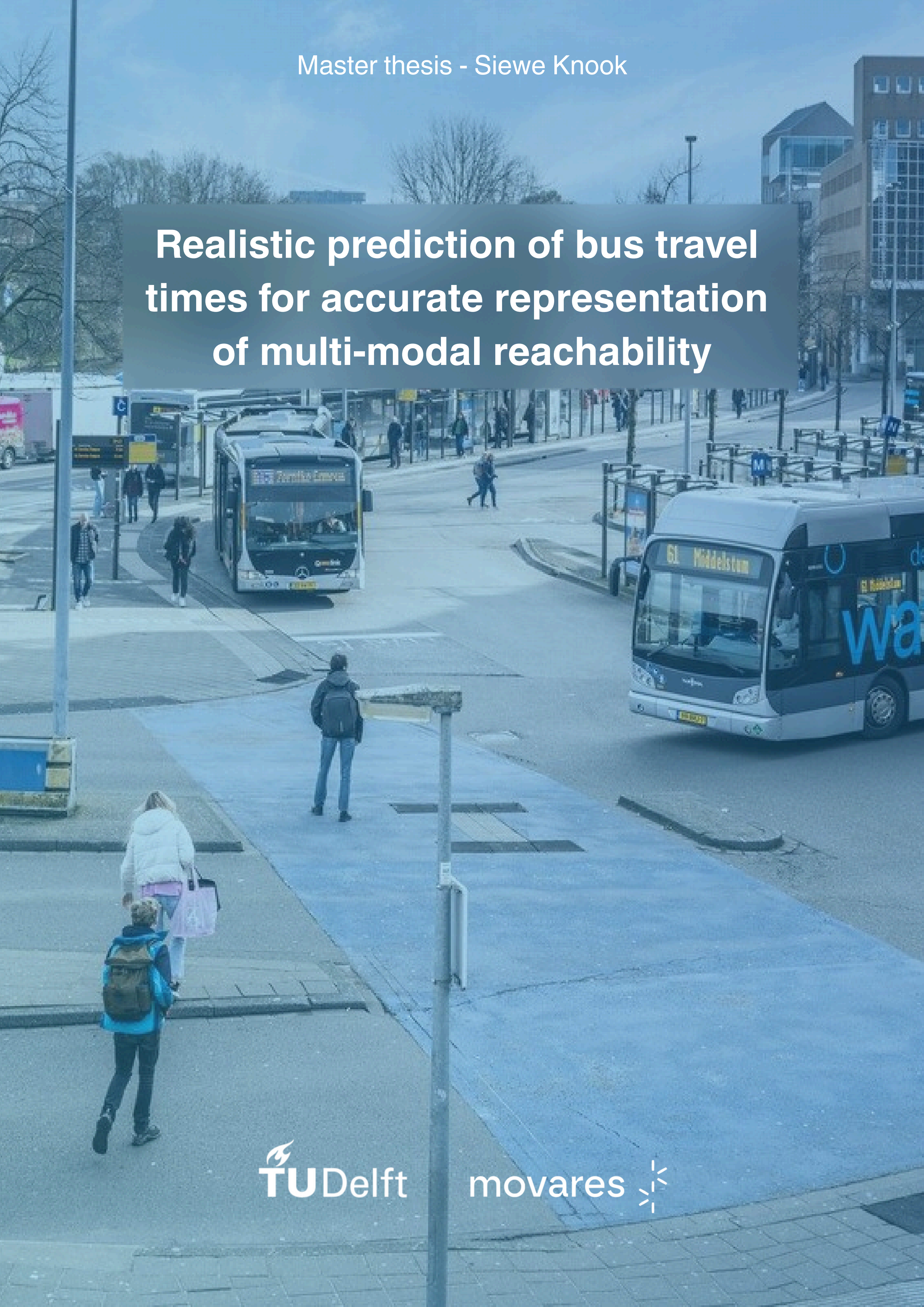
Master thesis - Siewe Knook

Realistic prediction of bus travel times for accurate representation of multi-modal reachability



TUDelft movares

Realistic prediction of bus travel times for accurate representation of multi-modal reachability

By

Siewe Knook

Master thesis

in partial fulfilment of the requirements for the degree of

Master of Science
in Mechanical Engineering

at the Department Maritime and Transport Technology of Faculty Mechanical Engineering of
Delft University of Technology
to be defended on May 2, 2025

Student number:	4892984	
MSc track:	Multi-Machine Engineering	
Report number:	2025.MME.9054	
Committee chair:	Dr. Bilge Atasoy	Faculty Mechanical Engineering
Thesis committee:	Dr. Jie Gao	Faculty Civil Engineering and Geosciences
	Dr. Javier Durán-Micco	Faculty Mechanical Engineering
	MSc Nynke Brouwer	Company supervisor, Movares
	Dr. Noreen Walker	Company supervisor, Movares
Date:	April 25, 2025	

Abstract

Public transportation systems operate in dynamic and unpredictable environments, necessitating close monitoring of real-world operations. Integrating data into route design, scheduling, and policy formulation is essential for creating an efficient and sustainable transit network. This thesis utilises historical bus travel time data to predict future bus travel times, enhancing the analysis of network reachability.

A literature review identified four prediction models to be investigated: Historical Average, Vector AutoRegression, Random Forest, and Long Short-Term Memory deep neural network. A case study involving six bus lines in Groningen was formulated, providing the necessary KoppelVlak 6 and GTFS schedule datasets to be used as inputs for the prediction models. During model development, patterns in historical travel time data were identified, and prediction accuracy was evaluated. The output from the most accurate prediction model was then utilised as input for the reachability analysis.

The analysis demonstrated that complex machine learning models, such as Random Forest and Long Short-Term Memory deep neural networks, yielded the most accurate predictions. The time of day when the journey occurs is particularly predictive of travel and dwell times. Integrating these predictions into a reachability analysis revealed instances of increased reachability, decreased reachability, and missed transfers compared to the original schedule.

The findings demonstrate that travel time prediction models can significantly enhance reachability analyses. This more accurate representation of reachability can be used to identify systemic issues in the design of the public transportation network, allowing for interventions to improve performance.

Preface

Dear reader,

This is the master thesis ’Realistic prediction of bus travel times for accurate representation of multi-modal reachability.’ It has been written to fulfil the graduation requirements of the Multi-Machine Engineering master at the Delft University of Technology. I was engaged in researching and writing this thesis from September 2024 to May 2025.

During my studies, I was drawn to the extensive transportation networks that enable people to lead mobile lives. The technology behind these systems has always fascinated me, and I believe it is crucial for a sustainable, healthy, and enjoyable future. Writing this thesis on bus operations has been incredibly interesting. Before concluding, I would like to express my gratitude to everyone who has supported me.

First of all, I would like to thank my supervisors. My sincere gratitude to Bilge for your excellent feedback and insightful suggestions. Jie, your clear and sharp guidance significantly enhanced the quality of this work, and I appreciate your support.

I am also extremely grateful for the involvement of Nynke and Noreen, my company supervisors. You have offered important guidance on the practical side of my master’s thesis, and your input was instrumental in shaping this work. Movares’ data team, of which I was a part for almost nine months, welcomed me and always helped whenever necessary. Being part of the data operations within a large company has been incredibly fascinating.

Finally, a huge thanks to my friends, family, and girlfriend, who listened to my stories about the complications of predicting bus travel times. Your encouragement and feedback were invaluable, and your presence made a significant difference in the completion of this work.

Enjoy the read!

Siewe

Contents

Glossary

Term	Abbreviation	Definition
Artificial Intelligence	AI	The simulation of human intelligence processes by machines, especially computer systems.
Conveyal	-	Software tool to assess the reachability of public transport networks and ordinary street networks.
Dwell time	-	Time stopped at stop
Deep learning	-	A type of machine learning using multi-layered neural networks to analyse complex data.
General Transit Feed Specification	GTFS	A standardised data format for PT schedules and geographic information.
Historical Average	HA	The average value of a dataset over a historical period, often used as a baseline comparison.
Kalman Filtering	KF	An algorithm that uses a series of measurements observed over time to estimate unknown, noisy variables.
Long Short-Term Memory	LSTM	A type of RNN architecture that is capable of learning long-term dependencies, particularly useful in time series prediction.
Machine Learning	ML	A field of AI that uses statistical techniques to allow computer systems to learn from data and improve performance over time without being explicitly programmed.
Mean Absolute Error	MAE	Average of absolute differences between predicted and actual values.
KoppelVlak 6	KV6	An interface for real-time PT data, providing vehicle locations and punctuality information for buses, trams, and metros in the Netherlands.
Nationaal Data Openbaar Vervoer	NDOV	Dutch real-time transit data.
Random Forest	RF	A ML algorithm that uses multiple decision trees to improve prediction accuracy.
Reachability	-	The ability to reach a destination from a given starting point within a certain time or distance.

Continues on next page

Term	**Abbreviation**	**Definition**
Recurrent Neural Network	RNN	A neural network designed for processing sequences, like time series or text, by using loops to maintain information.
Time series	-	A sequence of data points collected or recorded at successive points in time.
Travel time	-	The time taken to travel from one stop to another.
Vector AutoRegression	VAR	A statistical model used to capture the linear interdependencies among multiple time series.
Verbindingswijzer	-	Movares' reachability tool built in Conveyal.

1 Introduction

High-quality Public Transport (PT) improve the lives of citizens considerably. It can reduce traffic congestion and improve air quality by reducing carbon emissions [1]. Implementing an efficient and well-managed PT system will enhance a city's sustainability and improve mobility for its residents [2]. A key part of efficient PT is providing passengers and PT engineers with accurate information on the arrival and departure times of PT services. For passengers, accurate information reduces waiting times, enables better planning for connections and improves the overall travel experience [3]. Passengers can make informed decisions about their routes, adjust for delays, or choose alternative modes of transport if needed.

For transport engineers, accurate information on the performance of the PT network is essential for the design and optimisation of services. The schedule often does not reflect the real-world operation of the PT network well [4]. Utilising accurate historical information on arrival times and departure times is beneficial when designing routes, schedules or vehicle allocation for a PT network. These insights enable better scheduling, bottleneck identification and infrastructure planning [5]. By integrating historical data into design processes, engineers can create networks that minimise delays and enhance passenger satisfaction.

Incorporating this historical data is essential, because PT journeys frequently deviate from their scheduled times. PT is operating in an urban environment where disturbances are likely [4]. This means that it is difficult to maintain a deterministic travel time, because traffic and weather conditions can vary greatly. There has been a great development in collecting data on the GPS location of PT vehicles and leveraging it for the analysis of these uncertain PT networks [6]. For almost all PT journeys in the Netherlands, there is historical data on arrival and departure times at stops along the route [7]. There has been significant research in the 20th century to use this data specifically to make accurate decisions on the travel time of PT [8][9][10][11].

The research of this thesis will focus on the prediction of travel and dwell times of bus journeys. Travel time is defined as the time to reach a final destination or cross a link between two stops of the PT network. Travel time prediction refers to the prediction of current or future travel times. Dwell time is the time that a bus is stationary at a stop, meaning the difference between the arrival and departure time at the stop. Four prediction models will be developed in this thesis: a baseline Historical Average (HA) prediction model, a Vector Auto-Regression (VAR) prediction model, a Random Forest (RF) prediction model and a deep learning Long Short-Term Memory (LSTM) neural network prediction model. These prediction models will be compared and evaluated based on the patterns they identify.

The second focus point of this thesis is utilising the output of the prediction models to make reachability analysis more realistic. Reachability, in the context of PT, refers to the ability to reach a certain destination in a certain amount of time. Visual representation, such as an isochrone map or graph, is a key method to convey reachability information. Interactive software tools which generate and evaluate these visualisations are of great value to PT engineers. One of those tools is Conveyal, which utilises pre-planned schedules to present the travel times for PT. This thesis aims to predict more accurate travel and dwell times using historical travel time data to implement in this reachability tool.

1.1 Contributions

The four prediction models developed will predict potential future journeys. Current literature mainly focuses on predicting arrival time for bus journeys currently in operation. This means that current weather and traffic data can be incorporated into these models [12][13]. In this thesis, the training data for the prediction models will be the historical arrival and departure time data. In addition, extra variables derived from this dataset, such as hour of the day, day of the week and lagged variables, will be valuable for improving the prediction accuracy. This process ensures that all available information in this historical arrival and departure time dataset is utilised and will indicate whether this kind of data is sufficient for predicting future journeys. The four models tasked to utilise this training data and engineered features are increasingly more complex. Complexity refers to their ability to capture intricate patterns and relationships in the data, with HA and VAR being simple and RF and LSTM being complex. The evaluation of these models will contribute to the understanding of the nature of the travel and dwell times of a bus network.

Additionally, this thesis aims to showcase the benefits of incorporating these travel and dwell time predictions in reachability analysis. The size of the reachability region is often calculated using the planned PT schedules [14]. This research will calculate the reachability region using the predicted travel and dwell times. Comparisons between reachability based on the schedule and the predictions will be investigated. Incorporating these predictions will provide a more accurate representation of the real world than solely investigating the planned schedule. This will enable PT engineers and urban planners to make more informed network design and scheduling decisions.

1.2 Research questions

This thesis combines a proposed travel time prediction model of bus journeys and incorporates it to be used for reachability analysis. The main research question of this thesis is

How can a predictive model for public transport travel time be developed to improve reachability analysis?

Below are the sub-research questions of this thesis.

RQ1. What is the state-of-the-art for public transport travel time prediction?

RQ2. How can a real-world case study be designed to validate the output of a public transport travel time prediction model?

RQ3. Which travel time prediction models achieve high accuracy in predicting public transport travel times, and what factors influence their performance?

RQ4. What are the implications of integrating prediction models into reachability analysis tools?

Section 1.3 will present the overview of the methodological approach of this thesis to answer these research questions.

1.3 Methodological approach

1. **Literature review and technical methodology**

The research starts with a comprehensive literature review on the topics of PT travel time prediction and reachability analysis. This will address RQ1 and provide a good foundation for RQ3. Insights gained from the literature study will be used to formulate the technical methodology for developing the four prediction models, HA, VAR, RF and LSTM.

2. **Case study**

   In addition to the literature study, a real-world case study will be formulated to provide contextual data for building PT travel time prediction models. This step will address RQ2. This case study will be about a Dutch PT network. The raw data collected from the case study will undergo preprocessing and preparation, contributing to the development of the prediction models as training data and input data.

3. **Exploratory data analysis**

   An Exploratory Data Analysis (EDA) will be conducted to better understand the historical travel time data. The goal is to visualise and analyse patterns in the dataset, ensuring a solid understanding before starting the development of the prediction models.

4. **Prediction model development**

   Using the insights of the literature study and case study, the development of PT travel time prediction models will commence. Four different prediction models, HA, VAR, RF and LSTM, will be developed and evaluated. Comparing their outcomes with the EDA results provides a deeper understanding of their performance. This will address RQ3.

5. **Reachability analysis**

   The predictions generated by the models will be integrated to enhance the understanding of PT reachability. Here, the implications of these realistic travel and dwell times on reachability are investigated. This stage will answer the final RQ4.

1.4 Report outline

The report is organised as follows: Section 2 presents the literature review, with Section 2.5 summarising the key findings and identifying the research gap. Section 3 outlines the methodology for conducting EDA and developing the prediction models. Section 4 provides a detailed description of the case study, including data preprocessing, which is covered in Section 4.6. Section 5 is the results section of the research, which starts with Section 5.1 presenting the findings of the EDA. Sections 5.2-5.6 discuss the results of the four developed prediction models. Section 5.7 explores the outcomes of the reachability analysis. To conclude the research, Section 6 discusses the interpretations and implications of the results. This is followed by Section 7, which will offer the conclusions of this thesis.

2 Literature review

This section presents the relevant literature for this research. An overview of the state-of-the-art is presented to answer RQ1. The goal of this section is to identify the research gap and formulate the building blocks for the methodology and case study.

Section 2.1 highlights the critical role of data in PT, with a focus on Automatic Vehicle Location (AVL) systems. Section 2.2 explores various travel time prediction models, including parametric models (2.2.1), Kalman Filtering (KF) models (2.2.2), Machine Learning (ML) approaches (2.2.3) and deep learning models (2.2.4). Model evaluation is discussed in Section 2.3, covering methods for case study-based validation and key performance metrics. The concept of reachability analysis in urban PT systems is examined in Section 2.4. Finally, the literature review is summarised in Section 2.5.

2.1 Data in public transport

The widespread implementation of AVL provides large amounts of data on the operation of PT. Agencies and researchers can use this data to observe, collect and analyse location information about a vehicle. Ultimately, this data can be used to make informed decisions on network planning and improving passengers' experience [15]. Modern AVL systems rely on GPS systems to receive the longitude and latitude of the bus in real-time. This data is often enriched with arrivals and departures at stops during the PT journey [16].

This data has been applied to a wide variety of tasks. Performance analysis of the PT bus network, for example, is useful for the operators. Yan *et al.* [17] utilised AVL data in statistical analysis to assess spatial and temporal patterns during various route segments and time-of-day intervals. D'Acierno *et al.* [18] propose a method to estimate the urban traffic conditions based on the AVL data of the buses in the city. The method was able to accurately monitor traffic conditions not only in the bus lane, but throughout the entire road network.

In the Netherlands, national data standards are utilised. These are available for download through NDOV, which is the Dutch historical real-time transit data [7]. This database provides the schedules, actual travel information and rates for all PT services in the Netherlands.

2.2 Travel time prediction models

The application of AVL data that this thesis focuses on is travel time prediction. This section lays out the prediction models that have been used extensively for this task. These models include parametric models (2.2.1), KF models (2.2.2), ML models (2.2.3) and deep learning-based models (2.2.4). Furthermore, the datasets used for these prediction models are also discussed in this section.

2.2.1 Parametric models

In parametric models, the set of parameters estimated to establish a relationship between dependent and independent variables is predefined. These parameters will be calculated using data. The most popular parametric models are time series models and regression models. With time series models, the assumption is made that a pattern exists between historically observed data and future travel time patterns. Popular methods that fall under this category are Moving Average (MA), AutoRegressive (AR) and AutoRegressive Integrated Moving Average (ARIMA). The main advantage of these methods is their fast computational speed and ease of implementation. An overview of the literature that deploys historical average techniques is presented in Table 2.1.

Table 2.1 – Literature overview on bus travel time using analytical model approaches and historical average approaches. The prediction model column displays the prediction model used in the paper, and the dataset column displays what kind of data is used for predicting travel times.

Paper	**Prediction model**	**Dataset**
Chung and Shalaby [8], 2007	Regression	Historical GPS data, weather data, schedule
Suwardo *et al.* [19], 2010	ARIMA	Historical arrival and departure time data, schedule
Maiti *et al.* [20], 2014	MA, regression	Historical GPS data
Ma *et al.* [9], 2017	Markov chain	Historical arrival and departure time data

Chung and Shalaby [8] propose an estimated arrival time model which incorporates GPS data of the last five days of operations and the present day's operational conditions. The model that incorporated a regression model performed the best. This study concluded that data on the present operational conditions is necessary for accurate real-time prediction; solely relying on the past values is insufficient for prediction.

Suwardo *et al.* [19] propose an ARIMA method for predicting bus travel time solely based on past observations. The proposed models could effectively predict these times and could be leveraged for timetable setup.

Maiti *et al.* [20] propose a historical data model which considers vehicle trajectory and timestamps as input features. They have modelled using simple physical equations, such as HA and regression for vehicle velocity. The model has been shown to outperform alternatives such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM) prediction models in training and testing time while retaining accuracy.

Ma *et al.* [9] developed a model that captures correlations among link travel times conditional on the underlying traffic states. A Markov chain process is proposed to estimate the traffic transition probability model. The method is effective when correlations in the historical data exist.

These parametric models, such as ARIMA, MA and regression models, have a proven track record for time series forecasting. Their simplicity in implementation and ease of interpretation make them valuable tools for initial forecasting results. The papers discussed in this section show that they have acquired good results for PT travel time prediction. These models can establish a baseline prediction, which can then be improved by more sophisticated ML or deep-learning models.

A common limitation of parametric models, such as AR or MA, is their reliance on the last n previous observations. While this is often sufficient for short-term prediction, it can overlook important indicators from further in the past. More advanced ML methods discussed in Section

2.2.3, however, can generalise across the entire dataset without requiring a fixed context window.

Additionally, historical data must often be filtered by factors such as weekdays, the time of day, or other relevant characteristics to improve prediction accuracy. This process requires significant domain knowledge. Another key drawback of parametric models is their linear nature. This makes them less effective at capturing non-linear relationships in the data.

2.2.2 Kalman filter

KF has been extensively applied to the prediction of bus travel times. Table 2.2 presents a brief overview of the literature on this topic. KF consists of a state-space model of a linear stochastic system. It uses a series of measurements observed over time to estimate unknown variables in the future. Statistical noise and variance are also incorporated to produce a distribution for the output estimate.

Table 2.2 – Literature overview on bus travel time prediction using a KF approach. The prediction models column for all papers is KF approaches, but these are occasionally used in tandem with another model.

Paper	Prediction model	Dataset
Kumar *et al.* [10], 2017	Ensemble KF	Real-time GPS data
Li *et al.* [21], 2017	KF, k-NN	Historical and real-time GPS data
B. Anil Kumar and Vanajakshi [22], 2019	KF, k-NN	Historical and real-time GPS data
Zhang *et al.* [23], 2022	KF	Historical travel time data, GPS data and IC cards

Kumar *et al.* [10] propose a prediction method that considers both temporal and spatial variations in travel time. A Godunov schema was used to discretise the speed-based equation, and the prediction scheme was based on KF. The proposed method outperformed the HA and ANN methods.

Li *et al.* [21] propose a real-time mixed model for bus arrival time prediction. Accurate short-term predictions are made by considering traffic flow and delay jitter patterns, which are mined by a K-Nearest Neighbours (k-NN) algorithm. KF was applied to real-time data flow, and multi-step predictions can be obtained by combining it with a Markov Transfer database on historical data.

B. Anil Kumar and Vanajakshi [22] propose a method with a k-NN classification algorithm for data mining. A pattern between the real-time input and historical data is sought and is used for arrival time prediction. The model is based on a KF framework. It was shown that the proposed method outperforms a historic average prediction model.

Zhang *et al.* [23] developed a short-term bus travel time prediction using a KF approach. Using a large data mining approach from multiple data sources, such as GPS signals, historical travel time and IC cards.

KF is unsuitable for the task of predicting travel time in the future, which is the objective of this thesis. A KF model needs to be updated with the new measurements on the system, and this makes it suitable for real-time prediction. Applying KF on a model with a larger prediction horizon, the model will not be accurate, as predictions farther in the future will solely rely on the physical model. There won't be corrections using the system measurements. This section is included in the literature review, because it is widely used for travel time prediction.

2.2.3 Machine learning

In Section 2.2.1 parametric prediction models are discussed; this section discusses ML models, which are non-parametric models. This means that the relationship between the historical observed data and the future travel time patterns will be obtained from the data itself, along with corresponding parameters. Popular models are ANN, RF and SVM. This section will discuss ML models that are not deep learning models. Those will be discussed in Section 2.2.4.

An ANN is an ML model inspired by the brain's neural networks [24]. It consists of layers of interconnected nodes that process data and adjust their connections based on learning patterns in the data. It is an ML model which can learn complex non-linear relationships. RF is an ensemble ML model which builds multiple decision trees during training and merges their output for accurate prediction [25]. This also controls overfitting and enables it to handle large, complex datasets. Lastly, SVM is a supervised ML model, which works by finding the optimal hyperplane that best divides the data points [26]. Based on this division, it can predict travel and dwell times, for example. It is effective for complex datasets when the data is not linearly separable.

Table 2.3 – Literature overview on bus travel time prediction using diverse ML methods. The prediction model column displays the model that was found to yield the best results; in most papers, different models are also tested.

Paper	**Prediction model**	**Dataset**
Amita *et al.* [27], 2015	ANN	Historical arrival and departure time data, schedule, historical GPS data
Yu *et al.* [28], 2017	RF, k-NN	Historical AVL data
Ma *et al.* [29], 2019	SVM	Historical taxi data, historical travel time data, historical bus smart card data
Chondrodima *et al.* [30], 2022	RF	Historical arrival and departure time data
García-Mauriño *et al.* [11], 2024	ANN	Bus schedule, real-time travel time data

Amita *et al.* [27] developed an ANN which takes dwell time, delays and distance between bus stops as input to predict bus travel time. The model demonstrated superior pattern recognition compared to a linear regression model.

Yu *et al.* [28] propose a hybridisation approach of an RF model based on a k-NN model. That is, this RFk-NN model contains two main procedures. The first process involves selecting the training set for the RF model from the original dataset, which is done using a k-NN algorithm. The second process consists of training and a regression procedure for the RF model. This approach achieves high accuracy compared to other techniques such as linear regression, SVM, and classic RF.

Ma *et al.* [29] propose a segment-based bus route graph with two independent prediction models, which predict transit time and dwelling time. An SVM model is trained on bus travel time data, taxi travel time data and bus smart card data. The model can achieve high prediction accuracy in both normal and abnormal traffic conditions.

Chondrodima *et al.* [30] propose a method to use GTFS data in a framework for predicting PT arrival time. This framework combines a GTFS schedule with a real-time GTFS feed. Several machine learning algorithms are tested on this framework and an ANN had the best performance. This was mainly due to the large amount of data available and the ability of the ANN to learn intricate patterns.

García-Mauriño *et al.* [11] propose a procedure to predict the time of arrival of a bus fleet that relies solely on historical arrival time records. One model tree and two RF techniques were selected based on low computational costs, superior structure interpretability and regression capabilities. Significant improvements in terms of error mean and standard deviation are achieved for the Madrid and Paris bus fleets.

These studies demonstrate that ML models can identify patterns in data to predict arrival times effectively. They excel at handling non-linear patterns, which are frequently encountered in PT datasets. For instance, Chondrodima *et al.* [30] found that in their dataset, the most complex model, an ANN, yielded the best prediction results.

Unlike VAR and ARIMA models, which are specifically designed for time-series data, ML models are often more general-purpose. Simpler models like VAR and ARIMA can sometimes better exploit the sequential nature of the data compared to ML algorithms. This distinction is not necessarily a drawback of either approach, but it is important to consider when developing PT travel time prediction models.

The quality of data is crucial for any ML research. Yu *et al.* [28] required extensive feature engineering and preprocessing to achieve good results with RF algorithms. One key aspect of high-quality data is that the selected features must accurately represent real-world scenarios and ensure that samples are not uncorrelated [28]. Additionally, collecting a large and diverse dataset helps prevent ML models from overfitting, which is particularly important for more complex models like ANNs.

2.2.4 Deep learning

Deep learning models are considered the state-of-the-art for public transport travel time prediction. These techniques include LSTM neural networks, Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN). These techniques are widely used for time-series forecasting problems. An overview of the literature deploying these techniques for bus travel time prediction is presented in Table 2.4.

RNNS are a type of deep learning model designed to handle sequential data by maintaining a hidden state that captures information from previous time steps. However, RNNs struggle with long-term dependencies due to issues like vanishing gradients [31]. LSTM networks address this problem by introducing memory cells and gating mechanisms that regulate the flow of information. This allows them to retain information of longer sequences.

Table 2.4 – Literature overview on prediction of bus travel times using deep-learning techniques.

Paper	Prediction model	Dataset
Pang *et al.* [32], 2019	RNN, LSTM	Historical and real-time GPS data, infrastructure data
He *et al.* [33], 2019	LSTM	Bus schedule, road characteristics, historical travel time data
Liu *et al.* [34], 2020	LSTM	Historical and real-time GPS data
Alam *et al.* [12], 2020	RNN	Historical and real-time GPS data, weather data
Han *et al.* [13], 2020	LSTM	Bus schedule, historical and real-time GPS data, weather, bus mechanical properties
Chondrodima *et al.* [35], 2022	RBF ANN	Bus schedule, historical and real-time arrival and departure time data

Pang *et al.* [32] deploys an RNN with an LSTM block to correct for the passing of the earlier bus stops. Real-time data is incorporated with data on the infrastructure to create an efficient deep-learning approach. They conclude that long-range dependencies in time are necessary to accurately predict bus travel time.

He *et al.* [33] predict bus journey time for an individual passenger by separately predicting riding and waiting time. An LSTM network is used to predict the riding time of each segment of the bus/lines and routes. They show that journey time is significantly impacted by waiting time and is therefore estimated by taking the historical average. Their approach is able to outperform six baseline approaches in a case study on a part of the Singaporean bus network.

Liu *et al.* [34] propose a hybrid model of LSTM and ANN based on a spatio-temporal feature vector. Long-distance arrival-to-station prediction is performed by the temporal feature vector. Short-distance arrival-to-station prediction is realised by the spatial feature. The proposed approach is highly accurate in solving bus arrival time prediction problems.

Alam *et al.* [12] propose an LSTM model which is trained on GPS coordinates of transit buses and hourly weather data. They found that incorporating weather data drastically improved the predictive performance of the model. It even captured extreme delays and early arrivals in the predictions.

Han *et al.* [13] propose a GPS calibration method and introduces projection rules for specific road shapes. Using both historical data and real-time GPS information, an LSTM-based training model is employed for bus arrival prediction. Experimental results show that the proposed method outperforms traditional time-of-arrival techniques.

Chondrodima *et al.* [35] propose a data-driven approach for predicting arrival time based on RBF neural networks, using a modified version of the successful PSO-NSFM algorithm for training. The proposed model utilises open real-world data feeds. This RBF NN can outperform state-of-the-art prediction models in prediction accuracy and computational times.

In general, deep learning methods have demonstrated superior predictive performance compared to traditional models such as HA, ARIMA and KF. These conventional models often struggle to capture the temporal dependencies and nonlinear relationships present in real-world bus data. There is a strong consensus that incorporating long-term historical patterns can significantly enhance forecasting accuracy. This is because travel times are influenced by complex factors such as recurring traffic patterns, weather conditions and seasonal influences. This may not be fully captured by short-term dependencies alone.

In Table 2.4 it can be observed that Pang *et al.* [32] and He *et al.* [33] already incorporate infrastructure data. This suggests that these deep learning approaches are powerful enough to make informed decisions utilising more diverse data. This is also demonstrated by Alam *et al.* [12] and Han *et al.* [13], which utilise weather data in their LSTM approach.

2.3 Prediction model evaluation

The prediction models outlined in the previous sections require an input data source, which is provided by a real-world case study. This process is described in Section 2.3.1. Additionally, the standards for evaluating the performance of these prediction models are discussed in Section 2.3.2.

2.3.1 Case studies in the literature

After formulating the methodology for the PT travel time prediction, the methods must be tested on real-world scenarios. In all papers discussed in Sections 2.2.1-2.2.4, a case study is necessary to test and evaluate the models. The main goal is to use the real-world data for the prediction model.

Ma *et al.* [29] and He *et al.* [33] analyse multiple bus routes with varying lengths and road characteristics in the same urban network, as this ensures diverse and comprehensive testing. Different routes exhibit varying traffic patterns, road conditions and passenger demand. A model trained and tested on a single route may overfit to its specific characteristics, limiting its generalisability to other routes. García-Mauriño *et al.* [11] also has this approach, but takes bus routes from two different European cities, namely Madrid and Paris.

The data sourced from the case study must be of high quality and complete. This ensures that the model captures true patterns and relationships rather than noise or errors. Also, features that predict the bus travel time must be available in the case study. This can be historical records of the network [27][17][30], weather data [12][13] or traffic data [29][32].

2.3.2 Evaluation metrics

When evaluating and making improvements to the prediction models, it is crucial to select a meaningful evaluation metric. When comparing the performance on an unseen part of the dataset, comparing it to the actual target variables will quantify how well a regression model predicts outcomes. This section will discuss four of the evaluation metrics used by the literature discussed in Sections 2.2.1-2.2.4.

Mean Absolute Error (MAE) measures the average absolute difference between predicted values and actual values. MAE is useful because it maintains interpretability in the original units of the target variable and is less sensitive to outliers compared to other metrics like Mean Squared Error (MSE). For these reasons, it is widely used [10][23][28][33][30]. An alteration of the MAE metric is the MAE/distance, which is the average error of travel time per km. This compensates for the larger error, which may be caused by the longer travel distance [28].

Another metric is the Mean Absolute Percentage Error (MAPE), which measures the average percentage difference between predicted and actual values. It provides a relative measure of error, making it useful for comparing models across different datasets. Two drawbacks are that it is undefined when actual values are equal to zero, and it can over-penalise when the actual values are small [10][28][33].

The last metric that is used for the evaluation of bus travel time prediction models is Root Mean Squared Error (RMSE). This metric measures the standard deviation of prediction errors. Unlike MAE, RMSE gives more weight to larger errors due to squaring the differences, making it more sensitive to outliers [28].

2.4 Reachability analysis

Reachability refers to the time that it takes a passenger to reach certain destinations. It is a crucial factor in urban planning and mobility, as it affects economic activity, social inclusion and overall quality of life. Section 2.4.1 explains the factors influencing reachability and Section 2.4.2 outlines the analytical methods used for the analysis of reachability.

2.4.1 Reachability factors

Reachability is influenced by various factors, both spatial and temporal. Spatial network factors include the layout of routes, stops, and stations. The logical design of transfer points and hubs is crucial for enhancing multi-modal reachability. Decisions must be made based on population centres and Points of Interest (POIs) [36]. In addition to spatial factors, temporal factors also play a significant role in urban reachability. These include service frequency, operating hours, and schedule variations, all of which impact how accessible urban areas are.

Besides ensuring good reachability during normal operations, it is crucial for a PT network to be robust against random errors and systematic attacks [37]. A PT network can easily become dysfunctional if a single node is disabled. Therefore, designing the network to include viable alternative routes is imperative to account for these potential disruptions.

Reachability analysis often involves social and inclusion-related considerations. It is essential to ensure that every citizen in an urban environment has the opportunity to access necessary destinations. For instance, the layout of bus stops significantly influences accessibility to healthcare services [38]. Also, it has been found that crowding issues during peak hours can negatively impact reachability to jobs [39]. Olsson *et al.* [40] found that efficient reachable PT reduces the reliance on private cars and therefore supports environmental sustainability.

2.4.2 Analytical methods

Computing reachability analysis is a complex computational task that involves analysing numerous factors. This is often achieved using graph theory, where nodes represent stops and edges represent routes. Algorithms such as Dijkstra's or A* (A-star) can be utilised to calculate the shortest paths. There are ongoing advancements in reachability analysis techniques; for example, Tesfaye *et al.* [14] propose a cell partition method for analysing PT networks.

Various data sources can be utilised for reachability analysis. Most commonly, the provided schedule is analysed. Kujala *et al.* [41] investigate Pareto-optimal PT journeys to compare reachability. Other data sources include credit card information and AVL data, as leveraged by Arbex and Cunha [39] to investigate reachability.

2.5 Conclusion

This section aims to conclude the findings and set out design directions for the methodology and case study. As mentioned in the introduction in Section 1, the research gap of this thesis revolves around three points, namely the usage of minimal historical travel time data, predictions of future journeys and realistic predictions in the context of reachability.

2.5.1 Data

In the studied literature, the data sources used for PT travel time prediction can be ellobarote. This means that besides the historical travel time data, weather, traffic, taxi, fare card, and infrastructure data are also utilised. The input of these datasets is often leveraged by complex deep learning methods [13][32][33]. Reliable prediction results might be achieved by solely investigating the past travel time data. This data can be sourced from setting up a case study of a real-world bus line in an urban environment. This will often yield AVL data for bus travel time prediction. A form of this data is GTFS data. Chondrodima *et al.* [35] and Chondrodima *et al.* [35] utilise GTFS datasets for their training data representation, but more importantly, to convey the prediction information for more advanced analysis.

2.5.2 Prediction models

A widely accepted practice is to have a baseline prediction model applied to the data of the specific case study of the research. A historical average model is often used, due to its simplicity and intuitive interpretability. Time series models, such as MA, ARIMA and VAR, would be the next step, as these would be able to capture more intricate patterns.

A more complex approach is ML and deep-learning approaches [28] and [30] use an RF on historical datasets and yield good results. In Section 2.2.4, various deep learning methodologies are comprehensively discussed. The discussion highlights their superiority in achieving higher prediction accuracy and enhanced robustness. Deep learning techniques are widely recognised as the state-of-the-art approach for predicting PT arrival times. This establishes a strong foundation for the proposed research.

2.5.3 Research gap

The existing literature often focuses on the arrival time prediction of a bus that is currently in operation [13]. In these approaches, the prediction models consider real-time traffic and weather conditions, and the model can also use the arrival and departure times of the current bus trip. This thesis will focus on the prediction of arrival time for future bus trips. He *et al.* [33] does consider passenger journeys in the future, but not the prediction of the future bus schedule. Short-term future prediction is applied by Zhang *et al.* [23].

Another aspect of the research gap is that these future predictions will be incorporated into a reachability analysis. Patterns in travel and dwell times will influence the speed of the bus network and, therefore, reachability. This temporal factor will be expressed more realistically. Current literature mainly utilises travel time prediction to provide passengers with arrival time information of an ongoing bus journey, not to enhance reachability analysis.

3 Methodology

Section 2 provides an overview of the existing literature on the topic of PT travel time prediction and reachability analysis. This will be the theoretical foundation for the methodology section. Based on the literature review in Section 2, this section lays out the methodology for the preprocessing, EDA and prediction models. This will form the foundation for answering RQ3 and RQ4.

