Technische Hochschule Ingolstadt

Bachelor Thesis

in the study program Autonomous Vehicle Engineering
Faculty of Electrical Engineering and Computer Science

Predicting Bus Arrival Time at Traffic Light Stop Lines using Machine Learning Methods

First- und Last name : **Elsayed Fared Ibrahim Khalil Ibrahim**

Registered on : 15.05.2024

Submitted on : 23.05.2024

First Examiner : Schmidtner, Prof. Dr. Stefanie

Second Examiner : Ebert, Prof. Dr. Bernd Martin

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, that I have not presented it elsewhere for examination purposes, and that I have explicitly indicated all material which has been quoted either literally or by consent from the sources used. I have marked verbatim and indirect quotations as such.

Ingolstadt, _____

Elsayed Ibrahim

Acknowledgments

I would like to thank everyone who contributed to this work, and to Professor Dr. Stefanie Schmidtner for having this opportunity to do my thesis with her.

In addition, I would like to extend my gratitude to Philip-Roman Adam for his great support.

Finally, a special thanks to my wife, Elizaveta, whose love, patience, and unwavering support were invaluable throughout this journey. Her understanding and encouragement provided me with the strength and determination to persevere, even during the most challenging times.

Elsayed Ibrahim
Ingolstadt, Germany
July 16 2024

Abstract

Accurate and efficient prediction of the estimated arrival time at traffic lights for public transportation systems is a core factor. This thesis studies the application of machine learning techniques, in particular Linear Regression, Multi-Layer Perceptron, and Long Short-Term Memory networks to predict bus arrival times utilizing Floating Car Data. In this work, the data were focused on seven months within a particular urban area, and preprocessing was carried out to handle invalid runs and congestion effects. Other features engineered for the enhancement of model accuracy involved distance, time of day, day of the week, and month of the year.

The current study shows that LSTM performs relatively better than the Linear Regression and the MLP models, having the lowest RMSE and the highest R-squared value. Notably, the accuracy of the data increases significantly when dwell times are eliminated, indicative of the fact that predicting bus stop probability or their durations is indeed, which is a very complicated task. Evidence from my results suggests that ML models could greatly enhance prediction for bus arrival time compared with traditional mean-based methods.

These results were promising, but the one obvious limitation in this work is that it focuses on only one stop line of the traffic light and does not consider any external parameters, such as weather or passenger load. Therefore, future work will focus on developing models that can span wider scopes for multiple traffic lights, bring more external factors, and learn and adapt in real-time. This research sets a way for future developments in bus-arrival-time prediction; it helps in increasing the overall efficiency and reliability of public transport systems.

Acronyms

Adam Adaptive Moment Estimation.

EAT Estimated Arrival Time.

EDA Exploratory Data Analysis.

FCD Floating Car Data.

KDTree K-Dimensional Tree.

LSTM Long Short-Term Memory.

MAE Mean Absolute Error.

MLP Multi-Layer Perceptron.

MSE mean squared erro.

ReLU Rectified Linear Unit.

tanh hyperbolic tangent.

VIF Variance Inflation Factor.

List of figures

List of tables

Table of contents

1 Introduction

This chapter presents the background and motivation for my study in predicting bus estimated arrival times at the traffic light stop line using machine learning techniques. The increasing demand for effective, reliable public transportation in urban areas has necessitated the need for more adaptive and accurate prediction methods. Traditional approaches often fall short of accounting for the real-time variation and complexities of urban traffic. This chapter provides the statement of the problem, objectives, scope, and organization of the dissertation; thus, it sets the stage for a detailed exploratory journey.

1.1 Background and Motivation

Efficient public transportation is a cornerstone of urban development, particularly as cities continue to grow and the demand for reliable transit increases. Buses, a primary mode of public transport, are essential for daily commuting. However, predicting the arrival times of buses, especially at traffic light stop lines, remains a significant challenge due to the inherent variability of urban traffic conditions. An accurate prediction of bus arrival times can dramatically improve the efficiency of public transportation systems, reduce passenger wait times, and enhance overall user satisfaction.Policy [2023].

Traditionally, bus arrival time predictions have relied on two main methods: schedule-based predictions and historical data analysis. Schedule-based predictions assume that buses will adhere strictly to a predefined timetable. While this method is straightforward and easy to implement, it often falls short in accounting for real-time variations such as traffic congestion, weather conditions, and unexpected delays. On the other hand, historical data analysis uses past travel times to forecast future arrivals by averaging travel times under similar conditions. Although this method can be more reliable than schedule-based predictions, it tends to smooth out significant variations and can be slow to adapt to real-time changes.Aidan O'Sullivan [2016].

Another technique employed to enhance bus arrival predictions is the bus priority signal system. This method prioritizes buses at traffic lights, aiming to reduce delays and improve schedule adherence. While bus priority signals can effectively reduce delays at intersections, they are not always practical due to infrastructure limitations and the potential impact on other traffic.Wu [2018].

1.2 Problem Statement

Despite the advancements in traditional methods and the implementation of bus priority signals, predicting bus arrival times with high accuracy remains a challenge. The dynamic nature of urban traffic requires more sophisticated and adaptive approaches that can handle real-time variations effectively.

1.3 Objectives

The main objective of this study is to develop and evaluate machine learning models to predict bus arrival times at traffic light stop lines accurately. To achieve this, the study focuses on the following goals:

- **Data Preprocessing**: Cleaning and preparing the data for analysis.
- **Exploratory Data Analysis (EDA)**: Understanding the data and identifying significant patterns.
- **Feature Engineering**: Creating meaningful features to improve model performance.
- **Model Development**: Developing and evaluating various machine learning models.
- **Model Evaluation**: Comparing the performance of different models to determine the most effective approach.

1.4 Scope of the Study

This research is concentrated on predicting bus arrival times at traffic light stop lines using a dataset collected from a specific urban area. The study involves data preprocessing, exploratory data analysis, feature engineering, and the development and evaluation of multiple machine learning models. The performance of these models will be compared against a baseline mean model, traditional linear regression, multi-layer perceptron (MLP), and long short-term memory (LSTM) networks. Additionally, the study will assess the impact of including and excluding dwell times at bus stops on model performance.

1.5 Thesis Organization

The thesis is structured into several chapters, each addressing a different aspect of the research:

- **Chapter 1: Introduction** – Provides the background, motivation, problem statement, objectives, scope, and organization of the thesis.
- **Chapter 2: Literature Review** – Examines existing methods for predicting bus arrival times and highlights gaps in current research.
- **Chapter 3: Methodology** – Describes the data collection, preprocessing steps, exploratory data analysis, feature engineering, and the machine learning models used.
- **Chapter 4: Implementation** – Details the software, tools, model training, validation processes, and evaluation metrics.
- **Chapter 5: Results and Discussion** – Presents the results of the models, compares their performance, and discusses the findings.
- **Chapter 6: Conclusion and Future Work** – Summarizes the key findings, contributions, limitations, and suggests directions for future research.

2 Literature Review

This chapter reviews existing research work on bus arrival time prediction. The chapter begins by looking at traditional methods of schedule-based predictions, historical data analysis, and bus priority signal systems. It then proceeds to discuss recent machine learning techniques for Linear Regression, ANN, and LSTM. Here, identification where research might be lacking and thus where this study would be needed and could contribute.

2.1 Traditional Methods

In the field of bus arrival time prediction, traditional methods have long been utilized to estimate when buses will reach their stops. These methods include schedule-based predictions, historical data analysis, and bus priority signal systems. Each of these approaches has its own set of advantages and limitations.

2.1.1 Schedule-Based Calculation.

Schedule-based predictions are among the oldest and simplest methods used for predicting bus arrival times. This approach relies on predefined timetables that specify the expected arrival times of buses at various stops. The primary advantage of schedule-based predictions is their simplicity and ease of implementation. Passengers and transit authorities can easily understand and follow a fixed schedule. Aidan O'Sullivan [2016].

However, this method has significant limitations. It assumes that buses will adhere strictly to the timetable, which rarely happens in real-world conditions. Factors such as traffic congestion, roadworks, accidents, and weather conditions can cause substantial deviations from the schedule. As a result, schedule-based predictions often fail to provide accurate arrival times, leading to passenger dissatisfaction and inefficient bus operations.

2.1.2 Historical Data Analysis.

Another traditional method for calculating bus arrival times is historical data analysis. This approach utilizes past travel time data to forecast future arrivals. By averaging travel times under similar conditions (e.g., time of day, day of the week, weather conditions), historical data analysis can provide more reliable predictions compared to schedule-based methods. Yongpei Huang [2021].

The main advantage of this approach is its ability to incorporate real travel time data, making it more adaptive to typical traffic patterns. However, historical data analysis also has its drawbacks. It tends to smooth out significant variations and may not accurately capture unusual or unexpected conditions. Additionally, this method can be slow to adapt to real-time changes, as it relies on historical averages rather than current conditions.

2.1.3 Bus Priority Signal System

The bus priority signal system is a more advanced traditional method aimed at improving the punctuality of bus services. This approach involves adjusting traffic signals to prioritize buses, allowing them to pass through intersections more quickly and reducing delays. By giving buses priority at traffic lights, this system helps to improve schedule adherence and reduce overall travel time.Smith [2005].

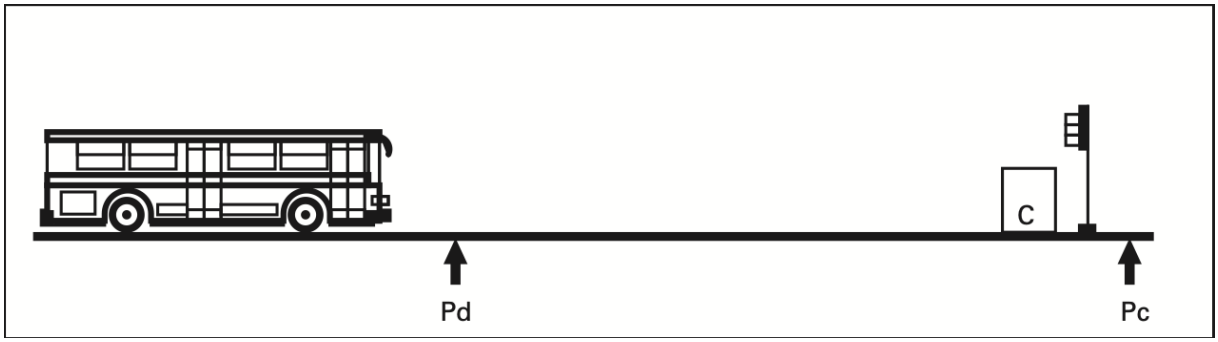

Figure 1: Bus Priority Signal System
Smith [2005]

Figure 1 shows the General Steps in Providing Bus Priority at Intersections.

1. Bus Detection and Priority Request: The bus is detected by the bus detection system **Pc** in some point **Pd** upstream of the intersection. A priority request is sent to the traffic control system **C**, which decides whether to grant priority based on predefined conditions.
2. Traffic Signal Adjustment: If the bus is approaching during a green light, the green phase is extended. If the bus is approaching during a red light, the green phase for the cross street is shortened to provide an earlier green phase for the bus.
3. Restoration of Normal Timing: Once the bus clears the intersection, detected by the downstream system, the traffic controller restores normal signal timing using predetermined logic.

The advantages of bus priority signal systems are clear: they can significantly reduce delays at intersections and enhance the reliability of bus services. However, implementing such systems requires substantial infrastructure changes and investment. Additionally, prioritizing buses at traffic signals can disrupt the flow of other traffic, potentially causing congestion for other vehicles. This trade-off needs careful consideration when planning and implementing bus priority systems.

2.2 Related works

Recent advancements in machine learning have significantly enhanced the accuracy of bus arrival time predictions. This section reviews three key studies that utilize Linear Regression Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM) networks to predict bus arrival times.

2.2.1 Linear Regression

Taparia and Brady [2021] developed models to predict bus journey and arrival times using historical GPS data. Their research aimed to tackle urban traffic congestion by providing commuters with reliable travel time and arrival information, which in turn, encourages public transportation use. They focused on building models for both overall journey times and bus arrival times at stops.

Data Used

- Dataset: Dublin City Council provided a dataset that covered GPS data for buses in Dublin from November 6 to November 30, 2012. The dataset was vast, with about 35 million rows, each containing 15 features such as timestamp, line ID, direction, journey pattern ID, production time frame, vehicle journey ID, operator, longitude, latitude, delay, block ID, vehicle ID, stop ID, and bus stop status.
- Preprocessing: They derived new features like 'day of the week' and 'hour' from the timestamps and calculated 'distance from the city center' using the Haversine formula to improve the model's accuracy.

They concentrated on three bus routes with the most data, particularly Route 46, which had 1989 journey records. For model training, they used 80% of these records and reserved the remaining 20% for testing.

Results

The Linear Regression model's performance was evaluated using Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error (RMSE). The results were:

MAE: 7.6130 minutes

MAPE: 7.2406%

RMSE: 10.0342 minutes

2.2.2 Artificial Neural Networks

In a 2019 study by Lam, Ng, and Leong, the researchers developed a system to predict bus arrival times in Macao, a densely populated city. They leveraged real-time data from sensors installed on buses and at bus stops. The researchers explored three models: Simple Moving Average (SMA), Artificial Neural Network (ANN), and a hybrid model combining both SMA and ANN.C. T. Lam and Leong [2019]

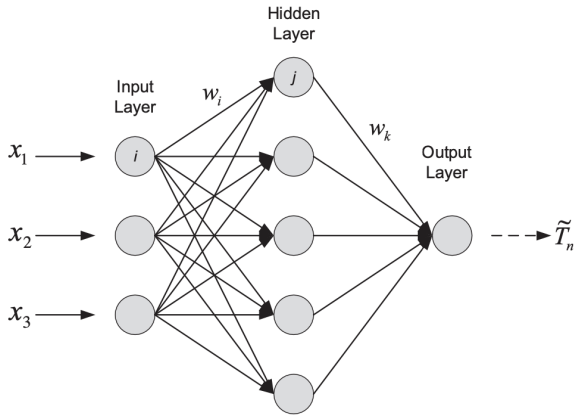

Figure 2: Structure of the proposed artificial neural networks.
C. T. Lam and Leong [2019]

Figure 2 shows the Model Structure:

Input Layer, the ANN model used the two most recent link times (the time it takes for a bus to travel between two stops) and weather conditions, which significantly impact traffic. Hidden Layer, the model featured one hidden layer with five neurons, using the Sigmoid function for activation. Output Layer, this layer predicted the bus arrival time at the next stop.

