

FIGURE 5. LSTM Performance on Unseen Data

The graph shows the LSTM model's performance when tested on unseen data, the reliability and the robustness of the model suggests that it performed somehow good, since the graph still indicates that the predicted values are quite close to the ground truth, although with many deviations. At the same time, the model was not able to capture some of the values in the ground truth, upon scrutinizing it. This supports the final result in Table 4, where the RMSE and MAE is higher for the unseen data compared to the training data. It could be inferred that the model was experiencing some difficulty on new data. Figure 5 summarizes that the model's predictions are less accurate, being equipped with larger average errors and greater variance, which is the opposite expected outcome, since RMSE and MAE are negatively-oriented values; therefore, numbers approaching 0 would be better and indicate that the model is performing well.

TABLE V. SUMMARY OF AVERAGE EXECUTION TIME AND DISTANCE METRICS

Algorithm	Execution Time	Distance (km)
Ant Colony Optimization	150.78	2592.71
Genetic Algorithm	14.73	1990.35
Particle Swarm Optimization	13.63	2266.47

The table shown is the summary of results after testing and running the algorithms for 10 instances. Simply, the Genetic Algorithm managed to obtain the lowest possible distance for optimized alternative routes while Ant Colony Optimization fell behind, obtaining the largest distance for the alternative routes. The results in Figure 5 would be tested using Pareto Optimization.

TABLE VI. RESULTS OF PARETO OPTIMAL

GOAL: Minimized	Ant Colony	Genetic Algorithm	Particle Swarm
Time	0.470588	4.671914	4.705882
Distance	0.529412	5.294118	3.109991
Score	1	9.966031	7.815873

To obtain the optimal route, a pareto optimization method was performed; wherein, table 6 would be the total score of each of the Metaheuristic algorithms used. Evidently, among the three algorithms, Ant Colony was the only one that had a small score. In this case, to seek minimization, the highest score tends to be the most optimal; however, in Ant Colony, it did not reach a high score. In the context of the study, Ant Colony as an algorithm would not be able to compute the optimal route. On the other hand, two algorithms stood out upon looking at the table, but one algorithm managed to obtain almost a score of 10. Particle Swarm fell short by roughly 3 but performed better than Ant Colony, having Genetic Algorithm as the highest total score. When the goal is to minimize, the highest total score is the most optimal one; ergo, Genetic Algorithm would be the best algorithm to compute for the minimized alternative optimal route in flight delays.

Moving on, results regarding the recommendation itself would be discussed in this part, where it focuses on explaining the obtained precision and average difference coming from the recommended dates and distances to the passenger.

Using the evaluation metric of precision, which is used to compute for the recommended dates, the recommendation system was able to garner an 86% precision score, which indicates that the recommendation system was able to recommend flight dates that have no delays in most cases.

To reiterate the formula, where the number of correct recommendations is divided by the number of recommendations, where the former is referenced to the ground truth, and the latter is the predicted dates based on the user-input. Again, a threshold of 15 is observed - once the model predicts that the passenger's flight date is delayed, the recommendation system will proceed in recommending flight dates that have no flight delays. However, if the value is 0, indicating that the flight has no delay, the recommendation system will simply notify the passenger that their flight has no delay. The recommended flight dates are monthly, suppose that the passenger's flight date has a delay and is within January; therefore, the recommendation system will recommend alternative flight dates within the month that has no delay. The precision score was obtained through using the deployed recommendation system in Streamlit.

In table 7, a sample tabular results of computing for the mean difference of the recommended alternative routes.

TABLE VII. AVERAGE DIFFERENCE OF RECOMMENDED ALTERNATIVE ROUTES

# of Instances	OD	AD	Difference
1	2791.42	2890.96	99.54
2	994.96	1582.96	588
3	269.91	666.09	396.18
4	2371.54	2394.97	23.43
5	5345.83	5921.48	575.65
5 Instances, AVG DIFF =		**336.56**	
50 Instances, AVG DIFF =		**612.1952**	

To emphasize, the table was shown as to show the process in obtaining the average difference, the first 5 instances are a part of the 50 instances— where the final result for computing the average difference is 612.1952. AD refers to the alternative distance and OD refers to the original distance, where the original distance is the actual route chosen by the passenger (A to C), and the alternative distance are the routes generated by the Genetic algorithm (A to B to C). The output of this metric is to simply show the efficiency of the recommended alternative routes (to verify if the recommended routes are truly optimized, since the goal of the recommended routes is the minimized distance).