Four prediction models have been selected for analysis and comparison of outcomes. The best-performing model will be utilised for reachability analysis. The first model to be developed is the baseline HA model. Next, a time-series VAR model will be set up. Additionally, two ML models will be developed: an RF model and an LSTM deep learning model. This selection of models represents a step-wise increase in complexity to better understand the trends in travel and dwell times within the bus network. Further motivation is outlined in section 3.4.

This methodology begins with a description of the symbols that are used in this Methodology section, which is presented in section 3.1. Secondly, the dataset that is required as input for the prediction models is described, this is outlined in Section 3.2. Analysing this dataset will be done in the EDA, discussed in Section 3.3. After this the four prediction models are described, starting with the HA model in Section 3.5. Next, the time series model VAR is outlined in Section 3.6 and the RF model in Section 3.7. Finally, the most complex model LSTM is described in Section 3.8. The evaluation techniques that will be used for these prediction models are documented in Section 3.9. Finally, Section 3.10 describes how the output of the prediction models is utilised for the reachability analysis.

3.1 Symbol definitions

Table 3.1 defines the symbols used throughout the methodology to ensure clarity and consistency.

Table 3.1 – Symbol definitions used in the methodology.

Symbol	Definition	Unit
j	Index for journey in the dataset	-
k	Index for stop in a certain journey	-
l	Index for link in a certain journey	-
$\mathbf{k}_j$	Vector of stops in journey j	-
$\mathbf{l}_j$	Vector of links in journey j	-
n_j	Number of stops in journey j	-
N	Number of journeys in dataset	-
at_k	Arrival time at stop k	(yyyy/mm/dd: hh/mm/ss)
dt_k	Departure time from stop k	(yyyy/mm/dd: hh/mm/ss)
y_k	Dwell time at stop k	s
y_l	Travel time of link l	s
$\mathbf{y}_{\mathbf{k},j}$	Vector of dwell times of journey j	s
$\mathbf{y}_{\mathbf{l},j}$	Vector of travel times of journey j	s
$\hat{y}_k$	Predicted dwell time at stop k	s
$\hat{y}_l$	Predicted travel time of link l	s
$\hat{\mathbf{y}}_{\mathbf{k},j}$	Vector of predicted dwell times in journey j	s
$\hat{\mathbf{y}}_{\mathbf{l},j}$	Vector of predicted travel times in journey j	s

3.2 Dataset description

Two data types are necessary for accurate bus travel and dwell time prediction, namely the schedule and the historical travel and dwell times. For the historical data, the source data can be in the form of arrival and departure times at certain stops along the bus routes or the dwell and travel times of the bus journey. This data should be represented as shown in Table 3.2. Each row of this dataset is a journey along a certain bus route, the columns are the travel and dwell times of that journey. This data should be presented as a time series indexed by the departure time from the first stop of the bus journey.

Table 3.2 – This is an example of one direction of a single bus line presented in a tabular format. Dwell time y_{k_1} is the difference between the arrival time at_{k_1} at the first stop and the departure time dt_{k_1} from the first stop. This is done for all the stops in $\mathbf{k}_j$ in a journey j. Travel time y_{l_1} is the difference between the departure time dt_{k_1} at the first stop and the arrival time at_{k_2} at the second stop. This is also done for all links in $\mathbf{l}_j$ in a journey j. The table is indexed by the departure time dt_{k_1} from the first stop.

Journey	**Index**	y_{k_1}	y_{l_1}	y_{k_2}	y_{l_2}	$\cdots$	$y_{l_{n-1}}$	y_{k_n}
$j=1$	$t_0(1)$	30	80	0	65	$\cdots$	125	20
$j=2$	$t_0(2)$	45	70	50	90	$\cdots$	135	10
$j=3$	$t_0(3)$	0	95	40	85	$\cdots$	140	15
$j=4$	$t_0(4)$	45	75	0	75	$\cdots$	130	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$j=N$	$t_0(N)$	35	90	0	70	$\cdots$	140	0

The data described in Table 3.2 can be constructed from a logged AVL dataset. This dataset should include the historical arrival and departure times at each stop. Using this information, the Equations 3.1 and 3.2 calculate dwell time $y_{k,j}$ and travel time $y_{l,j}$, which form the columns of the dataset.

$$y_{k,j} = dt_{k,j} - at_{k,j} \tag{3.1}$$

Where $y_{k,j}$ is the dwell time at stop k for journey j, which is calculated by taking the difference between the arrival time $at_{k,j}$ and departure time $dt_{k,j}$ at stop k for journey j.

$$y_{l,j} = at_{k,j} - dt_{k-1,j} \tag{3.2}$$

Where $y_{l,j}$ is the travel time for link l for journey j which spans between stop k and the preceding stop $k-1$. Travel time $y_{l,j}$ is calculated by taking the difference between the departure time $dt_{k-1,j}$ from stop $k-1$ and the arrival time $at_{k,j}$ at stop k for journey j.

Table 3.2 should not have any missing values. This means that preprocessing steps such as imputation, duplicate handling and checking schedule adherence might be necessary steps to acquire this data. These steps for the specific case study are explained in Section 4.6.

Two journeys j and j' follow the same route when the stop vector $\mathbf{k}_j$ is identical to the stop vector $\mathbf{k}_{j'}$. Stop vector $\mathbf{k}_j = \{k_1, k_2, ..., k_n\}$ is defined as the stops visited by a bus in a journey j, where the first element of the vector is the first stop and the second element is the second visited stop and so on. The corresponding dwell times at the stops in $\mathbf{k}_j$ are defined as $\mathbf{y}_{\mathbf{k},j} = \{y_{k_1,j}, y_{k_2,j}, ..., y_{k_n,j}\}$.

When the stop vector $\mathbf{k}_j$ is identical, then the link vector $\mathbf{l}_j$ will also be identical. This is because the link vector $\mathbf{l}_j = \{l_1, l_2, ..., l_{n-1}\}$ is defined as the first element being the link between the first and second stop and the second link the link between the second and third stop and so on. The corresponding travel times at the links in $\mathbf{l}_j$ are defined as $\mathbf{y}_{\mathbf{l},j} = \{y_{l_1,j}, y_{l_2,j}, ..., y_{k_n,j}\}$.

3.3 Exploratory data analysis

The main objective of the EDA is to gain insights that inform subsequent prediction modelling and decision-making, which is split into two parts. Firstly, investigating the AVL data used to create the data format presented in Table 3.2 must be done. It is beneficial to identify missing values, outliers and duplicate messages in the AVL data source. With this knowledge, precise data cleaning and data transformation steps can be undertaken.

The second step of the EDA is summarising, visualising and interpreting the created dataset of travel and dwell times. This begins with generating a summary of the statistics, such as mean, median, standard deviation and percentiles. After this, histograms, box plots and density plots can be used to analyse single variables. Scatter plots and time-indexed line plots may reveal more about the patterns in time. Analysing the z-scores will reveal more about the outliers.

3.4 Prediction model selection

Four prediction models are selected for analysis, increasing in complexity from a simple baseline model. The goal is for each model to better capture the underlying patterns of travel and dwell times, thereby increasing prediction accuracy and effectively handling the numerous variables involved in bus travel time prediction.

The baseline model will be an HA model, chosen for its simplicity as it doesn't require complex models or assumptions. Maiti *et al.* [20] demonstrated the effectiveness of using average travel times for predictions. They also suggested making these averages dependent on the time of day, an approach that will be developed in this thesis.

The second model will be a VAR model. This multivariate time series prediction model can capture interdependencies between a large set of variables, which is crucial for predicting travel and dwell times, where each stop and link is represented as a single variable. With its simplicity and lag structure, VAR can effectively capture temporal dependencies. Although ARIMA has been shown to forecast public transport travel times effectively [19], it is applied to a single variable and does not account for interdependencies between variables.

The third model, RF, is the first ML model selected due to its ensemble-based nature, which combines multiple decision trees to reduce variance and enhance generalisation. Another advantage of RF is its inherent ability to provide feature importance metrics, enabling the selection of the most predictive features. This reduces dimensionality and enhances model efficiency [25]. Promising results have been observed when applying RF to historical travel time data [28][30].

The most complex model will be an LSTM neural network. Deep learning methods have been shown to capture deep underlying patterns [34]. LSTM, in particular, is advanced in predicting time-series data and can easily incorporate other variables such as weather or traffic information [12][13]. This versatility will enable us to extract the most from the engineered features of the historical travel time data.

3.5 Historical average

This section lays out the mathematical design of an efficient HA prediction model. Two variants are proposed in this research, the general HA model is described in Section 3.5.1 and the time-dependent HA model is outlined in Section 5.2.3.

3.5.1 General historical average

The predictions for a certain route, such that the stop vectors $\mathbf{k}_j$ and $\mathbf{l}_j$ are identical for all journeys considered, are the averages of travel times of the specific links and dwell times of specific stops. Equation 3.3 is used to calculate the average travel time $\hat{y}_l$ for a link l.

$$\hat{y}_l = \frac{1}{N}\sum_{j=1}^{N} y_{l,j} \tag{3.3}$$

In Equation 3.3, link l has been travelled N times in journeys in the training dataset. $y_{l,j}$ is the travel time of a certain link l for a journey j. The average is taken over all the instances N that the link l has been travelled $\{y_{l,1}, y_{l,2}, ...y_{l,N}\}$. This will give the prediction for the travel time $\hat{y}_l$ for a link l. Similar to predicting travel times, Equation 3.4 is used to predict the dwell time $\hat{y}_k$ at a stop k.

$$\hat{y}_k = \frac{1}{N}\sum_{j=1}^{N} y_{k,j} \tag{3.4}$$

In this equation, the predicted dwell time $\hat{y}_k$ at stop k is calculated by taking the average of all the dwell times at this stop for all journeys in the dataset. The average is taken over all the instances that this stop has been used in a certain route $\{y_{k,1}, y_{k,2}, ...y_{k,N}\}$.

Applying Equations 3.3 and 3.4 to all links in $\mathbf{l}$ and stops in $\mathbf{k}$, respectively, will yield predicted travel and dwell times for the complete route. The predicted vector travel times $\hat{\mathbf{y}}_\mathbf{l} = \{\hat{y}_{l_1}, \hat{y}_{l_2}, ..., \hat{y}_{l_{n-1}}\}$ and predicted vector dwell times $\hat{\mathbf{y}}_\mathbf{k} = \{\hat{y}_{k_1}, \hat{y}_{k_2}, ..., \hat{y}_{k_n}\}$ can be used as predictions for the reachability analysis.

3.5.2 Time-dependent historic average

The average dwell and travel times will vary during different times of the day [20]. To exploit this, an advancement on the HA model is developed. The aim of the time-dependent HA model is to take the average dwell and travel times of journeys with a start time t_0 that falls in a certain interval $\langle t, t + dt\rangle$ to predict feature journeys that also have start times t_0 in that same interval [20]. This way, the day is divided into similar-sized intervals to predict all journeys with varying starting times. The size of the interval is decided by the parameter dt. This should be determined by the interval size dt, which produces the best results on the test dataset.

The Equations 3.3 and 3.4 are adjusted to Equations 3.5 and 3.6. In these equations, t relates to a specific similar-sized interval $\langle t, t+dt\rangle$, the day is divided into. This means that $y_{l,t,j}$ and $y_{k,t,j}$ refer to the past travel and dwell times in certain time interval $\langle t, t + dt\rangle$. Furthermore, $\hat{y}_{l,t}$ and $\hat{y}_{k,t}$ are the predicted travel and dwell times for that time interval. This is similar to the equations presented in the previous Section 3.5.1.

$$\hat{y}_{l,t} = \frac{1}{N}\sum_{j=1}^{N} y_{l,t,j} \tag{3.5}$$

$$\hat{y}_{k,t} = \frac{1}{N}\sum_{j=1}^{N} y_{k,t,j} \tag{3.6}$$

3.6 Vector AutoRegression

This section describes the statistical method VAR that can be used to analyse and forecast multivariate time series. This method is able to model more intricate patterns in the data than the HA model. The mathematical formulation in this section for the VAR model and forecasting methods is based on Lütkepohl [42].

Section 3.6.1 outlines the definition of a VAR model. Following this, Section 3.6.2 details the optimisation process for the VAR model. Once the model is optimised, it is crucial to determine the optimal lag order, which is discussed in Section 3.6.3. Finally, Section 3.6.4 explains how the optimized VAR model can be utilised for forecasting travel and dwell times.

3.6.1 Model definition

The basic p-lag VAR model (denoted as VAR(p)) has the form as displayed in Equation 3.7.

$$Y_t = A_1 Y_{t-1} + A_2 Y_{t-2} + \cdots + A_p Y_{t-p} + u_t \tag{3.7}$$

Where:

- n is the number of stops and $n-1$ is the number of links for a bus line in a certain direction. This means that $(2n-1)$ is the number of variables in the VAR model.
- Y_t is a $(2n-1)$-dimensional vector of dwell times y_k and travel times y_l at time step t, s.t. $Y_t = \{y_{k_1}, y_{l_1}, \dots, y_{l_{n-1}}, y_{k_n}\}$. This is a combination of the vectors $\mathbf{y_k}$ and $\mathbf{y_l}$.
- $A_i \quad (1, \dots, p)$ are $(2n-1) \times (2n-1)$ coefficient matrices.
- $u_t \sim \mathcal{N}(0, \Sigma_u)$ is a white noise error term with zero mean and covariance matrix Σ_u.

3.6.2 Estimation procedure

Firstly, residual tests, such as checking for autocorrelation and normality, are conducted to ensure model adequacy. The training data input of the VAR prediction model is assumed to be stationary and the periods should be equally spread. Preprocessing steps must be undertaken to ensure these features for the prediction model.

After the estimation procedure is performed on the training data. The parameters A_i are estimated using Ordinary Least Squares (OLS).

$$\hat{A}_i = (X'X)^{-1}X'Y_i \tag{3.8}$$

where for each time series i the estimated $\hat{A}_i$ are calculated using this equation. Where X is the matrix of the lagged variables. Y_i is the vector of current values of the i-th time series.

3.6.3 Lag order optimisation

The optimal lag order p is determined using the criteria Akaike Information Criterion (AIC) [43]. AIC is a statistical measure used to compare models by balancing the quality of the fit with model complexity. It is calculated as:

$$AIC = 2(2n-1) - 2\ln(L) \tag{3.9}$$

Where:

- $(2n-1)$ is the number of estimated parameters in the model (i.e. the coefficient matrices)
- L is the maximum likelihood of the model.

A lower AIC value indicates a better trade-off between model accuracy and complexity. When using it to compare VAR models, it ensures that the model is neither underfitting nor overfitting.

3.6.4 Forecasting

After attaining an estimated VAR(p) model with an optimal lag value, the future dwell and travel times can be forecasted iteratively. Equation 3.10 is used for this process.

$$\hat{Y}_{t+h} = A_1\hat{Y}_{t+h-1} + A_2\hat{Y}_{t+h-2} + \cdots + A_p\hat{Y}_{t+h-p} \tag{3.10}$$

Where h denotes the forecast horizon. Here, the optimised $A_1, \ldots, A_i$ are used for future predictions. Similarly, $\hat{Y}_t$ is the combination of vectors $\hat{\mathbf{y}}_\mathbf{k}$ and $\hat{\mathbf{y}}_\mathbf{l}$.

3.7 Random Forest regression

This section describes the RF model that will be developed. Section 3.7.1 explains the inner workings of a decision tree which make up the RF algorithm. These decision trees make up the RF, which is explained in Section 3.7.2. RF is not strictly a time-series model. This means that lagged variables must be created, this is described in section 3.7.3. Section 3.7.4 explains the process of tuning the hyperparameters of the RF model, which is important to improve prediction accuracy. For model evaluation, feature importance analysis can be done on the trained RF model. This is outlined in the evaluation Section 3.9.2.

3.7.1 Decision trees

A decision tree is a flowchart-like structure where each internal node represents a decision based on a feature. The features in this case are travel and dwell times. Decisions typically involve whether a specific feature is higher or lower than a particular value. Depending on the outcome, the sample is led to another branch of the decision tree. The structure of the decision tree is as follows:

1. **Root node:** The topmost node in the decision tree and this represents the entire dataset. The dataset is split into child notes based on the feature that provides the best split.
2. **Internal nodes:** These nodes represent decisions based on features and sections of the dataset. These nodes split into further child nodes.
3. **Leaf nodes:** Terminal nodes that represent the final output of the regression model. These nodes take a continuous value to output.

When training the decision tree, the algorithm selects the feature that provides the highest quality split based on the criterion. The criterion that was selected for splitting is to minimise the Mean Squared Error when trying to predict the required outcomes. This criterion penalises larger errors more, which means the model is encouraged to focus on reducing significant errors. The dataset is split based on the selected feature and the process is repeated recursively. This stops when a stopping condition is met, such as maximum depth, minimum samples per leaf or no further information gain. Equation 3.11 displays how the MSE is calculated.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2 \tag{3.11}$$

Where $\hat{Y}_i$ is a vector of the predicted travel and dwell times of a journey in the training dataset, and Y_i are the true values. The goal is to minimise the MSE value in each split made during the training process.

3.7.2 Random forest

An RF algorithm fits many decision tree regressors on sub-samples of the training dataset and uses averaging to improve the predictive accuracy [25]. The first step of the process is bootstrap sampling, where random rows of the training data are selected. After feature sampling, only a random subset of features is used for each decision tree. This means a selection of travel and dwell times is used to predict future travel and dwell times. This ensures the diversity of the decision trees and avoids overfitting. The RF model is trained when every decision tree is configured.

The features that the model will train on are the travel times $\mathbf{y_l}$ and dwell times $\mathbf{y_k}$. The model is implemented using the `scikit-learn` library, namely the `RandomForestRegressor` class is used. Figure 3.1 displays the prediction process when the model is trained.

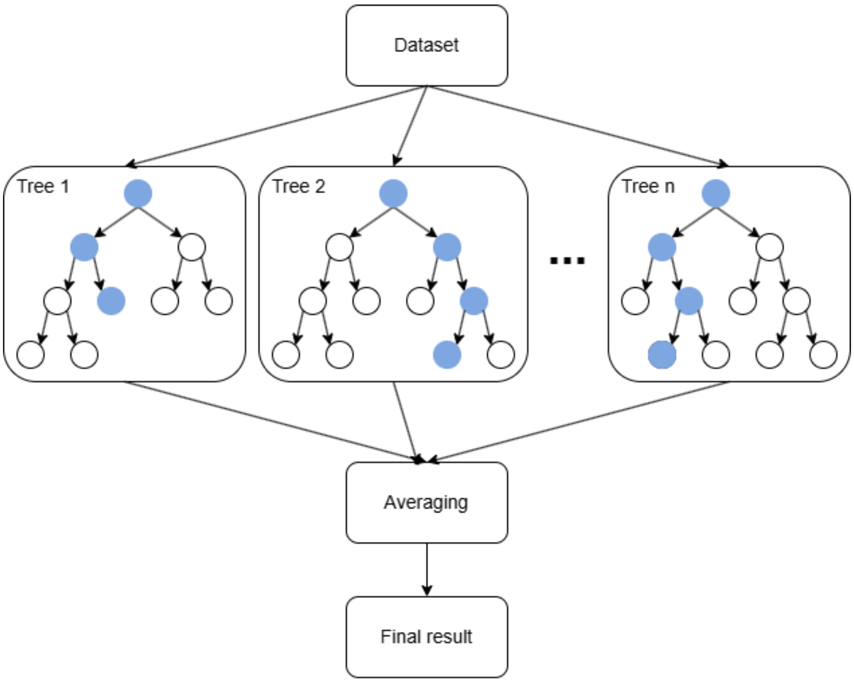


Figure 3.1 – Diagram illustrating the process of predicting when the whole RF model is trained (i.e., the decision trees are trained). The blue path is the path with certain values for input features and each decision tree outputs its predicted travel and dwell times. All the outcomes are averaged and this leads to the final prediction.

3.7.3 Lagged features

RF is not inherently a time series forecasting model. This means that lagged features must be created to enable the model to predict the time series based on historical travel and dwell times. The lagged features of a certain journey are the travel and dwell times of the previous n_{lags} journeys. The travel and dwell times of the current journey are variables that need to be predicted. n_{lags} is the number of previous journeys incorporated for which lagged features are created. This can be thought of as how far back the RF model can look.

The parameter n_{lags} can be optimised like a standard hyperparameter, by trying different values and evaluating the results. However, in the case of this specific hyperparameter, it can be determined using domain knowledge on the amount of past data that is necessary to capture patterns of interest in the data. For example, 168 lags are needed to capture the patterns for a whole week when the data is sampled hourly.

3.7.4 Hyperparameter tuning

After training the model, it is imperative to optimise its hyperparameters. Hyperparameters are the configuration of the RF model that is set before training the model. Finding the optimal set of hyperparameters will guarantee the best predictive performance for the dataset. The hyperparameters tuned for the RF model are displayed in Table 3.3. These can be tuned by Grid Search or Random Search, which are ways to test a set of hyperparameters and assess their prediction outcomes. The goal is to find the set of hyperparameters that achieves the lowest MAE.

Table 3.3 – Hyperparameters of the RF algorithm.

Parameter name	**Description**
n_estimators	The number of trees in the forest. This generally improves model performance, but also increases computational cost.
max_depth	Maximum depth of the tree. This helps prevent overfitting, but may reduce model complexity.
min_sampels_split	The minimum number of samples required to split an internal node. Higher values prevent the model from learning overly specific patterns, reducing overfitting.
min_samples_leaf	The minimum number of samples required to be at a leaf node. This helps the model generalise better.
max_features	The number of features to consider when looking for the best split. This helps to balance model accuracy and computational efficiency.
bootstrap	True or False to allow the model to use a different subset of data for training, improving robustness.

3.8 Long Short-Term Memory deep neural network

The most advanced prediction model will be an LSTM deep neural network. Section 3.8.1 explains the inner workings of a single LSTM unit. These are used in the LSTM layers of the complete neural network, which is described in Section 3.8.2. Section 3.8.3 describes the algorithm used to train the LSTM model. Afterwards, the hyperparameters of the model must be optimised. This is explained in Section 3.8.4. Finally, Section 3.9.3 explains how the evaluation of the loss function during training indicate wether the model is underfitting or overfitting.

3.8.1 Long Short-Term Memory unit

An LSTM unit is a type of recurrent neural network, designed to overcome the vanishing gradient problem. This unit is designed to handle sequential data and long-range dependencies more effectively than standard RNNs. This is done by incorporating memory cells and gating mechanisms that regulate the flow of information. The key feature is their ability to selectively remember and forget information through a set of gating mechanisms [31]. An LSTM unit consists of a memory cell, which is controlled by the following three gates:

- **Forget gate:** Determines which past information to discard from the memory cell.
- **Input gate:** Determines which new information to store in the memory cell.
- **Output gate:** Determines what information should be passed to the next time step as the hidden state

These gates allow LSTM networks to retain or discard information selectively. This behaviour enables the network to learn long-term dependencies. The forget gate is defined as Equation 3.12. It determines what portion of the previous cell state (C_{t-1}) should be forgotten. It uses a sigmoid function σ to output values between 0 (forget completely) and 1 (keep completely)

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{3.12}$$

where W_f and b_f are the weight matrix and bias, and h_{t-1} is the previous hidden state. The input gate i_t determines what portion of the new candidate cell state should be added to the current memory. The input gate is computed in Equation 3.13.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{3.13}$$

Where W_i and b_i are the weight matrix and bias. A candidate cell state $\tilde{C_t}$ is computed using Equation 3.14.

$$\tilde{C_t} = \tanh(W_C[h_{t-1}, x_t] + b_C) \tag{3.14}$$

Where W_C and b_C are the weight matrix and bias. The cell state update C_t combines the forget and input gates to update the cell state using equation 3.15

$$C_t = f_t * C_{t-1} + i_t * \tilde{C_t} \tag{3.15}$$

The output gate o_t determines how much of the new cell state should be exposed as the hidden state and is computed by Equation 3.16.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{3.16}$$

Where W_o and b_o are the weight matrix and bias. The hidden state h_t is computed as Equation 3.17. This is the short-term memory that carries information from previous time steps to the current one. This is also the data flow that can be used as output of the unit.

$$h_t = o_t \tanh(C_t) \tag{3.17}$$

The weight matrices and biases in Equations 3.12, 3.13, 3.14 and 3.16 are to be configured during training. To summarise the inner workings of an LSTM unit and how these equations produce the next cell state and hidden state, Figure 3.2 is a flow diagram of this process.

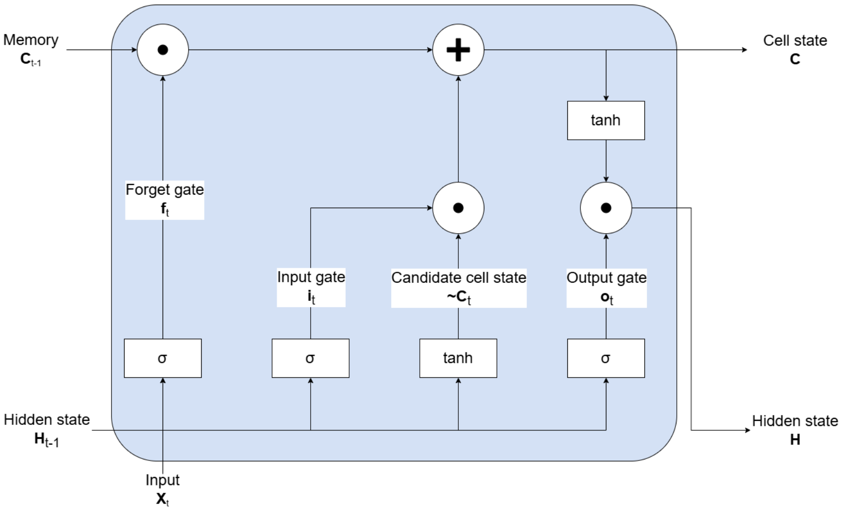


Figure 3.2 – Diagram of single LSTM unit [34]. Key components include the memory cell, input gate, forget gate, candidate cell state, output gate and hidden state. The input gate, forget gate and output gate use sigmoid functions, while the candidate memory uses a tanh function. The cell state is updated through addition and multiplication operations, and the hidden state is updated using tanh and sigmoid functions. Arrows in the diagram indicate the flow of information between these components.

3.8.2 Network layers

A deep neural network is built from different layers that propagate data. Each layer extracts and refines features from the data, gradually building up more abstract and high-level representations. The structure consists of an Input layer, LSTM layers, Dense layer and Reshape layer [34].

- **Input layer:** This is the first layer that receives the raw data. Each neuron in this layer represents a feature of the input data.
- **LSTM layers:** These layers contain the LSTM units, this is where long-term sequential patterns are captured and prepared for prediction.
- **Dense layer:** A regular densely-connected NN layer. This layer helps to transform the hidden states outputted by the LSTM layer into suitable predictions. This layer can capture non-linear relationships and ensures that the final output is in the desired shape.
- **Reshape/output layer:** This is the final layer that produces the prediction. The number of neurons is the number of target variables for regression.

Adding more LSTM layers will increase the depth of the network, which can enhance the capacity to learn from data. Deeper networks can capture more complex patterns, given sufficient data and computational resources. However, the depth must be optimal for the task of predicting bus travel and dwell times. This modularity of network structures enables the researcher to creatively design and structure the network. The goal is to find a suitable structure.

By varying the number of LSTM layers and evaluating the MAE results on the test dataset. The optimal number of LSTM layers can be found for the complexity of the data patterns. By testing multiple lines, a good structure can be found. It is also essential to investigate whether the models are not overfitting or underfitting. This process is explained in Section 3.9.3.

3.8.3 Model training

The training of an LSTM network follows a gradient-based optimisation approach using Backpropagation Through Time [44]. The cell states, weights and biases of the LSTM and the weights of the Dense layers are initialised. For the learning process, there are four steps:

1. **Forward pass:** Process each time step through the network with the current LSTM and Dense parameters.
2. **Compute loss function:** With outputs, the loss function (mean squared-error) is computed on the unseen validation dataset (part of the training dataset).
3. **Backwards pass:** The loss gradients must be computed through time, propagating errors backwards.
4. **Parameter update:** The LSTM and Dense weights are updated using stochastic gradient descent.

This is repeated over multiple epochs until convergence. Convergence can be defined based on the validation loss or early stopping parameter. Validation loss indicates how well the model generalises to new, unseen data and should be minimised. An epoch is one complete pass of the training dataset through the algorithm.

3.8.4 Hyperparameter tuning

When a suitable structure of the LSTM deep learning network is determined for the task of travel and dwell time prediction, the hyperparameters of the network should be optimised to ensure effective learning and reliable results. Table 3.4 presents several hyperparameters that must be tuned when training the model.

Table 3.4 – Important hyperparameters that must be tuned for an LSTM deep neural network. This is a non-exhaustive list of hyperparameters that can be set [45].

Parameter name	Description
# of LSTM units	Number of units in the LSTM layer. More units can capture more complex patterns, but may increase the risk of overfitting.
Learning rate	Controls how much the model's weights are adjusted with respect to the loss gradient. The learning rate is a trade-off between training speed and precision of adjustments during training.
# of epochs	Number of times the entire training dataset is passed through the model during training. More epochs can improve model performance, but may also lead to overfitting.
Optimiser	Algorithm that is used to update the model's weights based on the loss function. Common optimisers include Adam, SGD, and RMSprop, each with different strategies for adjusting weights.
Activation function	Function that determines the output of a neuron given an input or set of inputs. Common activation functions include `ReLU`, `sigmoid`, and `tanh`, each affecting the model's learning and performance differently.
Dropout rate	Fraction of neurons that are randomly dropped during training to prevent overfitting. A higher dropout rate can improve generalisation but may slow down training.
Kernel initialiser	Algorithm used to initialise the weights of the model. Common initialisers include `'glorot_uniform'`, and `'he_normal'`.

Exploring the hyperparameter space can be done by a Random Search and Bayesian optimisation. Random search is trying random sets of hyperparameters and evaluating the best one. A more refined Grid Search can follow this.

Bayesian optimisation is a probabilistic model-based approach for hyperparameter tuning that uses a Gaussian process to predict the performance of different hyperparameter settings [46]. It iteratively updates this model and aims to balance exploration of new hyperparameters and exploitation of well-performing hyperparameters. This method requires fewer evaluations compared to grid or random search. It is useful for tuning LSTM deep learning models where evaluations are computationally expensive.

3.9 Prediction model evaluation

This section describes how the prediction models will be evaluated and compared. These evaluation methods will be used to conclude the best-performing travel time prediction model. Firstly, Section 3.9.1 outlines how the MAE will be calculated on the test dataset which enables the comparison between prediction models. Section 3.9.2 describes the feature importance evaluation that can be done for the trained RF model. Finally, Section 3.9.3 introduces the analysis of the loss of the validation and training dataset during the LSTM training process. This will indicate how reliable the trained models are.

3.9.1 Actual travel times

Evaluating the model's ability to predict future travel times is important to ensure that it is capable of capturing real-life patterns. The data that is being tested on must be more recent than the travel time data with which the prediction model was trained. This is to avoid lookaheads, the phenomenon of future information that is improperly or deliberately used when making predictions on current or past events. This results in data leakage, where the model gains access to information it would not realistically have in a real-world prediction scenario, leading to overly optimistic performance metrics during training and evaluation.

To avoid this a test dataset is data of the last 20 percent of the journeys. These trips will be predicted using the models and evaluated using MAE. This is done by comparing the travel time of links and the dwell time at stops of the test dataset with the predicted travel and dwell times. MAE was selected because it is intuitive and robust for outliers. The actual travel time of a link y_l and the predicted travel time of the link $\hat{y}_l$. The MAE for link l is calculated using Equation 3.18, which is travelled N times in the test dataset. Equation 3.19, is the MAE for dwell time at stop k when travelled in a certain direction.

$$MAE_l = \sum_{i=1}^{N} \left| \frac{y_{l,i} - \hat{y}_{l,i}}{N} \right| \tag{3.18}$$

$$MAE_k = \sum_{i=1}^{N} \left| \frac{y_{k,i} - \hat{y}_{k,i}}{N} \right| \tag{3.19}$$

The MAE will be calculated on the unseen test dataset and serve as an indication of the prediction model's performance. A lower MAE indicates a better prediction.

3.9.2 Feature importance evaluation

When the RF is trained, there will be prediction decisions made in the decision trees on certain features. These features can be analysed through feature importance evaluation. The goal is to investigate which features are determinative for the regression output of the RF model. This evaluation analyses the Mean Decrease in Impurity (MDI).

MDI is also known as Gini importance. This method measures the average decrease in node impurity caused by a certain feature. Decision trees use node impurity to decide splits. When a certain feature is used to split a node, the impurity decreases, and the decision tree gets closer to an output. This decrease in impurity can be attributed to the feature on which this node is split. When averaging the impurity decrease of every node in all decision trees in the RF, the mean impurity decrease caused by each feature is calculated. A higher decrease in impurity caused by a feature indicates a greater contribution to the model's decision-making process.

3.9.3 Loss analysis

The loss function measures how well the model's predictions match the validation dataset during the training of an LSTM model. Essentially, it quantifies the difference between the predicted values and the true values. This loss function is calculated using a validation dataset, and the model tries to minimise the loss value. In addition, the loss function can also be calculated when trying to predict the outcomes of the training dataset. Analysing the loss functions on the training and validation sets will give important insights into the model's behaviour.

The loss function can be any function. However, it is typically the MSE for regression tasks. During training, the training and validation losses can be calculated every epoch. A line plot can be generated with epochs on the x-axis and loss values on the y-axis. When the trends of the lines are observed, the following things can be identified.

- **Convergence:** Both losses decrease and stabilise at similar values, the model is well-trained.
- **Underfitting:** If both losses remain high, the model is not learning adequately, requiring changes to the model's architecture or hyperparameter tuning.
- **Overfitting:** Validation loss is significantly higher than training loss; the model memorises the training data rather than generalising.

3.10 Reachability analysis

Section 3.10.1 explains the mathematical definition of reachability and Section 3.10.2 provides several examples of reachability analysis that can be performed with more accurate travel and dwell times predictions.

3.10.1 Mathematical definition

Reachability is defined as the ability to access a set of destinations within a given time frame from a specific starting point. Let:

- S be the set of all possible starting points (e.g., bus stops).
- D be the set of all possible destinations.
- $T(s,d)$ be the travel time function, which gives the travel time from starting point $s \in S$ to destination $d \in D$

The reachability set $R(s,t)$ from a starting point s within a time threshold t is defined as:

$$R(s,t) = \{d \in D \mid T(s,d) \leq t\} \tag{3.20}$$

This set $R(s,t)$ includes all destinations d that can be reached from s within time t. Another set that can be defined is the ischrone $I(s,t)$, which is a contour that represents the boundaries of the reachability set. For a given starting point s and time threshold t, the isochrone $I(s,t)$ is the set of points that are exactly t units of away from s, defined in Equation 3.21

$$I(s,t) = \{d \in D \mid T(s,d) = t\} \tag{3.21}$$

The travel time function T estimates the time required to travel between two points in a transportation network. This function will be influenced by the predicted travel times $\mathbf{\hat{y}_l}$ and dwell times $\mathbf{\hat{y}_k}$. This mathematical framework helps to visualise and analyse how accessible different parts of an urban PT network are.

3.10.2 Types of analysis

Plotting the isochrones on a map will visualise the areas that are reachable within specific time thresholds. This is part of the spatial analysis of reachability. Analysing the variation in travel and dwell times during the day will help to understand how reachability changes throughout the day. This would be temporal analysis.

Another powerful analysis is scenario analysis, where different interventions in the bus network are performed (e.g., increased frequency, new routes). The impact on the reachability can be analysed. This way also the prediction of travel and dwell times can be validated.

4 Case study

To validate the proposed methodology, a case study is conducted, focusing on the bus network of Groningen. This case study is introduced in Section 4.1. This section also outlines the two key datasets in this case study. The first dataset consists of historical arrival and departure time data, sourced from the Nationale Data Openbaar Vervoer (NDOV) database [7]. Specifically, this data is formatted as KoppelVlak 6 (KV6), which provides real-time insights into the actual state of the PT network. The KV6 data format is further detailed in Section 4.2. The second dataset contains the planned schedules used by PT operators to structure the network. These schedules are presented in the General Transit Feed Specification (GTFS) format, which is discussed in Section 4.3. Additionally, Section 4.4 describes Conveyal, a reachability tool that serves as the input for our predicted dwell and travel times. Section 4.5 outlines the integration of KV6 and GTFS datasets to create the input for Conveyal, illustrated by a data flow diagram. Finally, Section 4.6 explains the preprocessing steps necessary for the KV6 and GTFS datasets. This section aims to answer RQ2 by outlining and designing a real-world case study.

4.1 Public transport Groningen

Groningen's bus network will be analysed for the case study. The transportation network has been examined and optimised recently for the province of Groningen [47]. Several mobility issues regarding the university campus and train stations were solved. In this particular evaluation, reachability analysis has been conducted based on the schedule [48]. Additionally, the choice of this city and network ensures that the necessary PT travel data is present, as it was also utilised in this project. This previous work provides context for the results and conclusions of this thesis.

Since 2019, the bus network in Groningen has been operated by Qbuzz. For this analysis, lines 1, 2, 3, 4, 5, and 6 are selected. These lines belong to the Q-link network of the city [49]. This network is designed as a fast bus network connecting the city of Groningen and its neighbouring towns. Lines 11 and 15 also belong to this network, but were not selected for this case study. These lines drive a similar route as line 1, offering a more direct bus to the university campus of the city. This overlap makes it unnecessary to assess lines 11 and 15 for the reachability analysis in this thesis.