The ANN was trained using the back-propagation algorithm. The hybrid model combined the simple averaging of SMA with the predictive capabilities of the ANN.

Data Used:

Real-time Bus Locations, Data was sourced from the Macao Transport Bureau (DSAT) website, providing accurate, real-time bus locations. Weather Conditions, Real-time weather data was mapped to numerical values and used as inputs to the model. Link Times Database, historical link times between bus stops were stored and used for training and validation.

The study found that the hybrid model achieved a mean absolute percentage error (MAPE) of 17%, with a mean absolute error (MAE) and root mean square error (RMSE) both under one minute, outperforming the individual models.

2.2.3 Long Short-Term Memory (LSTM)

In another study, Zeng and colleagues focused on predicting bus arrival times using LSTM networks. Their research targeted the public bus system in Chongqing, China, using a dataset comprising driving records from 30 buses collected over one year.Z. Lingqiu and Lidong [2019]

Model Structure:

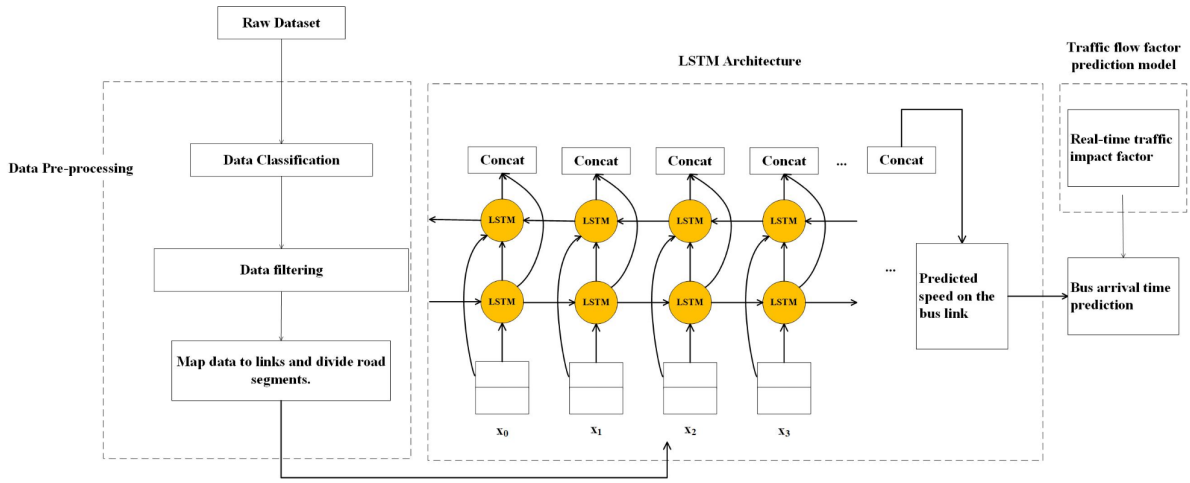

Figure 3: Overall System Architecture.
Z. Lingqiu and Lidong [2019]

- **Input Layer:** The model included features such as bus ID, speed, timestamp, longitude, and latitude, processed into segments representing specific road sections.
- **Hidden Layers:** The LSTM model consisted of:
  - Input dimensions: 20 (transformed into a 20x20 matrix).
  - Hidden layers: 100 LSTM cells with a tanh activation function, arranged in two layers.
  - Dropout layers to prevent overfitting.
- **Output Layer:** This layer predicted the bus arrival time for the next 20 links.

The model was trained using the Adam optimizer, with mean squared error (MSE) as the loss function.

Data Used:

- **Raw Dataset:** The dataset included 46 variables such as mechanical properties, operational status, and location information.
- **Traffic Impact Factors:** Two real-time traffic factors were defined: the link factor (current traffic state of the road segment) and the bus stop factor (dwell time at bus stops).
- **Training and Test Sets:** The data was divided into peak and off-peak times. The training data covered April to December 2016, with 5,000 records for testing and the remainder for training.

The LSTM-based method showed a significant improvement in prediction accuracy, achieving lower RMSE and MAPE values during both peak and off-peak hours compared to traditional methods. Including real-time traffic data further enhanced the model's performance.

2.3 Gaps in Existing Research

But still, there is an ample amount that can be worked on in the following areas for betterment and future studies. Identifying the gaps might end up increasing precision or may lead to more reliability regarding the prediction.

Complex Traffic Pattern

One issue is dealing with the non-linear complex relationships within the traffic data. Simple models, for example, Linear Regression, fail to capture the complexities. Even if some more advanced models, such as Artificial Neural Networks or Long Short-Term Memory networks, have shown to be promising, their application is not so popular. The great potential of such advanced models requires more research to realize how they may be used to tackle the nuances of urban traffic. Taparia and Brady [2021].

Scalability and Generalization

One more issue is that most research activity is generally carried out by concentrating on a particular route or a very small expanse of land. In doing so, these findings do not hold true for other cities or routes, to a large extent. For instance, a model performing well in one city can be an utter failure in another—all due to differences in traffic patterns and infrastructure. Future research should focus on the development of scalable and generalizable models across other urban settings. Testing a model in multiple cities and along routes is an effective way to learn if it is robust and adaptable.

Influence of External Factors

Although there has been some research on the influence of weather, a lot is yet to be known on how bus-arrival times are influenced by other external factors. Passenger load, traffic signal timings, and road conditions are some of the various factors that affect travel time but usually are not included in predictive models. How these factors together influence bus arrival times could be understood better and ways to include them in models devised to improve accuracy.

Potential of Hybrid and Ensemble Models

Various modeling approaches are hybrid or ensemble nature. However, their application is yet to be more developed when applied in bus arrival time prediction. Research is needed on the combination of techniques, such as adding the statistical techniques into the machine learning models to increase performance, where borrowing the best from both models is achieved. Amer Shalaby [2004].

Learning and Real-Time Adaptation

Most current models suffer from an inability to adapt to changes in traffic patterns over time. The models ought to learn and update in real-time through streaming data, thus providing adaptive predictions. Developing algorithms that can process and learn from streaming data efficiently makes the updating of new instance data accurately and updated with every instance added at all points. S. Basak and Dubey [2019]

The Last Concern: Standardized Evaluation Metrics

The last concern is about standardized metrics and validation methods in the current literature. The comparison of results and the inference of the most effective approach are also difficult because different studies provide results in different metrics. To establish normalized metrics of evaluation and validation protocols will be important areas for future research. This will ensure consistency and make comparison easier between previous work. Prakhar Balasubramanian [2015].

3 Methodology

This chapter explains how the objectives of our research were reached. Description of the data collection process in detail, along with the sources and features of the floating car data to be used in this study. This chapter also provides the data preprocessing procedures for cleaning and preparing the data for analysis, exploratory data analysis conducted to understand patterns of the dataset, and the feature engineering techniques that are used in improving the model performance. Finally, presenting the machine-learning models used in this paper: the Mean Model, Linear Regression, Multi-Layer Perceptron (MLP), and Long Short-Term Memory (LSTM) networks.

3.1 Data Collection

The foundation of this study rests on the Floating Car Data (FCD) collected from a bus network. FCD provides a wealth of information by tracking the movements of vehicles using GPS technology. This data includes precise details about the location, speed, and direction of buses at various points in time can be seen in Table 1. For this study, data was collected over 7 months (24949 runs) to ensure robustness and reliability in the findings. The used data is in DataFrames form.

Column Name	Description
run	A unique identifier for each bus run.
utcTime	The timestamp of when the data was recorded in UTC.
offset	An offset value
speed	The speed of the bus at the time of recording (in meters per second).
longitude	The longitude coordinate of the bus location.
latitude	The latitude coordinate of the bus location.
link	The identifier for the road segment or link the bus is currently on.
linkPos	The position of the bus on the current link, measured in meters from the start of the link.
route	The route number the bus is serving.
vehicle	A unique identifier for the bus vehicle.
trip	A unique identifier for the bus trip.
geometry	The geometric location of the bus represented as a POINT (in meters Easting and Northing)

Table 1: The available Floating Car Data from the bus

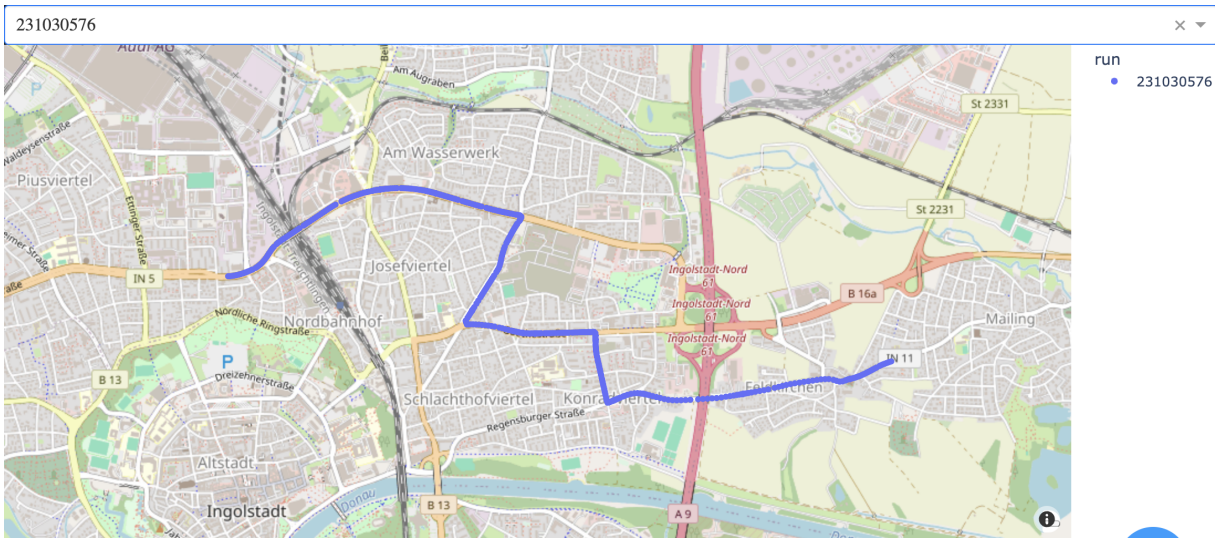

Figure 4: Example of a complete run points

An example of the used data after visualising one run can be seen in figure 4. A map for part of Ingolstadt shows most of the Bus line 70 after visualising one run of data points, from which it is easy to understand this line route.

3.2 Data Preprocessing

Figure 5: The traffic lights stop line considered in this study

Data preprocessing represents a very important aspect in the process of having a high quality, clean and relevant data for the study. In this study, we focus only on one traffic lights which can be seen in Figure 5. Therefore, it is better to keep only the data before this stop line. The steps followed during preprocessing are as below:

Removing Invalid Runs:

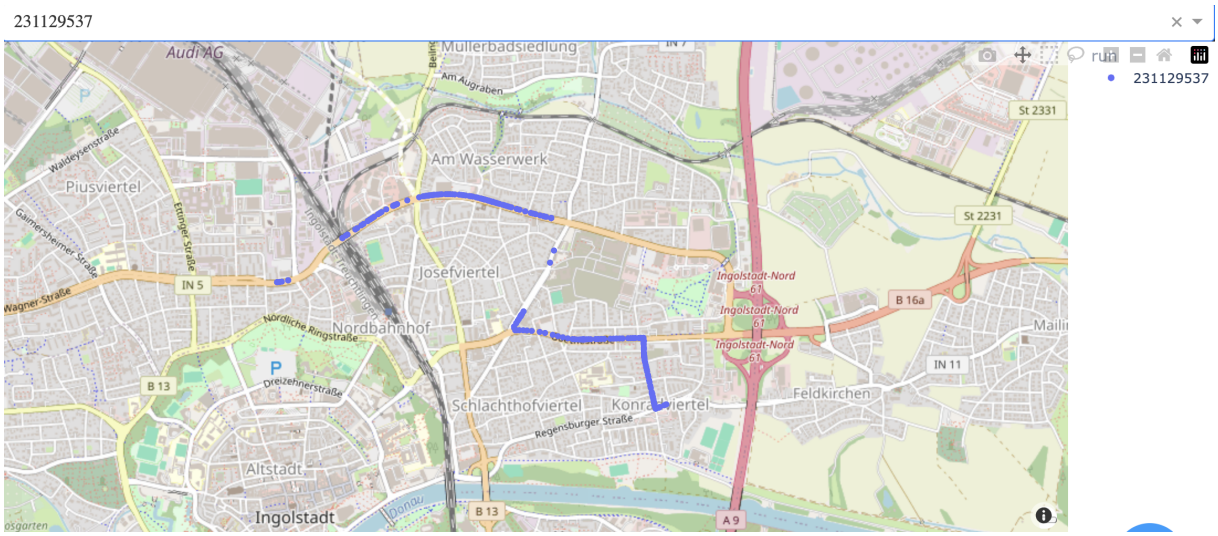

Figure 6: An Example of an invalid run

Figure 6 shows an example for an invalid run, in which the run misses some data points for a period of time. Entries from the data that did not follow expected patterns or with obvious errors, such as missing points or lack of sending data for an unacceptable distance or period of time, have been identified and removed. Another consideration is using runs in a specific direction and eliminating the ones in the opposite direction. The main aim of this step is to eliminate noise or inaccuracies from the dataset.