IV. CONCLUSION AND RECOMMENDATIONS

This study is intended to make predictions about the delay of flights— which gives the passenger prior knowledge whether their respective flight is going to be delayed, and for how many minutes.

Through a flight booking recommendation system, the passenger will be prompted to input their flight details, namely: the date of flight along with its origin and destination (in IATA codes). Through the information collected from the user-input, it will proceed in predicting the delay of flight. If the date is identified to have no delays, it will not recommend anything and will simply tell that the flight is less likely to be delayed. However, if it does detect a delay on the supposed chosen date, the recommendation system— where LSTM is integrated, is able to infer how many minutes it will be delayed. Equipped with a Genetic algorithm, it would then also recommend an optimized alternative route for the passenger.

To synthesize, an inference is drawn where it highlights the impact of flight delays to passengers, along with possible solutions to mitigate it. With this paper, the outcome may have been a little bit different if a data set with more attributes had been worked on— which is not the case for this study since the model was only given 4 attributes. The study has the potential to be drastically improved, where other attributes such as weather may be incorporated, which may further refine the model when predicting the flight delays. Results-wise, LSTM was the time-series model that performed better among the three that was tested; however, that is only the case with the given data set, LSTM may or may not be improved when predicting flight delays using other data sets. Different utilization of deep learning models may also contribute to detect the flight delays, since only three were performed and compared with. Another challenge to this research is the hardware limitations, since LSTM is generally a model that requires a lot of historical data, which means that the architecture contains a lot of trainable parameters; hence this study may be tested on a high-ended personal computer, to explore for other time series models, alternative network structures, and parameters. The results that are gathered may have been improved if it was not because of the restrictions of hardware. Where the GPU offered at Colab did not suffice, even the dataset that is used caused the Colab to crash because of its high memory. Consequently, exploring other optimization algorithms, aside from Metaheuristic, may also perform well in finding the alternative optimized route— a guaranteed route where it will exactly infer that it will experience zero delay. The recommendation system also has areas for improvement, since it did not have the details to why the flight would be delayed or the factors behind why their flight is likely delayed within that day. It merely informs the passenger that their flight is delayed in n minutes and presents an optimized alternative route where even the route is not guaranteed to be delay-free.

REFERENCES

[1] Y. Liu, M. Yin, and M. Hansen, “Economic costs of air cargo flight delays related to late package deliveries,” Transportation Research. Part E, Logistics and Transportation Review, vol. 125, pp. 388–401, May 2019, doi: 10.1016/j.tre.2019.03.017.

[2] “Flying in 2023,” *Civil Aviation Authority*, October 2, 2023. [Online]. Available: https://www.caa.co.uk/publication/download/20986

[3] “Airline On-Time Statistics and Delay Causes”, *Bureau of Transportation Statistics*, Retrieved May 15, 2024, from https://www.bts.gov/content/airline-time-performance

[4] S. Almufti, A. Shaban, R. Ali, and J. Fuente, "Overview of Metaheuristic Algorithms," *Polaris Global Journal of Scholarly Research and Trends*, vol. 2, pp. 10-32, April 2023, 2023, doi: 10.58429/pgjsrt.v2n2a144.

[5] T. Bhattacharyya, B. Chatterjee, P. Singh, J. Yoon, Z. W. Geem, and R. Sarkar, "Mayfly in Harmony: A New Hybrid Meta-Heuristic Feature Selection Algorithm," *IEEE Access*, vol. 8, pp. 195642-195657, 2020, doi: 10.1109/ACCESS.2020.3031718.