The selected six lines ensure comprehensive coverage in all directions out of the city. These lines have a consistent, frequent schedule, providing quality data for the prediction models. Predicting these lines allows for a meaningful reachability analysis in diverse regions of Groningen's bus network. The lines are referred to in the data and official documentation as g501, g502, g503, g504, g505, and g506, respectively. This notation will be used throughout the thesis. Figure 4.1 displays all six bus lines on a map.

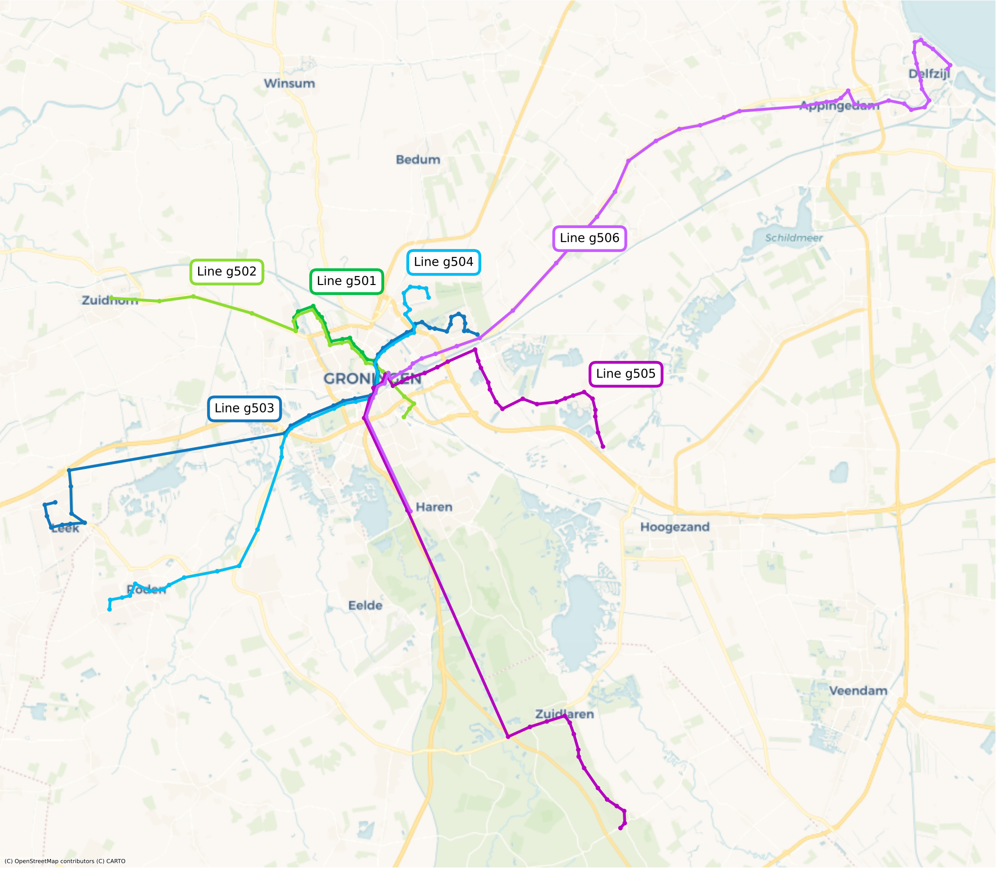


Figure 4.1 – The six bus lines in Groningen that are used in the case study. These lines fan out in different directions to neighbouring towns. All lines, except g502, stop at the main train station of Groningen.

Figure 4.1 offers a spatial overview of the six bus lines, Table 4.1 outlines the outer stops of the bus lines, the number of stops along them and the length.

Table 4.1 – The six Qbuzz lines in Groningen in the case study. Outer stop 1 to outer stop 2 is indicated as direction 1 and outer stop 2 to outer stop 1 is indicated as direction 2.

Line	**Outer stop 1**	**Outer stop 2**	**# of stops**	**Length (km)**
g501	Groningen, P+R Reitdiep	Groningen, Hoofdstation	19	7.73
g502	Zuidhorn, Station	Groningen, Station Europapark	26	17.90
g503	Groningen, Ruischerbrug	Leek, Oostindie	39	29.64
g504	Groningen, Wibenaheerd	Roden, Kastelenlaan	39	26.70
g505	Annen, Zuid	Scharmer, Goldberweg	33	33.80
g506	Delfzijl, Station	Groningen, Hoofdstation	48	36.65

The analysed data covers the period from September 1st to October 25th. This timeframe was selected to capture regular usage of the PT network. September 1st marks the end of the summer vacation, when people resume their normal work routines, leading to consistent PT usage. October 25th was chosen as the endpoint because the wintertime transition occurs on October 27th, which would shift travel time data and require adjustments. By using data up to October 25th, we ensure homogeneity in the dataset.

As explained in Section 3.2, historical travel times are necessary as input for the prediction models. This case study provides this data in the form of KV6 data. The connected schedule is presented as GTFS bundles.

4.2 KV6 dataset description

The KV6 dataset, updated in real-time, captures the arrival and departure times at each stop for all PT journeys in the Netherlands. Table 4.2 provides an overview of the eight types of messages in the KV6 dataset. When these messages are correctly reconstructed for a PT journey, they provide a precise overview of its performance and operational patterns.

Table 4.2 – Data message types of KV6 dataset. Each message is sent with a corresponding combination of keys as defined in Table 4.3 and a timestamp.

Message type	Description
Delay	Expected delay of journey that has not been initialised yet.
Init	Journey is initialised and the vehicle is assigned.
Departure	Leaves or passes stop.
Onroute	Underway on planned route.
Arrival	Arrival at stop.
Onstop	Vehicle is at stop (sent when vehicle is longer at stop than message interval).
Offroute	Vehicle is on unknown route.
End	Vehicle is uncoupled.

Each message presented in Table 4.2 is indexed with the keys in Table 4.3. The primary keys can uniquely identify the PT journey and its events. Each message contains other variables with additional information that can be useful for analysis.

Table 4.3 – Variables of a message in the KV6 dataset. The first five keys identify a unique PT journey. The two stop keys are used to identify messages in Table 4.2 that are related to a certain stop along the PT journey.

	Element	Description
Primary keys	DataOwnerCode	PT agency
	LinePlanningNumber	Line number as defined by the PT agency.
	OperatingDay	Day of operation.
	JourneyNumber	Public journey number as defined by the PT agency.
	ReinforcementNumber	0=normal schedule, >0=extra scheduled PT journey.
Stop keys	UserStopCode	Stop code of stop defined by the PT agency, which is being arrived at or departed from
	PassageSequenceNumber	Passage number corresponding to the UserStopCode
Other	Timestamp	Time of sending of the message
	Vehicle number	Vehicle identification number
	Punctuality	Divergence of the schedule in seconds
	RD-X	RDS in meters
	RD-Y	RDS in meters

For the case study, the KV6 messages are gathered for the six lines in Groningen between the 1st of September and the 25th of October. This dataset contains 1,808,374 messages, with only the Init, Arrival, Departure and End messages. The other messages were already deleted as they are not necessary for the creation of the data format.

Using these messages, the data structure described in Section 3.2 can be constructed. Dwell times for a journey are derived by calculating the difference between arrival and departure times at each stop. Similarly, travel times between consecutive stops are determined by subtracting the departure time of the previous stop from the arrival time at the next stop. This preprocessing step is further detailed in Section 4.6.4.

4.3 GTFS schedule dataset description

Besides the historical travel times of the KV6 dataset, NDOV also contains the planned PT schedule of the Netherlands in the NeTEx dataset. In this project, the planned PT schedule is needed in a GTFS format, as this is required for the Conveyal input. OVapi [50] transforms the NeTEx schedule from NDOV to GTFS format.

GTFS is a standardised data format that provides a structure for public transit agencies to describe the details of their services, such as schedules, stops, fares, etc. [51]. The open data standard is the go-to standard for many public transport agencies to represent the schedules. It contains information about routes, schedules and fares. Table 4.4 presents an overview of the files that are present in a GTFS schedule.

Table 4.4 – Text files that make up a GTFS schedule dataset. These files are interconnected through common identifiers, such as `route_id`, `trip_id` and `stop_id`, allowing for a cohesive representation of the PT system. The purpose of the GTFS schedule is to provide a standardised format for PT agencies to share their transit data.

File name	**Description**
agency.txt	Transit agencies with service represented in this dataset.
stops.txt	Stops where vehicles pick up or drop off riders.
routes.txt	Transit routes.
trips.txt	Trips for each route.
stop_times.txt	Times that a vehicle arrives at and departs from stops for each trip.
calendar_dates.txt	Service dates specified using a weekly schedule with start and end dates.
shapes.txt	Rules for mapping vehicle travel paths.
transfers.txt	Rules for making connections at transfer points between routes.
feed_info.txt	Dataset metadata, including publisher, version and expiration information.

The different files in Table 4.4 relate to each with specific ids, an overview is presented in Figure 4.2. These files contain all information on the planned schedule of the six bus lines of Groningen.

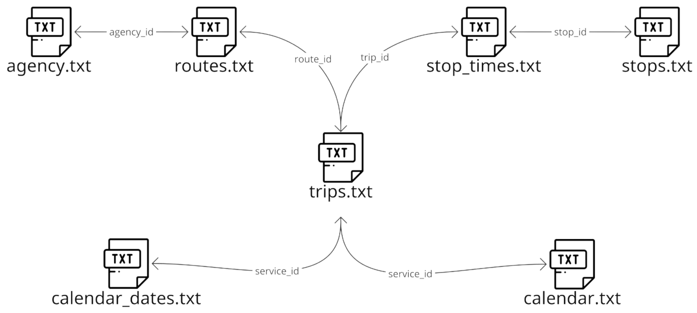


Figure 4.2 – Diagram of how the ids are used in a GTFS Schedule dataset to inter-relate information between files [51].

4.4 Conveyal

For this case study, Conveyal is used as the reachability analysis tool [52]. This tool uses GTFS datasets as input for PT data and Open Street Map as input for private modes of transport. The KV6 dataset provides the historical travel data, which will be input for the prediction models. The predicted schedules are written in the GTFS dataset. The goal is to use these predictions to improve reachability analysis.

Conveyal is a web-based analysis tool. This tool enables the visualisation of multi-modal transportation networks. The real strength of the tool is the ability to configure custom scenarios and studies. The comparison between different scenarios empowers PT engineers to understand and improve the reachability of urban environments. For visualisations, reachability is displayed using coloured isochrones on a map. An example of this is shown in Figure 4.3.

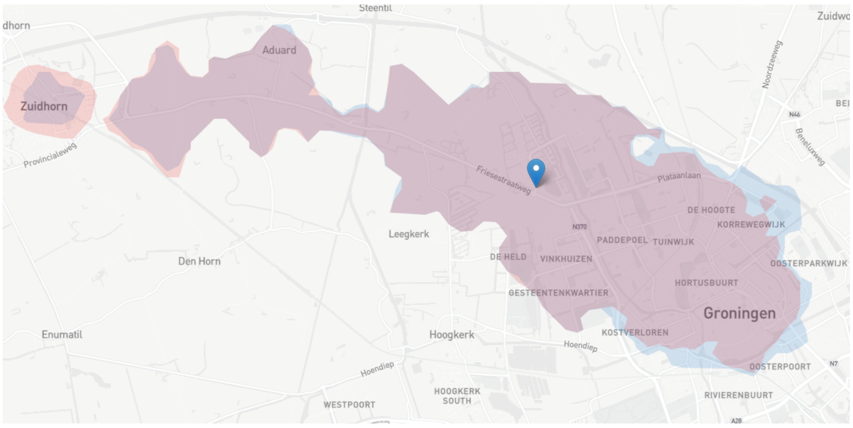


Figure 4.3 – Example of reachability analysis run in Conveyal. The different coloured areas depict how far you can travel in 45 minutes from the starting point using walking and public transport. The red and blue colours display the reachability of different configured scenarios. The isochrone is coloured purple when the isochrones overlap.

Conveyal calculates door-to-door travel time through the actual street plan and public transport options presented in the uploaded GTFS dataset. It uses the centre of a regular grid of cells as potential destinations that can be reached. Travel time includes reaching nearby transit stops (walking, bicycling or driving a car), waiting to board PT, riding in PT and travelling from a transit stop to the destination. The tool calculates travel times for all possible departure times within a specific departure window for every destination grid cell. All these different travel times form a statistical distribution, from which a certain percentile of travel time, set in the user interface, can be displayed.

4.5 Data framework

This section sets out to present the design of the data framework in more detail. The aim is to show how the KV6 dataset will be used as training data for the travel time prediction model. The GTFS schedule will be used as query data for the prediction model, and the output of the prediction model will overwrite certain files in the GTFS schedule. This altered GTFS schedule will be used as input for the analysis in Conveyal. Figure 4.4 shows the complete overview of the data framework of the project. The rest of this section explains subparts of this framework in more detail.

An important note, as outlined in Section 3.2, the prediction models will predict travel times between stops and dwell times at stops. In these diagrams, predicted arrival and departure times at each stop along the route are discussed. These arrival and departure times are cumulatively constructed from the predicted travel and dwell times. This process is further detailed in Section 4.6.12.

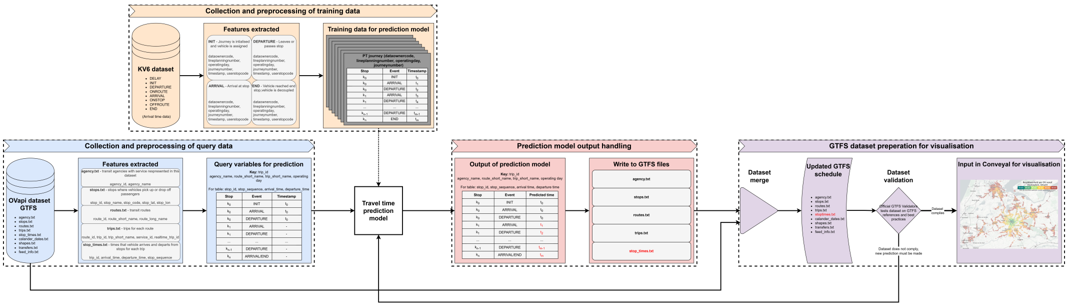
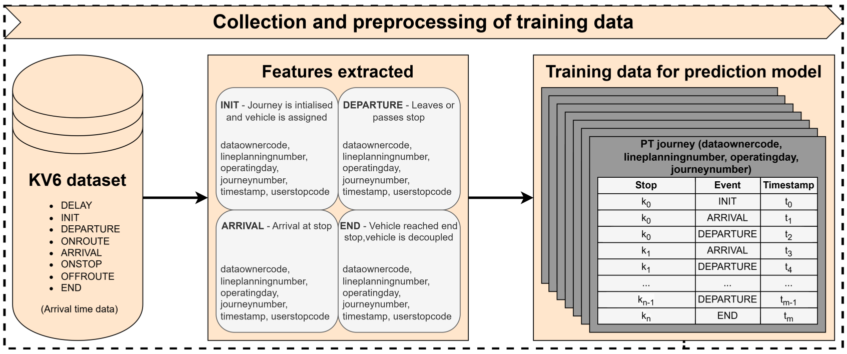

Collection and preprocessing of training data
KV6 dataset
Features extracted
Training data for prediction model
Collection and preprocessing of query data
OVapi dataset GTFS
Features extracted
Query variables for prediction
Travel time prediction model
Prediction model output handling
Output of prediction model
Write to GTFS files
agency.txt
stops.txt
routes.txt
trips.txt
stop_times.txt
GTFS dataset preperation for visualisation
Dataset merge
Updated GTFS schedule
Dataset validation
Input in Conveyal for visualisation

Figure 4.4 – Overview of how the KV6 and GTFS datasets will be leveraged to predict travel and dwell times, which will be used as input for the reachability analysis in Conveyal. In this diagram, the prediction model is assumed to be a black box. The yellow diagram on top is the training data where the KV6 messages are converted to journeys. The blue box displays a journey from the GTFS dataset that the prediction model will predict. In the red box, the output of the prediction model is shown, which will be used to overwrite `stop_times.txt`. This new GTFS dataset will serve as input for the reachability analysis in Conveyal.

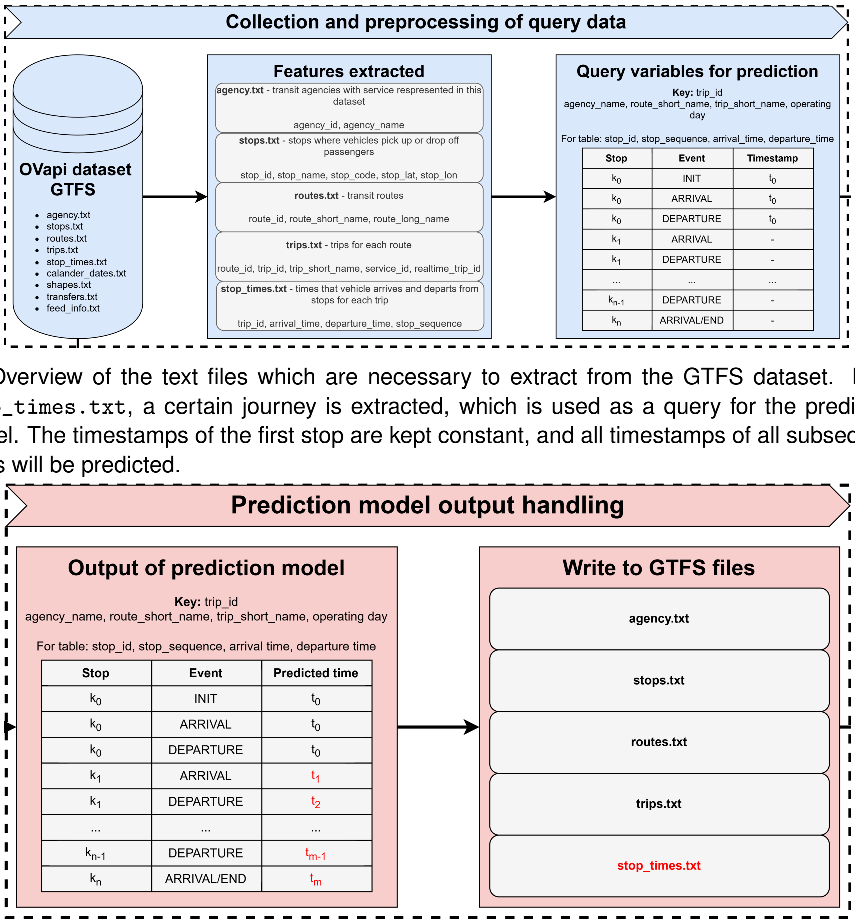
The KV6 dataset will be used as training data for the prediction model. Only Init, Arrival, Departure and End messages will be used as input for this analysis. With these messages, a PT journey can be constructed using the arrival and departure messages for each stop the journey visits. The first and last stops are the Init and End messages, respectively. This approach is visualised in Figure 4.5.


Collection and preprocessing of training data
KV6 dataset
DELAY
INIT
DEPARTURE
ONROUTE
ARRIVAL
ONSTOP
OFFROUTE
END
(Arrival time data)
Features extracted
INIT - Journey is intialised and vehicle is assigned
dataownercode, lineplanningnumber, operatingday, journeynumber, timestamp, userstopcode
DEPARTURE - Leaves or passes stop
dataownercode, lineplanningnumber, operatingday, journeynumber, timestamp, userstopcode
ARRIVAL - Arrival at stop
dataownercode, lineplanningnumber, operatingday, journeynumber, timestamp, userstopcode
END - Vehicle reached end stop,vehicle is decoupled
dataownercode, lineplanningnumber, operatingday, journeynumber, timestamp, userstopcode
Training data for prediction model
PT journey (dataownercode, lineplanningnumber, operatingday, journeynumber)

Stop	Event	Timestamp
k_0	INIT	t_0
k_0	ARRIVAL	t_1
k_0	DEPARTURE	t_2
k_1	ARRIVAL	t_3
k_1	DEPARTURE	t_4
...	...	...
k_{n-1}	DEPARTURE	t_{m-1}
k_n	END	t_m

Figure 4.5 – Overview of the collection of the training data from the KV6 dataset. Using the Init, Arrival, Departure and End messages, the journeys are reconstructed.

As mentioned in Section 4.4, Conveyal takes a GTFS schedule as input. Using the same dataset as the query data of the prediction model is an efficient choice. In this proposed setup, the specific PT journeys that must be predicted are extracted from the GTFS schedule. This means that a journey specified by `trip_id` will only keep its value for t_0 of the first stop, and the timestamps of the rest of the journey will be predicted. This process can be seen in Figures 4.6a and 4.6b.


Collection and preprocessing of query data

OVapi dataset GTFS

- agency.txt
- stops.txt
- routes.txt
- trips.txt
- stop_times.txt
- calander_dates.txt
- shapes.txt
- transfers.txt
- feed_info.txt

Features extracted

agency.txt - transit agencies with service respresented in this dataset

agency_id, agency_name

stops.txt - stops where vehicles pick up or drop off passengers

stop_id, stop_name, stop_code, stop_lat, stop_lon

routes.txt - transit routes

route_id, route_short_name, route_long_name

trips.txt - trips for each route

route_id, trip_id, trip_short_name, service_id, realtime_trip_id

stop_times.txt - times that vehicle arrives and departs from stops for each trip

trip_id, arrival_time, departure_time, stop_sequence

Query variables for prediction

Key: trip_id
agency_name, route_short_name, trip_short_name, operating day

For table: stop_id, stop_sequence, arrival_time, departure_time

Stop	Event	Timestamp
k_0	INIT	t_0
k_0	ARRIVAL	t_0
k_0	DEPARTURE	t_0
k_1	ARRIVAL	-
k_1	DEPARTURE	-
...	...	...
k_{n-1}	DEPARTURE	-
k_n	ARRIVAL/END	-

(a) Overview of the text files which are necessary to extract from the GTFS dataset. From `stop_times.txt`, a certain journey is extracted, which is used as a query for the prediction model. The timestamps of the first stop are kept constant, and all timestamps of all subsequent stops will be predicted.

Prediction model output handling

Output of prediction model

Key: trip_id
agency_name, route_short_name, trip_short_name, operating day

For table: stop_id, stop_sequence, arrival time, departure time

Stop	Event	Predicted time
k_0	INIT	t_0
k_0	ARRIVAL	t_0
k_0	DEPARTURE	t_0
k_1	ARRIVAL	t_1
k_1	DEPARTURE	t_2
...	...	...
k_{n-1}	DEPARTURE	t_{m-1}
k_n	ARRIVAL/END	t_m

Write to GTFS files

- agency.txt
- stops.txt
- routes.txt
- trips.txt
- stop_times.txt

(b) Output of prediction model with predicted arrival and departure times of the queried journey. The predicted timestamps are shown in red in the diagram. These red timestamps overwrite the original schedule in `stop_times.txt` of the query GTFS dataset.

Figure 4.6 – Diagrams showcasing the process of the GTFS schedule being predicted by the prediction model.

After this process, the datasets are combined to create a complete and functional GTFS schedule. This will be uploaded to Conveyal, where scenario studies can be done on the predicted schedule. This is shown in the last subdiagram in Figure 4.7

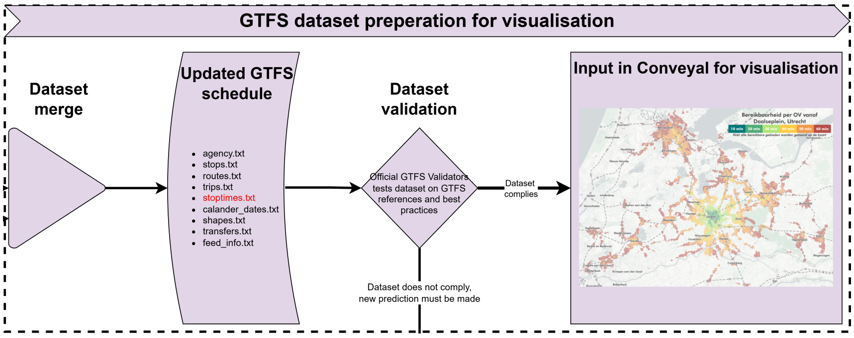

GTFS dataset preperation for visualisation
Dataset merge
Updated GTFS schedule
agency.txt
stops.txt
routes.txt
trips.txt
stoptimes.txt
calander_dates.txt
shapes.txt
transfers.txt
feed_info.txt
Dataset validation
Official GTFS Validators tests dataset on GTFS references and best practices
Dataset complies
Dataset does not comply, new prediction must be made
Input in Conveyal for visualisation

Figure 4.7 – Merging the original GTFS schedule with the changed `stop_times.txt` to create a complete GTFS dataset to upload to Conveyal.

4.6 Data preprocessing

This section and Section 5.1 are closely related, as results from the EDA have influenced design choices for the preprocessing steps presented in this section. However, this section focuses on the practical necessities of the data handling and processes presented by the KV6 and GTFS data. Section 5.1 presents more of the underlying patterns that are found in the EDA, which impact the decisions made for the ML models.

This section details the preprocessing steps required to convert the KV6 dataset into the format described in Section 3.2. The KV6 dataset, an AVL dataset, must first be transformed into a time-series format capturing travel and dwell times, as outlined in Sections 4.6.1-4.6.4. Following this transformation, additional preprocessing steps specific to each prediction model are necessary, as described in Sections 4.6.5-4.6.10. All KV6-related preprocessing steps are summarised in Section 4.6.11.

In addition to preprocessing the KV6 training data, the query and output data also require minor adjustments. The GTFS dataset, which will be used as input for the reachability analysis, needs small transformations. These transformations are detailed in Section 4.6.12.

4.6.1 KV6 message imputation

For the proposed data format described in Section 3.2, it is necessary to take the differences between timestamps to calculate dwell and travel times. This means that for a journey, for every stop, there should be an arrival and a departure message with a timestamp to compute these variables. However, in the KV6 dataset, arrival and departure messages are missing occasionally. To optimise the dataset usage and avoid removing too many journeys, an effort is made to impute these missing messages. This is done only when a single necessary arrival or departure message is missing for a stop, not when both are missing.

Arrival messages

Missing arrival messages are due to the way the KV6 data is collected. By design, when a bus skips a stop, there is no arrival message for that stop. This occurs when no passengers are boarding or exiting the bus at a certain stop. There will only be a departure message for this skipped stop and this will prevent the correct calculation for travel time between the previous stop and this stop.

The solution is to impute an arrival message for each unmatched departure message. This imputed arrival message has the same dataownercode, lineplanningnumber, userstopcode, operatingday, journeynumber, reinforcementnumber and timestamp as the unmatched departure message. This means that the dwell time for an imputed arrival message stop is considered zero seconds, as the timestamps of the arrival and departure messages are identical. This process is illustrated in Table 4.5b.

Table 4.5 – Demonstration of an arrival message imputation. This shows an arbitrary subpart of a bus journey in the case study dataset. In this journey, the bus didn't stop at stop 6.

(a) Unmatched departure message in journey at stop 6 of the journey.

Journey	Messagetype	Stop	Timestamp
⋮	⋮	⋮	⋮
QBuzz, line 1, journey 25	Arrival	5	2024-09-02 11:22:21
QBuzz, line 1, journey 25	Departure	5	2024-09-02 11:22:54
QBuzz, line 1, journey 25	Departure	6	2024-09-02 11:23:48
QBuzz, line 1, journey 25	Arrival	7	2024-09-02 11:24:32
QBuzz, line 1, journey 25	Departure	7	2024-09-02 11:25:03
⋮	⋮	⋮	⋮

(b) Imputed arrival message with the same journey information, stop and timestamp as the unmatched departure message.

Journey	Messagetype	Stop	Timestamp
⋮	⋮	⋮	⋮
QBuzz, line 1, journey 25	Arrival	5	2024-09-02 11:22:21
QBuzz, line 1, journey 25	Departure	5	2024-09-02 11:22:54
QBuzz, line 1, journey 25	Arrival	6	2024-09-02 11:23:48
QBuzz, line 1, journey 25	Departure	6	2024-09-02 11:23:48
QBuzz, line 1, journey 25	Arrival	7	2024-09-02 11:24:32
QBuzz, line 1, journey 25	Departure	7	2024-09-02 11:25:03
⋮	⋮	⋮	⋮

When this preprocessing step was applied to the case study dataset, 507,782 arrival messages were imputed. The dataset now contains 2,162,655 messages in total.

Departure messages

The KV6 dataset also has missing departure messages. In contrast to the missing arrival messages, there is no good explanation for these messages to be missing. This likely happens when there is a malfunction in the computer that tracks the vehicle's location. When there is a missing departure message, there is an arrival message for that stop. This means that the bus did visit the stop. The same solution is applied to the missing departure messages, meaning copying the data of the unmatched arrival message and imputing it as a departure message. This means that these stops are assumed to be skipped stops with a dwell time of zero seconds. This is mainly done to retain as many journeys in the dataset for training as possible. The process of departure message imputation is illustrated in Table 4.6b.

Table 4.6 – Visualisation of an example of a departure message imputation. This shows an arbitrary subpart of a bus journey in the case study dataset.

(a) Unmatched arrival message in journey.

Journey	Messagetype	Stop	Timestamp
⋮	⋮	⋮	⋮
QBuzz, line 1, journey 30	Arrival	10	2024-09-04 14:11:55
QBuzz, line 1, journey 30	Departure	10	2024-09-04 14:12:24
QBuzz, line 1, journey 30	Arrival	11	2024-09-04 14:13:34
QBuzz, line 1, journey 30	Arrival	12	2024-09-04 14:14:04
QBuzz, line 1, journey 30	Departure	12	2024-09-04 14:14:38
⋮	⋮	⋮	⋮

(b) Imputed departure message with the same journey information, stop and timestamp as the unmatched departure message.

Journey	Messagetype	Stop	Timestamp
⋮	⋮	⋮	⋮
QBuzz, line 1, journey 30	Arrival	10	2024-09-04 14:11:55
QBuzz, line 1, journey 30	Departure	10	2024-09-04 14:12:24
QBuzz, line 1, journey 30	Arrival	11	2024-09-04 14:13:34
QBuzz, line 1, journey 30	Departure	11	2024-09-04 14:13:34
QBuzz, line 1, journey 30	Arrival	12	2024-09-04 14:14:04
QBuzz, line 1, journey 30	Departure	12	2024-09-04 14:14:38
⋮	⋮	⋮	⋮

In the KV6 dataset of the case study, only 49 departure messages were imputed. These imputations ensured that for complete journeys, there is a departure message for each stop. For these 49 messages, the dwell time will be calculated as zero.

4.6.2 KV6 duplicate messages

In the KV6 dataset, duplicate messages frequently occur, which can cause issues when calculating travel and dwell times. For example, if there are duplicate departure messages, a decision must be made on which message to use for calculating the difference between the timestamps of the arrival message. The following rules have been set up for the removal of duplicate messages.

- Two messages are deemed duplicates when they have the same dataownercode, operatingday, lineplanningnumber, userstopcode, messagetype and reinforcement number. The timestamp does not have to be identical.
- For the message types 'init' and 'arrival,' the message with the earliest timestamp is retained, and duplicates are removed from the dataset. The first message is kept because,

for 'init' and 'arrival,' it is logical to assume that the system correctly detected the event at that first time.

- For the message types 'departure' and 'end,' the message with the latest timestamp is retained, and duplicates are removed from the dataset. This approach ensures that the system's final message confirms the bus's departure from the stop, indicating that the bus left the stop at that specific time.

The process following the rules above is demonstrated in Table 4.7.

Table 4.7 – Demonstration of an example of handling duplicate messages. In this table, the red messages are removed from the KV6 dataset.

(a) Example of duplicate handling of arrival messages. For arrival messages, the message with the earliest timestamp is kept. Here, the bus has a double arrival message for stop 3 along this route. This process is similar to init messages.

Journey	Messagetype	Stop	Timestamp
⋮	⋮	⋮	⋮
QBuzz, line 1, journey 12	Departure	2	2024-09-15 09:48:38
QBuzz, line 1, journey 12	Arrival	3	2024-09-15 09:49:51
QBuzz, line 1, journey 12	Arrival	3	2024-09-15 09:49:54
QBuzz, line 1, journey 12	Departure	3	2024-09-15 09:50:31
⋮	⋮	⋮	⋮

(b) Example of duplicate handling of departure messages, where there is a duplicate departure message for stop 14. For departure messages, the message with the latest timestamp is kept. This process is similar to end messages.

Journey	Messagetype	Stop	Timestamp
⋮	⋮	⋮	⋮
QBuzz, line 1, journey 3	Arrival	14	2024-09-23 06:07:01
QBuzz, line 1, journey 3	Departure	14	2024-09-23 06:07:41
QBuzz, line 1, journey 3	Departure	14	2024-09-23 06:07:43
QBuzz, line 1, journey 3	Arrival	15	2024-09-23 06:09:21
⋮	⋮	⋮	⋮

In the case study dataset, 153,590 messages were removed due to being duplicates.

4.6.3 Schedule adherence

For the end goal of calculating all the travel and dwell times of a bus journey, there are several requirements. Two of the requirements, namely no missing messages and no duplicate messages, are discussed in the Sections 4.6.1 and 4.6.2. Other issues arise when certain stops are not accounted for in the messages; this means that you cannot correctly calculate the travel times between the preceding and following stops. Also, there is no information on the dwell time of this stop. This issue is solved by the schedule adherence check. This is the final preprocessing step before the creation of the data format described in Section 3.2.

Schedule adherence is done by checking if every stop is present for a certain journey. That is to say, $\mathbf{k}_j$ is equal to the message sequence for that journey and what the schedule prescribes it to be. This is done with the GTFS schedule of the same journey. By matching the journey with the corresponding `service_id` and `realtime_trip_id` in the GTFS schedule. This combination can be matched with a `trip_id` and its corresponding schedule stop times in `stop_times.txt`. The rules for checking if the stop order is identical are formulated below.

- KV6 journey stop order $\mathbf{k}_j$ should be identical to the GTFS schedule stop order $\mathbf{k}_{j'}$ of the corresponding journey.
- For every stop along the journey, the arrival message's timestamp should precede the departure message's timestamp.
- Missing init or end messages for the first and last stop of the journey, respectively, can be ignored, and the dwell time of these stops is set at zero.
- KV6 journeys that do not adhere to the schedule are deleted from the KV6 dataset.

When applying these rules to the case study dataset, two main reasons for removal were identified. Firstly, there were 9554 journeys where several stops were not visited at all in the messages. Most of these journeys only included an 'init' message and lacked the information necessary for the prediction models. Additionally, journeys that missed one or two stops compared to the schedule were also removed. Although this led to the deletion of useful information, the decision was made for practical efficiency.

Secondly, 3196 journeys were removed because their stop order did not match the schedule. In these cases, all necessary stops were present in the messages, but were not in the correct order. This issue was most prevalent for line g504, due to a temporary deviation from the schedule. Consequently, valuable journeys were removed.

While these removals were necessary to advance the preprocessing, there is potential for future improvements. Refining the criteria for journey inclusion and developing methods to handle deviations more effectively could preserve more useful data and enhance the accuracy of the prediction models.

4.6.4 Data format creation

The three previous sections discuss message imputation, duplicate handling and schedule adherence of the KV6 messages dataset. At this stage, the KV6 messages dataset is transformed into a dataset with travel and dwell times per link and stop. This is done by looping over the messages of the journeys and applying Equations 3.1 and 3.2. The time series index is taken as the departure time from the first stop of the journey. This process created the dataset that will form the basis for the training data of the prediction models.

This has been done for the case study KV6 dataset, which initially contained 32094 journeys. 19344 complete journeys could be constructed from this dataset. This concludes the sections that describe the relevant preprocessing of training data for all prediction models. The next sections will discuss model-specific preprocessing steps.

4.6.5 Outliers handling

Outlier winsorization is a technique used to limit extreme values in a dataset to reduce the impact of outliers. In this method, we identify outliers using z-scores and replace them with a specified percentile value. Equation 4.1 states the definition of the z-score. The z-score measures how many standard deviations a data point is from the mean.

$$z = \frac{x - \mu}{\sigma} \tag{4.1}$$

where x is the data point, μ is the mean and σ is the standard deviation. Data points with z-scores higher than three are considered outliers. These data points are replaced with the 95th percentile value of the corresponding travel or dwell time. The 95th percentile is the value below which 95% of the data points fall. Low outliers were not removed, as these were often the zero values of dwell times, meaning they are significant for the analysis of the behaviour of the bus. Outliers were removed for the VAR and LSTM prediction models.

4.6.6 Periodic resampling

Prediction models such as VAR and RF require periodic data. The time series data is indexed by the departure time from the first stop of the journey, which, because of the schedule, means that it is approximately periodic (e.g. a bus departs from the first stop every 15 minutes). However, departure times of the bus can differ by a small amount, thus making it not periodic. This problem can be fixed by periodic resampling of the dataset. Another issue fixed with this preprocessing step is bus journeys that were removed due to not adhering to the schedule. Resampling imputes these removed journeys with the average travel and dwell times of the journeys before and after. This process is demonstrated in Table 4.8b.

Table 4.8 – Demonstration of periodic resampling, where multiple journeys falling in a certain time interval are combined to a single entry with the averaged travel and dwell times.

(a) The entries are not periodic. The bus seemingly leaves every 30 minutes, but in reality this will always deviate.