Considering congestion before traffic lights:

As we focus on one traffic lights, it is better to use data for suitable distance before this stop line. The considered data is only for 400 meters before the stop line. An issue arose that there may be a congestion before the traffic lights. The mean of the time taken for the runs to cross 50 meters before the traffic lights is 31.88 seconds. Predicting the arrival time of the bus to 50 meters before the traffic lights can simply overcome this issue. Therefore, the points which are 50-400 meters before the stop lines are considered.

Figure 7 shows the final data will be used for training and testing the model, each run includes only points 50-400 meters before the stop line, which is enough data to predict the EAT at the traffic lights.

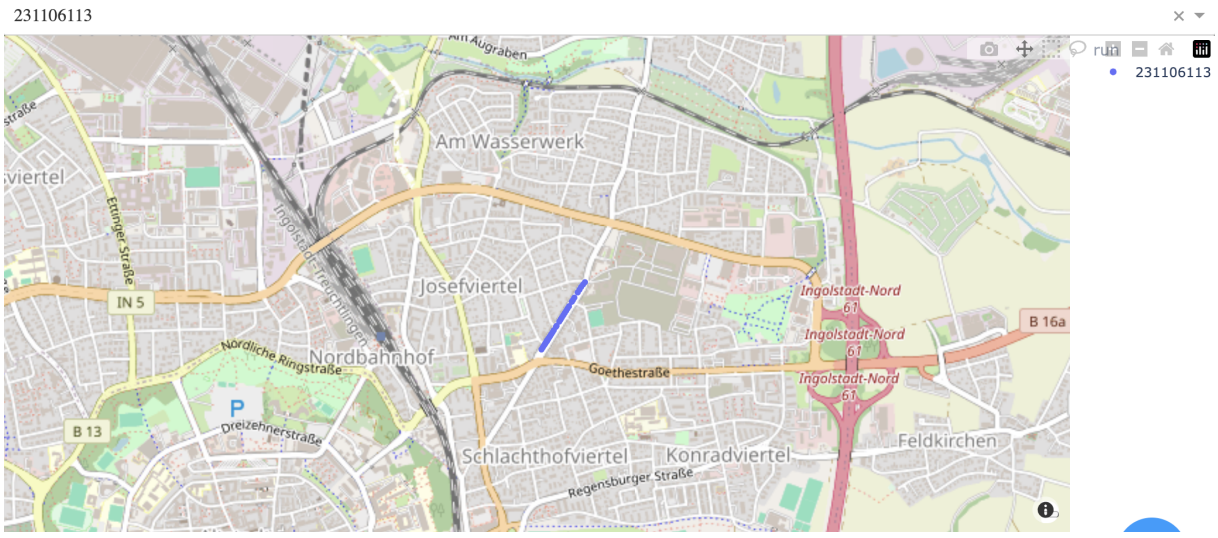

Figure 7: An example for a 400 meters run after considering the congestion

Remove Waiting Time at Bus Stops:

For special use cases such as evaluating models, a new set of data were created after removing the waiting (Dwell) time at bus stops to get the actual travel times between bus stops. This is very important, making sure that the prediction is from the movement of the bus and not how long it takes at each stop. Waiting times can introduce variability that has no relation to the traffic conditions or travel distances.

3.3 Exploratory Data Analysis (EDA)

All the analysis stated in this section was done on the points which are 400 to 50 meters before the traffic lights. After preprocessing, we have 11875 runs.

3.3.1 Points map with speed

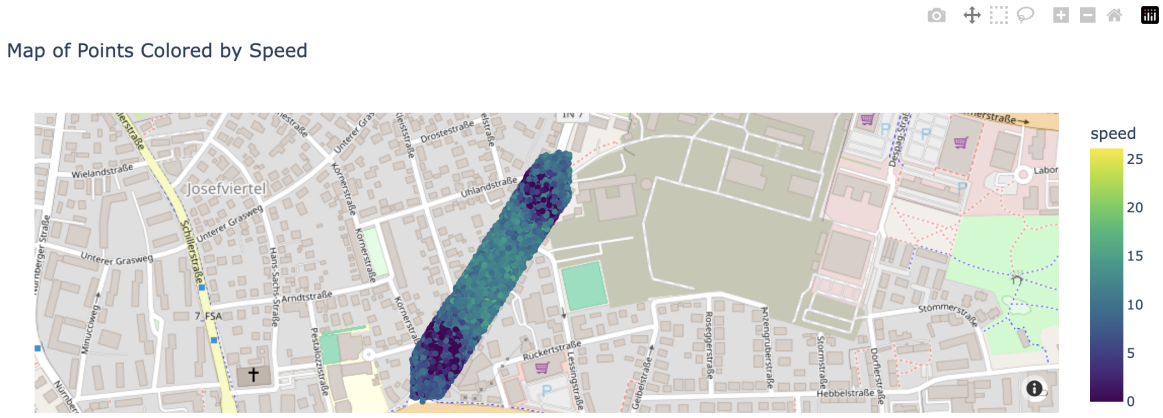

Figure 8: Map of Points colored by speed

Figure 8 shows visualising the data points on the map with an indication of the current speed in each points can help in identifying some patterns. In this figure, it is clear to recognise that the bus stops two times before the traffic lights, the bus stops two time because there are two Bus Stops on its way to the traffic lights. It can also be recognised that not all the runs stop at these Bus Stops, which indicates a variance in the taken period to reach the traffic lights. An important statistics shows that 41% of the busses stopped once, and 35% of the busses stopped twice.

3.3.2 Run Duration Analysis

- **Per Hour of the Day**

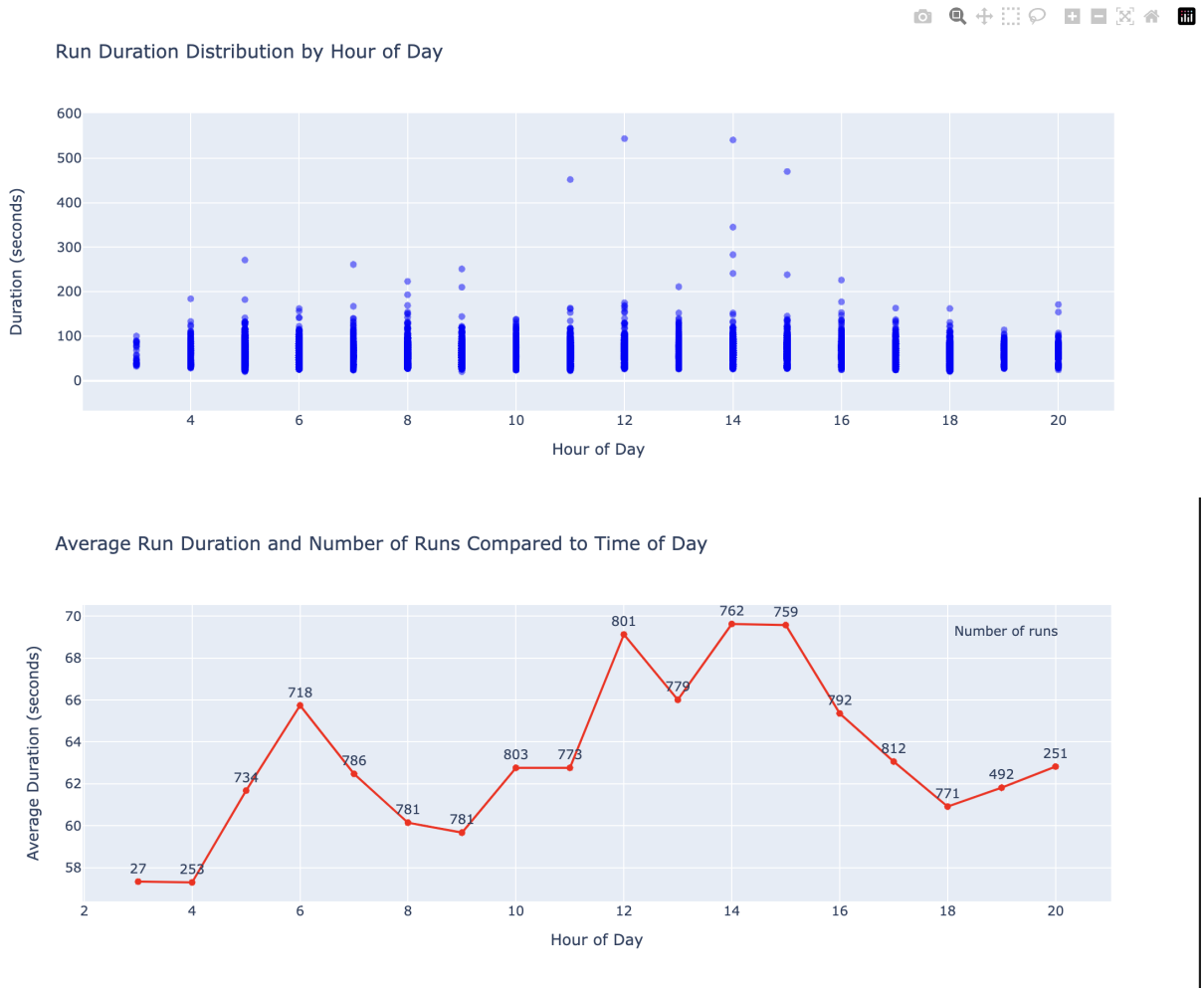

Figure 9: Run Duration distribution and average per hour of the day

Figure 9 shows two graphs, the first is run duration distribution by hour of day having hour of the day on the x-axis and duration in seconds on the y-axis. The second graph is average run duration in seconds on the y-axis and the hour of day on the x-axis, number of runs in each hour can be seen on the graph. A growing number of Runs can be seen in the early morning, and a decay in the late evening. The existence of the outliers is clear. Noticed higher average around 06:00 and 14:00, this is probably because of the rush hours.

- **Per Day of the Week**

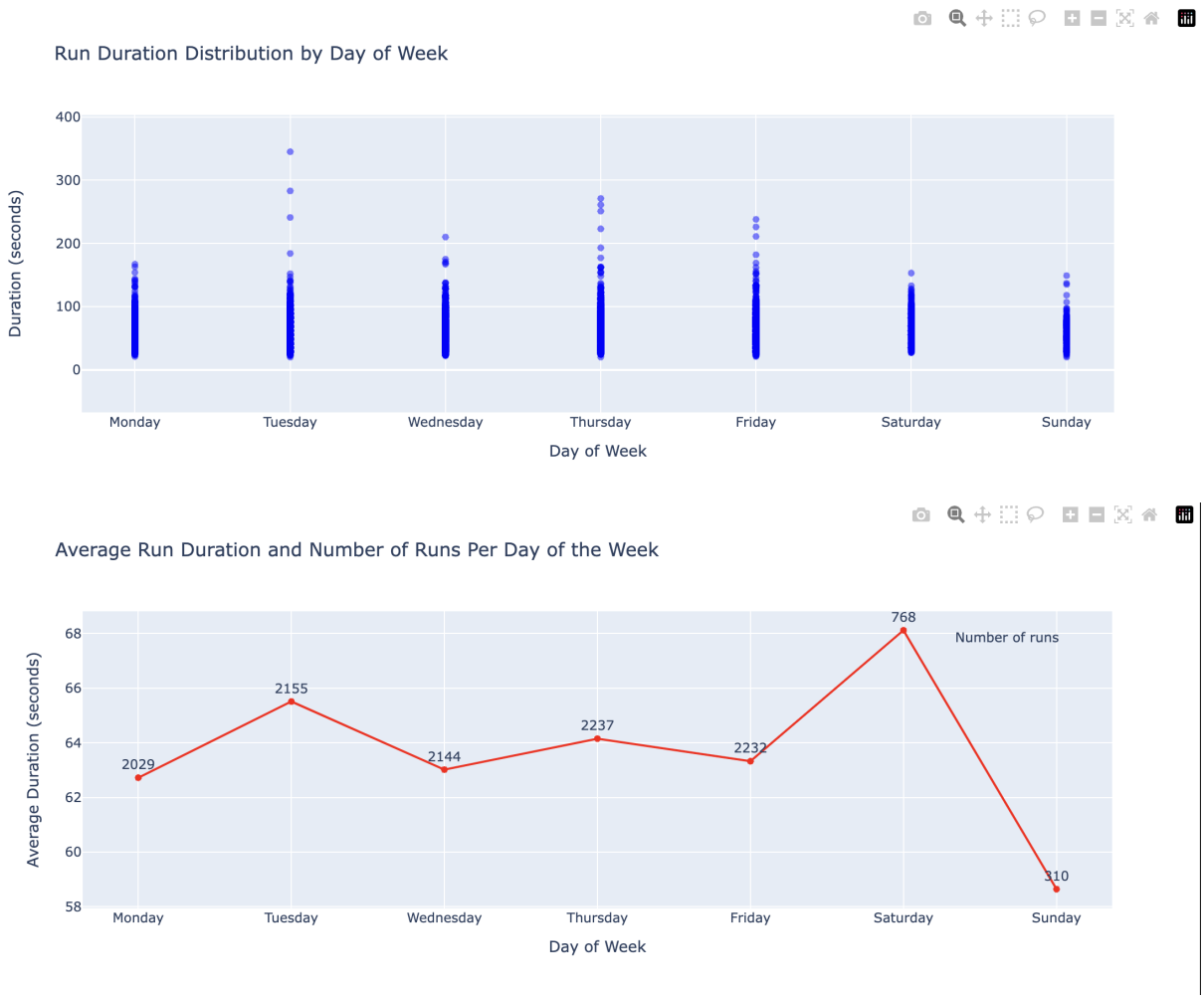

Figure 10: Run Duration distribution and average per day of the week

Figure 10 shows two graphs, the first is run duration distribution by day of week having day of the week on the x-axis and duration in seconds on the y-axis. The second graph shows average run duration in seconds on the y-axis and the day of week on the x-axis, number of runs on each day can be seen on the graph. Almost consistent Average during the weekdays, the highest average is noticed on Saturday as everyone goes out, and the lowest average on Sunday as people prefer to stay home. Consistent high number of Runs on weekdays as bus comes every 10 minutes, lower on Saturday as it comes every 30 minutes, and the lowest number of runs on Sunday as it comes every one hour.