[6] Aumhpatel, “Flight status prediction,” *Kaggle*, Apr. 28, 2024, from https://www.kaggle.com/code/aumhpatel/flight-status-prediction/input?select=Combined_Flights_2022.csv

[7] “airportsdata,” *PyPI*, Apr. 15, 2024, from https://pypi.org/project/airportsdata/

[8] “haversine,” *PyPI*, Jan. 16, 2024, from https://pypi.org/project/haversine/

[9] “mealpy”, *PyPI*, Nov. 5, 2023, from https://pypi.org/project/mealpy/

[10] “IATA airline and location codes,” *IATA,* Retrieved May 16, 2024*,* from https://www.iata.org/en/services/codes/

[11] “sklearn.preprocessing.StandardScaler,” *Scikit-learn,* Retrieved May 16, 2024, from https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[12] S. Saxena, “What is LSTM? Introduction to Long Short-Term Memory,” *Analytics Vidhya*, Jan. 04, 2024. https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/

[13] D. Kalita, “A brief overview of Recurrent Neural networks (RNN),” *Analytics Vidhya*, Feb. 07, 2024. https://www.analyticsvidhya.com/blog/2022/03/a-brief-overview-of-recurrent-neural-networks-rnn/

[14] S. Saxena, “Introduction to Gated Recurrent Unit (GRU),” *Analytics Vidhya*, Dec. 20, 2023. https://www.analyticsvidhya.com/blog/2021/03/introduction-to-gated-recurrent-unit-gru/#:~:text=GRUs%20are%20very%20similar%20to,LSTM%20and%20have%20simpler%20architecture.

[15] GeeksforGeeks, “Haversine formula to find distance between two points on a sphere,” *GeeksforGeeks*, Sep. 05, 2022, from https://www.geeksforgeeks.org/haversine-formula-to-find-distance-between-two-points-on-a-sphere/

[16] J. Nivash,“Understanding ant colony optimization algorithms,” *INDIAai*. https://indiaai.gov.in/article/understanding-ant-colony-optimization-algorithms

[17] V. Mallawaarachchi, “Introduction to genetic algorithms,” *Medium*, Sep. 21, 2023, [Online]. Available: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3

[18] A. Tam, “A gentle introduction to particle swarm optimization,” *MachineLearningMastery.com*, Oct. 11, 2021. https://machinelearningmastery.com/a-gentle-introduction-to-particle-swarm-optimization/

[19] N. Mhadhbi, “Python Tutorial: Streamlit,” *Datacamp,* Dec. 21, 2021, from https://www.datacamp.com/tutorial/streamlit

[20] S. Acharya, “What are RMSE and MAE? - Towards Data Science,” *Medium*, Jan. 06, 2022. [Online]. Available: https://towardsdatascience.com/what-are-rmse-and-mae-e405ce230383

[21] S. Suffian, R. Dzombak, and K. Mehta, “Future directions for nonconventional and vernacular material research and applications,” in *Elsevier eBooks*, 2020, pp. 63–80. doi: 10.1016/b978-0-08-102704-2.00003-2.

[22] P. Aher, “Evaluation Metrics for Recommendation Systems — An overview,” *Medium*, Aug. 16, 2023. [Online]. Available: https://towardsdatascience.com/evaluation-metrics-for-recommendation-systems-an-overview-71290690ecba

[23] D. Benveniste, “How to ARCHITECT a search engine like Google Search,” *The AiEdge Newsletter*, Jan. 25, 2023, from

https://newsletter.theaiedge.io/p/how-to-architect-a-search-engine

[24] “Airline On-Time Performance - Defining late,” *OAG,* Retrieved date: May 17, 2024, from https://www.oag.com/airline-on-time-performance-defining-late

[25] J. Qu, M. Xiao, L. Yang, and W. Xie, "Flight Delay Regression Prediction Model Based on Att-Conv-LSTM," *Entropy (Basel, Switzerland)*, May 8, 2023, vol. 25, no. 5, p. 770, 2023. doi: 10.3390/e25050770.

[26] V. Rastogi, “RMSE and MAE - Vaibhav Rastogi - Medium,” *Medium*, Aug. 13, 2023. [Online]. Available: https://medium.com/@vaibhav1403/rmse-and-mae-415470f52b58

[27] “Mean absolute error (MAE) and root mean squared error (RMSE),” *eumetrain,* Retrieved May 17, 2024, from https://resources.eumetrain.org/data/4/451/english/msg/ver_cont_var/uos3/uos3_ko1.htm#:~:text=The%20MAE%20is%20a%20linear,weighted%20equally%20in%20the%20average.&text=The%20RMSE%20is%20a%20quadratic,in%20both%20of%20the%20references.