Index	y_{k_1}	y_{l_1}	y_{k_2}	y_{l_2}	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\iddots$
2024-09-11 17:14:31	32	79	0	62	$\cdots$
2024-09-11 17:45:21	41	71	54	93	$\cdots$
2024-09-11 18:16:54	0	95	41	88	$\cdots$
2024-09-11 18:42:58	47	78	0	71	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

(b) The data is resampled to be periodic. In this case, the resampling period is one hour. The average values that fall in that hourly interval are taken from the previous table to calculate the new values for the dwell and travel times. This means that these four journeys are now represented as two journeys.

Index	y_{k_1}	y_{l_1}	y_{k_2}	y_{l_2}	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\iddots$
2024-09-11 17:00:00	36.5	75	27	77.5	$\cdots$
2024-09-11 18:00:00	23.5	86.5	20.5	79.5	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

4.6.7 Lagged features

RF is not inherently designed to forecast time series data. Consequently, it requires that the data be enriched with lagged features to provide the prediction models with information about the past. Lagged features are variables that represent past values of the time series. For example, a lagged feature could be the value of the travel time of the previous hour. The number of past time steps (lags) to include needs to be decided based on the patterns that are present in the data. Creating lagged features is as simple as shifting the dataset for that feature. After shifting the dataset, there will be missing values at the beginning of the time series. These rows are removed from the training data. This process of creating lagged features is shown in Table 4.9b.

Table 4.9 – Example of the creation of lagged features, where each row gets a lagged copy of the previous time sample variables. This is done for a dataset that already underwent periodic sampling.

(a) Initial dataset without lagged features.

Index	y_{k_1}	y_{l_1}	$\cdots$
$\vdots$	$\vdots$	$\vdots$	⋰
2024-09-28 12:00:00	28	73	$\cdots$
2024-09-28 13:00:00	36.5	75	$\cdots$
2024-09-28 14:00:00	23.5	86.5	$\cdots$
2024-09-28 15:00:00	30.5	70	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\ddots$

(b) For all travel and dwell times of the dataset lagged features are created. In this case, the lag is equal to 1. However, more lagged features might be needed to capture patterns from the past effectively.

Index	y_{k_1}	$y_{k_1}(lag = 1)$	y_{l_1}	$y_{l_1}(lag = 1)$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	⋰
2024-09-28 12:00:00	28	34	73	70.5	$\cdots$
2024-09-28 13:00:00	36.5	28	75	73	$\cdots$
2024-09-28 14:00:00	23.5	36.5	86.5	75	$\cdots$
2024-09-28 15:00:00	30.5	23.5	70	86.5	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

4.6.8 Time features

Besides creating lagged features, another way to provide more information to the prediction models is to encode the time index in so-called time features. These features can function as the exogenous features of a VAR model or be used by ML algorithms such as RF and LSTM.

The time features are computed by encoding the datetime index as an integer. For hours, this is achieved by mapping each hour of the day to an integer in the range $0, \ldots, 23$. Similarly, for days of the week, the mapping assigns integers $0, \ldots, 6$ to Monday through Sunday.

Sinusoidal transformations are performed to capture the periodic behaviour of hours of the day and days of the week. For example, this ensures that hour 0 and hour 23, which appear far apart but are actually adjacent, are encoded with values that are near each other. It has been shown that these enable ML models to capture temporal patterns more easily [53]. Equations 4.2-4.5 have been used to encode the hour of the day and day of the week for this case study.

$$hour\_sin = \sin(2\pi\frac{h}{24}) \tag{4.2}$$

$$hour\_cos = \cos(2\pi\frac{h}{24}) \tag{4.3}$$

$$day\_sin = \sin(2\pi\frac{d}{7}) \tag{4.4}$$

$$day\_cos = \cos(2\pi\frac{d}{7}) \tag{4.5}$$

Where h denotes the encoded hour of the day and d is the day of the week. An example of time feature engineering can be seen in Table 4.10.

Table 4.10 – Example of the sinusoidal time features. These features are calculated by first encoding the hour of the index to an integer $0, \dots, 23$ and the day to an integer $0, \dots, 6$. Then, using equations 4.2-4.5 to calculate the values displayed in this table.

Index	**Hour_sin**	**Hour_cos**	**Day_sin**	**Day_cos**
⋮	⋮	⋮	⋮	⋮
2024-10-02 13:00:00	-0.259	-0.966	0.975	-0.223
2024-10-02 14:00:00	-0.500	-0.866	0.975	-0.223
⋮	⋮	⋮	⋮	⋮
2024-10-05 07:00:00	0.966	-0.259	-0.975	-0.223
2024-10-05 08:00:00	0.866	-0.500	-0.975	-0.223
⋮	⋮	⋮	⋮	⋮

4.6.9 Feature scaling

Feature scaling is a crucial preprocessing step for LSTM, as these models are sensitive to the scale of the input data. The method commonly used for scaling is the `MinMaxScaler`, which transforms data into the range $[0, 1]$. This ensures that all input features contribute equally to the learning process. When data is not scaled, some features with large values might have a greater impact on the learning process. This will lead to issues like slow convergence or poor generalisation. Equation 4.6 shows the equation used for scaling the input features of the training data.

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{4.6}$$

Where:

- X is the original feature.
- X_{min} and X_{max} are the minimum and maximum value of the feature.
- X_{scaled} is the transformed feature with a value in the range $[0, 1]$.

4.6.10 Sequence creation

The raw data format must be converted into meaningful input-output sequences for the LSTM model. This is crucial for this model to capture temporal dependencies. The input data must be structured as a 3d array with the shape `(samples, timesteps (input), features (input)`. The input has an associated 3D target array with the shape `(samples, timesteps (target), features (target)`

- **Samples:** represents the number of training examples. The number of samples is the same for the input and target arrays.
- **Timesteps (input):** denote the number of time steps in each input sequence. These are the number of timesteps a prediction is based on.
- **Features (input):** indicates the number of variables in each time step of the input.
- **Timesteps (target):** denotes the number of time steps in each output sequence. This is how many timesteps in the feature are predicted.
- **Features (target):** indicates the number of variables in each time step of the target.

4.6.11 Summary of preprocessing of training data

The previous sections presented in detail the preprocessing steps that were undertaken for the creation of the training data. The first four sections apply to creating the data format and this is identical for all the prediction models that are used in this research. The sections after this are model-specific preprocessing steps. This information is summarised in the Table 4.11. Upon completing these steps, the historical travel data from the KV6 dataset will be prepared for the prediction models.

Table 4.11 – This table outlines the necessary preprocessing steps for each prediction model. The steps are listed in the order they should be applied for each respective model.

Preprocessing technique	HA	VAR	RF	LSTM
KV6 message imputation 4.6.1	✓	✓	✓	✓
KV6 duplicate messages 4.6.2	✓	✓	✓	✓
Schedule adherence 4.6.3	✓	✓	✓	✓
Data format creation 4.6.4	✓	✓	✓	✓
Outlier handling 4.6.5		✓		✓
Resampling 4.6.6		✓	✓	✓
Lagged features 4.6.7			✓	
Time features 4.6.8			✓	✓
Feature scaling 4.6.9				✓
Sequence creation 4.6.10				✓

4.6.12 GTFS preprocessing

The previous sections discussed the preprocessing steps necessary for the KV6 training data. As explained in Section 4.5, the GTFS schedule will be overwritten by the output of the prediction models.

The following steps must be completed:

1. Every KV6 journey must be matched with the correct trip in the GTFS schedule. (i.e. matching `trip_id`.
2. Arrival and departure times in `stop_times.txt` are in string format and must be converted to a timestamp.
3. For each predicted journey with corresponding `trip_id`, the arrival and departure times are overwritten. This process is shown in Table 4.12b.
4. The new `stop_times.txt` file can be integrated into the complete dataset and uploaded as a zip folder to Conveyal.

Upon completing these steps, the GTFS dataset is ready to serve as input data for the reachability analysis in Conveyal.

Table 4.12 – The process of overwriting the GTFS schedule with the output of a prediction model. This is done for every journey that is predicted in the dataset.

(a) The original schedule for a journey in `stop_times.txt`.

Stop	Arrival time	Departure time
k_1	at_{k_1}	dt_{k_1}
k_2	at_{k_2}	dt_{k_2}
k_3	at_{k_3}	dt_{k_3}
k_4	at_{k_4}	dt_{k_4}
$\vdots$	$\vdots$	$\vdots$

(b) The new schedule in `stop_times.txt` with the predictions $\mathbf{\hat{y}_k}$ and $\mathbf{\hat{y}_l}$.

Stop	Arrival time	Departure time
k_1	at_{k_1}	dt_{k_1}
k_2	$\hat{at}_{k_2} = dt_{k_1} + \hat{y}_{l_1}$	$\hat{dt}_{k_2} = \hat{at}_{k_2} + \hat{y}_{k_2}$
k_3	$\hat{at}_{k_3} = \hat{dt}_{k_2} + \hat{y}_{l_2}$	$\hat{dt}_{k_3} = \hat{at}_{k_3} + \hat{y}_{k_3}$
k_4	$\hat{at}_{k_4} = \hat{dt}_{k_3} + \hat{y}_{l_3}$	$\hat{dt}_{k_4} = \hat{at}_{k_4} + \hat{y}_{k_4}$
$\vdots$	$\vdots$	$\vdots$

5 Results

Section 3 sets up the Methodology for the design of the four prediction models and reachability analysis. The results of developing these models and conducting the reachability analysis are presented in this section. Section 4 described the case study and preprocessing steps of the datasets that will be used for the development of these models.

The results section aims to answer the final two research questions RQ3 and RQ4. Answering RQ3 means assessing the accuracy of the four models predicting future travel and dwell times. Besides accuracy, it is also important to investigate the factors influencing the decision-making of the models to unveil patterns in the data. To answer RQ4, the reachability analysis is conducted with the predicted travel and dwell times. The goal is to investigate the implications of the more accurate representation of the reachability of the PT network.

The results section begins with Section 5.1 presenting the outcomes of the EDA. The results of HA (5.2), VAR (5.3), RF (5.4) and LSTM (5.5) are discussed. Section 5.6 summarises and compares the four prediction models. Finally, Section 5.7 discusses the reachability analysis, which incorporates the predictions of the best-performing prediction model.

5.1 Exploratory data analysis

Having a preliminary understanding of underlying patterns in the data is crucial before developing prediction models. This understanding aids in evaluating the models and explaining their outcomes. The EDA consists of four parts: rush hours, day of the week, outliers, and skipped stops. This selection is based on domain knowledge and insights gained from preliminary investigations.

Section 5.1.1 explores the patterns in travel and dwell times during rush hours. Section 5.1.2 examines how weekdays and weekends affect journey duration. Section 5.1.3 investigates the presence of outliers in the data. Finally, Section 5.1.4 outlines the occurrences of skipped stops.

5.1.1 Rush hours

During rush hours, both travel and dwell times are expected to increase. The analysis conducted in this section will explore not only rush hours but also general behaviour during other parts of the day. There are two peak periods during the day: the morning rush from 07:00 to 09:00 and the evening rush from 16:30 to 18:30. When visualising the cumulative travel time, it becomes evident that a significant number of journeys during these periods take longer than those outside of rush hours. This trend is illustrated for line g506 in direction 1, in Figure 5.1. In this figure, it can be observed that the red rush hour journeys tend to be higher than the blue non-rush hour journeys. This means that these journeys are generally slower. In this plot, it can also be observed that some blue journeys are high, meaning that slow journeys can happen outside rush hours.

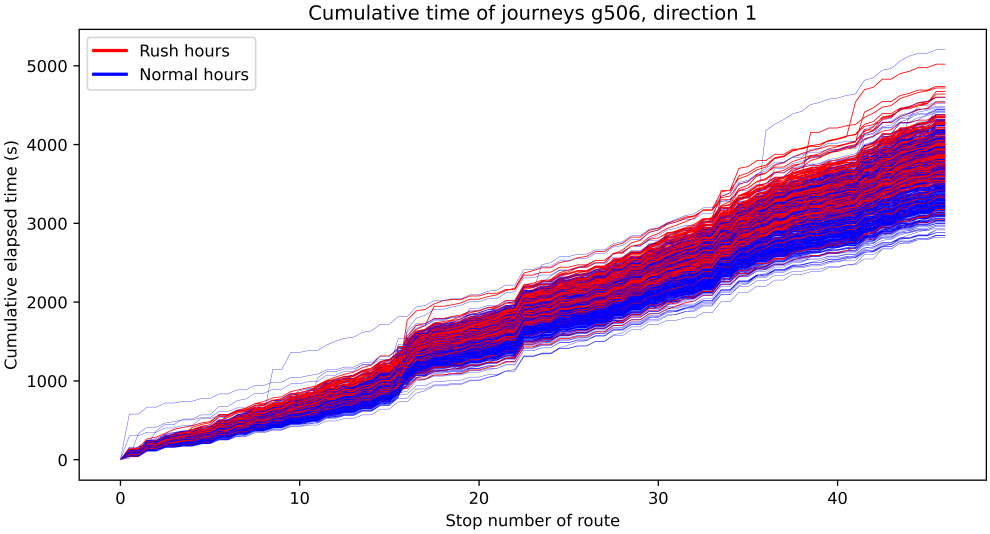
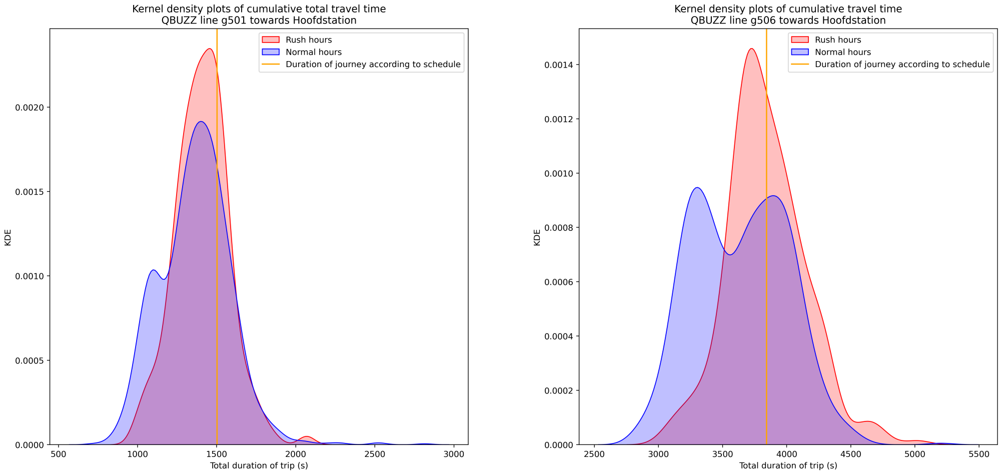


Figure 5.1 – The cumulative travel time is measured from the first stop of the route. The x-axis represents the stop number relative to the starting point, while the y-axis shows the cumulative travel time taken to reach each stop from the first stop.

When plotting the Kernel Density Estimation (KDE) of the total travel time of the journey (i.e. the travel time from the first stop to the last stop of the journey) with the same definition of rush hours. It can be observed that journeys during rush hour exhibit a different pattern. Figure 5.2 shows that the non-rush hour journeys tend to be faster than the journeys during rush hours. However, in both KDE plots, it can be observed that slow journeys happen during non-rush hours, and not every rush hour journey is slow.

Also, in Figure 5.2, the orange vertical line displays the time the bus journeys should take according to the schedule. The majority of the mass of the KDEs, for both rush hours and non-rush hours, is to the left of this line. This means that the schedule is always conservative on the time the journeys take.



(a) G501, direction 1

(b) G506, direction 2

Figure 5.2 – KDE visualisation of total travel times of the bus. On the x-axis is the total duration of the journey in seconds. The y-axis displays the distribution.

To investigate more closely the travel and dwell time behaviour per hour, the z-score is calculated. This measures how many standard deviations a data point is from the mean of a dataset. It helps to understand the position of a value within a distribution. The definition is given in Equation 4.1, but is also displayed below.

$$z = \frac{x - \mu}{\sigma} \tag{5.1}$$

Figure 5.3 illustrates the average z-score of travel and dwell times for each hour of the day. A high z-score indicates slower journeys, while a low z-score indicates faster journeys. The distribution of these averages is represented by boxplots, which are displayed side-by-side in Figure 5.3.

Figure 5.3 visualises the travel and dwell times for line g501 in direction 1. The boxplots show that daytime values are generally higher than those for the evening and early morning, likely due to increased traffic during the day. Notably, the boxplot for 8:00 is higher than the others, indicating that the slowest journeys occur during this time. However, a similar peak is not observed for the afternoon rush hour, suggesting that journeys throughout the daytime are consistently slow. The fastest journeys seem to occur in the time interval of 6:00 and 7:00.

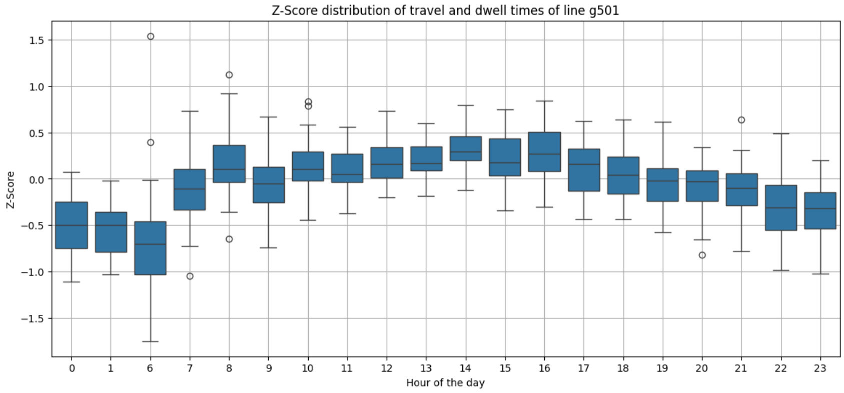


Figure 5.3 – Boxplots per hour of the average z-score of the travel and dwell times of line g501 in direction 1. The higher the boxplot is, the slower the journeys are for that hour; the lower the boxplot is, the faster the journeys are.

In conclusion, the duration of journeys, along with their travel and dwell times, is influenced by the time of day. This suggests that incorporating time features, such as the hour of the day, into the prediction models will be beneficial.

5.1.2 Weekdays

In the same manner that Figure 5.3 is constructed in Section 5.1.1, the impact of the day of the week is analysed. This can be seen for line g501 in direction 2 in Figure 5.4.

Figure 5.4 shows that the boxplots for Monday through Thursday are quite similar. Friday's boxplot is higher, indicating that the slowest journeys of the week occur on that day. Saturday's boxplot is more stretched, suggesting that bus journeys can be either fast or quite slow. The boxplot for Sunday has the lowest z-score distribution, indicating that the fastest journeys of the week occur on that day, likely due to decreased ridership during the weekend.

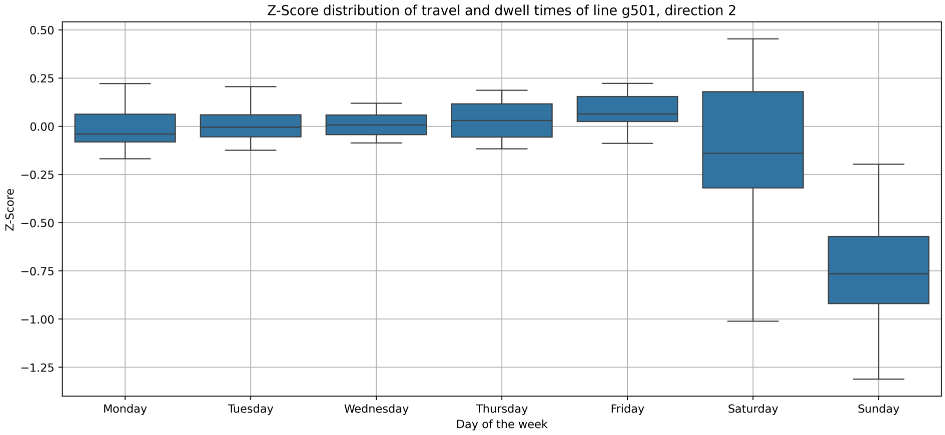


Figure 5.4 – Boxplots per weekday of the average Z-score of the travel and dwell times of line g501 in direction 1.

The EDA results in Sections 5.1.1 and 5.1.2 indicate that the time of day and the day of the week significantly impact overall journey performance. For instance, travel and dwell times are higher during the day and midweek. Therefore, incorporating this information into prediction models through time feature engineering can yield positive prediction results. The process of time feature engineering is detailed in Section 4.6.8.

5.1.3 Outliers

During the initial data analysis, numerous outliers were identified, stemming from various causes. These outliers can be categorised into two main groups.

1. **Real-world incidents:** These include genuine disruptions such as traffic incidents, delays caused by rush hours and extended dwell times due to unforeseen circumstances.
2. **Data-related issues:** Errors arising from data collection faults or systems malfunction. This was evident during the preprocessing of the KV6 messages, where duplicate and missing messages were frequently observed.

A common threshold for defining data points as outliers is having a z-score $|Z| > 3$. Z-score z is defined as

$$z = \frac{x - \mu}{\sigma} \tag{5.2}$$

Where x is the travel or dwell time, μ and σ are the mean and standard deviation, respectively, of the travel time between two stops or dwell time at a certain stop. The percentage of data points that can be considered outliers is displayed in Table 5.1.

Table 5.1 – The percentage of travel and dwell times that are considered outliers when defining outliers of having a $|Z| > 3$.

Line	Direction 1	Direction 2
g501	1.171	1.256
g502	1.516	1.375
g503	1.213	1.176
g504	1.334	1.273
g505	1.503	1.849
g506	1.393	1.389

Another way to visualise outliers is by using boxplots. Outliers are common across all features for the various bus lines and directions. Figure 5.5 illustrates the spread of values for four features of bus line g504 in direction 1. The dots outside the boxplot are outliers, which can be severe for some travel and dwell times, as shown in the figure. Outliers in a boxplot are data points that fall outside the whiskers, which extend to 1.5 times the InterQuartile Range from the first and third quartiles (Q1 and Q3). Specifically, any data point below $Q1 - 1.5 \cdot IQR$ or above $Q3 + 1.5 \cdot IQR$ is considered an outlier. These outliers are represented as individual dots in the plot.

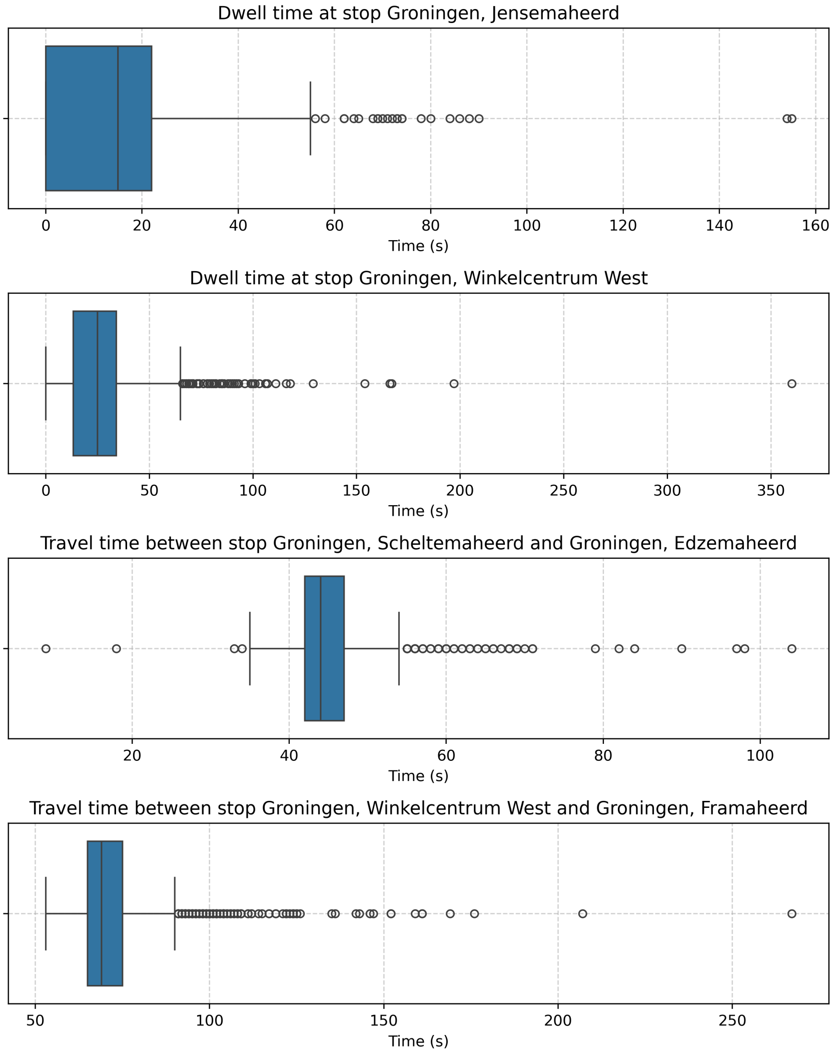


Figure 5.5 – Singular boxplots of two travel times and two dwell times of line g504 in direction 1. The x-axes show the times in seconds.

Figure 5.5 serves solely as an example of outliers. However, these kinds of extreme outliers are present in the travel time data for all directions and lines. The existence of outliers can negatively impact the predictive power of models such as HA, VAR and LSTM. RF is more robust to outliers.

5.1.4 Skipped stops

Skipped stops are represented in the data as dwell time equal to zero. Such zero values can have a great impact on the performance of prediction models. For example, it can heavily skew the average dwell time of a certain stop. Therefore, it is necessary to analyse and investigate the occurrences of skipped stops. Also, other ML learning models will have difficulties with predicting zero values.

In Figure 5.6, it is evident that the frequency of skipped stops varies depending on the time of day. Fewer stops are skipped during the day, while more stops are skipped in the evening and early morning. This pattern can be explained by the varying number of public transport users at different times of the day.

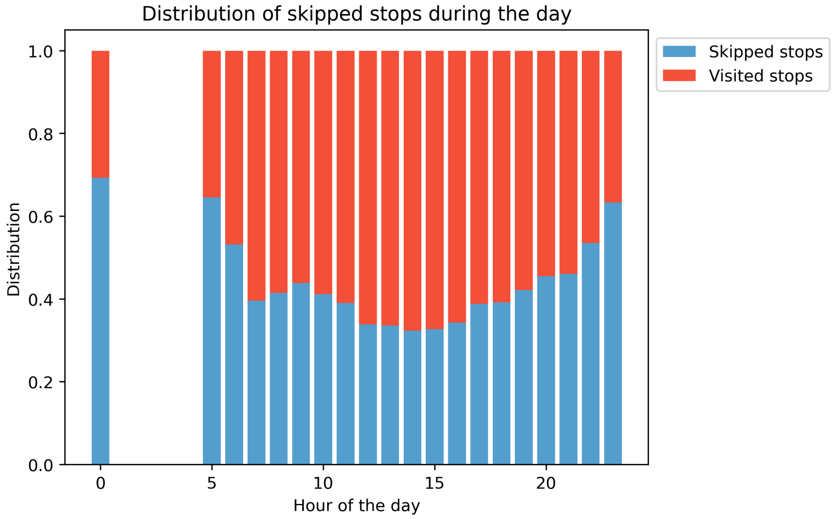


Figure 5.6 – Distribution of skipped stops vs visited stop per hour for line g504, direction 1.

Figure 5.7 displays that not every stop is skipped equally. Some stops will be relatively easy to predict, as they are skipped roughly 80 percent of the time.

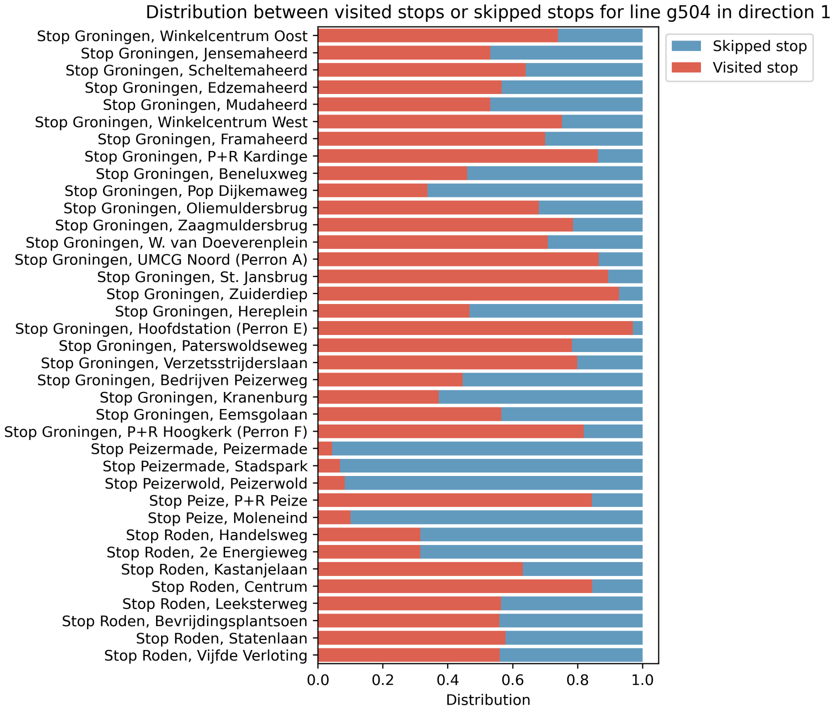


Figure 5.7 – Distribution of skipped stop vs visited stop per stop for line g504, direction 1.

To conclude the EDA section, this overview highlights key patterns in the travel and dwell time data used for training prediction models. Firstly, travel and dwell times heavily depend on when the journey takes place. Secondly, the data contains many outliers, which must be handled effectively for good prediction performance. Finally, there are identifiable patterns in skipped stops, suggesting that prediction models could efficiently predict zero values for dwell times. These conclusions will aid in evaluating the prediction models.

5.2 Historical average

This section presents the results of the baseline HA prediction model. The methodology outlined in Section 3.5 is applied to the travel time dataset of the Groningen bus network. This is the first step in predicting travel and dwell times and therefore addresses RQ3. The primary goal is to establish a baseline for prediction evaluation and outline expectations for more advanced models.

As explained in Section 3.5, the HA model predicts future travel and dwell times by averaging the corresponding times from past journeys. This prediction will be evaluated against a test dataset, yielding an MAE result. After this first approach, a more advanced time-dependent HA model will be developed. This section will also present key insights and evaluations.

Section 5.2.1 explains the process of averaging travel and dwell times to get the prediction value. This initial version's MAE results are evaluated in Section 5.2.2. Section 5.2.3 will introduce the results of the time-dependent HA model. Finally, Section 5.2.4 will offer concluding remarks on the HA models.

5.2.1 Averaging dwell and travel times

The data format presented in Table 3.2 enables us to take the averages of the columns to get the average travel and dwell times. These will be the predictions $\hat{\mathbf{y}}_k$ and $\hat{\mathbf{y}}_l$ calculated using Equations 3.3 and 3.4. This is done for all 6 lines in both directions. Figure 5.8 visualises the cumulative predicted journey for line g506 in direction 1. The blue lines are all the cumulative journeys on which this average journey is based. This figure shows how the prediction reflects the average journey of the dataset.

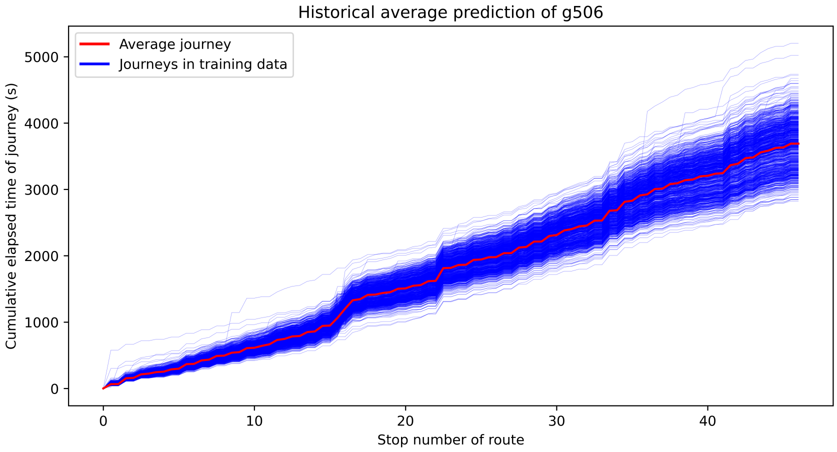


Figure 5.8 – Average journey that is used for the prediction of future journeys visualised between the training journeys on which it is based.

5.2.2 Evaluation

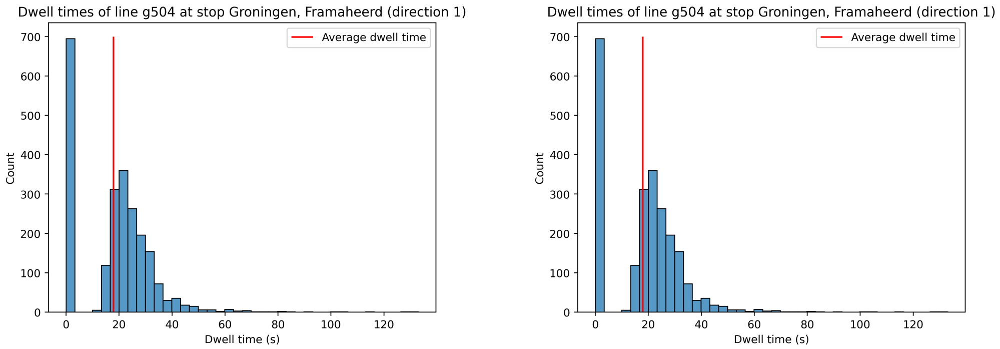
To evaluate the performance of the HA model, the MAE will be calculated for all lines and directions. MAE is calculated by averaging the training dataset and evaluating it against the test dataset. The training data consists of the first 80% of journeys, while the test dataset comprises the most recent 20% of journeys. This is done for both directions of every line of the dataset. The results are shown in Table 5.2.

Table 5.2 – MAE results of the HA model. Overall, MAE is evaluated on dwell and travel times. Dwell MAE and travel MAE are evaluated on only dwell and travel times, respectively.

Bus route	Overall MAE	Dwell MAE	Travel MAE
g501, 1	11.344	12.285	10.495
g501, 2	11.533	13.473	9.739
g502, 1	13.628	13.206	14.000
g502, 2	12.563	10.699	14.427
g503, 1	11.527	13.827	9.226
g503, 2	10.860	12.823	8.948
g504, 1	11.485	13.235	9.782
g504, 2	15.103	17.346	12.765
g505, 1	14.857	14.023	15.665
g505, 2	12.563	10.699	14.427
g506, 1	10.407	11.701	9.197
g506, 2	10.396	11.733	9.116

In Table 5.2, it can be observed that overall MAE is similar for all lines when comparing direction 1 and direction 2. G506 has the best MAE results, and G502 has the worst. Comparing MAE results between lines, however, is not meaningful. Table 4.1 shows that there is a large difference between the length and number of stops of each bus line. Increasing the size of the bus line will introduce variability and randomness and therefore increase the MAE. Comparing MAE is only justifiable between the different prediction models trained on the same dataset, which is the purpose of this baseline HA model.

Overall, the predictive performance of the dwell times is lower, primarily due to the presence of zero values in the data from skipped stops. These zero values skew the distribution. In Figure 5.9, the impact of these zero values on the predicted mean dwell time can be observed. The histograms show that the average dwell time does not represent the non-zero distribution well, and it can not predict any zero values (i.e. skipped stops) for future journeys.



(a) Dwell time at stop Framaheerd along bus line g504 in direction 1.

(b) Dwell time at stop Hereplein along bus line g501 in direction 1.

Figure 5.9 – In these dwell time histograms, the distribution is not well represented by the average value due to the presence of zero values. This also means that the baseline HA model cannot predict skipped stops for future journeys.

When plotting the MAE of the travel times on a map of Groningen, it can be observed that the links in the city centre have the highest MAE. This means that taking the average for links in these dense urban regions is less predictive than for regions outside of the city. This can be seen for line g501 in Figure 5.10. For this route specifically, the first link from Hoofdstation to Hereplein has the highest MAE when tested against the test dataset.

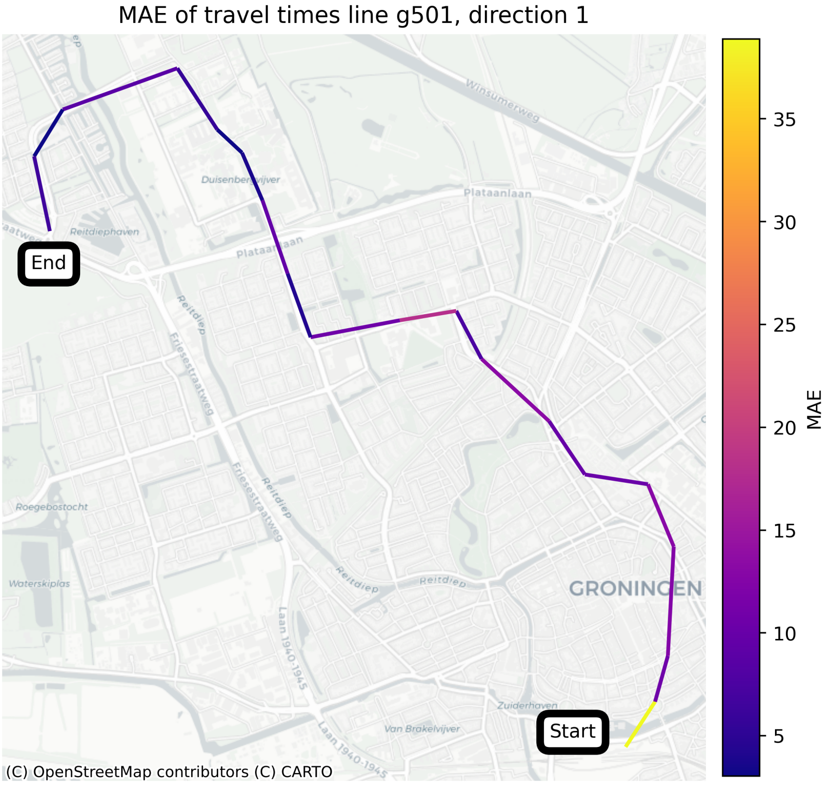

MAE of travel times line g501, direction 1
End
Start
GRONINGEN
(C) OpenStreetMap contributors (C) CARTO
MAE
5
10
15
20
25
30
35

Figure 5.10 – MAE of travel time of the links along line g501 in direction 1 for the HA model. The MAE of the of the first link (Hoofdstation to Hereplein) is especially high.