- **Per Week of the Year**

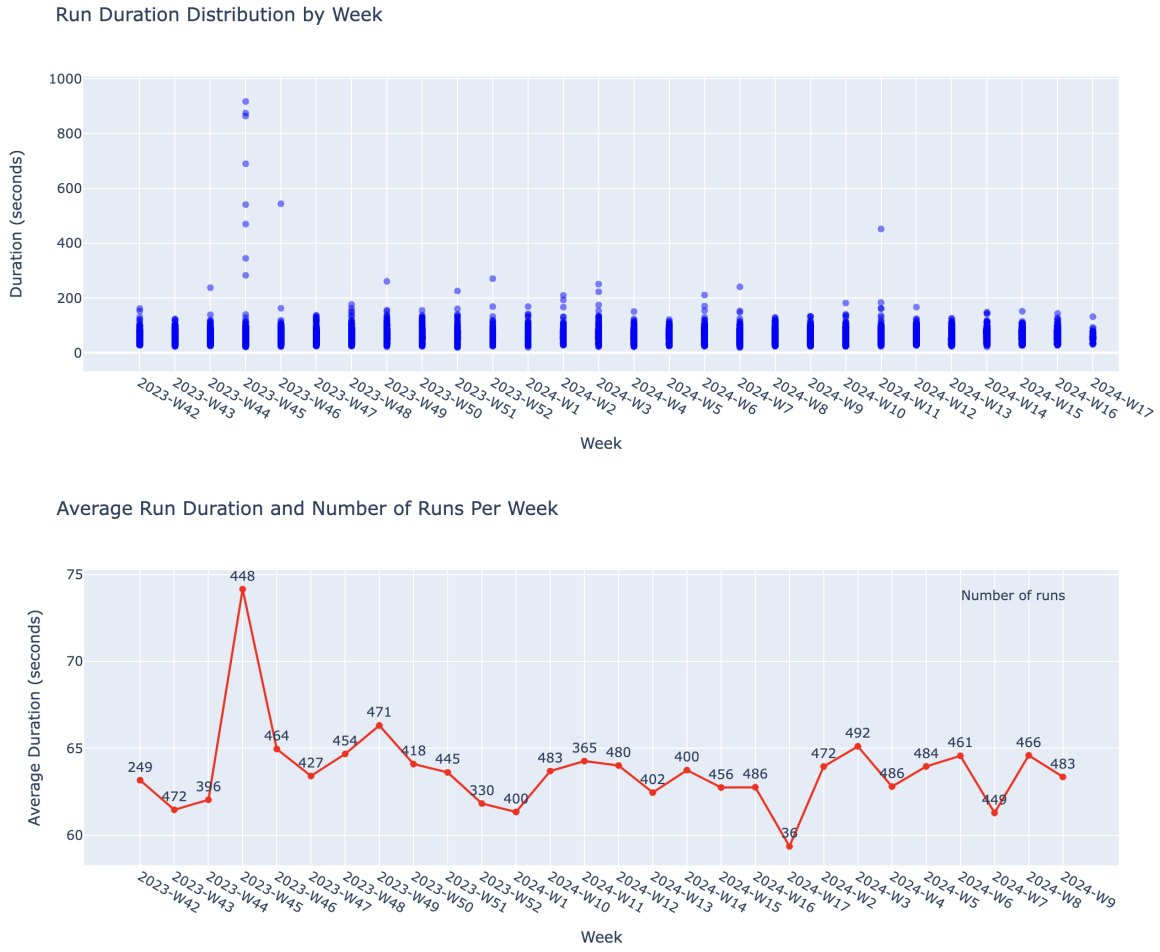

Figure 11: Run Duration distribution and average per week of the year

Figure 11 shows two graphs, the first is run duration distribution by week of the year having week of the year on the x-axis and duration in seconds on the y-axis. The second graph shows average run duration in seconds on the y-axis and the week of year on the x-axis, number of runs on each week can be seen on the graph. Consistency with fluctuations, there is a peak in run times average around week 45, but it smooths out after that. The number of runs stays almost steady week to week except weeks in which there was public holidays.The outliers are obvious here, they extremely exist in the week 45, which is affecting the average.

- **Per Month of the Year**

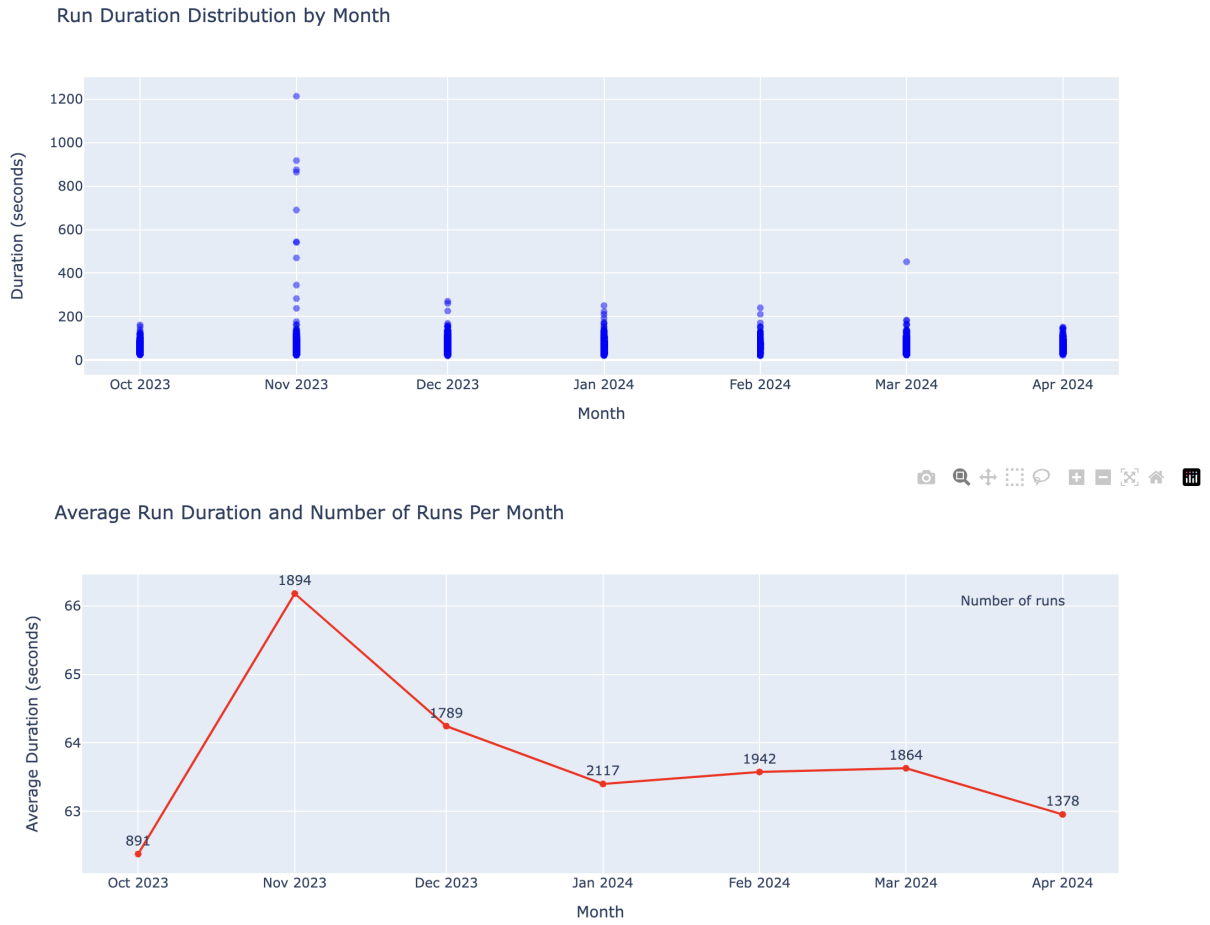

Figure 12: Run Duration distribution and average per month of the year

Figure 12 shows two graphs, the first is run duration distribution by month of the year having month of the year on the x-axis and duration in seconds on the y-axis. The second graph shows average run duration in seconds on the y-axis and the month of year on the x-axis, number of runs on each month can be seen on the graph. November has the highest average among the months, and it is mainly because the week 45, and the outliers can be seen here too, the number of runs was the highest in November as well. The rest of the months almost have similar values.

3.3.3 Descriptive Analysis

Statistic	Value
Count	11,875
Mean	63.91 seconds
Std Dev	28.09 seconds
Min	20.00 seconds
25 percent Q1	52.00 seconds
Median Q2	61.00 seconds
75 percent Q3	76.00 seconds
Max	1213.00 seconds
Variance	789.03
Range	1193.00 seconds

Table 2: Runs duration descriptive analysis

Table 2 shows the main descriptive analysis of the used data. Doing descriptive analysis helps in understanding the data in a better way, and can help in identifying the outliers.

- **Identifying the Outliers**

Remove or unconsider the outliers can help in removing very rare situation, especially when there is no explanation for this behaviour. Identifying the outliers can be done using Interquartile Range **IQR**. Dekking et al. [2005]

Given Values:

$$\text{Q1 (25th percentile)} : 52.00 seconds$$
$$\text{Q3 (75th percentile)} : 76.00 seconds$$

Calculate the Interquartile Range (IQR):

$$\text{IQR} = Q3 - Q1$$
$$\text{IQR} = 76 - 52$$
$$\text{IQR} = 24.00 \text{ seconds}$$

Calculate the Lower Bound:

$$\text{Lower Bound} = Q1 - 1.5 \times \text{IQR}$$
$$\text{Lower Bound} = 52.00 - 1.5 \times 24.00$$
$$\text{Lower Bound} = 16.00 \text{ seconds}$$

Calculate the Upper Bound:

$$\text{Upper Bound} = Q3 + 1.5 \times \text{IQR}$$
$$\text{Upper Bound} = 76.00 + 1.5 \times 24.00$$
$$\text{Upper Bound} = 112.00 \text{ seconds}$$

3.4 Feature Engineering

Feature engineering is the process of taking raw data and extracting features that can be put into a model. The creation of useful features may dramatically empower the performance of a model. In this study, the following features were engineered from the collected data:

- **Distance:**

Description: The distance between consecutive data points along the bus route.

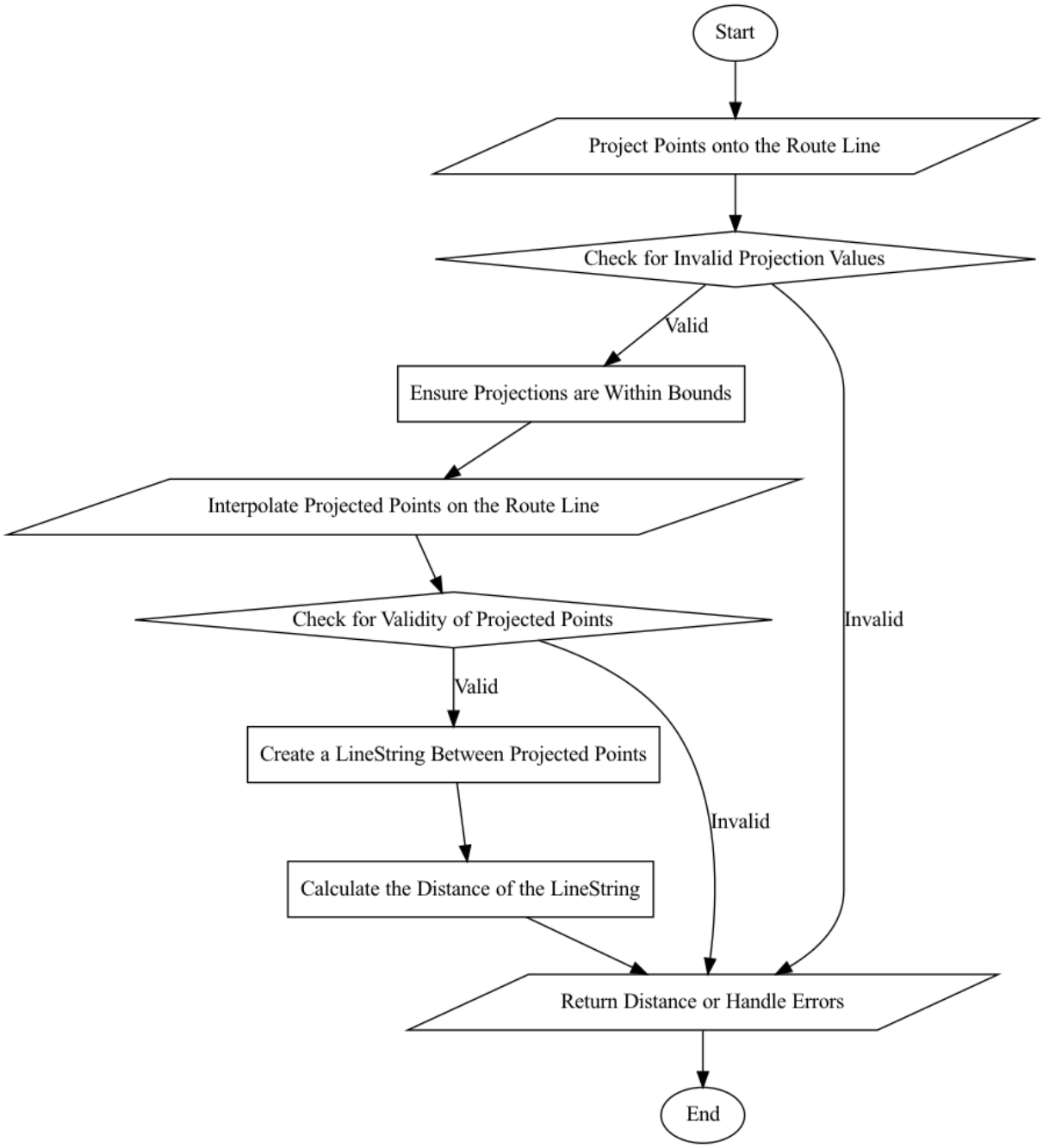

Figure 13: Distance calculation flowchart

Figure 13 shows the methodology for Distance calculation. This feature was calculated using a function, which calculates the real distance of the route between the two points. What has been ensured here is that the GPS point is projected onto a LineString representing the bus route. More precisely, the actual distance along the bus route will be calculated, as opposed to the straight-line distance which would be naive and unrealistic. In general, the function does the following:

Projection of each point over the bus route line. This way, the points used in calculating distance are directly on the bus route. Clamp projections within bounds of the route so that no invalid calculation is done. The projected points are interpolated to get the exact positions on the route. Two interpolated points create a segment between them and distance is calculated as the length of this segment. The function contains robust error handling to take care of any unexpected problems that could potentially occur during the calculation process in order to guarantee accuracy and reliability.

- **Time of Day:**

Description: at what particular hour of the day was the data point taken. Method: From each of the utcTime, I extracted the hour of the day. This is an essential feature because there is so much variation in traffic conditions at different hours—for example, rush hours in the morning and evening times compared to late-night hours.

- **Day of Week:**

Description: on which day of the week the data point was. Approach: Extracting the day of the week from the utcTime so as to determine if the data point was one where it was recorded on a weekday or during the weekend. Traffic flow and passenger load are typically varied on weekdays and weekends. Hence this variable adds predictive value to estimate bus arrival times.

- **Week of the Year:**

Description: In which week the point was collected. Method: Extracting the week from the utcTime to capture any patterns happening as the impact of holidays or semester break from schools and universities. This feature will not be used in training the model, as the model will be validated using randomly chosen complete weeks.

- **Month of the Year:**

Description: The month when data points were gotten. Method: The month was extracted from the utcTime to capture any seasonal variations in traffic patterns. Different months have different traffic conditions due to holidays, school sessions, and weather changes.

3.5 Used Models

3.5.1 Mean Model

The mean model just calculates the average taken time to reach the stop line from the same distance

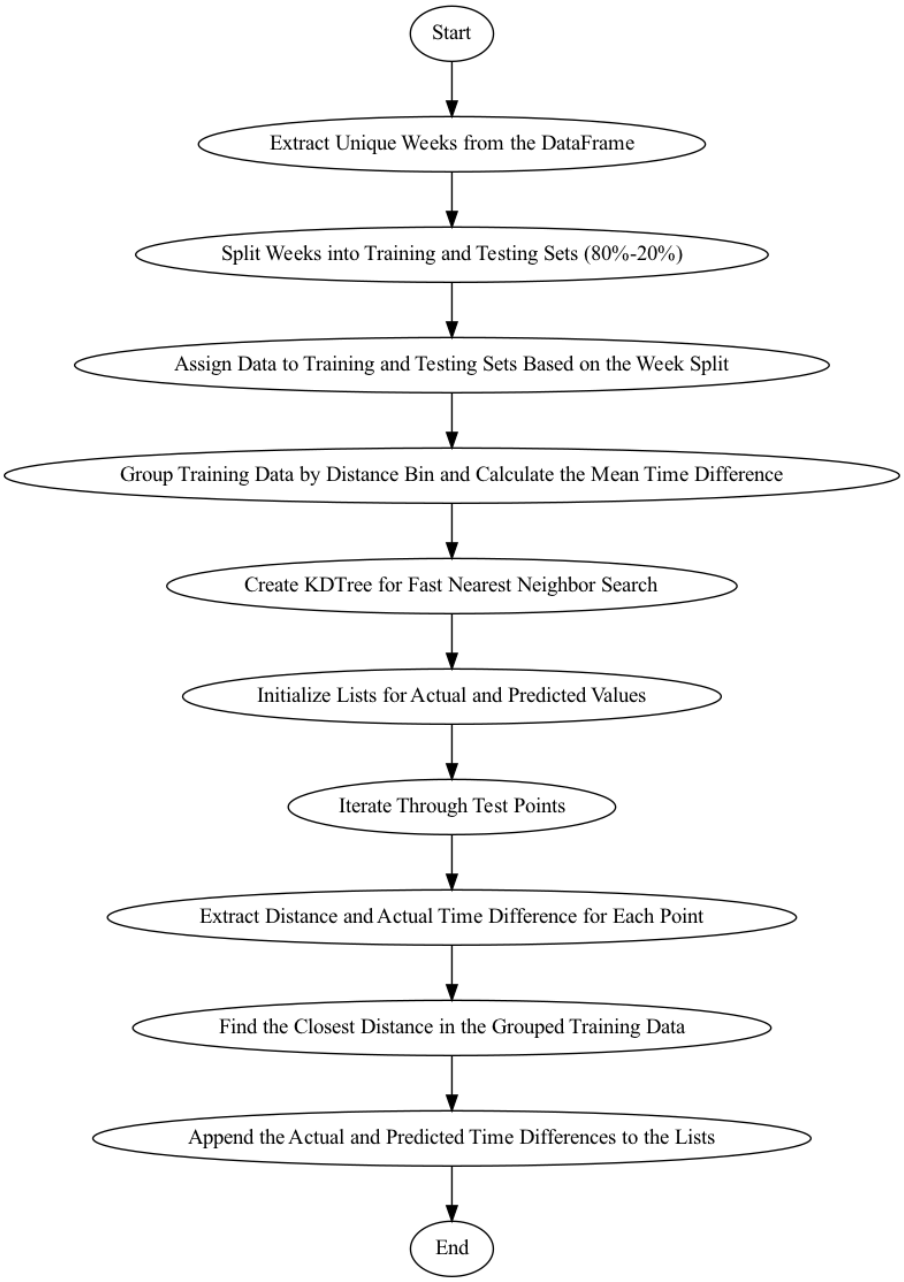

Figure 14: Mean Model Algorithm Flowchart

Figure 14 shows the used Algorithm for the mean model. For each point, the distance to the stop line, the week, and the time difference were added. Combined all monthly data into a GeoDataFrame. Randomly split the data by weeks, using 20% for testing. Group the training data by distance. Create a KDTree using the grouped training data for fast nearest-neighbor

search based on the distance. For each point in the test set, find the closest distance in the training data using the KDTree. Get the time difference based on the nearest distance. Compare the calculated and actual time differences to evaluate performance.

3.5.2 Linear Regression

Linear regression model assumes a straight-line relationship between the input features and the target variable. Therefore, this model is the simplest Machine Learning model. Implementing this model needs identifying the features and the target variable. Then normalising the features, to ensure that each feature had a mean of 0 and a standard deviation of 1. It is commonly used as a start in Machine Learning, as it is easy to be evaluated, in addition to the ability of evaluating the used features because of the short training and testing time.

3.5.3 Multi-Layer Perceptron (MLP)

Is a class of feedforward artificial neural network. It consists of at least three layers of nodes: an input layer, one or more hidden layers, and an output layer. In each node or neuron of one layer, there is a connection to every node in the next layer with some weight. This will allow for a network to learn complex patterns. The MLP uses activation functions, such as ReLU or sigmoid, that bring non-linearity into the system and allow it to handle non-linear relations within the data. Normally, backpropagation trains an MLP in an iterative manner of fine-tuning its weights so that the error between output prediction and desired output is minimized. It can model intricate data structures; therefore, it is commonly applied in practice to accomplish tasks of classification, regression, and forecasting. Jaiswal [2024]

C. T. Lam and Leong [2019] Found in practical that using simpler models achieves the same performance. They used Sigmoid activation function.

Practically, using simpler architecture achieved the same performance as the data is not very complex. After research and experiments, ReLU activation function is more suitable for Linear tasks. Batch size 64 and default learning rate were used.

- **Model Structure**

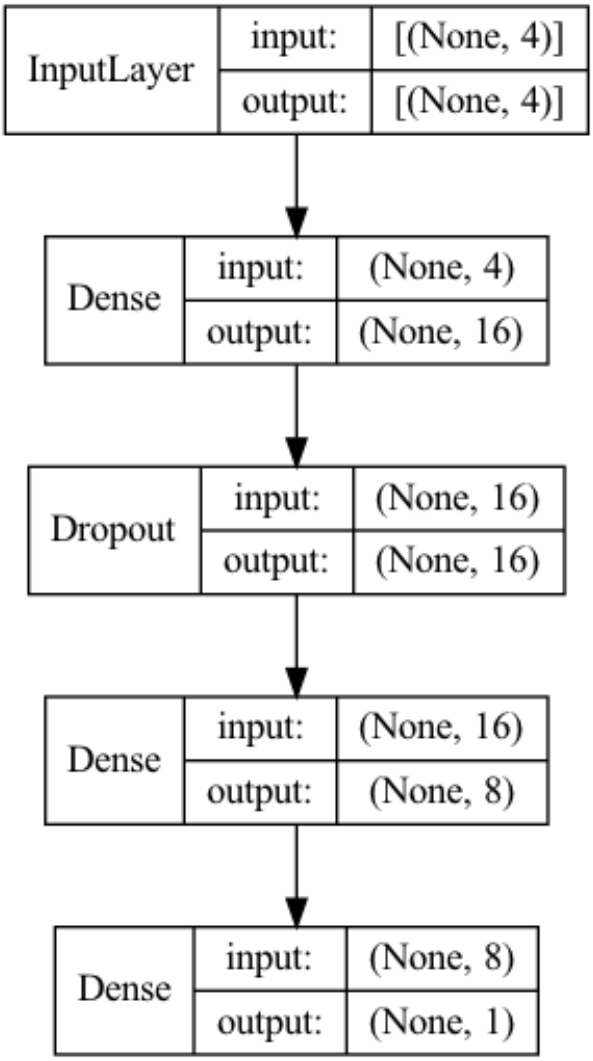

Figure 15: Used MLP model structure

Figure 15 shows the used model structure which includes 5 layers. Input Layer, gets the normalized features. Hidden Layers, two hidden layers of 16 and 8 neurons, respectively; both have a ReLU activation function. A Dropout layer was used to avoid overfitting by randomly setting 0.1 of the cells to 0. Output Layer, a single neuron with a linear activation function is used to predict the time difference.

- **Compilation of the Model**

The Adaptive Moment Estimation **Adam** optimizer was applied for training due to its effectiveness and adaptive learning rate capabilities. The implemented loss function is Mean Squared Error (MSE), measuring the mean of the squares of differences between predicted time and actual time. An extra metric integrated for model evaluation was Mean Absolute Error (MAE).

3.5.4 Long Short-Term Memory (LSTM)

Recurrent neural network, which includes Long Short-Term Memory, is designed to deal with sequential data in capturing long-term dependencies. It is done majorly well for time series prediction, e.g., predicting bus arrival times given historical travel data. Olah [2015]

Z. Lingqiu and Lidong [2019] used 100 cells for their LSTM, with tanh activation function in addition to a dense layer with same number of cells. The main reason for that number of cells is how complex is their data, as they used 20 input features.

The data used in this study is much less complicated with only 4 input features. Therefore, a simpler LSTM model with fewer cells is used. After doing many experiments with various configuration, the best performance was achieved with Batch size 64 and default learning rate with the following structure.

- **Model Structure**

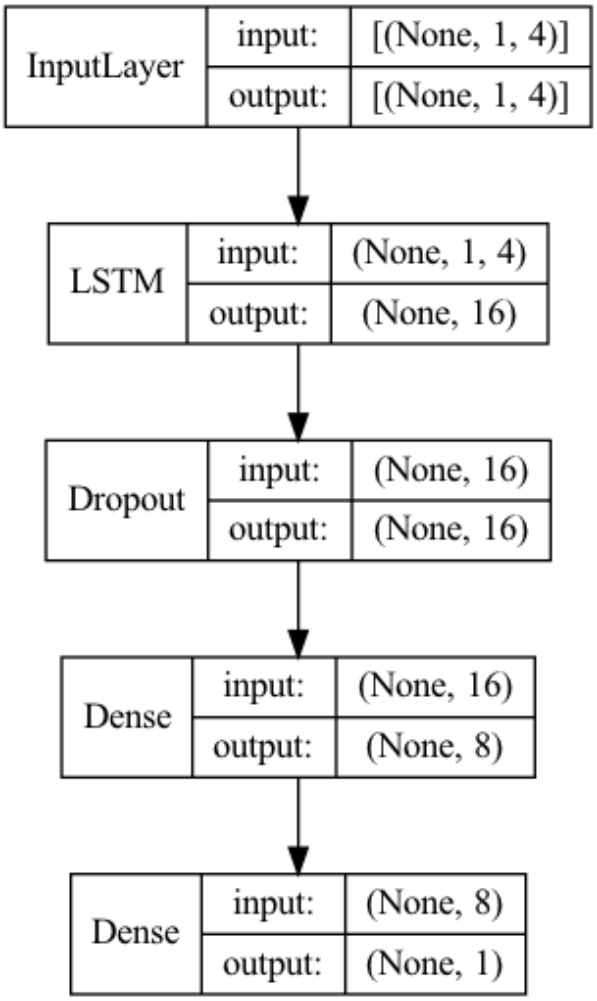

Figure 16: Used LSTM model structure

Figure 16 shows the used model structure, One layer is being applied in LSTM with 16 cells, which uses tanh activation. This kind of LSTM layer is pretty good at capturing temporal dependencies and patterns in data that is sequential in nature. A dropout layer used to avoid

overfitting by randomly set 0.1 of the cells to 0. One fully connected (Dense) layer with 16 neurons and 'relu' activation was added after the LSTM layer. The high-level of the hierarchy helps learn complex representations of the data. The final layer included in this model is a Dense layer, with a single neuron to give the output prediction of bus arrival time.