The same plot has been made for the line g503 in direction 1, which is shown in Figure 5.11. In this figure, it can be observed that the MAE increases around the city centre of Groningen. The travel times near the centre are more variable, thus making it more difficult to predict. The links in the starting town of Leek and the end in Groningen have relatively low MAE values. Also, the long link on the highway between Midwolde and Hoogkerk has a high MAE. This is the longest link, and according to the schedule, this link takes around 9 minutes to travel. This will also inflate the MAE, compared to the shorter links.

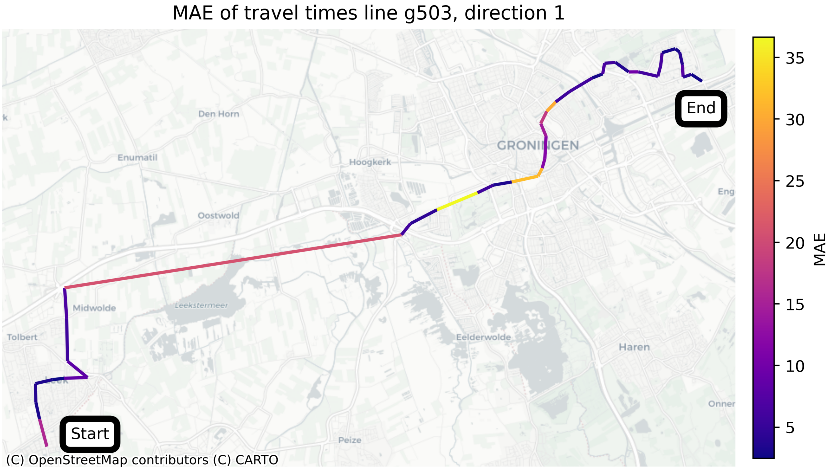


Figure 5.11 – MAE of travel time of different links along g503 in direction 1 for the HA model.

The MAE travel time maps of the lines and directions is displayed in Appendix B.1.

5.2.3 Time-dependent historical average

A more advanced HA model is the time-dependent HA model. In this model, the predictions $\hat{\mathbf{y}}_{k,t}$ and $\hat{\mathbf{y}}_{l,t}$ are based on the average of journeys within a specific time of day. These are calculated using Equations 3.5 and 3.6 for all 6 lines in both directions. The EDA (Section 5.1.1) suggests that travel and dwell times heavily depend on the time of day the journey occurs. The time-specific HA model aims to exploit this pattern.

The first step is investigating the optimal size of the time interval. This is done by computing the MAE results for each time interval in the same manner that the ordinary HA model is evaluated. The results of this analysis are displayed in Table 5.3.

Table 5.3 – The overall MAE results of different time intervals for the time-dependent HA model. For each line and direction, the lowest MAE is bold.

Direction	30 min	60 min	120 min	180 min
g501, 1	10.177	**10.165**	10.293	10.447
g501, 2	**10.467**	13.345	13.426	13.534
g502, 1	**12.432**	14.105	14.177	14.222
g502, 2	**12.778**	13.387	13.721	13.698
g503, 1	10.781	**10.722**	10.774	10.772
g503, 2	10.928	10.213	**10.195**	10.198
g504, 1	**10.746**	11.448	11.496	11.548
g504, 2	15.492	15.148	15.126	**15.118**
g505, 1	14.102	15.720	15.697	**11.822**
g505, 2	**12.329**	12.418	12.365	12.358
g506, 1	9.652	**9.530**	9.533	9.547
g506, 2	**9.989**	10.636	10.666	10.678

In Table 5.3, it is shown that overall, the best MAE results are achieved with the smallest time window of 30 minutes. This is the case for 6 out of the 12 lines in both directions, which is the most of all time intervals. For uniformity, this time interval is selected for all lines and directions for the analysis of the time-dependent HA model. Reducing this 30-minute window will be nonsensical for most lines, as there is a bus every half hour. This means an even smaller window will not differ from the 30-minute time interval. This would be a form of overfitting as the model memorises the training data so closely that it will not be able to generalise to new data. The model might start to capture irrelevant variations or noise, rather than any consistent patterns present for the 30-minute interval.

Figure 5.12 shows several examples of 30-minute averages for two predicted travel times and two predicted dwell times. It can be seen that there is a clear peak for these features around the morning rush hour. This means that a higher travel and dwell time will be predicted around this time. This is in agreement with what is expected from the EDA.

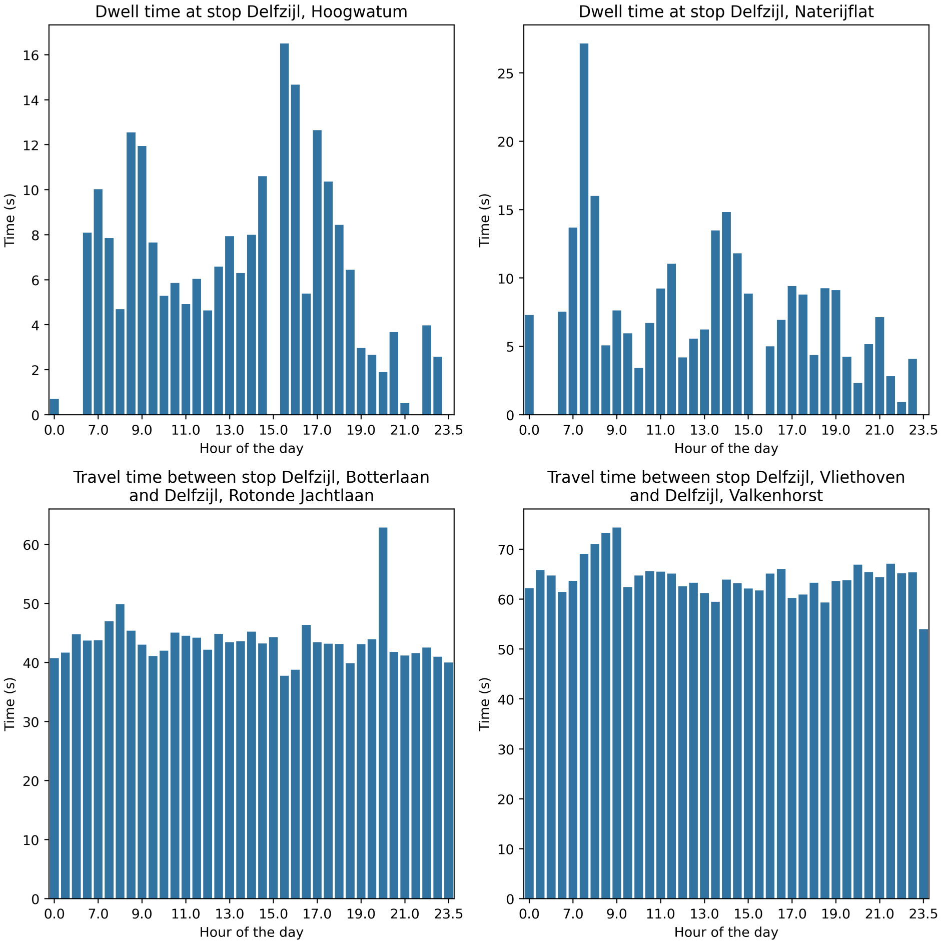


Figure 5.12 – Examples of travel and dwell times averages per 30 minutes for line g506 in direction 1.

The 30-minute MAE in Table 5.3 is the same as the overall MAE column in Table 5.4. In addition, this table also displays the dwell and travel MAEs. The 30-minute HA model improves on every line and direction compared to the ordinary HA model. This can be seen by the overall MAE, which is lower for the 30-minute HA model. Also, for the 30-minute HA model, the dwell MAE is higher than the travel MAE for most lines and directions. This means the same problem persists when predicting these zero values for the dwell times described in Section 5.2.2.

Table 5.4 – MAE results of 30-minute HA model for 6 lines in both directions.

Bus route	Overal MAE	Dwell MAE	Travel MAE
G501, 1	10.177	10.738	9.649
G501, 2	10.467	11.795	9.212
G502, 1	12.432	12.099	12.738
G502, 2	12.778	11.884	13.564
G503, 1	10.781	12.351	9.210
G503, 2	10.928	12.231	9.660
G504, 1	10.746	11.980	9.544
G504, 2	15.492	17.447	13.455
G505, 1	14.102	12.673	15.487
G505, 2	12.329	9.884	14.774
G506, 1	9.652	10.570	8.793
G506, 2	9.989	10.798	9.215

5.2.4 Summary of historical average models

This section aimed to develop baseline prediction results, which were done by setting up the time-dependent HA model. This model performed slightly better than the ordinary HA model, demonstrating its effectiveness in leveraging time-of-day patterns. Additionally, the spatial interpretation of the MAE results provided insights for the reachability analysis. The findings for this baseline HA model will be instrumental in refining prediction models and improving overall performance.

As suggested by the EDA on skipped stops and dwell times equal to zero, there is some difficulty with predicting these occurrences. Taking the median or using a zero-inflated distribution might give better results. However, the current version of the model will be sufficient as a baseline model.

5.3 Vector autoregression

The second prediction model developed in this research is the VAR model. This time series model is a relatively computationally simple model. Besides producing a meaningful prediction of travel and dwell times, it also gives a good insight into whether the travel and dwell times have any significant autoregressive patterns. A VAR prediction model is a statistical model used to capture the linear interdependencies among multiple travel and dwell time series. The goal is to optimise the coefficient matrices, which can in turn be used for future prediction.

Section 5.3.1 provides an overview of how the model is implemented and the preprocessing steps that are necessary. Section 5.3.2 presents the MAE results of the prediction model. This is followed by Section 5.3.3, which investigates more closely the prediction behaviour of single travel and dwell times. Section 5.3.4 discusses the addition of time features in the form of exogenous variables to the VAR model. Finally, Section 5.3.5 concludes the findings of the implementation of the VAR prediction model.

5.3.1 Model implementation

The training data represented in the format of Table 3.2 must undergo two preprocessing steps to make it suitable for training. Firstly, travel and dwell time prediction will significantly improve when outliers are handled. VAR's performance is suboptimal when many outliers are present in the data. This process is described in Section 4.6.5. Secondly, the data must be resampled to become periodic, as this is assumed for a VAR model. This is outlined in Section 4.6.6.

For a VAR model, the primary hyperparameter to tune is the maximum lag order used for autoregression. This is determined by fitting the VAR model to the training data, which involves estimating the optimal coefficients for prediction. For each lag order, the AIC (Equation 3.9 is computed and the goal is to select the lag order with the lowest AIC. This indicates the model with the best predictive performance, and this VAR model with the corresponding maximum lag order is used for the prediction of future travel and dwell times. The model is implemented using the `VAR` model from Statsmodels [54].

5.3.2 Evaluation

The VAR models were fitted to the training data of the six bus lines in both directions. These VAR models were used to predict future travel and dwell times and were tested against the test dataset. The MAE results of this analysis are shown in Table 5.5.

Table 5.5 – The MAE results of the ordinary VAR prediction model with the corresponding optimal lag order.

Bus route	Overall MAE	Lag order
G501, 1	8.040	1
G501, 2	8.387	1
G502, 1	55.485	9
G502, 2	65.516	9
G503, 1	43.755	8
G503, 2	20.089	7
G504, 1	17.508	10
G504, 2	9.902	1
G505, 1	11.057	1
G505, 2	12.172	1
G506, 1	12.958	7
G506, 2	12.667	6

Comparing Table 5.5 to the MAE results of the time-dependent HA model (Table 5.4), for several directions the VAR model can improve on the baseline model. This is the case for g501 in both directions, g504 in direction 2 and g505 in both directions. For all the other lines and directions, the MAE on the test dataset did not improve.

The lag order is equal to one for all improved directions compared to the baseline model. Also, it was found that in the other directions, the lag order was higher than one. When the optimal lag order is found to be equal to one, the prediction model uses the values from just one previous time period to make predictions. This implies that these models are relatively simple and the current values of the variables are highly dependent on the values from the previous time period.

For lines g502 and g503 in both directions, it can be observed that the VAR model performs extremely poorly on the test dataset. These models were not able to find meaningful linear patterns in the historical travel and dwell times and often completely broke down. This resulted in MAE values which increased between 84-413% increase compared to the baseline model. For all models that did not improve on the baseline HA model, the optimal lag order that was found ranges between 6 and 10. There can be several reasons for poor performance for these models, such as overfitting, data quality and incorrect lag selection criteria.

5.3.3 Individual travel and dwell times

To better understand the prediction behaviour of VAR models, it is crucial to investigate individual travel and dwell times. This section will highlight how the best-performing algorithms predict travel and dwell times. Additionally, examples will illustrate instances where a VAR model fails to generalise and predict properly.

The premise is that the following day's travel and dwell times are predicted based on the previous day's travel and dwell times using the optimal VAR. These VAR models have been sampled hourly. The operating hours of a specific line determine which hours are predicted. For directions where the lag value is smaller than the number of hours to be predicted, future time steps are predicted using previously predicted time steps. This can lead to the prediction converging to a single value over time. Multiple days in the test dataset are incorporated into this analysis for a better overview of patterns.

In Table 5.5, it can be observed that line g501 in direction 1 is one of the few instances where the baseline HA model has been improved upon. For this model, the optimal AIC value was achieved with a lag value of 1. This VAR model has been analysed more thoroughly. Figure 5.13 shows that the predictions for the travel time between Nijenborgh and Zernikeplein remain relatively stable, consistently forecasting a travel time of approximately 24 seconds. What causes the good prediction results is that the travel time of the test dataset is relatively stable, ranging from 20 to 30 seconds. As a result, the low MAE gives the impression of strong predictive performance. However, closer inspection of the graph reveals that the VAR model doesn't seem able to generalise the patterns in the data effectively.

Additionally, Figure 5.13 suggests that the test data appears almost erratic, with no clear rush hour patterns, as analysed in the EDA in Section 5.1.1. This lack of discernible trends at the individual link level highlights an important consideration for the model's practical application: it is more effective when used for the prediction of whole bus lines than for predicting individual travel and dwell times.

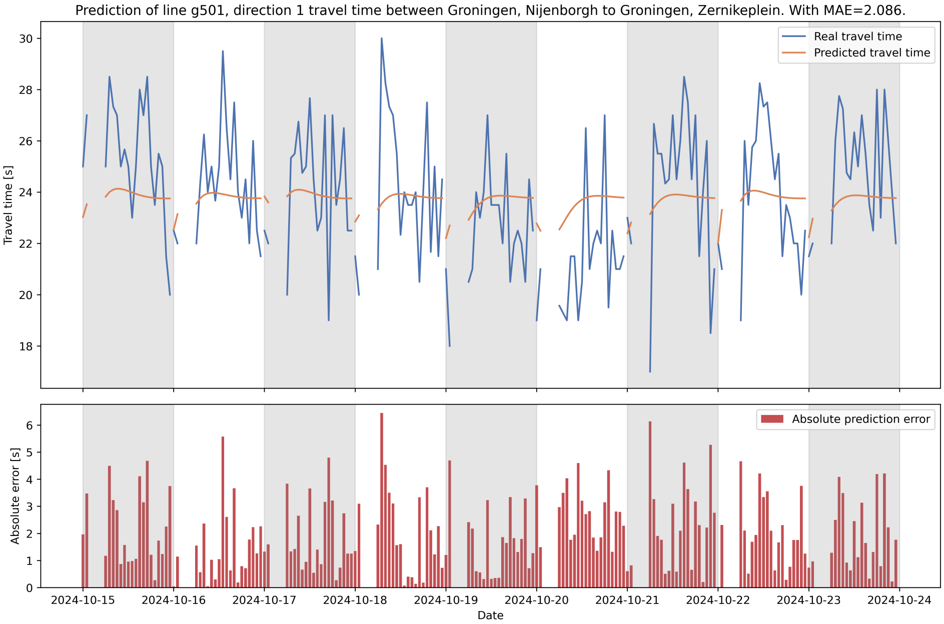


Figure 5.13 – Travel time predictions between the Nijenborgh and Zernikeplein stops from October 15, 2024, to October 24, 2024. The upper plot shows the actual travel times from the test dataset alongside the predictions made by the VAR model. The lower plot presents the MAE for each hourly prediction.

Besides improving on the baseline HA model, it can also happen that the VAR model seems to completely fall apart on the test data. This, for example, happens with line g502 in direction 1 with an MAE of 55.485. When optimising the VAR model using the training data in this direction, it was found that a lag order of 9 was the most optimal. Figure 5.14 shows one of the worst-performing travel time predictions of this line, which is the link between the Boumaboulevard and P+R Euroborg P3.

A first thing to note when looking at Figure 5.14 is that this link is longer than the link analysed in Figure 5.13. A longer link with larger travel times can inflate the MAE figure significantly. However, when inspecting Figure 5.14, it can be observed that the predictions for this travel time are poor for the test results. In the plot, the test travel times range between 60 seconds and 120 seconds. However, the predictions range from -200 to 400 seconds.

As mentioned, the predictions can also be negative. This is, of course, not possible. This will negatively impact the MAE results of the model. This happens because outliers or extreme values can largely influence the optimised coefficient matrices in the test data, which is used as input for the prediction. This can, for example, be fixed by making all the negative predictions equal to zero.

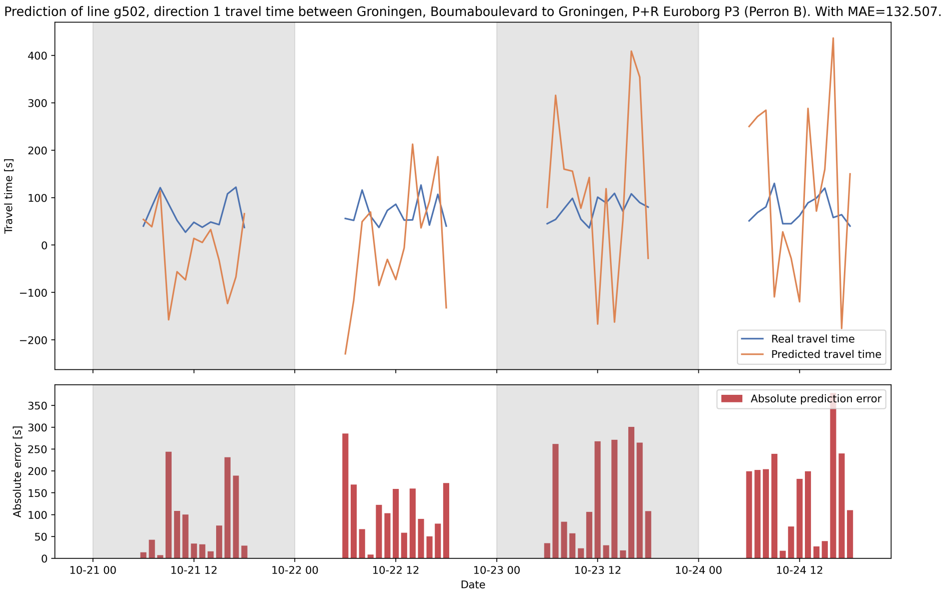


Figure 5.14 – Travel time predictions between the Boumaboulevard and P+R Euroborg P3 stops from October 21, 2024, to October 24, 2024. The upper plot shows the actual travel times from the test dataset alongside the predictions made by the VAR model. The lower plot presents the MAE for each hourly prediction.

A limitation of this VAR model approach is its assumption that journeys are continuously connected. This implies that the autoregressive components for early journeys on a given day are calculated using data from the latest journeys of the previous day. For some lines, these journeys are relatively close together (e.g., there are no buses from 01:00 to 04:00 at night). However, as shown in Figure 5.14, the schedule for this line spans from 08:00 to 18:00. This can lead to issues, as the journeys from the previous day do not appear to have predictive significance for the following day.

5.3.4 Exogenous time features

In the VAR model of Statsmodels, exogenous features can be added to improve predictive power [54]. For this analysis, time features discussed in Section 4.6.8 are added to the model as the exogenous features. These were primarily engineered for the more complex models, RF and LSTM, which can utilise this extra information to find patterns in the dataset. Equation 5.3 displays how the VAR model equations change when exogenous variables are incorporated into the model.

$$Y_t = A_1 Y_{t-1} + \cdots + A_p Y_{t-p} + B_0 X_{t-1} + \cdots + B_s X_{t-s} + u_t \tag{5.3}$$

In these equations, X_t is a vector of these exogenous time features, and B_j is the corresponding coefficient matrix. These need to be optimised in the same manner as A_i matrices. To investigate the effect of these time features on the VAR model with exogenous features, the model has been fitted and optimised in the same way as the ordinary VAR model. The MAE results are shown in Table 5.6. It can be observed that the MAE results do not significantly improve compared to Table 5.5.

Table 5.6 – MAE results of VAR with exogenous variables, which are the time features presented in Section 4.6.8. The MAE does not improve for any line or direction.

Bus route	Overall MAE	Stop order
g501, 1	18.856	22
g501, 2	25.947	23
g502, 1	62.790	9
g502, 2	79.481	9
g503, 1	43.636	8
g503, 2	20.176	7
g504, 1	17.699	10
g504, 2	17.894	8
g505, 1	37.943	10
g505, 2	15.942	5
g506, 1	12.968	7
g506, 2	12.810	6

The VAR model does not seem to benefit from the extra information the time features provide. This can be caused by the fact that it overcomplicates the simple VAR model, and it cannot generalise well with these time features.

5.3.5 Summary of vector autoregression

The development of the VAR model has provided valuable insights into the auto-regressive properties of travel and dwell times within the Groningen dataset. When analysing the MAE results, it is evident that the VAR models have outperformed the baseline HA prediction model for the lines g501 and g505. However, in many other directions, the predictions have been poor. This suggests that a more complex ML model may be required to train on the data, as the short-term values do not appear to be predictive of travel and dwell times.

One issue with the VAR model is its inability to predict zero values for dwell times. This often resulted in the model overshooting and predicting negative values. This could be addressed by ensuring all predictions are zero or higher, potentially improving prediction performance. Additionally, incorporating time features as exogenous variables did not yield fruitful results. Enhancing the model with features such as weather and traffic conditions, which may be more indicative of bus performance, could lead to better predictions.

5.4 Random Forest regression

The first ML model applied to the data is the RF regression model. This is an ensemble learning method that builds multiple decision trees. By combining the predictions of all decision trees, the accuracy is improved and the risk of overfitting is mitigated. The main reason to investigate this model is its ability to handle non-linear relationships and good predictive performance on incohesive datasets. This is necessary because HA and VAR couldn’t capture patterns in the dwell and travel times efficiently.

Section 5.4.1 explains the process of training the RF model. The hyperparameter tuning is presented in Section 5.4.2. After this, Section 5.4.3 investigates whether the tuned models are overfitting on the training data. Section 5.4.4 outlines the feature importance evaluation. Finally, Section 5.4.5 offers concluding remarks on the RF model.

5.4.1 Model training

As described in Section 4.6.11, for the RF model, the following preprocessing steps were necessary. How these preprocessing mutations are executed is described in that section; however, below, the essential preprocessing steps for RF are reiterated. These choices stem from domain knowledge and EDA.

- **Resampling (4.6.6):** This step ensures the data is periodic and complete for analysis. Since the RF model is not sequential, structuring the prediction steps coherently enhances its predictive performance.

  The data is resampled by the hour, which means that every day has as many hours as the bus is in operation. Hours where there is no bus service are removed from the data (i.e. hours at night).

- **Lag features (4.6.7):** These features provide the prediction model with the corresponding historic dwell and travel times for that corresponding journey.

  To capture the information of the past two weeks, 280 lag features per feature are added.

- **Time features (4.6.8):** Adding time-related features provides the model with valuable context about the journey, improving its ability to make accurate predictions.

  There are four time features (i.e. `hour_sin`, `hour_cos`, `day_sin`, `day_cos`) appended to the training data.

The initial dataset is divided into a training set and a test set to ensure the model is evaluated on unseen data. When performing the split, the input features (X) consist of all lag features and time-related features, while the target variables (y) are the travel times and dwell times at the time of the journey. The model that is used for the training is SKLearn's `RandomForestRegressor` [55].

5.4.2 Hyperparameter tuning

There are six hyperparameters of the RF prediction model selected to be optimised, namely `n_estimators`, `max_depth`, `min_samples_split`, `min_samples_leaf`, `max_features` and `bootstrap`. `n_estimators` refers to the number of trees, with more trees generally improving predictions but increasing computational cost. `max_depth` limits the depth of each tree to prevent overfitting. `min_samples_split` sets the minimum number of samples needed to split a node, balancing complexity and overfitting. `min_samples_leaf` ensures a minimum number of samples in leaf nodes, promoting simpler trees. `max_features` controls the number of features considered for splitting a node, affecting randomness and accuracy. `bootstrap` determines if bootstrap sampling is used, enhancing robustness by training each tree on a random subset of data. Table 5.7 displays the values that will be searched for the hyperparameter optimisation. This selection is based on preliminary trial-and-error using the RF model on the historical travel and dwell times.

Table 5.7 – Hyperparameters grid used for RandomGridSearchCV of RF algorithm.

Parameter name	**Search values**
n_estimators	$[50, 100, 200, 400, 800, 1200]$
max_depth	$[10, 20, 40, \mathrm{None}]$
min_sampels_split	$[2, 4, 8, 16]$
min_samples_leaf	$[1, 5, 10]$
max_featuresv	$[\text{'sqrt'}, \text{'log2'}]$
bootstrap	$[\mathrm{True}, \mathrm{False}]$

To efficiently explore the hyperparameter space for the RF model, first, a `RandomizedSearchCV` was applied, followed by a more precise `GridSearchCV`. The primary reason for using `RandomizedSearchCV` initially is computational efficiency. Unlike `GridSearchCV`, which evaluates every possible combination of hyperparameters, `RandomizedSearchCV` samples a subset of parameter combinations. Given the search grid in Table 5.7, training all possible models would require evaluating 1,600 configurations. Therefore, a randomised search provides a more practical way to identify promising hyperparameter values before refining them with a grid search.

A different RF model is trained for each direction, which means that a different set of hyperparameters will be optimal for each direction. A set of hyperparameters is optimal when it achieves the lowest MAE on the test dataset. The optimal hyperparameters for each direction are displayed in Table 5.8.

Table 5.8 – Tuned RF hyperparameters for the 6 lines in Groningen. For all directions, bootstrap is set as false.

Direction	**n_estimators**	**max_depth**	**min_samples _split**	**min_samples _leaf**	**max_features**
g501, 1	800	20	4	5	sqrt
g501, 2	800	20	4	5	sqrt
g502, 1	200	40	4	5	sqrt
g502, 2	800	20	4	5	sqrt
g503, 1	800	20	4	5	sqrt
g503, 2	50	40	8	10	sqrt
g504, 1	800	20	4	5	sqrt
g504, 2	1200	20	4	5	log2
g505, 1	50	40	8	10	sqrt
g505, 2	200	40	4	5	sqrt
g506, 1	800	20	4	5	sqrt
g506, 2	800	20	4	5	sqrt

In Table 5.8, it can be observed that for most directions, except g503 in direction 2 and g505 in both directions, the models prefer a high number of estimators in the RF. Also, it was found that the max_depth parameter was never set to None, which means the models generalise better on the test dataset when the depth is limited, reducing overfitting. For the hyperparemteres min_samples_split and min_samples_leaf, were both in the middle of the possible values. For max_features, the square root was preferred over log_2 for 11 out of 12 lines and directions.

5.4.3 Overfitting analysis

This section presents the predictive performance of the trained models from the previous Section 5.4.1. For this section, the best models with the best-performing hyperparameters are tested on the test dataset. The MAE results on this dataset are presented in Table 5.9. Besides the MAE on the test dataset, the MAE on the training dataset is also displayed in this table. When the MAE on the training dataset is significantly lower than the MAE on the test dataset, it may indicate that the model is overfitting. Overfitting occurs when a model learns the historical travel and dwell times too well, including their noise and outliers, which negatively impacts its performance on new, unseen data. On the other hand, a slightly lower MAE on the training dataset compared to the test dataset is expected and not a sign of overfitting. This slight difference is normal because the model is optimised to perform well on the training data.

Table 5.9 – RF MAE evaluation.

Direction	MAE test dataset	MAE training dataset	Overfitting
g501, 1	7.527	5.495	
g501, 2	7.872	4.943	
g502, 1	9.991	6.171	
g502, 2	11.287	6.25	
g503, 1	9.726	5.335	
g503, 2	9.122	5.926	
g504, 1	7.997	5.766	
g504, 2	13.34	1.788	✓
g505, 1	11.183	7.036	
g505, 2	9.391	5.481	
g506, 1	8.726	6.139	
g506, 2	8.961	6.629	

In Table 5.9, it can be observed that for g504, direction 2, the MAE on the training dataset is significantly lower than the MAE on the unseen dataset. This discrepancy indicates that the model may be overfitting the training data. To address this, the model's hyperparameters, such as max_depth, min_samples_split, or min_samples_leaf, can be adjusted.

However, for the specific case of g504, direction 2, the preprocessing step for schedule adherence (Section 4.6.3) removed many journeys from the dataset. This issue arose due to a mismatch between the end station in the GTFS schedule and the KV6 dataset. Manually fixing this would be time-consuming, resulting in incomplete data for this bus line. Consequently, this may lead to poor predictive results for the RF model.

When comparing the MAE results of the RF model to the baseline HA model (5.2), it can be observed that the more complex MAE model was able improve the prediction accuracy for all lines and directions.

5.4.4 Feature importance evaluation

RF models allow us to evaluate the importance of the features. The goal is to get an insight into what features influence the end outcome of the regression model the most. The first step would be a decrease in impurity analysis.

Mean Decrease in Impurity (MDI), also known as Gini Importance, is a method used to measure the reduction in impurity of a feature. This reduction in impurity refers to the variance for regression that a particular feature provides across all decision trees in a Random Forest (RF) regression model. During the training of an RF model, decision trees split nodes based on features that most effectively reduce impurity. This can be calculated for the entire RF model, giving each feature in the training dataset a performance score. Features with higher scores contribute more significantly to the model's prediction of travel and dwell times, while features with lower scores are less influential.

The MDI has been analysed for all 12 directional RF models. Notably, in 5 out of the 12 models, a time feature emerged as one of the most influential features. This suggests that while the travel and dwell times of past journeys are important, the journey departure time of the predicted journey is even more critical. For these 5 models, an hourly time feature consistently played a prominent role in decision-making. This aligns with the findings in Section 5.1.1, which indicate that the time of day significantly impacts the length of travel and dwell times. Figures 5.15 and 5.16 illustrate this, showing the high influence of time features in these RF models.

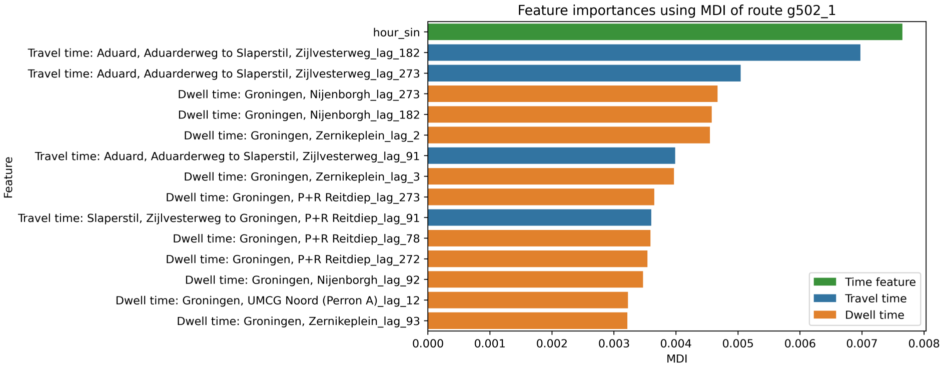


Figure 5.15 – Feature importance analysis of line g502, direction 1. The time feature `hour_sin` is influential for the prediction of future travel and dwell times.

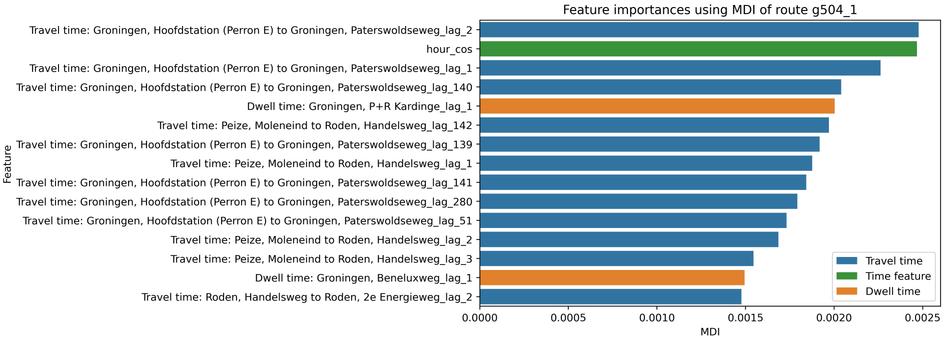


Figure 5.16 – Feature importance analysis of line g504, direction 1. The time feature `hour_cos` is influential for the prediction of future travel and dwell times. Here it can be seen that

Another pattern that can be observed is that, for some RF models, the same feature is very influential at different lags. This means that the value of a specific past feature at various points in time is quite indicative of the predicted journey. This is evident in Figure 5.16, where the travel time between Hoofdstation (Perron E) and Paterswoldseweg has a high influence. Lags 1 and 2, representing the two most recent data points, are at the top. Besides, lag 1 and 2, lags around 140 are also present, these are the same journeys from a week ago. The dataset was sampled hourly and four hours in the night were removed, so this comes to 20 entries per day. Which results in weeks of 140 data entries.

The phenomenon of the same feature being utilised a lot by the model can also be observed in Figure 5.17, where the feature travel time from Julianaplein to Hoofdstation (Perron J) and dwell time at Julianaplein are the most dominant features for the prediction for line g505 in direction 1.

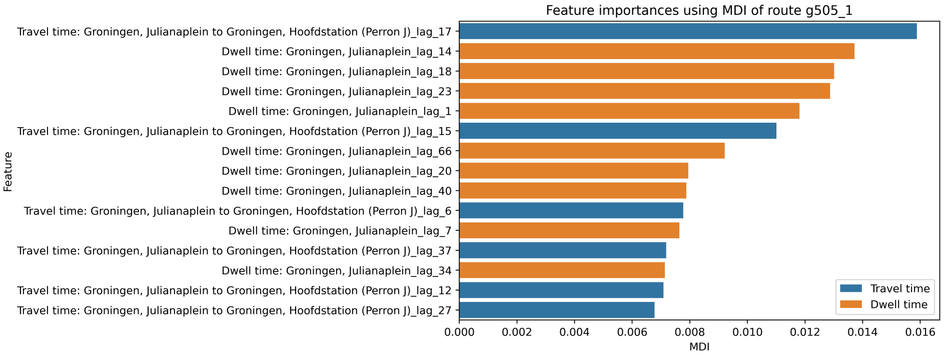


Figure 5.17 – Feature importance analysis of line g505, direction 1. Travel time from Julianaplein to Hoofdstation (Perron J) and the dwell time at the Julianalaan are the most dominant features.

The feature importance analysis figures for the other lines and directions can be found in Appendix B.2.

5.4.5 Summary of Random Forest regression

The RF model, not inherently designed for time-series prediction, demonstrated promising results for predicting travel and dwell times. RF's ability to uncover complex patterns, where relationships are not immediately obvious, was leveraged for the development of the model. The MAE results of the RF model significantly outperformed the time-dependent HA baseline model. Incorporating time features notably enhanced the accuracy of the predictions, proven by the feature importance analysis. This was also anticipated based on the results of the EDA.

The case study provides a relatively small dataset in relation to the number of features it aims to predict. Although the MAE results on the test dataset appear to be good for the RF prediction model, it is crucial to be aware of the noise found in the VAR analysis in Section 5.3.3. These patterns, even for a complex model such as RF, are difficult to pick up.

5.5 Long Short-Term Memory deep neural network

LSTM is the final model that will be developed. This deep learning model is designed to handle sequences of data, which in our case are the historical travel and dwell times. Its strength is its ability to learn patterns in the data which are not readily apparent.

Section 5.4.1 outlines the choices that are made for the training of the LSTM model. After this the optimal network structure is investigated, which is described in Section 5.5.2. The model's hyperparameters are tuned in Section 5.5.3. Section 5.5.4 presents the MAE evaluation of the trained prediction models and Section 5.5.5 investigates the validation and accuracy plots. Section 5.5.6 concludes the LSTM results.

5.5.1 Model training

As explained in Section 4.6, several preprocessing steps are necessary for the LSTM deep learning model. These steps are outlined below. The first three are essential to improve model accuracy. The last two steps are essential for the way the LSTM network expects the data to be represented as sequences.

- **Outlier handling (4.6.5):** Outlier winsorization involves identifying outliers using z-scores and replacing them with the 95th percentile, to reduce their impact on the dataset. High outliers are replaced, while low outliers, often zero values, are retained.
- **Resampling (4.6.6):** This step ensures the data is periodic and complete for analysis. Since the RF model is not sequential, structuring the prediction steps coherently enhances its predictive performance.

  The data is resampled by the hour, which means that every day has as many hours as the bus is in operation. Hours where there is no bus service are removed from the data (i.e. hours at night).
- **Time features (4.6.8):** Adding time-related features provides the model with valuable context about the journey, improving its ability to make accurate predictions.

  There are four time features (i.e. `hour_sin`, `hour_cos`, `day_sin`, `day_cos`) appended to the training data.
- **Feature scaling (4.6.9):** The `MinMaxScaler` method transforms data into the range $[0, 1]$, ensuring all features contribute equally to the learning process and preventing issues like slow convergence or poor generalisation.
- **Sequence creation (4.6.10):** To prepare the historical travel and dwell times for the LSTM model, it must be converted into meaningful input-output sequences to capture temporal dependencies.

  The input data is structured as a 3D array with the shape `(samples, timesteps (input), features (input))`, and the target data is similarly structured as `(samples, timesteps (target), features (target))`.

The models model is trained as a Sequential linear model from the Keras package [45]. From this same package, the layers LSTM, Dropout, Dense and Reshape are used. The following hyperparameters are selected for the training processes until the hyperparameter tuning. These values are considered the standard values of the Keras package. Table 5.10 presents an overview of the hyperparameters which are kept constant between testing the different structures. Early stopping is turned on when the validation loss does not improve in 10 epochs.

Table 5.10 – Hyperparameters that are kept constant when testing before the hyperparameters are tuned.

Hyperparameter	**Value**
Epochs	200
Validation split	0.2
Early stopping	10 epochs without val_loss improvement
Loss	MSE
Batch size	64
Optimiser	adam

5.5.2 Network structure

When designing the structure of a neural network, it is crucial to determine the number and types of layers to include. This decision significantly impacts the network's ability to learn and generalise from data. Given the nature of sequential data, incorporating LSTM layers is essential, as they are effective at capturing temporal dependencies and patterns.

The literature review indicates that LSTM networks have been effective for PT travel time prediction. However, it is essential to evaluate the performance of various models on the specific data from the bus network in Groningen to determine the most suitable network structure for this case study.

Data from three bus lines have been tested to evaluate the performance of different network structures. These lines are g501 in both directions and g504 in direction 1, which are the bus lines that have the most data available in the case study dataset. This experiment aims to determine the necessary complexity of the deep neural network, which largely depends on any underlying patterns in the training data. Although the three previous predictions had difficulty with predicting the seemingly noisy data, the complex nature of LSTM networks may reveal patterns when the network complexity is increased. This complexity, in this case, is adding an LSTM layer.

Besides the LSTM layer, there is an input layer which receives the preprocessed training data in the 3d Numpy array sequences described in Section 4.6.10. When adding LSTM layers to the model, there is also a Dropout layer between each LSTM layer. At the end of the Sequential model, there are two Dense layers. These layers create the required output of the model using the output of the LSTM layers.

Table 5.11 – Three network structures were tested to determine the required network complexity. The main characteristic column of an LSTM layer is the number of LSTM units in the layer and of a dropout layer is the fraction of input units to drop.

Layer type	Main characteristic	Network 1	Network 2	Network 3
Input layer	-	✓	✓	✓
LSTM layer 1	256	✓	✓	✓
Dropout	0.2		✓	✓
LSTM layer 2	128		✓	✓
Dropout	0.2			✓
LSTM layer 3	64			✓
Dense	-	✓	✓	✓
Dense	-	✓	✓	✓
Reshape	-	✓	✓	✓

For training, the dataset is split into an 80 % training and a 20 % test dataset. The trained model is used on the test dataset and the MAE results are calculated. These results are shown in Table 5.12.

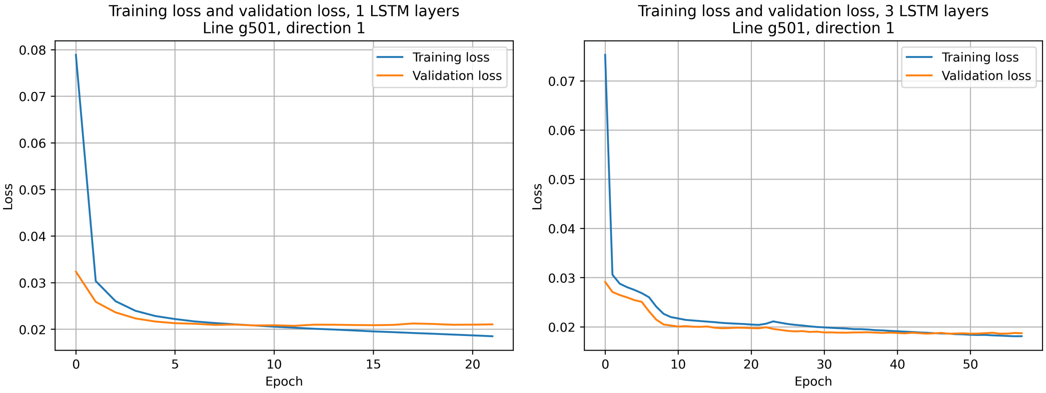
Table 5.12 – Line g501 direction 1, line g501 direction 2 and line g504 direction 1 are tested as these lines have the most journeys in the case study dataset. The network with three LSTM layers had the best MAE results on the test dataset. The lowest MAE for each line and direction is bold.

# of LSTM layers	g501, 1	g501, 2	g504, 1
1	7.172	7.300	7.574
2	6.808	6.945	7.314
3	**6.734**	**6.846**	**7.247**

It can be observed in Table 5.12, that for all three lines, the MAE decreases when increasing the complexity of the model. This suggests that a three-layer LSTM model seems to be optimal for capturing the patterns in the dataset. However, investigating the validation loss plot suggests that there are deeper issues with the training of these models. This is also suggested by not improving the MAE considerably compared to the other models.

When examining the validation loss plots in Figures 5.18a and 5.18b, it becomes evident that after only a few epochs, the validation loss ceases to improve significantly. This suggests that the model is not consistently learning and improving as training progresses. Several factors could explain this behaviour. One common reason is underfitting; however, in this case, underfitting can be ruled out. Specifically, in the loss plot for the three-layer LSTM in Figure 5.18b, the training loss is lower than the validation loss, which indicates that the model is capturing patterns from the training data. Additionally, for g501 in direction 1 of the three-layer LSTM, the training MAE is 6.372, slightly lower than the validation MAE of 6.734, which means that the model is not overfitting.

The plateauing behaviour observed in Figures 5.18a and 5.18b may indicate challenges in extracting meaningful patterns from the data. This could also be caused by the small amount of data provided in the case study dataset. The hyperparameter tuning discussed in the next section could help enhance prediction performance.



(a) Validation and training loss of a network of one LSTM layer.

(b) Validation and training loss of a network of three LSTM layers.

Figure 5.18 – Validation and training loss plots with epochs on the x-axis. Displayed for two different network structures for line g501 in direction 1.

To conclude this section, the network structure displayed in Table 5.13 will be analysed further in the hyperparameter tuning section. This table also displays the output shape of each layer. This way

Table 5.13 – Primary LSTM network structure for a certain direction.

Layer type	**Output shape**
Input layer	(input_timesteps, # of features)
LSTM layer 1	(input_timesteps, LSTM_units_1)
Dropout	(input_timesteps, LSTM_units_1)
LSTM layer 2	(input_timesteps, LSTM_units_2)
Dropout	(input_timesteps, LSTM_units_2)
LSTM layer 3	LSTM_units_3
Dense	pred_timesteps*# of pred features
Dense	pred_timesteps*# of pred features
Reshape	(pred_timesteps, # of pred features)

5.5.3 Hyperparameter tuning

Table 5.14 presents the hyperparameter space that will be searched for the optimised structure found in the previous section.

Table 5.14 – Hyperparameters space used for Bayesian Optimisation of LSTM algorithm. LSTM units refer to the units of each LSTM layer in the neural network. Dropout rates are for the two dropout layers present in the neural network. Activation 1 and 2 are the activation functions of the first and second LSTM layers, respectively. The kernel initialiser is for the first LSTM layer. Optimiser and learning rate control the behaviour during model training. The discrete hyperparameter options allow for fewer options for the Bayesian Optimisation algorithm to explore, meaning faster training.

Parameter name	**Search values**
LSTM units 1	$[128, 256, 384, 512]$
LSTM units 2	$[64, 128, 192, 256]$
LSTM units 3	$[32, 64, 96, 128]$
Dropout rate 1	$[0.1, 0.3, 0.5]$
Dropout rate 2	$[0.1, 0.3, 0.5]$
Activation 1	['relu', 'tanh', 'swish']
Activation 2	['relu', 'tanh', 'swish']
Kernel initialiser	['HeNormal', 'GlorotUniform', 'Orthogonal']
Optimiser	['adam', 'rmsprop', 'adamw']
Learning rate	log scale between 10^{-4} and 10^{-2}

In Table 5.14, three activation algorithms are tested, namely `relu`, `tanh` and `swish`. These activation functions are for the first two LSTM layers, respectively. `ReLu` means Rectified Linear Unit, which outputs the input directly if it is positive and otherwise outputs zero. This is shown in Equation 5.4.

$$f(x) = \max(0, x) \tag{5.4}$$

`Tanh` means hyperbolic tangent, which squashes the input to a range between -1 and 1. Equation 5.5 shows the definition.

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{5.5}$$

`Swish` is a smooth, non-monotonic activation function which has shown significant improvements compared to `ReLu` [56].

$$f(x) = x \cdot \sigma(x) = \frac{1}{1 + e^{-x}} \tag{5.6}$$

The kernel initialisers tested are `HeNormal`, `GlorotUniform` and `Orthogonal`, which are used to initialise the weights of the LSTM network. `HeNormal` aims to maintain proper variance across layers [57]. `GlorotUniform` balances variance across layers for sigmoid and tanh activations [58]. `Orthogonal` initialises the weights as an orthogonal matrix by using singular value decomposition [59].

The three optimisers tested for are `Adam`, `RMSprop` and `Adamw`. `Adam` uses the mean of gradients and the variance of gradients to adaptively update the model's weights and `Adamw` updates the weights outside the gradient update [60]. `RMSprop` keeps track of a moving average of square gradients to update the weights.

Table 5.15 shows the optimal hyperparameters found by the `BayesianOptimization`. This has been optimised for the same selected lines as in the analysis of the network structure in Section 5.5.2. These lines were selected, because they contained the most journeys in the case study dataset. No additional lines or directions were selected due to the high computational cost of running the `BayesianOptimization`.

Table 5.15 – Optimised hyperparameters for the three lines with the most data points. The last row shows the MAE results of the optimal network on the test dataset.

Parameter name	g501,1	g501,2	g504,1
LSTM units 1	256	256	384
LSTM units 2	192	192	128
LSTM units 3	64	96	128
Dropout 1	0.5	0.3	0.3
Dropout 2	0.1	0.1	0.3
Activation 1	tanh	tanh	tanh
Activation 2	tanh	swish	tanh
Kernel iniatilisator	HeNormal	GlorotUniform	HeNormal
Optimiser	adam	adamw	adamw
Learning rate	0.00327	0.00138	0.00561
MAE	6.776	7.025	7.326

In Table 5.15, it can be observed that the LSTM units are quite similar for both directions of line g501. The first layer of g504 has more LSTM units. This may be caused by the higher number of stops of line g504 and therefore more features to predict. An LSTM layer with more units could be beneficial in effectively capturing all the patterns in the data.

The most selected activation function is tanh. This activation function can be susceptible to the vanishing gradient problem. However, this doesn't seem problematic for our data and network setup, because tanh is found to be optimal.

Bayesian optimisation is inherently a stochastic process. When the algorithm is initialised, it begins with a set of initial hyperparameters to start the exploration. Consequently, each run of the model may yield different results. It is essential to run the model multiple times, to ensure reliable results. However, for the hyperparameters in Table 5.15, the algorithm was only executed once due to computational constraints. This could also be fixed by setting a good set of initial hyperparameters, based on domain knowledge or earlier exploration of the model and dataset.

5.5.4 Evaluation

With the optimised hyperparameter for the three directions, there was a good indication of which hyperparameter worked well for the historical travel and dwell time data. A combination of these hyperparameters presented in Table 5.15 was used with the optimised network structure to train on the data of each line and direction. The combination of hyperparameters was based on the optimal hyperparameters found by the `BayesianOptimization` and trial-and-error testing. This means that identical models were used for all lines and directions. Table 5.16 presents the MAE results for the 6 lines in Groningen, when tested against the test dataset.

Table 5.16 – MAE results of LSTM network on the test dataset.

Direction	MAE
g501, 1	6.743
g501, 2	7.034
g502, 1	9.485
g502, 2	10.435
g503, 1	8.818
g503, 2	9.086
g504, 1	7.276
g504, 2	8.422
g505, 1	11.054
g505, 2	11.332
g506, 1	7.729
g506, 2	7.961

These MAE values are the lowest MAE values found of all prediction models for 10 out of the 12 lines and directions. Only for line g505 in direction 2 and line g506 in direction 1 does the RF model perform better.

5.5.5 Overfitting analysis

For the 12 trained LSTM prediction models in Section 5.5.4, the training and validation loss functions have been plotted. This section outlines some of the well-trained models and problematic models. Figure 5.19 illustrates the model's training process, showing a consistent decrease in both training and validation loss. By the end of training at epoch 52, the losses converge closely, indicating that the model has not overfitted on the training data.

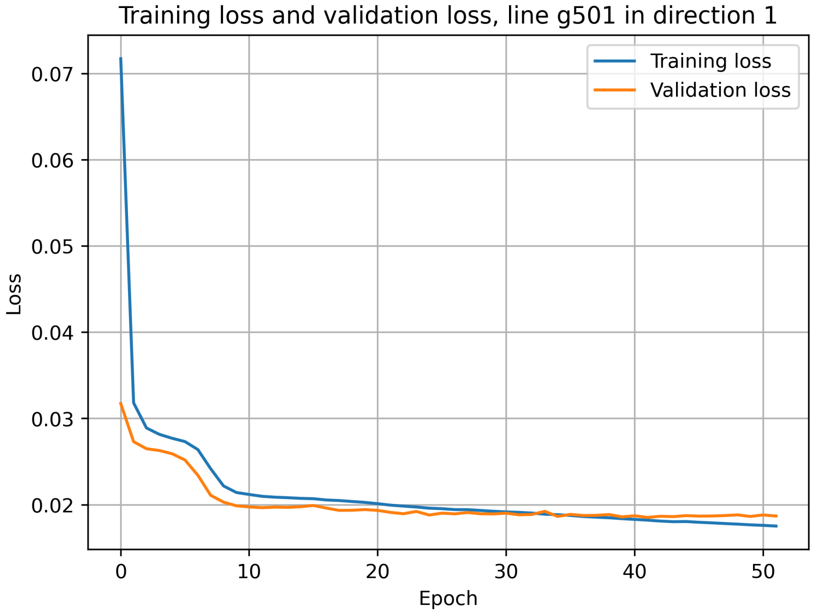


Figure 5.19 – Training set loss and validation set loss plotted during model training for each epoch. This is shown for the LSTM model of line g501 in direction 1.

Figure 5.20 illustrates the training process of the LSTM model on the data of line g504 in direction 1. At the end of training, there is a significant difference between the loss functions, which indicates overfitting. Additionally, the loss of the validation dataset stayed relatively constant during training. This means that the model did not pick up on any patterns, and it could not improve considerably from initial guesses. This same behaviour was observed for g501 in direction 2, g503 in both directions and g506 in direction 1. This suggests the model's performance is suboptimal, and its predictive accuracy on unseen data might be limited.

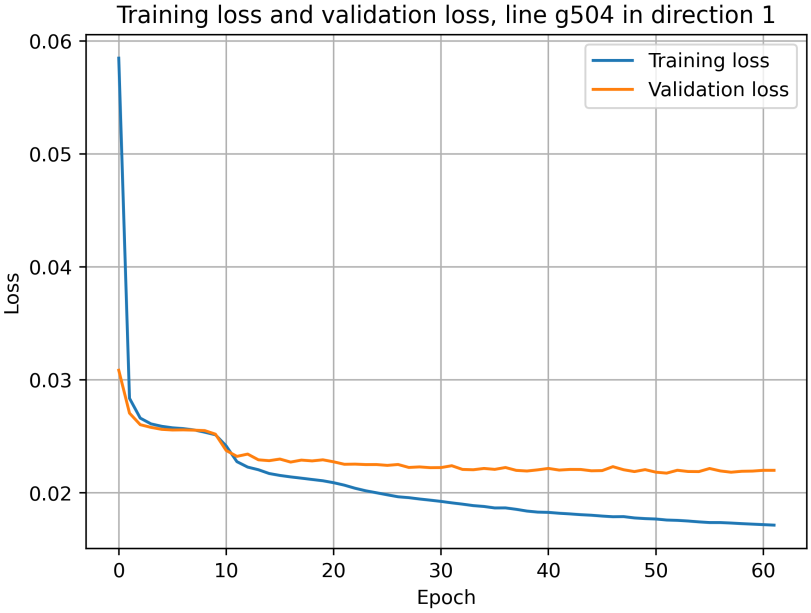


Figure 5.20 – Training set loss and validation set loss plotted during model training for each epoch. Training set accuracy and validation set accuracy during model training for each epoch. This is shown for the LSTM model of line g504 in direction 1.

This training behaviour indicates that there are potential issues in the travel and dwell time training data. The data might lack strong sequential dependencies and little correlation over timesteps. This is suggested by the validation loss, which plateaus for some lines and directions. Another problem can be that there is a high noise-to-signal ratio in the dataset. This causes the model to be unable to extract useful features.

The loss analysis figures for the other lines and directions can be found in Appendix B.3.

5.5.6 Summary of Long Short-Term Memory deep neural network

In this section, the optimal network structure was identified as having three layers of LSTM units, with hyperparameters optimised for this configuration using `BayesianOptimization`. The predictive performance was notably accurate. This was shown by the lowest MAE among the four prediction models. However, an analysis of the loss and accuracy plots revealed issues with the learning process, indicating the lack of strong predictive patterns in the data.

The hyperparameter tuning process could be enhanced by testing a broader range of configurations and conducting multiple `BayesianOptimization` runs. This approach may lead to improved performance on the dataset. Additionally, utilising a setup with higher computational speed would produce faster results.

5.6 Comparison of prediction models

Sections 5.2-5.5 presented the performance and training of the four prediction models selected in this research. In these sections, the strengths and shortcomings of the prediction models have been evaluated. However, more importantly, a better understanding of the nature of the data has been developed. This summary section aims to unify the outcomes of the four models and will answer RQ3.

5.6.1 MAE evaluation

For all four prediction models, the MAE on the test dataset was calculated. An overview of the MAE results is shown in Table 5.17.

Table 5.17 – Complete overview of MAE results of the HA, VAR, RF and LSTM prediction models. The lowest MAE for each line and direction is in bold.

Direction	30 min HA	VAR	RF	LSTM
g501, 1	10.177	8.040	7.572	**6.743**
g501, 2	10.467	8.387	7.872	**7.034**
g502, 1	12.432	55.485	9.991	**9.485**
g502, 2	12.778	65.516	11.287	**10.435**
g503, 1	10.781	43.755	9.726	**8.818**
g503, 2	10.928	20.089	9.122	**9.086**
g504, 1	10.746	17.508	7.997	**7.276**
g504, 2	15.492	9.902	13.340	**8.422**
g505, 1	14.102	11.057	11.183	**11.054**
g505, 2	12.329	12.172	**9.391**	11.332
g506, 1	9.652	12.958	**8.726**	7.729
g506, 2	9.989	12.667	8.961	**7.961**

The baseline HA model demonstrated solid predictions for travel and dwell times, with MAE values consistently ranging between 9.652 and 14.102 across all directions. In contrast, the VAR models performed poorly on the test dataset for lines g502, g503, and g504 in direction 1, failing to outperform the baseline model. Notably, the MAE for line g502 in direction 2 rose to 65.516, indicating that a short-term linear approach to predicting travel and dwell times is insufficient.

From Table 5.17, it is evident that complex models are more effective for predicting travel and dwell times. The LSTM model achieved the best results for all lines and directions, except g505 in direction 2 and g506 in direction 1. The deep learning model significantly improves upon the baseline HA model. This suggests that the relationship between travel and dwell times and their past values is non-linear.

The complex ML models can capture details but are also prone to overfitting, especially with smaller datasets, as observed in some LSTM models. This means these models fixate more on the noise or random fluctuations in the travel and dwell times rather than the underlying patterns. Another drawback of the well-performing RF and LSTM models is model interpretability. Understanding the inner workings of these ML models is challenging. While feature importance analysis of the RF model provides some insights, it remains difficult to diagnose the decisive patterns for travel and dwell times.

5.6.2 Time features

The time of day was already exploited by the time-dependent HA model, for which the prediction results improved considerably compared to the ordinary HA model. With the available historical time travel and dwell time data, time features were the most apparent to create, providing the more complex prediction models with additional context. These were designed as sinusoidal patterns to reflect the cyclical nature of daily and weekly rhythms accurately. The feature importance analysis of the RF model indicated that the hourly features were particularly predictive. Also, the complex LSTM model achieved good results incorporating the time features. Only the VAR model did not benefit from the time features being added as exogenous features, and the model's predictions were worse than those of the ordinary VAR model.

5.6.3 Noisy data

As suggested in the EDA, the data contains a considerable amount of noise. The prediction models set out to investigate whether meaningful patterns could be found despite this noise. Even though the MAE results on the test dataset have improved with the complex ML models, it is difficult to draw conclusions about the presence of reliable patterns. The feature importance analysis of the RF model and the loss analysis of the LSTM model indicate that the data presents a diverse range of patterns. This is further highlighted by the dynamic nature observed in the analysis of individual travel and dwell times of the VAR model.

5.7 Reachability analysis

Sections 5.2-5.6 present the results of the travel and dwell time prediction of four models. The predicted travel and dwell times can now be leveraged to analyse reachability. Reachability refers to how fast destinations can be reached from a certain starting point. This section will focus on the reachability influenced by the six lines predicted in the case study of the bus network of Groningen. This section aims to present several examples of reachability issues caused by the travel and dwell time predictions. This will not be an exhaustive list of all the abnormalities which can be found in the case study dataset.

Section 5.7.1 compares the MAE of the predicted values with the original schedule. This is followed by Section 5.7.2, which details the setup of scenarios in Conveyal. Subsequent sections provide examples of the enhanced reachability analysis, discussing the following instances: equal reachability (5.7.3), increased reachability (5.7.4), decreased reachability (5.7.5), missed transfers (5.7.6), and rush hours (5.7.7). Finally, Section 5.7.8 offers concluding remarks and recommendations.

5.7.1 Comperative analysis: Schedule vs. LSTM

A common practice, and in the case of the Conveyal, is to compute reachability based on the planned schedule. This is done by uploading a GTFS schedule of the PT network to the tool, which will be used to compute travel times of the transportation network. These GTFS schedules can also be tested against the test dataset in the same manner as the predictions models are tested following the methodology presented in Section 3.9. The MAE results of this analysis are displayed in Table 5.18. Also, this table shows the MAE results of the LSTM prediction model. This prediction model is chosen, because it achieved the best prediction results.

Table 5.18 – MAE evaluation of the GTFS dataset tested against the test dataset of the case study. The LSTM results are also shown in this table. The MAE of the LSTM prediction model are significantly lower than the MAE of the schedule.

Direction	Schedule MAE	LSTM MAE
g501, 1	30.233	6.743
g501, 2	30.710	7.034
g502, 1	31.329	9.485
g502, 2	28.455	10.435
g503, 1	27.876	8.818
g503, 2	26.733	9.086
g504, 1	27.390	7.276
g504, 2	29.940	8.422
g505, 1	26.953	11.054
g505, 2	26.152	11.332
g506, 1	20.017	7.729
g506, 2	21.492	7.961

In Table 5.18, it can be observed that for all lines in both directions, the MAE of the LSTM prediction model is significantly lower than the schedule. This means that the LSTM prediction offers a better representation of the real-world performance of the transportation network than the schedule does. This will be beneficial for analysing reachability. An important note is that the baseline time-dependent HA and RF models also achieve lower MAE values than the original schedule.

The high MAE of the GTFS schedule are likely caused by the fact that all arrival and departure times are rounded off to whole minutes. This rounding causes precision loss, leading to differences between the scheduled travel and dwell times and the actual travel and dwell times.

5.7.2 Conveyal

The reachability analysis will be conducted in Conveyal [52]. This is a tool which visualises the reachability of isochrones. The tool will calculate how long it will take to reach certain destinations. This section explains the settings for running a reachability analysis.

As explained in Section 4.4, Conveyal takes a GTFS schedule as input to calculate travel times. For the analysis in this section, two versions of the GTFS schedule are uploaded: the original and the LSTM predictions. The process of overwriting of the GTFS dataset with the predictions is outlined broadly in Section 4.5 and explained in detail in Section 4.6.12.

An analysis run in Conveyal must typically compare two scenarios with each other. Four scenarios have been configured, based on the two uploaded GTFS schedules. These scenarios are shown in Table 5.19.

Table 5.19 – Scenarios that are defined in the Conveyal tool. The column '6 lines' displays whether, in a scenario, the original schedule is used or the predicted schedule. The six lines for our research are g501, g502, g503, g504, g505 and g506. 'Rest of network' column displays whether other lines, besides the six lines, are included in that scenario.

Scenario	6 lines	Rest of network
Whole original	Original	✓
6 line original	Original	
Whole predicted	Predicted	✓
6 line predicted	Predicted	

The four scenarios displayed in Table 5.19 allow for a comprehensive analysis of the impact of the LSTM predictions on reachability. The original schedule means the original planned schedule of QBuzz will be used to calculate travel times. The predicted schedule means that the LSTM predictions are used to calculate travel times. Besides these distinctions, there is also a distinction made between whether, in a scenario, the rest of the PT network is included. This would entail the original schedule of buses, trams and trains, which are not line g501, g502, g503, g504, g505 or g506. Including these other modes of transport in a scenario would mean that transfers from or to the six analysed lines are possible.

Besides setting up the scenarios, the following things must be defined for each analysis in Conveyal:

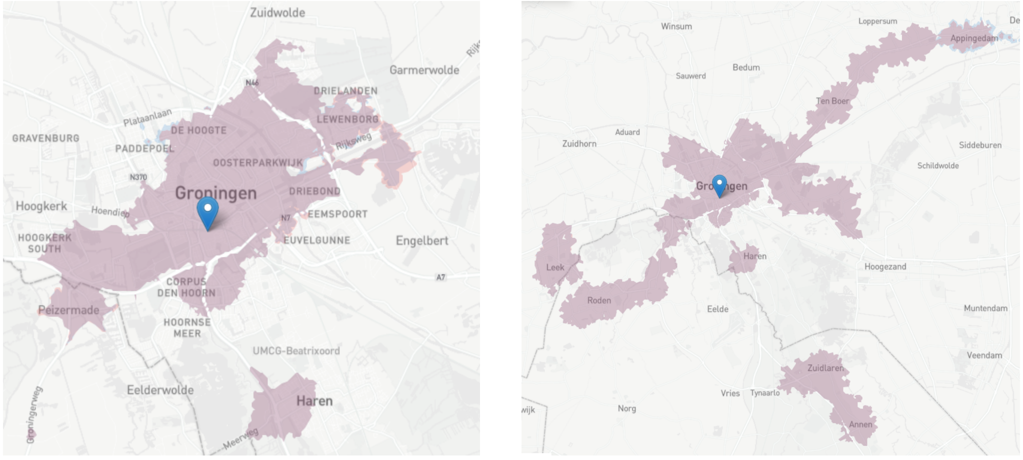
- **Travel time cutoff:** The edge of the displayed isochrone.
- **Time percentile:** How reliably people can reach destinations during a departure time window.
- **Departure time window**: Time interval during the journey from the starting point during which the journey should depart.
- **Day:** Day of the analysed journeys.
- **Starting point:** Starting point of the analysed journeys.
- **Region of analysis:** Region around the starting point for which the travel time is calculated.
- **Acces mode:** Mode (walking, cycling or by car) of transfer the starting point is being departed from.
- **Transit modes:** Modes (car, bus, tram, train and metro) are used in the journeys.
- **Egress mode:** Mode (walking, cycling or by car) of transfer to reach the destination.
- **Max # of transfers:** Maximum number of transfers between vehicles or modes within a journey.

When the scenarios and settings are defined for the analysis, Conveyal will calculate the travel times based on the road and PT network. It will display coloured isochrones on the map, indicating the percentile of passengers reaching that destination within the time cutoff. The isochrones are coloured red or blue depending on the selected scenario. When they overlap, they are coloured purple, indicating that a destination can be reached in both scenarios.

5.7.3 Equal reachability

The first analysis is when the original and predicted schedules display the same reachability. When comparing the original schedule with the predicted schedule, it is expected that there will be significant overlap for the majority of destinations. This is because the primary goal of a bus is to adhere to its schedule as closely as possible, and prediction models are designed to forecast these times accurately.

An investigation of multiple reachability analyses confirmed this expectation. For example, Figure 5.21 presents two figures from the same reachability analysis with different cutoff times. This analysis was run for a Saturday from Groningen Station. Figure 5.21a is cut off at 40 minutes, while Figure 5.21b is cut off at 80 minutes. The purple isochrones in both figures represent the overlapping isochrones, indicating similar reachability performance for the original schedule and the predicted schedule.



(a) Cutoff time 40 minutes.

(b) Cutoff time 80 minutes.

Figure 5.21 – Reachability analysis that shows equal isochrones for both the original schedule and predicted schedule. The starting point is Groningen Station on a Saturday between 12:00 and 14:00.

5.7.4 Increased reachability

The prediction model can predict that a journey will be travelled faster than the schedule expects. This can, for example, be seen in the reachability analysis run on Wednesday, which departed from Groningen Station between 15:00 and 16:00. Figure 5.22 shows the increased reachability caused by the shorter travel and dwell times. The red isochrone is the reachability according to the predicted schedule, which is larger than the purple isochrone of the original schedule. This difference in reachability is caused by the g502 line towards Zuidhorn, which was predicted to travel faster.

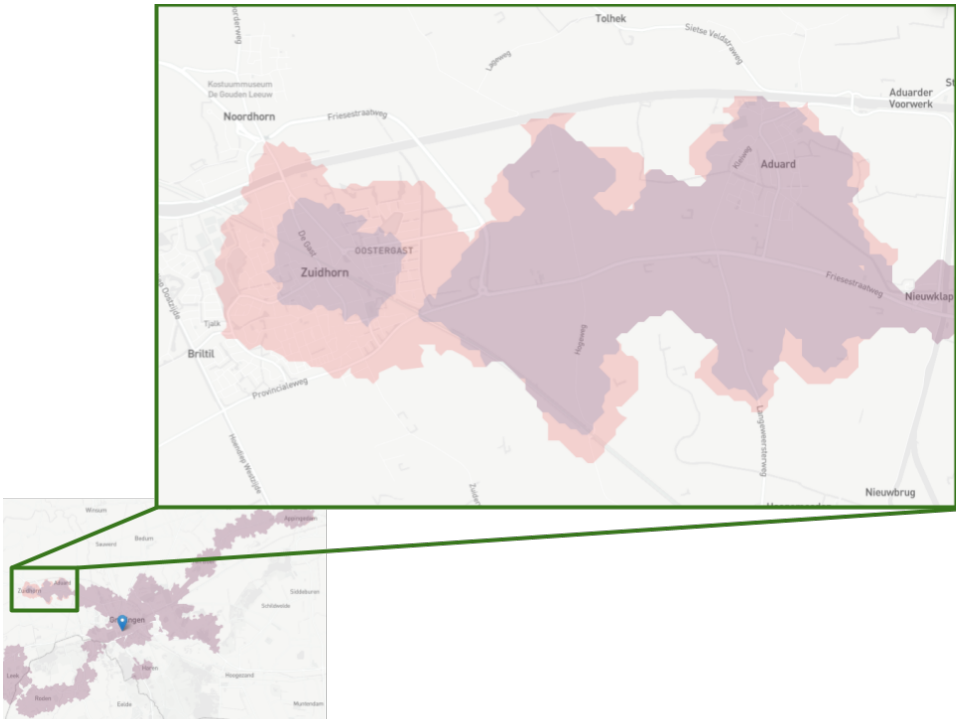


Figure 5.22 – Increased reachability example for an analysis run on a Wednesday with a starting point Groningen station between 15:00 and 16:00. Here, the larger red region is the predicted schedule by the LSTM model.

To validate the analysis in Figure 5.22, Table 5.20 displays the schedule of the corresponding bus journey for this reachability analysis. It can be observed that the journey g502, which left Groningen Station at 14:41:00, is predicted to arrive more than 6 minutes and 6 seconds earlier than the original schedule expects. This is the cause of the increased reachability area displayed in red.

Table 5.20 – The scheduled and predicted times of bus line g502 towards Zuidhorn. This journey departed from Groningen Station at 14:41:00. The predicted journey is ahead of the original schedule.

Stop	Original arrival time	Original departure time	Predicted arrival time	Predicted departure time
⋮	⋮	⋮	⋮	⋮
Slaperstil, Zijlvesterweg	15:12:00	15:12:00	15:10:50	15:10:52
Aduard, Aduarderweg	15:15:00	15:15:00	15:13:17	15:13:29
Zuidhorn, Spanjaardsdijk Zuid	15:17:00	15:17:00	15:14:46	15:14:47
Zuidhorn, Station (Perron)	15:25:00	15:25:00	15:18:54	15:18:54

5.7.5 Decreased reachability

When the prediction model predicts that a journey will be slower than the schedule, the reachability of that journey will decrease. An example of such behaviour can be seen in Figure 5.23, where line g505 towards Scharmer is predicted to be slower. This analysis is run on a Tuesday between 10:00 and 12:00. In this figure, the blue isochrone is the expected reachability according to the original schedule. The purple isochrone is what the predicted schedule expects to happen. The slower predicted bus journey causes the purple isochrone to be smaller.

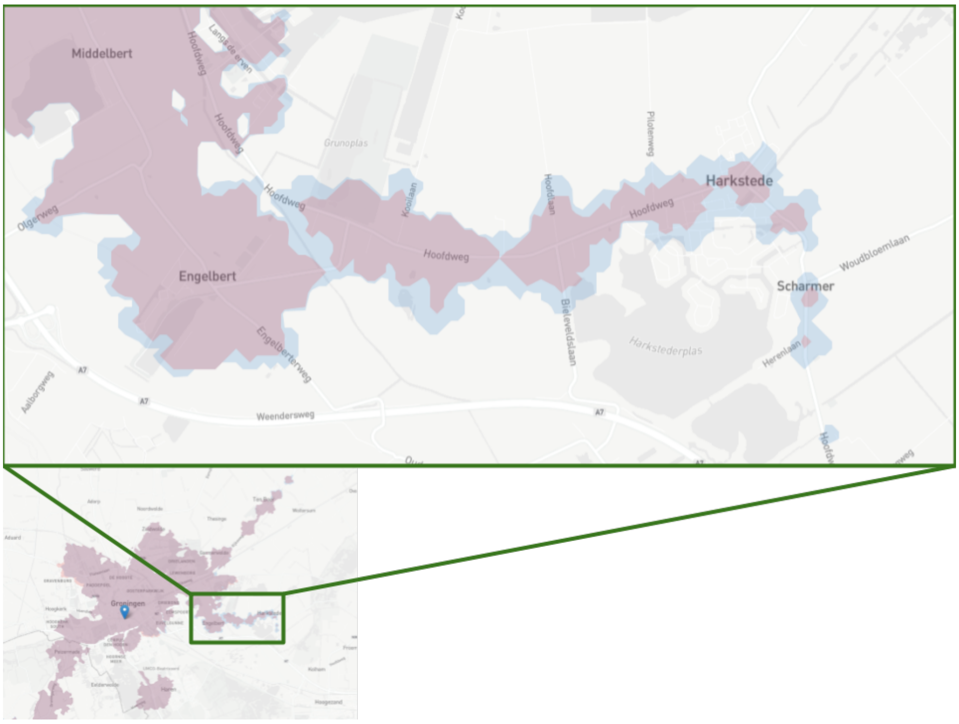


Figure 5.23 – Example of decreased reachability for an analysis run for a Tuesday from Groningen's station, which departed between 10:00 and 12:00. The blue isochrone is the reachability that the original schedule expects. The LSTM model predicts the reachability region to be the smaller purple isochrone.

To validate the analysis shown in Figure 5.23, Table 5.21 displays the original schedule and predicted schedule of the g505 journey towards Scharmer that causes this decreased reachability. This table shows that the LSTM model predicts a later arrival and departure time for the stops from Sportveld to Hoofdlaan. This means there is a decreased reachability for all stops of the town of Harkstede serviced by this journey.

Table 5.21 – The scheduled and predicted times of bus line g505 towards Scharmer. This journey departed from P+R Haren A28 at 10:05:00. The original schedule is ahead of the predicted schedule.