- **Model Compilation**

The model was compiled with Adaptive Moment Estimation **Adam** optimizer. Adam is known to be very efficient and effective in training deep learning models. The used loss function was the mean squared error (MSE), that is, the average of the square difference between predicted values and actuals. It also kept track of MAE as an additional performance metric.

4 Implementation

This chapter will go into implementation details on this work. Discussing the software and tools applied for creating the data instances and for the model training/evaluation with Pandas, GeoPandas, and TensorFlow/Keras. In this chapter will further present the training and validation of the models; splitting of the dataset into training and testing datasets, and efforts taken for the optimization of model performance. Additionally, the evaluation measures -through which the effectiveness and credibility of the used models were gauged- will be described.

4.1 Software and Tools

This subsection lists the software and tools for implementing, training, and testing the models.

- Python: The base programming language in which models are developed and all data analysis is run.
- Pandas: Library for data manipulation used to work with and process datasets.
- GeoPandas: An extension of Pandas that enables working with geospatial data.
- NumPy: Package for numerical data manipulation with large, multi-dimensional matrices and arrays.
- Shapely: A library for manipulation and analysis of geometric objects.
- Scikit-Learn:

  'StandardScaler': Used for feature normalization.

  'train_test_split': It is a data splitting function for partitioning the dataset into training and testing datasets.

  'mean_squared_error', 'r2_score': Model evaluation metrics.

  'LinearRegression': For building the linear regression model.

- TensorFlow/Keras:

  Sequential, Dense, Dropout, LSTM: Deep learning framework for the creation, training, and evaluation of neural network models.

- Datetime: For date and time manipulation.

4.2 Model Training and Validation

This subsection deals with training and validation. A split was done for the dataset based on the weeks, 0.80 of the weeks were used for training and 0.20 for testing. so that the training data and the validation data of the models would generalize well on new data. Here are the steps:

1. Data Preparation: Steps included feature and label extraction from the raw data, handling missing values in each case, and normalization to better model performance.
2. Data Splitting: Data was split into training and validation, ideally with an 80-20 split. This helped in training the model on a major share of the data while it would be validated by the remaining share in a cross-checking step for overfitting.
3. Training of Models:

   Linear Regression: With LinearRegression from Scikit-Learn.

   Neural Networks: Implemented both in MLP and LSTM using TensorFlow/Keras.
4. Hyperparameter Tuning: Hyperparameters such as learning rate, batch size, and the number of layers/neurons were tuned to optimize model performance.

4.3 Model Evaluation

We evaluate the models after training. The next sections detail the metrics and the methodologies of evaluation:

1. Evaluation Metrics: Among the most common evaluation metrics for the models are:

   Mean Absolute Error (MAE): Measures the average magnitude of the errors in the predictions but does not account for their direction.

   Mean Squared Error (MSE): Measures the mean difference between predicted and actual values squared.

   Root Mean Squared Error (RMSE): The square root of MSE gives an error metric in the same units as the target variable.

   R-squared (R^2): Measures the part of a dependent variable variation captured by the regression model.
2. Validation and Testing: The models were validated on a validation set and further exposed to an unseen test set in order to ascertain their robustness and generalization capability.
3. Comparison of Models: The performance of the compared models relied on evaluation metrics and their respective advantages and limitations analyzed in the various scenarios.
4. Visualization: Performance metrics and model predictions were visualized using plots to provide a clear understanding of the model behavior and accuracy.

5 Results and Discussion

In this chapter, the results of the used machine learning models will be presented, and provide a comparison of the performance of the Mean Model, Linear Regression, MLP, and LSTM models in predicting bus arrival times, taking into account the dwell times and without. The results were analyzed by way of the Root Mean Squared Error and R-squared value metrics. Additionally, a discussion of findings imply in terms of having a feature contribute to the strongest model is presented, and describe the strengths and limitations of each model.

5.1 Mean Model Results

The mean model is often used as a baseline to understand the performance improvement of more complex models. Including outliers in the mean model can highlight the robustness of more sophisticated models against anomalies in the data.

5.1.1 With Dwell Time

The mean model scored 19.47 seconds RMSE, and 0.43 R^2.

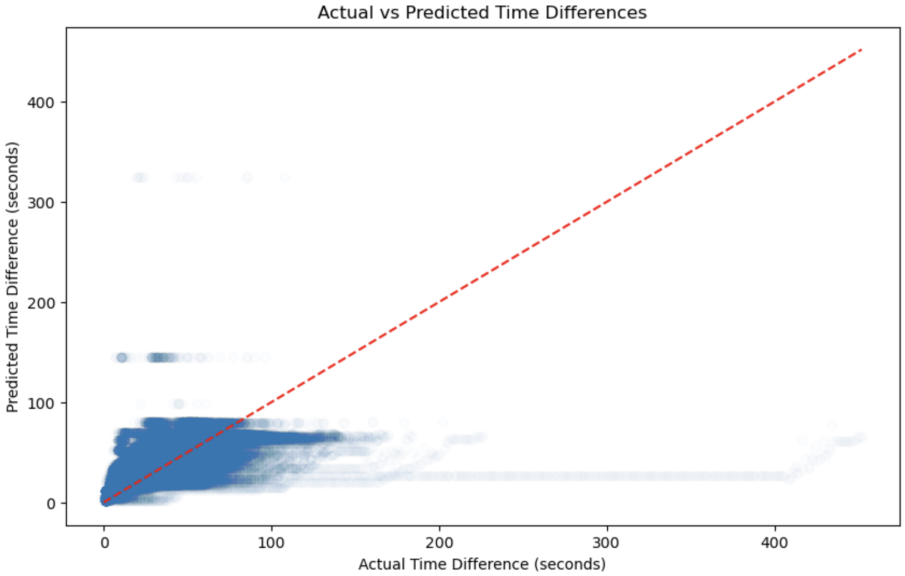

Figure 17: Mean Model Actual vs Predicted Travel Time With Dwell Time

Figure 17 shows the actual time value on the x-axis, and the predicted time value on the y-axis. The mean prediction error is 19.47 seconds; therefore, the accuracy is poor and can be improved. The R^2 score for the model is 0.43, a value representing that the model explains 43% of the variance in the actual time differences. As this number is rather low, it also means that 57% of the variance is not accounted for.

Most points collect in the lower left, which means that it most commonly predicts shorter times of arrival. It deviates notably from the diagonal to the ideal line, especially when the actual time taken is larger, indicating that the model considerably fails for those values in prediction.

5.1.2 Without Dwell Time

A score with 5.55 seconds RMSE and 0.86 R^2 was achieved.

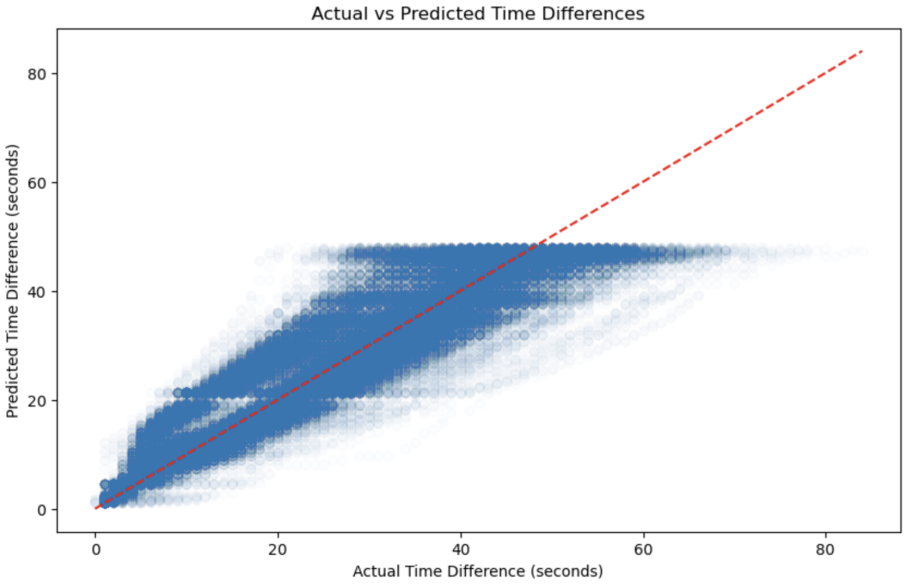

Figure 18: Mean Model Actual vs Predicted Travel Time Without Dwell Time

Figure 18 shows the actual time value on the x-axis, and the predicted time value on the y-axis. This result shows the impact of the bus stops, the model can not explain when the bus is going to stop, and it leads to a gap of the explained variance. After removing these stops, the bus became much more capable of explaining the output variance with the same features.

5.2 Linear Regression Results

5.2.1 With Dwell Time

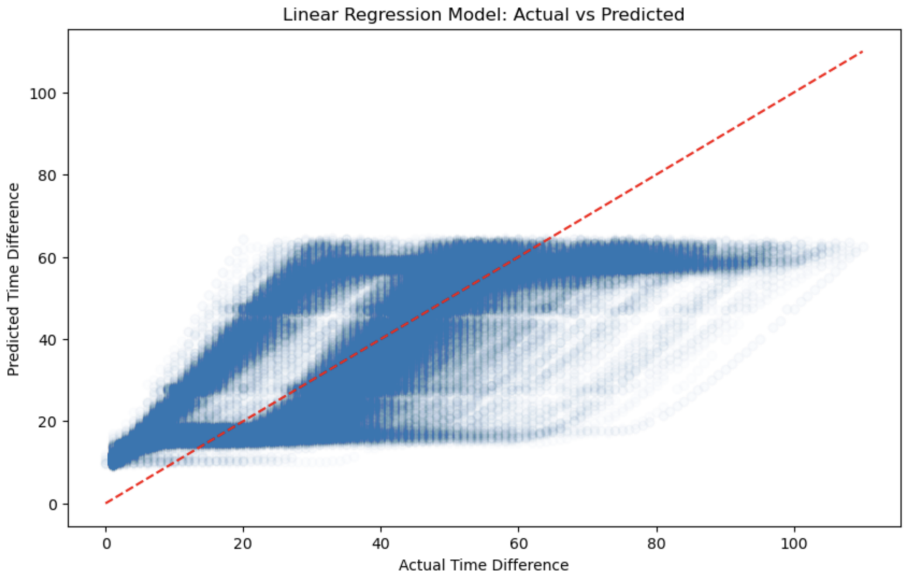

Figure 19: Linear Model Actual vs Predicted Travel Time With Dwell Time

Figure 19 shows the actual time value on the x-axis, and the predicted time value on the y-axis. A pattern can be seen between the predicted and the actual values. The linear model was able to achieve 12.4 seconds RMSE and 0.67 R^2. The model shows a real step forward in explaining the output variance with 67% of it, it recognise more patters. and has a less mean error of 12.4.

5.2.2 Without Dwell Time

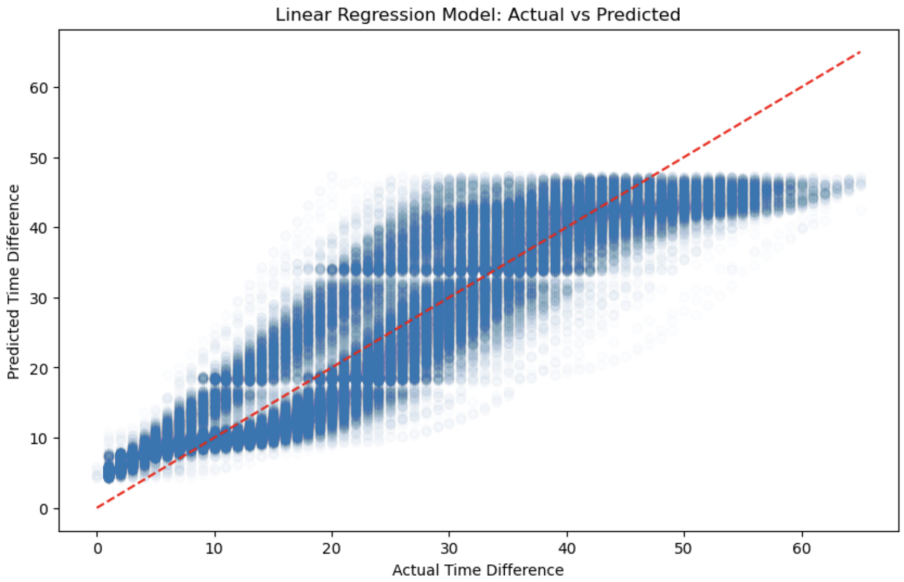

Figure 20: Linear Model Actual vs Predicted Travel Time Without Dwell Time

Similar to Mean Model, the Linear Model achieved 5.5 seconds RMSE and 0.86 R^2. Figure 20 shows the actual time value on the x-axis, and the predicted time value on the y-axis excluding the dwell time. The used features did not help in achieving better performance than the mean model after removing the Dwell time. But better performance is achieved after removing the dwell time.

5.2.3 Features Evaluation

As mentioned before, the Linear Regression Model can be used as a start in Machine Learning, especially to evaluate the models. For Features evaluation, the Variance Inflation model and the coefficients values were used.

Feature	Distance	day of week	time of day	month of year
VIF	1.000085	1.008825	1.000230	1.008753
Coefficients	17.99	0.317	0.319	-0.229

Table 3: VIF and Coefficient value of Features

Table 3 shows Variance Inflation factor and the coefficients value for the used features.

Variance Inflation Factor VIF

The Variance Inflation Factor VIF is used to indicate how much multicollinearity exists within a group of regression variables. It generally arises when there is a high correlation among independent variables in a regression model. An increased VIF greater than 5-10 shows a serious case of multicollinearity.

In this model, the independent variables, together with the constant term (intercept), record VIF values of around 1, so they display no multicollinearity among the independent variables. The VIF values for distance, day_of_week, time_of_day, and month_of_year are all very close to 1: 1.000085, 1.008825, 1.000230, and 1.008753, respectively. This means that every feature contributes independently to the model and is not overly correlated with any other features, which ensures stable and reliable coefficient estimates.

Coefficients

The coefficients in a linear regression model measure the amount of change in the dependent variable as a result of a unit change in the independent variable, while all other variables are constant.