Stop	Original arrival time	Original departure time	Predicted arrival time	Predicted departure time
⋮	⋮	⋮	⋮	⋮
Engelbert, Sportveld	10:38:00	10:38:00	10:39:58	10:40:04
Harkstede, Grunostrand	10:39:00	10:39:00	10:42:01	10:42:07
Harkstede, Veldzicht	10:40:00	10:40:00	10:42:39	10:42:40
Harkstede, Hoofdlaan	10:41:00	10:41:00	10:42:31	10:43:33
Harkstede, Pilotenweg	10:42:00	10:42:00	10:44:17	10:44:20
Harkstede, Dorpshuisweg	10:42:00	10:42:00	10:44:46	10:44:54
Harkstede, Hamweg	10:43:00	10:43:00	10:45:40	10:45:51
Harkstede, Hoofdlaan	10:44:00	10:44:00	10:46:24	10:46:28
⋮	⋮	⋮	⋮	⋮

5.7.6 Missed transfers

The previous examples only demonstrated the effect of the six bus lines in the case study. This analysis also includes other PT lines based on the original schedule. Slower predicted buses may cause missed transfers to other buses. This issue was observed in the reachability analysis conducted on Monday between 07:00 and 09:00, starting from Groningen station.

In Figure 5.24, a reachability difference between the schedule and the LSTM prediction can be observed towards the northwest of the City of Groningen. This area, serviced by line 564, was not included in the travel and dwell time prediction, meaning its data is based on the original schedule. The larger blue portion indicates that the original schedule suggests you can travel farther in the same amount of time compared to the predicted schedule.

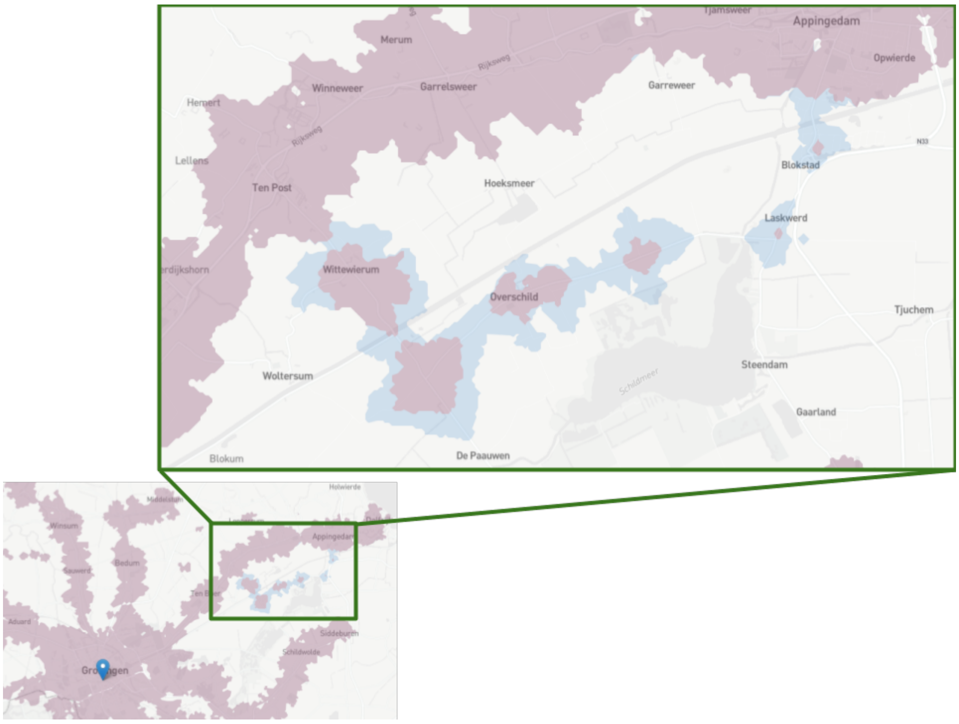


Figure 5.24 – Reachability difference caused by a missed transfer from line g506 to 564. This analysis was run on a Monday between 07:00 and 09:00 with a starting point of Groningen's station. The blue isochrone is the original schedule, and the smaller purple isochrone is the predicted schedule.

Further investigation shows that a transfer in the village of Ten Boer cannot be achieved due to the slower predicted line g506 towards Delfzijl. Table 5.22 shows the specific predicted timetable. This table indicates that a 52-second delay is already predicted at the stops before Ten Boer. In Ten Boer, this delay is expected to increase to 1 minute and 37 seconds. Consequently, the slower g506 bus reduces the reachability of the 564 bus.

Table 5.22 – The scheduled and predicted times of bus line g506 towards Delfzijl. This journey departed from Groningen station at 6:52:00. The journey is predicted to be slightly delayed throughout the town of Ten Boer.

Stop	Original arrival time	Original departure time	Predicted arrival time	Predicted departure time
⋮	⋮	⋮	⋮	⋮
Ten Boer, Bovenrijgeweg	07:09:00	07:09:00	07:09:52	07:09:53
Ten Boer, Groene Zoom	07:10:00	07:10:00	07:11:05	07:11:08
Ten Boer, Centrum	07:11:00	07:11:00	07:11:48	07:12:10
Ten Boer, Dijkshoorn	07:12:00	07:12:00	07:13:08	07:13:13
Ten Post, Rijksweg 113	07:13:00	07:13:00	07:14:37	07:14:38
⋮	⋮	⋮	⋮	⋮

5.7.7 Rush hours

The preceding sections compared the original schedule with the predicted schedule. This section sets out to predict two predicted schedules at different times of the day. EDA section 5.1.1 found that the bus travel and dwell times are influenced when the bus journey takes place. During the day, especially during rush hour, journeys tend to take longer. This can also be seen in the reachability analysis.

Figure 5.25 is an example of this kind of reachability analysis. This analysis was run on a Thursday from Groningen station. The red isochrone is the analysis between 14:00 and 15:00 and the blue isochrone is between 07:00 and 08:00. It can be observed that the red isochrone is larger, indicating that the reachability in the afternoon is higher. This is caused by the buses travelling faster than the buses in the rush hours between 07:00 and 08:00.

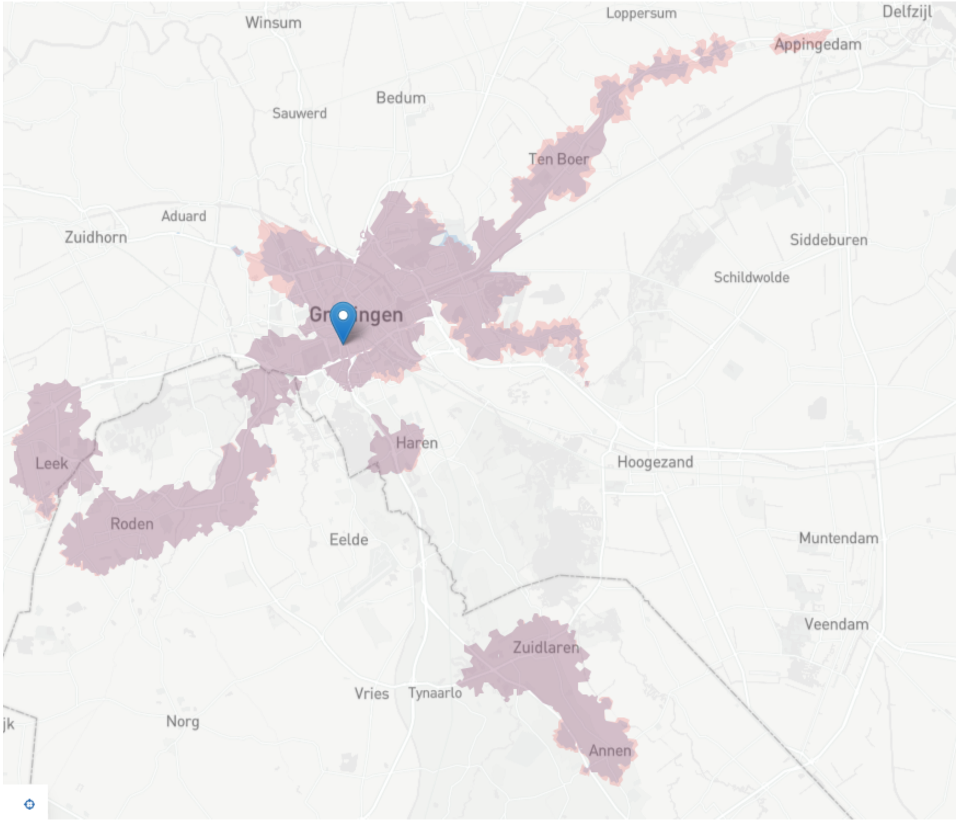


Figure 5.25 – Reachability analysis for Thursday with starting point Groningen Station. The red region is the analysis between 14:00 and 15:00 and the blue region is between 07:00 and 08:00.

These kinds of reachability analyses are of great value for PT engineers and are more accurate due to incorporating real-world bus travel data.

5.7.8 Summary of reachability analysis

This section demonstrates the improvement that the prediction models can offer compared to the original schedule. After this, the analyses in Conveyal are explained in four different scenarios. This section presents several examples illustrating the impact of predicted travel and dwell times on reachability analysis. In most analyses, the comparison between the original and predicted schedules shows an overlap, represented by a purple region in the figures. However, some specific analyses reveal differences due to the predicted travel and dwell times, which can either increase or decrease reachability. Besides, these analyses between the original and predicted schedules, Section 5.7.7 presented an example of how the rush hours tend to influence the reachability.

6 Discussion

This thesis set out to develop prediction models that predict bus travel and dwell times to improve reachability analysis. Firstly, the state-of-the-art of bus travel time prediction was formulated, which led to the selection of the four developed prediction models: HA, VAR, RF and LSTM. These prediction models range from simple to complex in the patterns that they can uncover in the data. To test these prediction models, the case study of the bus network of Groningen was set up, which provided the KV6 and GTFS datasets necessary as training data for the prediction models. The development of the four prediction models gave insight into the patterns in the training data and the prediction results of future journeys. The predicted travel and dwell times of the best-performing prediction models were integrated into a reachability analysis of the same region.

This section discusses the results presented in Section 5. It begins with Section 6.1, which summarises and interprets the acquired results. Next, Section 6.2 highlights the broader relevance and implications of these findings. Section 6.3 then outlines several limitations of the results. Finally, Section 6.4 provides suggestions for future research and Section 6.5 provides practical recommendations for PT engineers.

6.1 Results and interpretations

This section summarises and interprets the results presented in Section 5, which described the development of the prediction models and the reachability analysis. Answering RQ3 and RQ4 was the primary focus of the results section. Prior to the results section, Sections 2 and 4 addressed RQ1 and RQ2, respectively. Key takeaways are mentioned in this section for completeness; however, a comprehensive analysis of these research questions can be found in their respective sections.

6.1.1 State-of-the-art

To answer RQ1, the state-of-the-art in travel time prediction and reachability analysis was investigated through a review of relevant literature. The key findings from the literature review in Section 2 are summarised below.

- In contrast with existing methods, this thesis focused on travel time prediction using solely historical travel time data.
- This thesis focuses on predicting the travel and dwell times of future bus journeys, rather than those in operation. These predictions are integrated into reachability analysis.
- The study compares different prediction models, including HA models, time series models, and more complex ML and deep learning approaches. The technical methodology is built on this foundation.

6.1.2 Case study

To address 2.3.1, a real-world case study was formulated to contextualise the prediction models and reachability analysis. This case study also provided the necessary historical travel time data. The key aspects of the case study in Section 4 are outlined below.

- A case study was conducted on the bus network in Groningen, focusing on lines g501, g502, g503, g504, g505, and g506.
- Historical travel time data was sourced from the KV6 dataset, while the GTFS schedule provided the preplanned bus line schedules.
- A data framework was developed to overwrite GTFS schedules with the output of prediction models, using them as input for Conveyal, a reachability tool.
- Extensive preprocessing steps were performed, including imputing messages, handling duplicates, and checking schedule adherence, to convert KV6 messages into a travel and dwell time dataset.
- Model-specific preprocessing steps were applied to the travel and dwell time dataset to prepare it for the prediction models.

6.1.3 Prediction models

To answer RQ3, the study aimed to test four prediction models and investigate which of the selected prediction models was most suitable for predicting travel and dwell times of future journeys. Besides analysing predictive performance, this model development phase also aimed to gain insight into the factors influencing travel and dwell times of future journeys based on the historical travel time data. The research has led to the formulation of the following key findings, which are discussed in more detail below.

- The baseline time-dependent HA model provides reliable and insightful prediction results.
- The predictive performance improves with increased model complexity.
- Travel and dwell times are significantly affected by the departure time of the journey.
- The historical travel time data is noisy, which makes it challenging to predict individual journeys.

Baseline HA model

The baseline HA model, particularly the time-dependent version, demonstrated reliable results when analysing its MAE outcomes. Unlike a VAR model, its predictions are not able to diverge, as evidenced by the stable MAE values across all directions and lines. However, the HA model struggles with dwell time predictions that contain zero values. This issue can be addressed by using either the median as a prediction or a zero-inflated probability distribution.

This model is straightforward to interpret, making it easy to understand how the outputs are generated. Any anomalies in the predicted travel and dwell times can be identified and explained by examining historical travel times. This characteristic makes the model an excellent baseline and a valuable tool for initial analysis of travel time data.

Additionally, the time-dependent model offers a significant improvement over the original schedule in the context of reachability analysis. This is supported by the MAE results of the original schedule presented in Table 5.18, which are significantly higher for all lines and directions. The combination of improved accuracy and good interpretability provides urban planners or PT engineers with a practical starting point for conducting reachability analysis.

Model complexity

When analysing the MAE results of the four prediction models, it is evident that the more complex ML models achieve the lowest MAE when predicting the test dataset. While this does not necessarily indicate that RF and LSTM models are the most suitable for the task, it provides insight into the model complexity level required for accurate travel time prediction. The literature review also suggested that RF [28][30] or LSTM [13][32][33][34] would be suitable models for these complex time series forecasting tasks. RF's ability to handle non-linear relationships and LSTM's proficiency in capturing long-term dependencies are strengths that are particularly useful for travel time prediction.

It is challenging to manually dissect how historical travel and dwell times are predictive of future journeys. These complex ML models have learned significant patterns and improved on the baseline HA model. This indicates that there are meaningful patterns further back in the past than just the previous bus journey that can be indicative of the performance of future bus journeys. The divergence of VAR for some lines and directions and the well-performing RF and LSTM models suggest that relationships in the historical travel time data are likely non-linear.

For RF and LSTM, overfitting might occur. For example, the LSTM learning process was not optimal for some lines and directions, as evidenced by the plateauing behaviour of the validation loss plots. Nevertheless, both these models provide mechanisms to control overfitting. RF does this naturally by fitting a large number of decision trees, and it can also be controlled by setting hyperparameters such as `max_depth`, `min_samples_split`, or `min_samples_leaf`. Incorporating `Dropout` layers in the LSTM deep neural network also helps control overfitting.

Time-dependent travel times

An analysis of the departure times' impact on travel and dwell times in the EDA has already highlighted its significance. This highlighted that journeys during the daytime are relatively slower than in the early morning and evening. During the prediction model development, this was first confirmed by the improved MAE results achieved by the time-dependent HA model compared to the ordinary HA model. Secondly, the feature importance analysis of the RF models showed that the `hour_sin` and `hour_cos` time features were decisive for many directions and lines. Lastly, the LSTM model provided the best prediction while incorporating this additional data.

Bus performance is highly dependent on the environment in which they operate. The level of traffic and congestion differs throughout the day, which will also impact buses. Observing these patterns in the data aligns with expectations, and leveraging this knowledge for more accurate predictions is particularly useful.

Noisy data

Upon closely observing travel and dwell times, it became evident that they are relatively noisy. This conclusion is supported by the VAR patterns in Figures 5.13 and 5.14. Consequently, predicting an individual journey is challenging and unreliable. The prediction models offer reliable insights when considering multiple predicted journeys, which aligns with the consensus of leveraging ML for these tasks. However, an urban planner or PT engineer might want to investigate the reachability and performance of a specific journey. Ensuring that the prediction models provide reliable predictions that closely represent the real world is challenging.

6.1.4 Reachability analysis

To answer RQ4, the predictions of the best-performing model, LSTM, were used as input to the reachability tool Conveyal. In this tool, the predicted schedule was analysed together with the original schedule. This research has led to the formulation of the following key findings, which are discussed in more detail below.

- Leveraging historical travel time data improves reachability analysis by providing a more accurate representation of the real world.
- Discrepancies between the original and predicted schedule manifest as increased reachability, decreased reachability and missed transfers.
- The differences in reachability calculated using the original versus the predicted schedule are relatively minor.
- Reachability analysis reveals distinct time-of-day patterns.

Accurate real-word representation

It was investigated whether the travel and dwell times based on the original schedule are comparable to those in the test dataset. This experiment shows whether it represents the real world; this is similar to how the prediction models' output is tested. In Table 5.18, the MAE values of the best-performing model, LSTM, are significantly lower than the MAE values of the original schedule. This indicates that the predicted travel and dwell times more closely represent the real world because they incorporate historical travel times.

This result was expected, as the thesis's aim was to provide an improvement compared to calculating travel and dwell times based on the original schedule. The original schedule is also rounded to whole minutes, which is done for straightforward interpretation for passengers. This practical loss of precision is influential when utilising the schedule for other purposes, such as travel time calculations for reachability analysis.

Discrepancies between predicted and original

Figure 5.21 illustrates that the reachability regions calculated based on the original and predicted schedules often overlap. This is expected as the original schedule attempts to represent real-world bus operations closely. This overlap also validates that predictions are not extremely unrealistic.

What is of more interest for PT engineers and urban planners is where the original and predicted reachability are different. It was shown that for some specific times the predicted reachability isochrone could be larger (5.7.4) or smaller (5.7.5) than the original schedule isochrone. This means that either more or fewer destinations can be reached within a certain time. Section 5.7.6 also showed that a predicted delayed bus could result in missed transfers to other modes of transport. This also negatively impacts the destinations that the specific journey can reach. These examples are of great value for network design and illustrate the importance of leveraging historical travel time data in decision-making.

These individual predicted journeys will probably not lead to changes in the network design, scheduling, or policy. However, many of these observed discrepancies will be important to PT agencies. Proving the existence of these discrepancies was the scope of RQ4. For future research, the outcomes of this thesis can be utilised to explore whether systemic patterns in the discrepancies can be observed.

Small differences

While incorporating dwell and travel time prediction enhances the accuracy of reachability analysis and it was observed that there are discrepancies between the original and predicted schedules. The size differences between the isochrones in Figures 5.22, 5.23 and 5.24 are relatively small. This means that the reachability does not change considerably for some cases. The real-world isochrone difference ranges from approximately 200 to 1000 metres. This corresponds to 1 to 5 minutes of walking on a total journey time of 60 minutes.

This observation was discussed with a PT engineer who specialises in reachability analysis using Conveyal. He stated that these differences are significant enough to impact the efficiency of the analysed PT network. Understanding the actual predicted reachability is of great value when consulting on scheduling and routing to PT agencies.

Time-of-day patterns

In the EDA and prediction model development, it was concluded that the departure time from the first stop significantly impacts the travel and dwell times of that journey. This pattern could also be observed in the reachability analysis in Section 5.7.7. Here, the same pattern is that a journey in the morning rush hour between 07:00-08:00 takes longer than a journey departing between 14:00 and 15:00. This led to a decreased reachability of the journey in the morning of the day. It is significant for PT research that these patterns are also apparent when visualising the reachability.

6.2 Implications

This section describes the relevance and implications of the results in a broader context. First, the implications of predicting travel and dwell times of future journeys are explained. Second, the implications of enhanced reachability analysis are outlined.

6.2.1 Predicted travel times

Accurate travel and dwell time predictions can be used outside the context of reachability analysis. This information can provide PT agencies with valuable insights into travel patterns and operational inefficiencies. This information enables the agency to optimise routes, allocate resources and improve the passenger experience.

Trip-planning apps that passengers use can also leverage accurately predicted travel and dwell time. Solely using the schedule to compute travel time can be relatively misleading about what is actually happening in the real world. This is caused by the schedule, which is rounded to whole minutes. Adjusting for ongoing journeys that are already delayed is relatively easy and already happens in the trip-planning apps. However, it is also helpful to be aware of the expected performance of a passenger's journey in the future.

6.2.2 Realistic reachability

Reachability analysis is a powerful tool. The results showed significant differences when calculating travel time using the original schedule compared to predicted travel and dwell times. This approach provides a more realistic depiction of the actual achievable reachability of a region. This section will highlight three examples where this improved reachability analysis will impact designing networks and formulating policies.

Reachability analysis helps identify areas with large accessibility, indicating that the PT network effectively covers these regions. Conversely, areas with small reachability may highlight gaps in the network that need improvement. These gaps can also happen where it may seem the PT network is efficient, but bottlenecks or frequent delays negatively impact reachability.

This information is crucial for urban planners and PT engineers to optimise routes and schedules.

Besides analysing the regional coverage, examining the number of people efficiently served by the transportation network is essential. This means incorporating demographic data to better understand which segments of the population rely most on PT. Such insights can guide decisions on expanding services to underserved communities.

The next step is not only analysing the places where people live, but also taking into account the potential destinations of the population [39]. This reveals the economic and social benefits of the PT network. Measuring reachability is often done by analysing access to intermediate points (e.g. the nearest bus stop) rather than people's true destination. Jobs, businesses and services that can be reached in a certain amount of time will differ when the schedule or predicted travel and dwell times are used as input. Changing policies based on this analysis will improve social and economic inclusion by providing mobility options to disadvantaged groups.

In these increasingly elaborate reachability analysis examples, urban planners and PT engineers incorporate more factors for decision-making. A reachability tool that accurately represents the PT network is key for this. Interventions in policy and planning become more precise when real-world data is incorporated, rather than relying solely on schedules provided by public transportation companies.

6.3 Limitations

This section discusses four limitations of the results in this thesis. Firstly, the strict preprocessing approach is discussed. Secondly, the assumption of the same departure times as scheduled for overwriting the GTFS schedules is outlined. Lastly, the practical shortcomings of writing to the GTFS schedule is discussed.

6.3.1 Strict preprocessing

A strict preprocessing approach was used in the case study, where every journey in the historical travel time dataset had to adhere strictly to the schedule. This means the prediction models' training data did not consider defective journeys. Several of these journeys had no meaningful arrival and departure time data and were rightfully removed. However, there were also many journeys where only the stop order was incorrect, or a single stop would be missing. These journeys could be insightful as they most likely significantly influence reachability. For example, patterns in aborted bus journeys would be indicative of the predicted reachability. Devising more precise preprocessing algorithms that can handle a broad spectrum of anomalies in historical travel data would be a significant next step. However, for this thesis, only utilising fully completed bus journeys is insightful for analysing reachability.

6.3.2 First stop departure time

In this thesis, the assumption was made that for overwriting the original schedule with the predictions, the departure time from the first stop would be the same as the schedule. From here, the predicted travel and dwell times are cumulatively added to create the new schedule. However, in reality, buses may depart early or late, affecting the ability to make transfers and consequently impacting reachability. This does not necessarily mean that the bus drives slowly. This means that at each stop, there is a similar delay in the arrival and departure times caused by the late departure from the first stop, which impacts reachability.

The behaviour of a bus being delayed from the first stop departure is not currently captured in how the prediction models and the case study's data framework are set up. Solely predicting the travel and dwell times of the bus journeys still provides a more realistic reachability analysis and, therefore, answers the main research question. However, not modelling this behaviour means it does not entirely depict the real world.

6.3.3 Overwriting a GTFS schedule

A practical shortcoming involves writing the predicted travel and dwell times to a GTFS dataset not optimised for this specific usage. The issue arises because the same stop times corresponding to a single `trip_id` are used on different days, differentiated by their `service_id`. When writing the predicted travel and dwell times for journeys corresponding to this single trip in the GTFS dataset, it is impossible to have different predictions for these journeys on different days.

For example, two trips from g501 leaving Groningen station at 15:55 are defined by the same GTFS schedule, one on Wednesday and one on Thursday. A prediction model, such as RF or LSTM, will have different predictions for the Wednesday trip than the Thursday trip because it includes day-of-the-week features. Only one of the predictions can be written to the GTFS dataset, meaning a choice must be made regarding which prediction to implement. This issue can be resolved by creating the GTFS dataset from scratch with custom `trip_ids`, ensuring that no journeys on different days refer to the same stop times in `stop_times.txt`. However, this is a cumbersome task as the schedule is sourced from OVapi.nl and is set up this way.

6.4 Future research

This section provides several recommendations for future scientific research. Section 6.5 provides a more practical approach to how the results can be leveraged by, for example, a PT engineer or urban planner.

6.4.1 Delayed departure prediction

In Section 6.3.2, the limitation of taking the departure time of the first and cumulatively adding the predicted travel and dwell times was discussed. For future research, it is beneficial to consider delayed departures from the first stop, as it affects reachability. This could be altered by predicting the departure delay from the first stop compared to the schedule.

Practically, this could be done by adding a feature to every journey in the dataset: the delay between the scheduled departure from the first stop and the actual departure in the historical travel time dataset. This feature will also be predicted for future journeys. This will make the reachability analysis more realistic and provide the prediction model with extra information. For example, a journey with a delayed departure might demonstrate certain patterns in the travel and dwell times, which a complex model such as LSTM might pick up on.

6.4.2 Prediction models

This thesis offers an initial insight into the requirements for a prediction model to forecast future journeys based on historical travel times. It suggests that complex models, such as RF and LSTM, possess the necessary complexity for this task. The approaches in this thesis are relatively simple. More advanced deep learning models, such as ensemble models, gated recurring units and transformer-based models, could be essential to unveil the true patterns of the historical travel time data.

Enhancing the training dataset is crucial for the future development of prediction models, ensuring that true patterns are accurately identified. Below are several suggestions for improving the training dataset:

- Incorporate additional datasets such as weather, traffic, ticketing or exact GPS data.
- Collect a historical travel time dataset that spans a longer time interval than the dataset considered in this thesis's case study (>2 months).
- Include journeys that do not strictly adhere to the schedule, which involves enhancing the preprocessing algorithms used for historical travel time data.
- Engineer statistical and aggregated features, such as mean, median, or variance, over a rolling window.

Besides making the prediction models more elaborate, it would also be insightful if a more generalised approach could serve as a good input for reachability analysis. This could, for example, be done by predicting the same travel times for all journeys within a certain time window. Another example is to divide the journeys into larger segments comprising multiple stops, instead of dividing them into individual stops. A final example would be predicting the total time the journey takes from beginning to end and dividing this proportionally over all stops and links. These examples would decrease the detail level, but could still be useful and reliable as input for reachability analysis.

6.5 Practical recommendations

This section provides a PT engineer with practical and straightforward methods for assessing reachability using predicted bus travel times. These techniques can be efficiently applied to analyse new PT networks. The engineer does not need the comprehensive research depth of this thesis for their city's assessment. Below are four pointers for the initial setup of the research:

- **Analyse the historical travel times:** Carefully examine the AVL dataset's messages to decide on appropriate preprocessing techniques. Identifying historical travel time data anomalies is crucial, including handling defective, rerouted, and cancelled journeys.
- **Baseline prediction model:** Start with a simple, interpretable model like a time-dependent HA model. This model is easy to implement and provides a solid foundation.
- **Sufficient training data:** Ensure the training dataset is large enough, ideally encompassing up to a year of representative data.
- **Use analysis tools:** Utilise tools like Conveyal to display and analyse isochrones. Developing a program from scratch is time-intensive, whereas tools like Conveyal offer efficient analysis features. Overwriting the GTFS schedule with predictions is relatively straightforward.

These points provide a solid starting point for a comprehensive reachability analysis. The most critical step is analysing historical travel time data, which will reveal many patterns and insights. After this, consider the following advanced steps:

- **Advanced prediction models:** Implement more sophisticated models like RF or LSTM to enhance travel and dwell time predictions.
- **Delayed departure prediction**: Include the variable delayed departures from the first stop to represent real-world operations better.

- **Population-specific analysis:** Expand the analysis to assess the specific parts of the population served by the PT network. Investigate destinations of interest, such as jobs, schools, and medical facilities, within the reachability isochrone.

7 Conclusion

This thesis set out to enhance reachability analysis by providing a more realistic representation of the performance of the bus. This representation was achieved by predicting travel and dwell times of future journeys using the historical travel time data. The comparison between four prediction models indicated that the LSTM deep learning model had the best accuracy for this task. This model's predicted travel and dwell times better represented real-world bus reachability than the schedule could. This leads to cases of increased reachability, decreased reachability, or missed transfers.

Firstly, the literature research gave rise to the four prediction models that would be investigated for the task of travel time prediction. HA, VAR, RF and LSTM proved to be an insightful selection of prediction models during the model development phase. Each model improved the understanding of the patterns in the travel time data. Additionally, the effective case study data framework enabled the manipulation of the KV6 dataset and the GTFS dataset to be the input of Conveyal. This process was highly efficient and successfully facilitated the reachability analysis. In this reachability analysis, the expected discrepancies between the original and predicted schedule were found.

For future research, incorporating delayed departure prediction into the prediction models will be beneficial for a complete representation of the real-world operations. Additionally, employing more different prediction models, such as ensemble models, gated recurring units and transformer-based models can improve prediction accuracy. These models could also be combined with a more elaborate dataset, which, for example, combines weather data, traffic data and improved preprocessing of defective journeys.

Predicting future journeys provides valuable insights into the optimal design of the PT network. This can be achieved solely using historical travel time data. Besides, the primary goal of a transportation network is to achieve good reachability. It strives to connect people to their desired destinations quickly and conveniently. This thesis offers travel time prediction models that improve reachability analysis, which is invaluable to improve the sustainable and efficient PT networks of the future.

Bibliography

[1] J. Beaudoin, Y. H. Farzin, and C.-Y. C. Lin Lawell, “Public transit investment and sustainable transportation: A review of studies of transit’s impact on traffic congestion and air quality,” *Research in Transportation Economics*, vol. 52, pp. 15–22, 2015, Sustainable Transportation, ISSN: 0739-8859. DOI: https://doi.org/10.1016/j.retrec.2015.10.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0739885915000487.

[2] I. Makarova, A. Pashkevich, and K. Shubenkova, “Ensuring sustainability of public transport system through rational management,” *Procedia Engineering*, vol. 178, pp. 137–146, 2017, RelStat-2016: Proceedings of the 16th International Scientific Conference Reliability and Statistics in Transportation and Communication October 19-22, 2016. Transport and Telecommunication Institute, Riga, Latvia, ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2017.01.078. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705817300784.

[3] A. Carrel, A. Halvorsen, and J. L. Walker, “Passengers’ perception of and behavioral adaptation to unreliability in public transportation,” *Transportation Research Record*, vol. 2351, no. 1, pp. 153–162, 2013. DOI: 10.3141/2351-17. [Online]. Available: https://doi.org/10.3141/2351-17.

[4] M. D. Abkowitz and I. Engelstein, “Factors affecting running time on transit routes,” *Transportation Research Part A: General*, vol. 17, no. 2, pp. 107–113, 1983, ISSN: 0191-2607. DOI: https://doi.org/10.1016/0191-2607(83)90064-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/019126078390064X.

[5] B. Yao, P. Hu, X. Lu, J. Gao, and M. Zhang, “Transit network design based on travel time reliability,” *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 233–248, 2014, Special Issue on "Nature-Inspired Optimization Techniques in Transportation Planning and Operation", ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2013.12.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X13002647.

[6] P. G. Furth, T. H. J. Muller, J. G. Strathman, and B. Hemily, “Designing automated vehicle location systems for archived data analysis,” *Transportation Research Record*, vol. 1887, no. 1, pp. 62–70, 2004. DOI: 10.3141/1887-08. [Online]. Available: https://doi.org/10.3141/1887-08.

[7] Stichting OpenGeo, *Dutch real-time transit data*, 2013-2019. [Online]. Available: https://ndovloket.nl/.

[8] E.-H. Chung and A. Shalaby, “Expected time of arrival model for school bus transit using real-time global positioning system-based automatic vehicle location data,” *Journal of Intelligent Transportation Systems*, vol. 11, no. 4, pp. 157–167, 2007. DOI: 10.1080/15472450701649398. eprint: https://doi.org/10.1080/15472450701649398. [Online]. Available: https://doi.org/10.1080/15472450701649398.

[9] Z. Ma, H. N. Koutsopoulos, L. Ferreira, and M. Mesbah, “Estimation of trip travel time distribution using a generalized markov chain approach,” *Transportation Research Part C: Emerging Technologies*, vol. 74, pp. 1–21, 2017, ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2016.11.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X16302248.

[10] B. A. Kumar, L. Vanajakshi, and S. C. Subramanian, “Bus travel time prediction using a time-space discretization approach,” *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 308–332, 2017, ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2017.04.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X17301080.

[11] C. García-Mauriño, P. J. Zufiria, and A. Jarabo-Peñas, “Improving bus arrival time predictors using only public transport api data,” *Transportation Letters*, vol. 16, no. 8, pp. 804–813, 2024. DOI: 10.1080/19427867.2023.2245994. eprint: https://doi.org/10.1080/19427867.2023.2245994. [Online]. Available: https://doi.org/10.1080/19427867.2023.2245994.

[12] O. Alam, A. Kush, A. Emami, and P. Pouladzadeh, “Predicting irregularities in arrival times for transit buses with recurrent neural networks using gps coordinates and weather data,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 7, pp. 7813–7826, Sep. 2020. DOI: 10.1007/s12652-020-02507-9.

[13] Q. Han, K. Liu, L. Zeng, G. He, L. Ye, and F. Li, “A bus arrival time prediction method based on position calibration and lstm,” *IEEE Access*, vol. 8, pp. 42 372–42 383, 2020. DOI: 10.1109/ACCESS.2020.2976574.

[14] B. Tesfaye, N. Augsten, M. Pawlik, M. H. Böhlen, and C. S. Jensen, “Speeding up reachability queries in public transport networks using graph partitioning,” *Information Systems Frontiers*, vol. 24, no. 1, pp. 11–29, Aug. 2021. DOI: 10.1007/s10796-021-10164-2. [Online]. Available: https://doi.org/10.1007/s10796-021-10164-2.

[15] J. Greenfeld, “Automatic vehicle location (avl) for transit operation,” in *2000 10th Mediterranean Electrotechnical Conference. Information Technology and Electrotechnology for the Mediterranean Countries. Proceedings. MeleCon 2000 (Cat. No.00CH37099)*, vol. 2, 2000, 656–659 vol.2. DOI: 10.1109/MELCON.2000.880019.

[16] B. Predic, D. Rančic, D. Stojanovic, and A. Milosavljevic, “Automatic vehicle location in public bus transportation system,” *Annual Conference on Computers*, pp. 675–680, Jul. 2007. [Online]. Available: https://www.researchgate.net/profile/Aleksandar_Milosavljevic3/publication/228996934_Automatic_vehicle_location_in_public_bus_transportation_system/links/0046353cd723a76cee000000.pdf.

[17] Y. Yan, Z. Liu, and Y. Bie, “Performance evaluation of bus routes using automatic vehicle location data,” *Journal of Transportation Engineering*, vol. 142, no. 8, p. 04 016 029, 2016. DOI: 10.1061/(ASCE)TE.1943-5436.0000857. eprint: https://ascelibrary.org/doi/pdf/10.1061/%28ASCE%29TE.1943-5436.0000857. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29TE.1943-5436.0000857.

[18] L. D’Acierno, A. Cartenì, and B. Montella, “Estimation of urban traffic conditions using an automatic vehicle location (avl) system,” *European Journal of Operational Research*, vol. 196, no. 2, pp. 719–736, 2009, ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2007.12.053. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221708003482.

[19] Suwardo, N. Madzlan, and K. Ibrahim, “ARIMA models for bus travel time prediction,” *No journal*, Jun. 2010. [Online]. Available: http://dspace.unimap.edu.my:80/handle/123456789/13714.

[20] S. Maiti, A. Pal, A. Pal, T. Chattopadhyay, and A. Mukherjee, “Historical data based real time prediction of vehicle arrival time,” in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, pp. 1837–1842. DOI: 10.1109/ITSC.2014.6957960.

[21] J. Li, J. Gao, Y. Yang, and H. Wei, “Bus arrival time prediction based on mixed model,” *China Communications*, vol. 14, no. 5, pp. 38–47, 2017. DOI: 10.1109/CC.2017.7942193.

[22] S. S. A. B. Anil Kumar R. Jairam and L. Vanajakshi, "Real time bus travel time prediction using k-nn classifier," *Transportation Letters*, vol. 11, no. 7, pp. 362–372, 2019. DOI: 10.1080/19427867.2017.1366120. eprint: https://doi.org/10.1080/19427867.2017.1366120. [Online]. Available: https://doi.org/10.1080/19427867.2017.1366120.

[23] X. Zhang, L. Lauber, H. Liu, J. Shi, M. Xie, and Y. Pan, "Travel time prediction of urban public transportation based on detection of single routes," *PLOS ONE*, vol. 17, pp. 1–23, Jan. 2022. DOI: 10.1371/journal.pone.0262535. [Online]. Available: https://doi.org/10.1371/journal.pone.0262535.

[24] M. T. Hagan, H. B. Demuth, and M. Beale, *Neural network design*, 2nd ed. Dec. 1995. [Online]. Available: https://hagan.okstate.edu/NNDesign.pdf.