The intercept is when all the independent variables are equal to zero, 34.1814. The result for the coefficient of distance is 17.9000, where all other factors remain fixed; for each increase in a unit of distance, the dependent variable increases by 17.90 units. The day_of_week coefficient, which gives a positive but extremely small effect on the dependent variable for every one additional day of the week, is 0.3172. Furthermore, the dependent variable changes slightly for every one additional unit in the time of day, at 0.3198. On the other hand, month_of_year has a coefficient of -0.2294, hence, holding all variables constant, increasing the value of month_of_year by one unit will bring a reduction of one unit to the dependent variable.

5.3 MLP Model Results

5.3.1 With Dwell Time

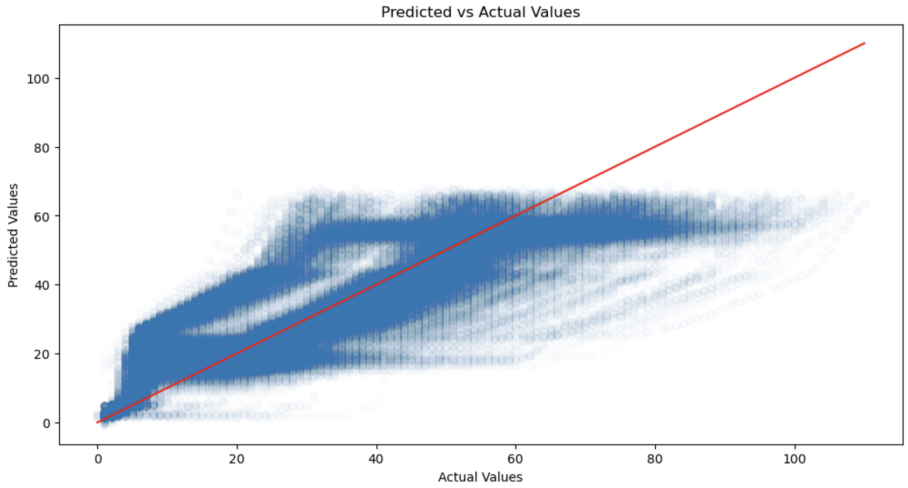

Figure 21: MLP Model Actual vs Predicted Travel Time With Dwell Time

Figure 21 shows the actual time value on the x-axis, and the predicted time value on the y-axis for MLP model including the dwell time. The MLP model was able to achieve 11.8 seconds RMSE and 0.70 R^2. The disribution shows a pattern which is most likely because of the stopping at the bus stops. The points are relatively closer to the ideal line, which can be explained with R-squared 0.70, so the model now is explaining more variance on the target variable.

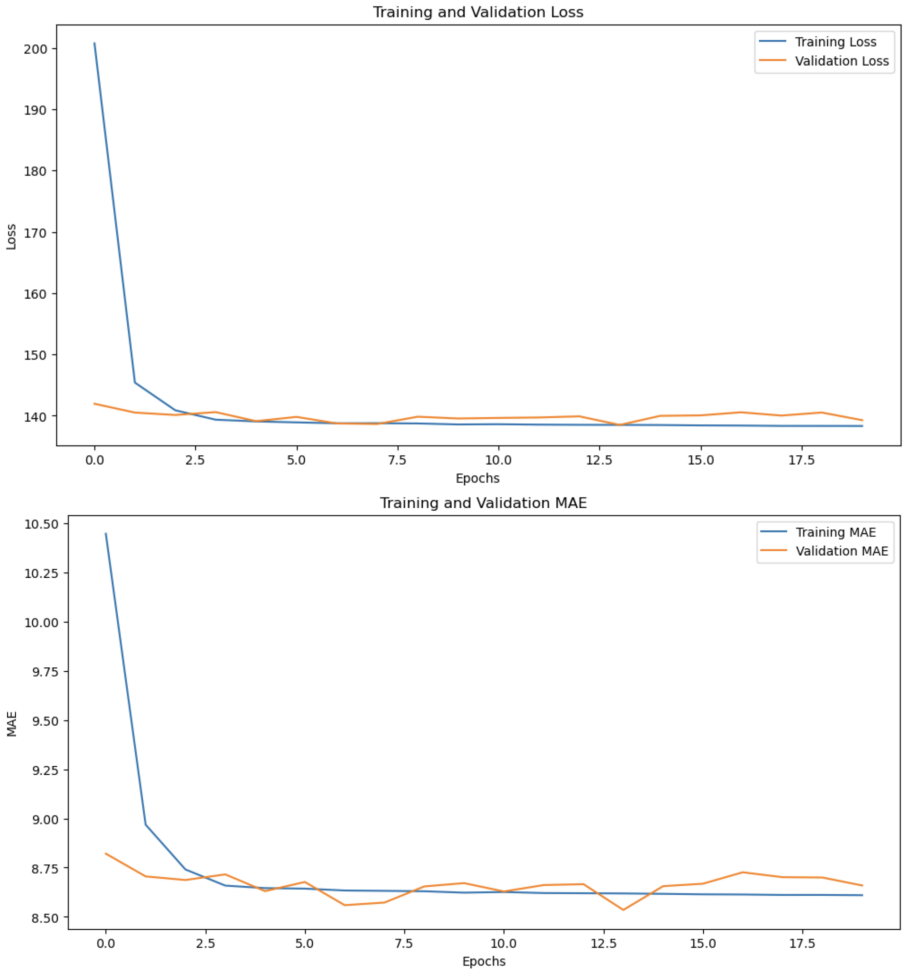

Figure 22: MLP training and validation loss with Dwell time

Figure 22 includes two graphs, the first shows epoch number on x-axis and training/validation loss on the y-axis, the second graph shows epoch number on x-axis and training/validation Mean Absolute Error on the y-axis. The training and validation plots show fast convergence, the model is learning fast which is most likely because of similarity in data by time. The training and validation curves are close over time, therefore, a model overfitting is unlikely. On the other hand, the model explains 70% of the target variance.

5.3.2 Without Dwell Time

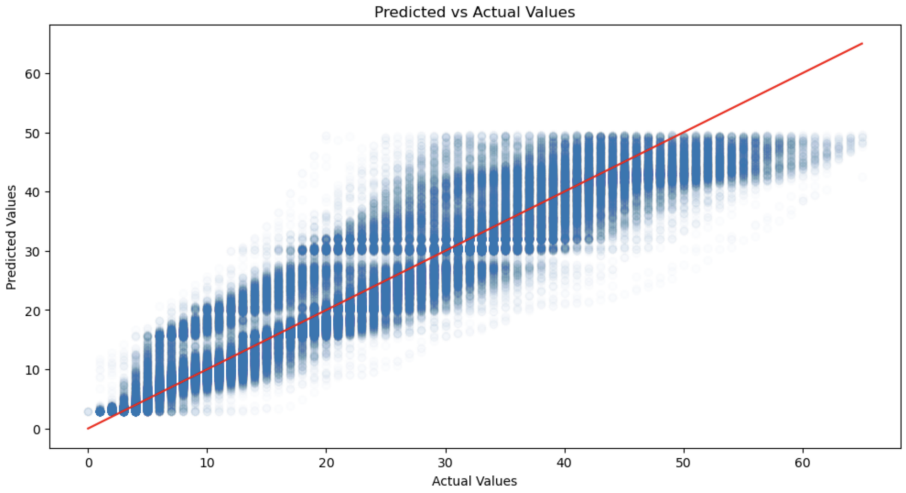

Figure 23: MLP Model Actual vs Predicted Travel Time Without Dwell Time

Figure 23 shows the actual time value on the x-axis, and the predicted time value on the y-axis for MLP model excluding the dwell time. In this case of excluding the Dwell time, the model achieved 5.2 seconds RMSE and 0.86 R^2. Closer distribution to the ideal line can be recognized.

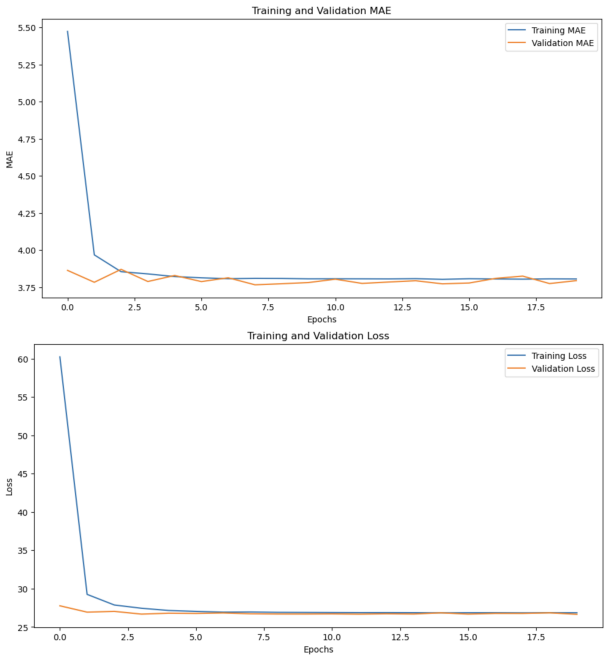

Figure 24: MLP training and validation loss without Dwell time

Figure 24 includes two graphs, the first shows epoch number on x-axis and training/validation loss on the y-axis, the second graph shows epoch number on x-axis and training/validation Mean Absolute Error on the y-axis. Again, quick convergence of the model occurred. the model is unlikely to overfit because of the close curves, and it shows a high ability for generalization.

5.4 LSTM Model Results

5.4.1 With Dwell Time

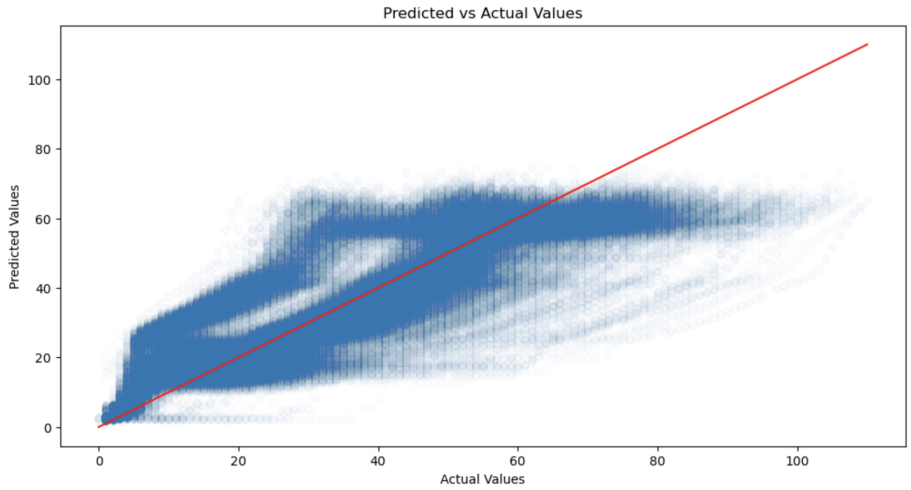

Figure 25: LSTM Model Actual vs Predicted Travel Time With Dwell Time

Figure 25 shows the actual time value on the x-axis, and the predicted time value on the y-axis for Long Short-Term Memory model including the dwell time. LSTM scored 11.7 RMSE and 0.71 R^2. The numbers shows a little improvement, although the Distribution is almost similar.

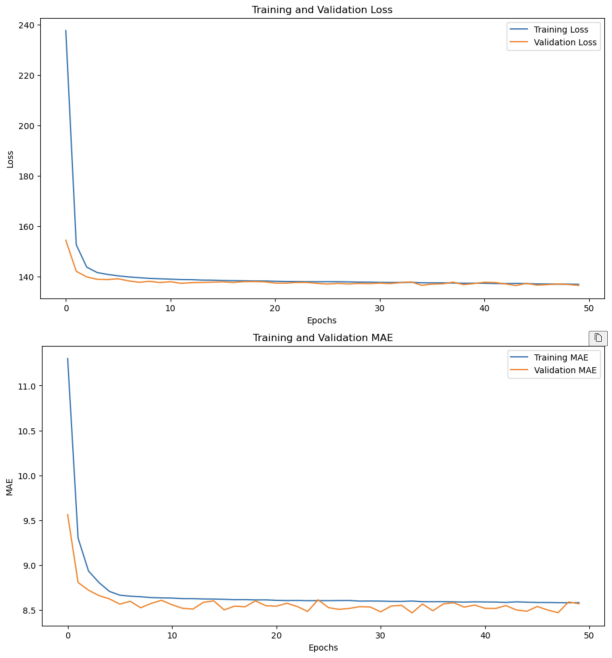

Figure 26: LSTM training and validation loss with Dwell time

Figure 26 includes two graphs, the first shows epoch number on x-axis and training/validation loss on the y-axis, the second graph shows epoch number on x-axis and training/validation Mean Absolute Error on the y-axis. Similar to the MLP model, fast convergence was noticed, and low probability of overfitting.

5.4.2 Without Dwell Time

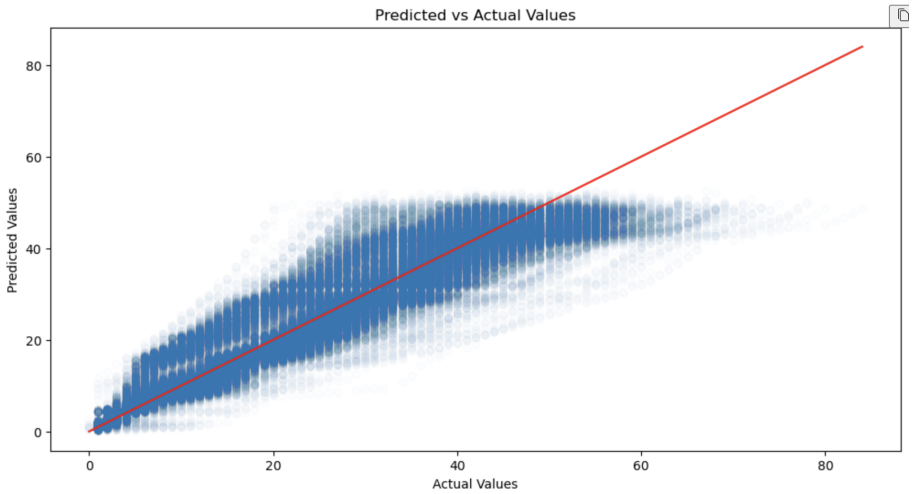

Figure 27: LSTM Model Actual vs Predicted Travel Time Without Dwell Time

Figure 27 shows the actual time value on the x-axis, and the predicted time value on the y-axis for Long Short-Term Memory model excluding the dwell time. After removing the Dwell time, the model achieved 5.5 seconds RMSE and 0.86 R^2. Closer distribution to the ideal line can be recognized. But similar performance to the other models using the same data is noticed.

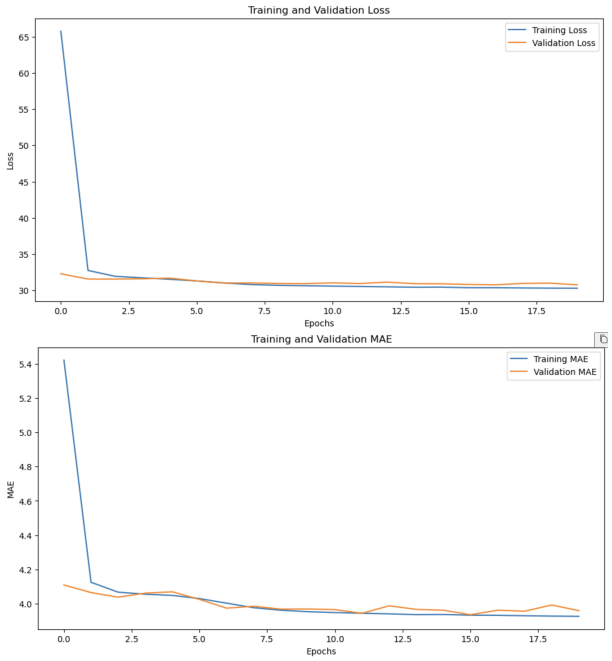

Figure 28: LSTM training and validation loss without Dwell time

Figure 28 includes two graphs, the first shows epoch number on x-axis and training/validation loss on the y-axis, the second graph shows epoch number on x-axis and training/validation Mean Absolute Error on the y-axis. Even faster convergence, Although closer values for training and validation were noticed.

5.5 Comparison of Models

5.5.1 With Dwell Time

Model	RMSE	R-squared
Mean	19.47	0.43
Linear Regression	12.4	0.67
MLP	11.8	0.70
LSTM	11.7	0.71

Table 4: Model comparison with Dwell Time by RMSE and R-squared

RMSE shows the mean average of the deviation in the output from the actual value. The Mean Model (which is used as a baseline and an example of a traditional method) performed poorly, as on average it deviates 19.47 seconds from the actual value. Introducing Machine learning improved the performance significantly, as the Linear model achieved 12.4 seconds which is 7 seconds less, an indication of capturing relationship between the features and target variable. Applying neural Networks in MLP Model allowed the model to capture non-linearity in the features, and it was clear in the score of 11.8 seconds. LSTM showed it is ability of preserving the long-term dependencies by slightly improving the performance to score 11.7 seconds.

R-squared shows how much the model can explain the variance in the dependent target variable by using the independent features. As the Mean model does not learn the data, but instead just taking the average no matter the current conditions, therefore, a poor performance by having the ability to explain only 43% of the output variance. Again, Machine learning enhanced this factor in the Linear Regression Model as it creates relationship between the different conditions and the resulting output, by explaining 67% of the variance, which is 24% more than the Mean Model. MLP model achieved a better performance by capturing the non-linear behaviour and explained 70% of the variance. LSTM did not introduce big further enhancement and scored 71%.

5.5.2 Without Dwell Time

Model	RMSE	R-squared
Mean	5.55	0.86
Linear Regression	5.5	0.86
MLP	5.2	0.86
LSTM	5.5	0.86

Table 5: Model comparison without Dwell Time by RMSE and R-squared

Without Dwell time, we consider the bus did not stop at all until reaching the traffic lights, which removes the issue of explaining when the bus is going to stop and when it is not going to stop. The performance of the Models does not show differences as they score almost 5.5 seconds RMSE and explain 86% of the output variance, which indicate that the features did not provide any useful data to the model to perform better than the Mean Model.

6 Conclusion and Future Work

The last chapter concludes the main findings of our study and discusses the contributions that this thesis has made to the field of bus arrival time prediction. I then offer a discussion of my contribution, point out research limitations, and suggest avenues for future work, including the extension to include more than one traffic light, additional variables of the external environment, and real-time adaptive models. This chapter concludes the thesis by pointing out the potential impacts that my research will have on the efficiency and reliability of public transport systems.

6.1 Summary of Findings

This paper has presented an investigation into predicting bus estimated arrival time at traffic light stop lines using various machine learning techniques. Major findings are:

Linear Regression, Multi-Layer Perceptron (MLP), and Long Short-Term Memory (LSTM) models applied to the data provided had substantial improvement over the traditional Mean model in predictive accuracy. The LSTM model slightly performed better than the MLP model, which means it is very important for the model to relate different features effectively with consideration for time relationships.

Model performances varied drastically through the inclusion or exclusion of dwell times, which are periods when buses stop at bus stops. Models had much better predictions when dwell times were excluded; therefore, the prediction of stops as well as their duration was a big challenge with the current set of features. The feature importance analysis showed that distance to the stop line, time of the day, day of the week, and month of the year are of high predictive value. The variance inflation factor (VIF) analysis confirmed minimal multicollinearity among these features, so model coefficients are reliable. Within these, distance was found to contribute most to the impact on predictions; the other temporal features played a smaller but still important role.

In the comparison of dwell times, the best result was with the LSTM model: a Root Mean Squared Error of 11.7 seconds and an R-squared value of 0.71. Notwithstanding, the model did not perform exceedingly well compared to the MLP model; therefore, both models do similarly better. All models, however, were performing more or less the same when the dwell times were excluded and resulted in an RMSE of around 5.5 seconds, with a corresponding R-squared value of 0.86. This further emphasizes the difficulty in integrating dwell times when predicting bus arrival times.

6.2 Contributions

This work has made several important contributions. Finally, the main contributions are the development and testing of Machine Learning predictive models—namely Linear Regression, MLP, and LSTM—where they have significant improvement over the previous mean-based calculations by many notches.

It was made using several methods in data preprocessing, the key to improvement in model performance. Preprocessing techniques involved the deletion of invalid runs, caution on dwell times with respect to analysis, narrowing the influence factor of traffic lights to data captured 50-400 meters down the street, and effects due to congestion. All the preceding measures ensured

that data fed into the models was of high quality and relevant toward enhancing the accuracy of predictions.

Feature engineering was done to create meaningful features from raw data, significantly boosting model performance. Some of the features pertaining to distance, time of the day, day of the week, and month of the year were useful in capturing the underlying patterns and periodicities in the bus arrival times. An evaluation of such features would indicate individual contributions to the model output and highlight the strong impact of distance on prediction accuracies.

A comparative analysis of different kinds of machine learning models was therefore carried out in varied conditions using them. It also provided a crucial understanding of their strengths and weaknesses, allowing a clear view of the way the models perform in forecasting bus arrival times. Such a detailed comparison is an instrumental source for future research and practical applications to allow optimal selection of models, which might be apt for different scenarios.

6.3 Limitations

Although this study showed some promises, it is not without flaws. One important limitation is the fact that the prediction of the arrival times is made for only one traffic light stop line; this may hence limit the generalization of the results to other places or even to greater traffic systems. In the future, the existing studies could add more stop lines of multiple traffic lights and extend the area more to improve the model's robustness and applicability.

Another limitation of the data is the scope of it. Data collection was done over the seven months in the same urban area. This may imply that not all possible traffic conditions and their variation were captured in both time and geographical space, reducing the robustness of the model. To use an example, there is a failure to properly take seasonal variations into account in the traffic pattern. Therefore, the model can run differently in each period throughout the year.

The models were not able to consider the fact that bus arrival times would be importantly affected by a number of external factors, including but not limited to weather, passenger load, and road incidents. These are sources of variation that the model does not explicitly use. Thus, they will be sensitive to mis-specifications under some conditions when predicting the future. The additional factors proposed can add more detail to the prediction framework.

All models relied on historical data and did not update in real-time. This drawback suggests that the models may be less effective in the drastically changing scenarios of traffic because of the real-time data required to make accurate predictions. A step change in the improvement of the accuracy and reliability of bus arrival time prediction would be achieved with models that are able to learn and adapt in real-time.

6.4 Future Work

There are a few ways in which the accuracy of prediction for bus arrival times can be improved by future research. Extending the scope of the study to one-year data on at least, multiple traffic lights and diversified urban areas will help in verifying the generalization and robustness of models. A wider dataset would capture a wider range of traffic conditions and seasonal variability, which gives a more representative basis for the training and testing of the developed models.

The modeling could be further enriched by including other characteristics, such as weather conditions, passenger loading, and incidents on the road. Notably, these factors can greatly affect bus arrival times, and modeling these impacts is likely to bring about an increase in the levels of accuracy of any prediction. Data integration on these external factors in real-time would make the models adaptive to the current situation.

Learn in real-time to develop models from streaming data. Thus, the models developed are much more accurate in prediction capabilities, making them much more applicable within the scope of dynamic and fast-changing traffic scenarios. Real-time learning therefore means that the models have a chance of being continually updated and further honed as new data comes in.

Hybrid and ensemble models warrant much attention in the further improvement of prediction accuracy. The best features of different machine learning techniques combined into one model will be able to leverage their individual strengths into a more resilient predictive framework. In fact, research in this direction could possibly lead to very sophisticated models surpassing individual techniques.

In these ways, the development of standardized evaluation metrics and validation protocols is fundamental in pushing the field ahead. It would serve the purpose of comparing across different studies over a common platform, whereby model performance can be measured. This standardization should reveal the best ways to drive further improvement in bus arrival time prediction.

Addressing those areas in future research could build a step further from the foundation that this study has laid, fostering more efficiency and reliability of public transportation systems by improving bus arrival time predictions.

Literature references

Jinhua Zhao Harilaos Koutsopoulos Aidan O'Sullivan, Francisco Pereira. Uncertainty in bus arrival time predictions: Treating heteroscedasticity with a metamodel approach. *IEEE Transactions on Intelligent Transportation Systems*, 99:1–11, 2016. URL https://mobility.mit.edu/publications/2016/osullivan-uncertainty-bus-arrival-time-predictions-treating-heteroscedasticity. Accessed: 19.07.2024.

Ali Farhan Amer Shalaby. Prediction model of bus arrival and departure times using avl and apc data. *Journal of Public Transportation*, 7(1):41–61, 2004. URL https://digitalcommons.usf.edu/jpt/vol7/iss1/3. Accessed: 20.07.2024.

B. Ng C. T. Lam and S. H. Leong. Prediction of bus arrival time using real-time on-line bus locations, 2019. URL https://ieeexplore.ieee.org/document/8947251. Accessed: 10.07.2024.

Frederik Michel Dekking, Cornelis Kraaikamp, Hen Paul Lopuhaä, and Ludolf Erwin Meester. *A Modern Introduction to Probability and Statistics*. Springer Texts in Statistics. Springer London, London, 2005. URL https://link.springer.com/book/10.1007/1-84628-168-7. Accessed: 16.06.2024.

Sejal Jaiswal. Multilayer perceptrons in machine learning, 2024. URL https://www.datacamp.com/tutorial/multilayer-perceptrons-in-machine-learning. Accessed: 2024-07-09.

Christopher Olah. Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/, 2015. Accessed: 2024-07-09.

Transportation Policy. Public transit, settlement patterns, and urban form. 2023. URL https://academic.oup.com/edited-volume/28314/chapter/215035204. Accessed: 18.07.2024.

K. Ramachandra Rao Prakhar Balasubramanian. An adaptive long-term bus arrival time prediction model with cyclic variations. *Journal of Public Transportation*, 18:1–18, 2015. URL https://digitalcommons.usf.edu/jpt/vol18/iss1/6. Accessed: 11.07.2024.

S. Sengupta S. Basak, F. Sun and A. Dubey. Real-time bus arrival prediction: A deep learning approach for enhanced urban mobility. *IEEE Transactions on Intelligent Transportation Systems*, 20:3283–3293, 2019. URL https://ar5iv.org/abs/2303.15495. Accessed: 20.07.2024.

Harriet R. Smith. Transit signal priority (tsp). May 2005. URL https://web.archive.org/web/20060923120521/http://www.fta.dot.gov/documents/TSPHandbook10-20-05.pdf. Accessed: 10.07.2024.

A. Taparia and M. Brady. Bus journey and arrival time prediction based on archived avl/gps data using machine learning. 2021. URL https://ieeexplore.ieee.org/document/9529328. Accessed: 11.07.2024.

et al. Wu. A review of research on public transport priority based on citespace. 2018. URL https://www.sciencedirect.com/science/article/pii/S235214651830005X. Accessed: 18.07.2024.

Hongzhi Yu Zijia Zhong Jinfei Yang Jinhua Zhao Yongpei Huang, Xiang Zhou. Bus arrival time prediction and reliability analysis: An experimental comparison of functional data analysis and bayesian support vector regression. *Applied Soft Computing*, 111:107663, 2021. URL https://www.sciencedirect.com/science/article/abs/pii/S1568494621005846. Accessed: 19.07.2024.

H. Qingwen Y. Lei L. Fengxi Z. Lingqiu, H. Guangyan and C. Lidong. A lstm based bus arrival time prediction method. *IEEE*, January 2019. URL https://ieeexplore.ieee.org/document/9060238. Accessed: 11.07.2024.