[25] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Jan. 2001. DOI: 10.1023/a:1010933404324. [Online]. Available: https://doi.org/10.1023/a:1010933404324.

[26] M. Awad and R. Khanna, *Efficient learning machines*. Jan. 2015. DOI: 10.1007/978-1-4302-5990-9. [Online]. Available: https://doi.org/10.1007/978-1-4302-5990-9.

[27] J. Amita, J. S. Singh, and G. P. Kumar, "Prediction of bus travel time using artifical neural network," *International Journal for Traffic and Transport Engineering*, vol. 5, no. 4, pp. 410–424, Dec. 2015. DOI: 10.7708/ijtte.2015.5(4).06. [Online]. Available: https://doi.org/10.7708/ijtte.2015.5(4).06.

[28] B. Yu, H. Wang, W. Shan, and B. Yao, "Prediction of Bus Travel Time Using Random Forests Based on Near Neighbors," *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 4, pp. 333–350, Nov. 2017. DOI: 10.1111/mice.12315. [Online]. Available: https://doi.org/10.1111/mice.12315.

[29] J. Ma, J. Chan, G. Ristanoski, S. Rajasegarar, and C. Leckie, "Bus travel time prediction with real-time traffic information," *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 536–549, 2019, ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2019.06.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X18309082.

[30] E. Chondrodima, H. Georgiou, N. Pelekis, and Y. Theodoridis, "Public transport arrival time prediction based on gtfs data," in *Machine Learning, Optimization, and Data Science*, Cham: Springer International Publishing, 2022, pp. 481–495, ISBN: 978-3-030-95470-3.

[31] S. Hochreiter and J. Schmidhuber, "Long Short-Term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. DOI: 10.1162/neco.1997.9.8.1735. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[32] J. Pang, J. Huang, Y. Du, H. Yu, Q. Huang, and B. Yin, "Learning to predict bus arrival time from heterogeneous measurements via recurrent neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3283–3293, 2019. DOI: 10.1109/TITS.2018.2873747.

[33] P. He, G. Jiang, S.-K. Lam, and D. Tang, "Travel-time prediction of bus journey with multiple bus trips," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 11, pp. 4192–4205, 2019. DOI: 10.1109/TITS.2018.2883342.

[34] H. Liu, H. Xu, Y. Yan, Z. Cai, T. Sun, and W. Li, "Bus arrival time prediction based on lstm and spatial-temporal feature vector," *IEEE Access*, vol. 8, pp. 11 917–11 929, 2020. DOI: 10.1109/ACCESS.2020.2965094.

[35] E. Chondrodima, H. Georgiou, N. Pelekis, and Y. Theodoridis, “Particle swarm optimization and rbf neural networks for public transport arrival time prediction using gtfs data,” *International Journal of Information Management Data Insights*, vol. 2, no. 2, p. 100 086, 2022, ISSN: 2667-0968. DOI: https://doi.org/10.1016/j.jjimei.2022.100086. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667096822000295.

[36] M. D. Domenico, A. Solé-Ribalta, S. Gómez, and A. Arenas, “Navigability of interconnected networks under random failures,” *Proceedings of the National Academy of Sciences*, vol. 111, no. 23, pp. 8351–8356, 2014. DOI: 10.1073/pnas.1318469111. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.1318469111.

[37] M. J. Williams and M. Musolesi, “Spatio-temporal networks: Reachability, centrality and robustness,” *Royal Society Open Science*, vol. 3, no. 6, p. 160 196, 2016. DOI: 10.1098/rsos.160196. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsos.160196. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rsos.160196.

[38] K. Hirako, S. Kani, Y. Morisaki, M. Fujiu, T. Nishino, and J. Takayama, “Estimations of Bus Stop Territories using Reachable Area Analysis Focusing on Travel Behavior of Elderly to Medical Facilities,” *International Journal of Engineering Research and*, vol. 9, no. 6, Jun. 2020. [Online]. Available: https://www.ijert.org/research/estimations-of-bus-stop-territories-using-reachable-area-analysis-focusing-on-travel-behavior-of-elderly-to-medical-facilities-IJERTV9IS060386.pdf.

[39] R. Arbex and C. B. Cunha, “Estimating the influence of crowding and travel time variability on accessibility to jobs in a large public transport network using smart card big data,” *Journal of Transport Geography*, vol. 85, p. 102 671, 2020, ISSN: 0966-6923. DOI: https://doi.org/10.1016/j.jtrangeo.2020.102671. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0966692319300092.

[40] L. E. Olsson, M. Friman, and K. Lättman, “Accessibility barriers and perceived accessibility: Implications for public transport,” *Urban Science*, vol. 5, no. 3, 2021, ISSN: 2413-8851. DOI: 10.3390/urbansci5030063. [Online]. Available: https://www.mdpi.com/2413-8851/5/3/63.

[41] R. Kujala, C. Weckström, M. N. Mladenović, and J. Saramäki, “Travel times and transfers in public transport: Comprehensive accessibility analysis based on pareto-optimal journeys,” *Computers, Environment and Urban Systems*, vol. 67, pp. 41–54, 2018, ISSN: 0198-9715. DOI: https://doi.org/10.1016/j.compenvurbsys.2017.08.012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0198971517300923.

[42] H. Lütkepohl, *New Introduction to Multiple Time Series Analysis*. Jan. 2005. DOI: 10.1007/978-3-540-27752-1. [Online]. Available: https://doi.org/10.1007/978-3-540-27752-1.

[43] H. Akaike, “A new look at the statistical model identification,” *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974. DOI: 10.1109/TAC.1974.1100705.

[44] P. Werbos, “Backpropagation through time: What it does and how to do it,” *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990. DOI: 10.1109/5.58337.

[45] Keras, *Keras documentation*. [Online]. Available: https://keras.io/api/.

[46] A. H. Victoria and G. Maragatham, “Automatic tuning of hyperparameters using Bayesian optimization,” *Evolving Systems*, vol. 12, no. 1, pp. 217–223, May 2020. DOI: 10.1007/s12530-020-09345-2. [Online]. Available: https://doi.org/10.1007/s12530-020-09345-2.

[47] P. Groningen, *Verbeteren OV-bereikbaarheid Zernike Campus Groningen*, Jan. 2022. [Online]. Available: https://groningen.stateninformatie.nl/document/11477237/1/3__OV-ontsluiting_Zernike_II_-eindrapportage_-_definitief_III_-_januari_2022.

[48] Movares, *Verkenning betere bereikbaarheid Zernike Campus Groningen - Movares Smart Urban Engineering*, Nov. 2024. [Online]. Available: https://movares.com/projecten/verkenning-betere-bereikbaarheid-zernike-campus-groningen/.

[49] Qbuzz, *Q-link*. [Online]. Available: https://www.qbuzz.nl/gd/reis-plannen/soortenbussen/q-link.

[50] *OVapi*. [Online]. Available: https://gtfs.ovapi.nl/.

[51] gtfs.org, *What is GTFS? - General Transit Feed Specification*, 2024. [Online]. Available: https://gtfs.org/getting_started/what_is_GTFS/.

[52] Conveyal, *Conveyal - Evaluate changes to your public transportation system*, 2024. [Online]. Available: https://conveyal.com/.

[53] E. Lewinson, *Three Approaches to Encoding Time Information as Features for ML Models | NVIDIA Technical Blog*, Aug. 2022. [Online]. Available: https://developer.nvidia.com/blog/three-approaches-to-encoding-time-information-as-features-for-ml-models/.

[54] Statsmodels, *statsmodels.tsa.vector*$_ar.var_model.VAR$, Apr. 2025. [Online]. Available: https://www.statsmodels.org/dev/generated/statsmodels.tsa.vector_ar.var_model.VAR.html.

[55] scikit-learn, *RandomForestRegressor*. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html.

[56] P. Ramachandran, B. Zoph, and Q. V. Le, "SWISH: a SELF-GATED ACTIVATION FUNCTION," *Google*, Oct. 2017. [Online]. Available: https://arxiv.org/pdf/1710.05941v1.

[57] K. He, X. Zhang, S. Ren, and J. Sun, *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification*, 2015. arXiv: 1502.01852 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1502.01852.

[58] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Y. W. Teh and M. Titterington, Eds., ser. Proceedings of Machine Learning Research, vol. 9, Chia Laguna Resort, Sardinia, Italy: PMLR, May 2010, pp. 249–256. [Online]. Available: https://proceedings.mlr.press/v9/glorot10a.html.

[59] A. M. Saxe, J. L. McClelland, and S. Ganguli, *Exact solutions to the nonlinear dynamics of learning in deep linear neural networks*, 2014. arXiv: 1312.6120 [cs.NE]. [Online]. Available: https://arxiv.org/abs/1312.6120.

[60] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1412.6980.

A Research paper

Realistic prediction of public transport travel times for accurate representation of multi-modal reachability

Siewe Knook

2025-04-25

Abstract

This paper uses historical bus travel time data to predict future travel times, aiming to improve public transportation network reachability. Four prediction models were evaluated: Historical Average, Vector AutoRegression, Random Forest, and Long Short-Term Memory (LSTM) deep neural network. A case study with six bus lines in Groningen used KoppelVlak 6 and GTFS schedule datasets. The analysis found that complex models like Random Forest and LSTM provided the most accurate predictions, with the time of day being a key factor. Integrating these predictions into reachability analysis showed changes in reachability and missed transfers compared to the original schedule. The findings highlight that accurate travel time predictions can enhance reachability analysis and help identify and address systemic issues in public transportation networks.

1 Introduction

High-quality Public Transport (PT) services improve the lives of citizens considerably. It can reduce traffic congestion and improve air quality by reducing carbon emissions [1]. Also, implementing efficient and well-managed public transport (PT) will make a city more sustainable [2]. A key part of efficient PT is providing passengers and engineers with accurate information on the arrival and departure times of PT services. For PT engineers, accurate predictions are transformative tools in the design and optimisation of services. The schedule often does not reflect the actual daily operation of the PT network well. Utilising accurate historical information on arrival times and departure times is beneficial when designing routes, schedules or vehicle allocation for a PT network. Engineers rely on predictive data to analyse a PT system. These insights enable better scheduling, bottleneck identification and infrastructure planning [3]. By integrating real-time data into design processes, engineers can create networks that minimise delays and enhance passenger satisfaction.

The need for the incorporation of this data is because PT journeys will often deviate from the schedule. PT is operating in an urban environment where disturbances are likely. This means that it is difficult to maintain a deterministic travel time because traffic and weather conditions can vary greatly. There has been a great development in collecting data on the GPS location of PT vehicles. This means that for almost all PT journeys in the Netherlands, there is historical data on arrival and departure times at stops along the route [4]. There has been significant research in the 20th century to leverage this kind of data to make accurate decisions on the travel time of PT [5][6][7][8].

The research of this paper will focus on the prediction of travel and dwell times of bus journeys. Travel time can be defined as the time to reach a destination or cross a link of the public transport network. Travel time prediction refers to the prediction of current or future travel times. Dwell time is the time that a bus is stationary at a stop, meaning the difference between the arrival and departure time at the stop. Four prediction models will be developed in this thesis: a baseline Historical Average (HA) prediction model, a Vector Auto-Regression (VAR) prediction model, a Random Forest (RF) prediction model and a deep learning Long Short-Term Memory (LSTM) neural network prediction model.

The second focus point of this paper is insightful reachability information, which is an important part of high-quality PT. Reachability, in the context of public transport, refers to the ability to reach a certain destination in a certain amount of time. Visual representation, such as an isochrone map or graph, is a key method to convey reachability information. Interactive software tools which generate and evaluate these visualisations are of great value to PT engineers. One of those tools is Conveyal, which utilises pre-planned schedules to present the travel times for PT. This paper aims to predict more accurate travel and dwell times using historical travel time data to implement in this reachability tool.

The contributions of this research can be summarised in two points. Firstly, the developed models will predict future journeys using solely historical travel time data. This data will be engineered accordingly to create time features, such as hour of the day, day of the week, and lagged variables. Secondly, this paper aims to showcase the benefits of incorporating these travel and dwell time predictions into reachability analysis.

The research is organised as follows: Section 2 presents the literature review. The insights gained from this section will be used to set up the methodology (3), which outlines the Exploratory Data Analysis (EDA), the prediction models development and reachability analysis. Section 4 provides a description of the case study of Groningen and the datasets. Afterwards, Section 5 presents the results of the EDA, prediction models and reachability analysis. Finally, Section 6 offers concluding remarks and discusses the implications of the research.

2 Literature review

The widespread implementation of AVL provides large amounts of data on the operation of PT. Agencies and researchers can use this data to observe, collect and analyse location information about a vehicle. Ultimately, this data can be used to make informed decisions on network planning and improving passengers' experience [9]. Modern AVL systems rely on GPS systems to receive the longitude and latitude of the bus in real-time. This data is often enriched with arrivals and departures at stops during the PT journey [10].

This data has been applied to a wide variety of tasks. Performance analysis of the PT bus network, for example, is useful for the operators. Yan *et al.* [11] utilised AVL data in statistical analysis to assess spatial and temporal patterns during various route segments and time-of-day intervals. D'Acierno *et al.* [12] propose a method to estimate the urban traffic conditions based on the AVL data of the buses in the city. The method was able to accurately monitor traffic conditions not only in the bus lane but throughout the entire road network. The application of AVL data that his thesis focuses on is travel time prediction.

The most popular parametric models are time series models and regression models. With time series models, the assumption is made that there exists a pattern between historically observed data and future travel time patterns. Popular methods that fall under this category are Moving Average (MA) [13][5], AutoRegressive (AR) and AutoRegressive Integrated Moving Average (ARIMA) [14]. The main advantage of these methods is their fast computational speed and ease of implementation. Chung and Shalaby [5] propose an estimated arrival time model which incorporates data on the last five days of operations and the present day's operational conditions. Maiti *et al.* [13] propose a historical data model which considers vehicle trajectory and timestamps as input features. A more advanced ARIMA model is proposed by Suwardo *et al.* [14] for predicting bus travel time solely based on past observations. The most complex would be Ma *et al.* [6], who developed a model that captures correlations among link travel times conditional on the underlying traffic states. The simplicity of HA and ARIMA in implementation and ease of interpretation make them valuable tools for initial forecasting results. A limitation of parametric models, such as AR or MA, is their reliance on the last n previous observations. While this is often sufficient for short-term linear prediction, it can overlook important indicators from further in the past.

More advanced are Machine Learning (ML) models. Yu *et al.* [15] propose a hybridization approach of a RF model based on a near neighbours model. García-Mauriño *et al.* [8] also proposes a RF model which predicts travel times based on historical data. Ma *et al.* [16] proposes a segment-based bus route graph with two independent prediction models, which predict transit time and dwelling time, respectively. Chondrodima *et al.* [17] propose a method to use GTFS data in a framework for predicting PT arrival time. This framework combines a GTFS schedule with a real-time GTFS feed. Several machine learning algorithms are tested on this framework, and an ANN had the best performance. ML excels at handling non-linear patterns, which are frequently encountered in PT datasets. For instance, Chondrodima *et al.* [17] found that in their dataset, the most complex model, an artificial neural network (ANN), yielded the best prediction results.

Deep learning models are considered the state-of-the-art for public transport travel time prediction. These techniques include LSTM neural networks, Recurrent Neural Networks (RNN) and convolutional neural networks (CNN). These techniques are widely used for time-series forecasting problems. Pang *et al.* [18] deploy an RNN with an LSTM block to correct for the passing of the earlier bus stops. He *et al.* [19] predict bus journey time for an individual passenger by separately predicting riding and waiting time. Liu *et al.* [20] propose a hybrid model of LSTM and ANN based on a spatio-temporal feature vector. Alam *et al.* [21] and Han *et al.* [22] propose an LSTM model which is trained on GPS coordinates of transit buses. In general, deep learning methods have demonstrated superior predictive performance compared to traditional models such as HA, ARIMA and KF. These conventional models often struggle to capture the temporal dependencies and nonlinear relationships present in real-world bus data.

Reachability refers to the time that it takes a passenger to reach certain destinations. It is a crucial factor in urban planning and mobility, as it affects economic activity, social inclusion and overall quality of life. Some factors which influence the reachability are: coverage and network density, service frequency, availability, connectivity, speed.

As mentioned in the introduction in Section 1, the existing literature often focuses on the arrival time prediction of a bus that is currently in operation [22]. In these approaches, the prediction models consider real-time traffic and weather conditions, and the model can also use the arrival and departure times of the current bus trip. This paper will focus on the prediction of arrival time for future bus trips. He *et al.* [19] does consider passenger journeys in the future, but not the prediction of the future bus schedule. Short-term future prediction is applied by Zhang *et al.* [23].

Another aspect of the research gap is that these future predictions will be incorporated into a reachability analysis. Patterns in travel and dwell times will influence the speed of the bus network and, therefore, reachability. This temporal factor will be expressed more realistically. Current literature mainly utilises travel time prediction to provide passengers with arrival time information of an ongoing bus journey, not to enhance reachability analysis.

3 Methodology

Four prediction models are selected for analysis, increasing in complexity from a simple baseline model. The goal is for each model to capture the underlying patterns of travel and dwell times better, thereby increasing prediction accuracy and effectively handling the numerous variables involved in bus travel time prediction. The baseline model will be an HA model, chosen for its simplicity as it doesn't require complex models or assumptions. Maiti *et al.* [13] demonstrated the effectiveness of using average travel times for predictions. They also suggested making these averages dependent on the time of day, an approach that will be developed in this thesis. The second model will be a VAR model. Although ARIMA has been shown to effectively forecast public transport travel times [14], it is applied to a single variable and does not account for interdependencies between variables. The third model, RF, is the first machine learning model selected due to its ensemble-based nature, which combines multiple decision trees to reduce variance and enhance generalisation [15]. The most complex model will be an LSTM neural network. LSTM, in particular, is advanced in predicting time-series data and can easily incorporate other variables such as weather or traffic information [21][22]. The list of symbols can be found in Appendix A.

3.1 Dataset creation and analysis

Two data types are necessary for accurate bus travel and dwell time prediction, namely the schedule and the historical travel and dwell times. For the historical data, the source data can be in the form of arrival and departure times at certain stops along the bus routes or the dwell and travel times of the bus journey. This dataset should be represented as shown in Table 3 in Appendix B. Each row of this dataset is a journey along a certain bus route, and the columns are the travel and dwell times of that journey. This dataset should be presented as a time series indexed by the departure time from the first stop of the bus journey. Using this information the arrival and departure time data, the Equations 1 and 2 calculate dwell time $y_{k,j}$ and travel time $y_{l,j}$, which form the columns of the dataset.

$$y_{k,j} = dt_{k,j} - at_{k,j} \tag{1}$$

Where $y_{k,j}$ is the dwell time at stop k for journey j, which is calculated by taking the difference between the arrival time $at_{k,j}$ and departure time $dt_{k,j}$ at stop k for journey j.

$$y_{l,j} = at_{k,j} - dt_{k-1,j} \tag{2}$$

Where $y_{l,j}$ is the travel time for link l for journey j which spans between stop k and the preceding stop $k-1$. Travel time $y_{l,j}$ is calculated by taking the difference between the departure time $dt_{k-1,j}$ from stop $k-1$ and the arrival time $at_{k,j}$ at stop k for journey j.

Table 3 should not have any missing values. This means that preprocessing steps such as imputation, duplicate handling and checking schedule adherence might be necessary steps to acquire this data. Summarising, visualising and interpreting the created dataset of travel and dwell times.

3.2 Historical Average

The predictions for a certain route, such that the stop vector $\mathbf{k}_j$ and $\mathbf{l}_j$ are identical for all journeys considered, are the averages of travel times of the specific links and dwell times of specific stops. Equation 3 is used to calculate the average travel time $\hat{y}_l$ for a link l.

$$\hat{y}_l = \frac{1}{N}\sum_{j=1}^{N} y_{l,j} \tag{3}$$

In Equation 3, link l has been travelled N times in journeys in the selected dataset. $y_{l,j}$ is the travel time of a certain link l for a journey j. The average is taken over all the instances N that the link l has been travelled $\{y_{l,1}, y_{l,2}, ...y_{l,N}\}$. This will give the prediction for the travel time $\hat{y}_l$ for a link l.

Similar to predicting travel times, Equation 4 is used to predict the dwell time $\hat{y}_k$ at a stop k.

$$\hat{y}_k = \frac{1}{N}\sum_{j=1}^{N} y_{k,j} \tag{4}$$

In this equation, the predicted dwell time $\hat{y}_k$ at stop k $\hat{y}_k$ is calculated by taking the average of all the dwell times at this stop for all journeys in the dataset. The average is taken over all the instances that this stop has been used in a certain route $\{y_{k,1}, y_{k,2}, ...y_{k,N}\}$.

Applying Equations 3 and 4 to all links in $\mathbf{l}$ and stops in $\mathbf{k}$, respectively, will yield predicted travel and dwell times for the complete route. The predicted vector travel times $\hat{\mathbf{y}}_\mathbf{l} = \{\hat{y}_{l_1}, \hat{y}_{l_2}, ..., \hat{y}_{l_{n-1}}\}$ and predicted vector dwell times $\hat{\mathbf{y}}_\mathbf{k} = \{\hat{y}_{k_1}, \hat{y}_{k_2}, ..., \hat{y}_{k_n}\}$ can be used as for the reachability analysis.

The Equations 3 and 4 are adjusted to Equations 5 and 6. In these equations, t relates to a specific similar-sized interval $\langle t, t+dt\rangle$, the day is divided into. This means that $y_{l,t,j}$ and $y_{k,t,j}$ refer to the past travel and dwell times, respectively. Furthermore, $\hat{y}_{l,t}$ and $\hat{y}_{k,t}$ are the predicted travel and dwell times, respectively. This is similar to the equations presented in the previous Section 3.2.

$$\hat{y}_{l,t} = \frac{1}{N}\sum_{j=1}^{N} y_{l,t,j} \tag{5}$$

$$\hat{y}_{k,t} = \frac{1}{N}\sum_{j=1}^{N} y_{k,t,j} \tag{6}$$

3.3 Vector AutoRegression

The basic p-lag VAR model (denoted as VAR(p)) has the form as displayed in Equation 7 [24].

$$Y_t = A_1 Y_{t-1} + A_2 Y_{t-2} + \cdots + A_p Y_{t-p} + u_t \tag{7}$$

where:

- n means that $(2n-1)$ is the number of variables, travel and dwell times of a journey in the VAR model.
- Y_t is a $(2n-1)$-dimensional vector of dwell times y_k and travel times y_l at time step t, s.t. $Y_t = \{y_{k_1}, y_{l_1}, \ldots, y_{l_{n-1}}, y_{k_n}\}$.
- $A_i \quad (1, \ldots, p)$ are $(2n-1) \times (2n-1)$ coefficient matrices.
- $u_t \sim \mathcal{N}(0, \Sigma_u)$ is a white noise error term with zero mean and covariance matrix Σ_u.

Firstly, when training the VAR model, residual tests, such as checking for autocorrelation and normality, are conducted to ensure model adequacy. Next, the estimation procedure is performed on the training data. The parameters A_i are estimated using Ordinary Least Squares (OLS). $\hat{A}_i = (X'X)^{-1}X'Y_i$ where for each time series i the estimated $\hat{A}_i$ are calculated using this equation. Where X is the matrix of the lagged variables. Y_i is the vector of current values of the i-th time series.

The optimal lag order p is determined using the criteria Akaike Information Criterion (AIC). AIC is a statistical measure used to compare models by balancing the quality of the fit with model complexity. A lower AIC value indicates a better trade-off between model accuracy and complexity. When using it to compare VAR models, it ensures that the model is neither underfitting nor overfitting. After attaining an estimated VAR(p) model with an optimal lag value, the future dwell and travel times can be forecasted iteratively.

3.4 Random Forest regression

A decision tree is a flowchart-like structure where each internal node represents a decision based on a feature. The features in this case are travel and dwell times. Decisions typically involve whether a certain feature is higher or lower than a certain value. Depending on the outcome, the sample is led to another branch of the decision tree. The structure of the decision tree is as follows: it contains *root nodes* (topmost), *internal nodes* and *leaf nodes* (terminal).

When creating the decision tree, the algorithm selects the feature that provides the best split based on the criterion. The criterion for splitting is mean squared, which is equal to variance reduction as a feature selection criterion and minimises the L2 loss using the mean of each leaf node. The dataset is split based on the selected feature and the process is repeated recursively. This stops when a stopping condition is met, such as maximum depth, minimum samples per leaf or no further information gain.

An RF algorithm fits many decision tree regressors on subsamples of the training dataset and uses averaging to improve the predictive accuracy. The first step of the process is bootstrap sampling, where random rows of the training data are selected. After feature sampling, only a random subset of features is used for each decision tree. This means a selection of travel times. This ensures the diversity of the decision trees and avoids overfitting.

The features that the model will train on are the travel times $\mathbf{y_l}$ and dwell times $\mathbf{y_k}$. The model is implemented using `scikit-learn` library, namely the `RandomForestRegressor` class is used.

RF is not inherently a time series forecasting model. This means that lagged features must be created to enable the model to predict the time series based on historic travel and dwell times. The number of journeys that are considered for the lagged variables is defined by the hyperparameter n_{lags}. This means that the n_{lags} previous journeys' travel and dwell times are the features of the RF algorithm, and the corresponding current journey travel and dwell times are the target variable. This model will also be enriched with time of day and day of the week variables, constructed as sinusoidal waves [25].

After training the model, it is imperative to optimise its hyperparameters. The following hyperparameters are important to train for an RF model, `n_estimators`, `max_depth`, `min_samples_split`, `min_samples_leaf`, `max_features` and `bootstrap`.

When the RF is trained, there will be prediction decisions made on certain training features, these features can be analysed through feature importance evaluation. The goal is to investigate which features are determinative for the regression output of the RF model. This is done by analysing the Mean Decrease in Impurity (MDI) evaluation. By combining both these methods, the risk of wrongly evaluating correlated features is mitigated.

3.5 Long Short-Term Memory neural network

An LSTM unit is a type of recurrent neural network, designed to overcome the vanishing gradient problem. This unit is designed to handle sequential data and long-range dependencies more effectively than standard RNNs. This is done by incorporating memory cells and gating mechanisms that regulate the flow of information. The key feature is their ability to selectively remember and forget information through a set of gating mechanisms [26].

An LSTM unit has a memory cell regulated by three gates. The *forget gate* decides which past information to discard, the *input gate* determines which new information to store, and the *output gate* decides what information to pass to the next time step as the hidden state. These gates work together to manage the flow of information within the LSTM unit.

These gates allow LSTM networks to selectively retain or discard information. This behaviour enables the network to learn long-term dependencies. A deep neural is built up from different layers that propagate data through them. Each layer extracts and refines features from the data, gradually building up more abstract and high-level representations.

A neural network for sequential data typically includes several key layers. The *input layer* receives raw data, with each neuron representing a feature. *LSTM layers* capture long-term sequential patterns, while the dense layer transforms hidden states into suitable predictions, capturing non-linear relationships. Finally, *the reshape/output* layer produces the final prediction, with neurons corresponding to the target variables.

By varying the number of LSTM layers and evaluating the MAE results on the test dataset. The optimal number of LSTM layers can be found for the complexity of the data patterns. By testing multiple lines, a good structure can be found. It is also important to investigate whether the models are overfitting or underfitting.

When the optimal structure of the LSTM deep learning network is determined, the hyperparameters # of LSTM units, learning rate, # of epochs, optimiser, activation function, dropout rate and kernel initialiser are trained. It is also important to investigate the loss of the validation set and the training dataset during the training process. The loss function measures how well the model's predictions match the validation dataset. The loss function is the mean squared error.

3.6 Prediction model evaluation

MAE was selected because it is intuitive and robust for outliers. The actual travel time of a link y_l and the predicted travel time of the link $\hat{y}_l$. The MAE for link l is calculated using Equation 8, which is travelled N times in the test dataset. Equation 9, is the MAE for dwell time at stop k when travelled in a certain direction.

$$MAE_l = \sum_{i=1}^{N} \left| \frac{y_{l,i} - \hat{y}_{l,i}}{N} \right| \tag{8}$$

$$MAE_k = \sum_{i=1}^{N} \left| \frac{y_{k,i} - \hat{y}_{k,i}}{N} \right| \tag{9}$$

3.7 Reachability analysis

Reachability is defined as the ability to access a set of destinations within a given time frame from a specific starting point.

Let:

- S be the set of all possible starting points (e.g., bus stops).
- D be the set of all possible destinations.
- $T(s, d)$ be the travel time function, which gives the travel time from starting point $s \in S$ to destination $d \in D$

The reachability set $R(s, t)$ from a starting point s within a time threshold t is defined as:

$$R(s, t) = \{d \in D \mid T(s, d) \leq t\} \tag{10}$$

This set $R(s, t)$ includes all destinations d that can be reached from s within time t. Another set that can be defined is the ischrone $I(s, t)$, which is a contour that represents the boundaries of the reachability set. For a given starting point s and time threshold t, the isochrone $I(s, t)$ is the set of points that are exactly t units of away from s, defined in Equation 11

$$I(s, t) = \{d \in D \mid T(s, d) = t\} \tag{11}$$

The travel time function T estimates the time required to travel between two points in a transportation network. This function will be influenced by the predicted travel times $\hat{\mathbf{y}}_\mathbf{l}$ and dwell times $\hat{\mathbf{y}}_\mathbf{k}$. This mathematical framework helps to visualise and analyse how accessible different parts of an urban PT network are.

Plotting the isochrones on a map will visualise the areas that are reachable within specific time thresholds. This is part of the spatial analysis of reachability. Analysing the variation in travel and dwell times during the day will help to understand how reachability changes throughout the day. This would be temporal analysis.

4 Case study

To validate the proposed methodology, a case study is conducted, focusing on the bus network of Groningen. In this paper, Groningen's bus network will be analysed. Since 2019, the bus network in Groningen has been operated by QBUZZ. For this analysis, lines 1, 2, 3, 4, 5, and 6 are selected. This ensures comprehensive coverage of most directions. These lines are referred to in the data and official documentation as g501, g502, g503, g504, g505, and g506, respectively. Table 4 in Appendix C contains the details of the lines.

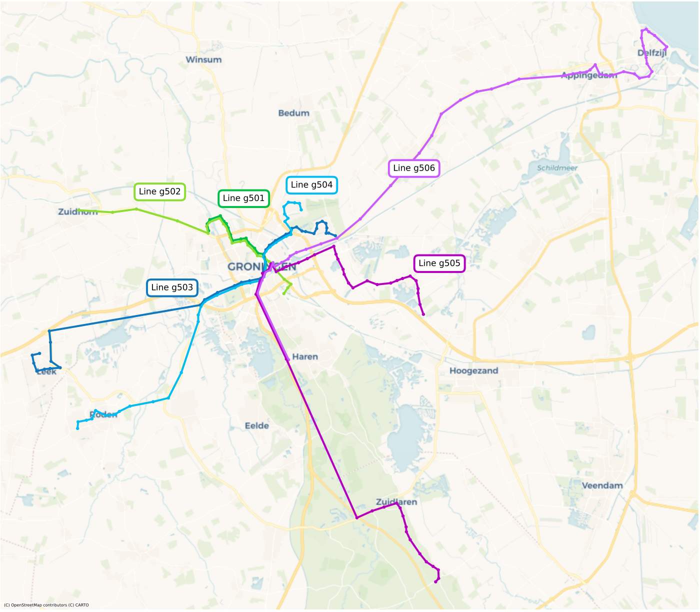


Figure 1 – The six bus lines in Groningen that are used in the case study.

Conveyal enables the visualisation of multi-modal transportation networks [27]. This will be used to assess the reachability of the Groningen network based on the predicted travel and dwell times. The datasets used for this are the KV6 dataset and the GTFS schedule sourced from the NDOV website [4]. The KV6 dataset provides the travel and dwell times of 19344 bus journeys that took place between September 1 and October 12, 2024. The GTFS schedule provides the schedule corresponding to these journeys.

5 Results

This section presents the results of the EDA (5.1), prediction models (5.2 and reachability analysis (5.3).

5.1 Exploratory data analysis

Figure 2 illustrates the average z-score of travel and dwell times for each hour of the day. A high z-score indicates slower journeys, while a low z-score indicates faster journeys. The distribution of these averages is represented by boxplots, which are displayed side-by-side in Figure 2.

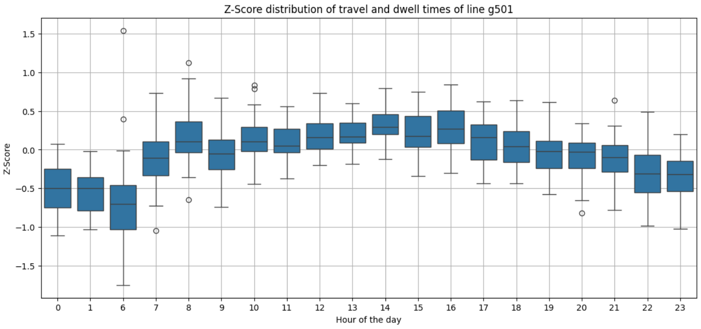


Figure 2 – Boxplots per hour of the average Z-score of the travel and dwell times of line g501 in direction 1.

Figure 2 visualises the travel and dwell times for line g501 in direction 1. The boxplots show that daytime values are generally higher than those for the evening and early morning, likely due to increased traffic during the day. Notably, the boxplot for 8:00 is higher than the others, indicating that the slowest journeys occur during this time. However, a similar peak is not observed for the afternoon rush hour, suggesting that journeys throughout the day are consistently slow. The fastest journeys seem to occur in the time interval of 6:00.

In the same manner as Figure 2, the impact of the day of the week is analysed. This can be seen for line g501 in direction 2 in Figure 3. Figure 3 shows that the boxplots for Monday through Thursday are quite similar. Friday's boxplot is higher, indicating that the slowest journeys of the week occur on that day. Saturday's boxplot is more stretched, suggesting that bus journeys can be either fast or quite slow. The boxplot for Sunday has the lowest z-score distribution, indicating that the fastest journeys of the week occur on that day, likely due to decreased ridership during the weekend.

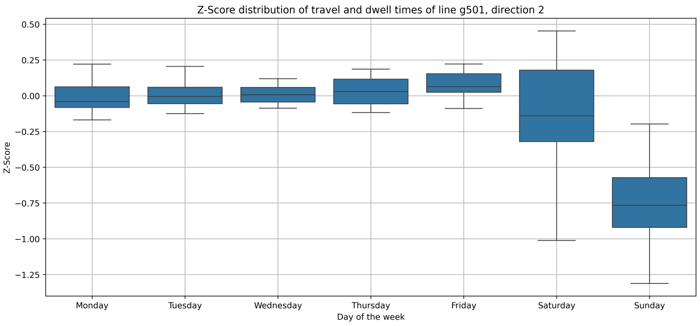


Figure 3 – Boxplots per weekday of the average Z-score of the travel and dwell times of line g501 in direction 1.

5.2 Prediction models

The four prediction models (HA, VAR, RF and LSTM) were trained. The predictions were tested against the test dataset and Table 1 presents the MAE results. The baseline HA model provided solid predictions for travel and dwell times, while the VAR models underperformed, especially for certain lines and directions. Complex models like RF and LSTM showed superior results, indicating that the relationship between travel and dwell times is non-linear. However, these models can overfit, particularly with smaller datasets, and their interpretability remains a challenge despite insights from feature importance analysis.

Table 1 – Complete overview of MAE results of the HA, VAR, RF and LSTM prediction models. The lowest MAE for each line and direction is in bold.

Direction	30 min HA	VAR	RF	LSTM
g501, 1	10.177	8.040	7.572	**6.743**
g501, 2	10.467	8.387	7.872	**7.034**
g502, 1	12.432	55.485	9.991	**9.485**
g502, 2	12.778	65.516	11.287	**10.435**
g503, 1	10.781	43.755	9.726	**8.818**
g503, 2	10.928	20.089	9.122	**9.086**
g504, 1	10.746	17.508	7.997	**7.276**
g504, 2	15.492	9.902	13.340	**8.422**
g505, 1	14.102	11.057	11.183	**11.054**
g505, 2	12.329	12.172	**9.391**	11.332
g506, 1	9.652	12.958	**8.726**	7.729
g506, 2	9.989	12.667	8.961	**7.961**

When plotting the MAE of the travel times on a map in Groningen of the HA model, it can be observed that the links in the city centre have the highest MAE. This means that taking the average for links in these dense urban regions is less predictive than for regions outside of the city. This can be seen for line 1 in Figure 4. This can mean that these busy links have a high variability of travel times.

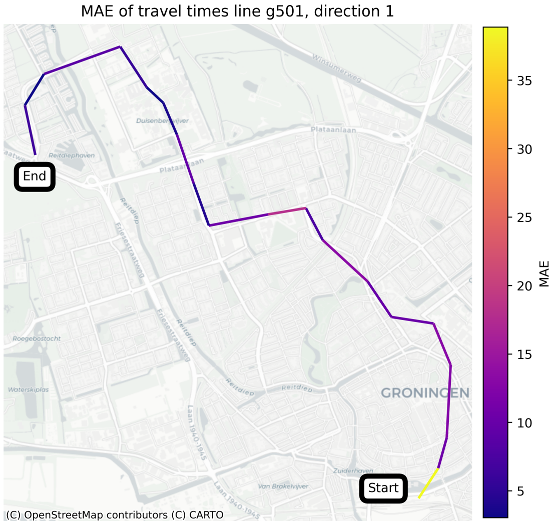

MAE of travel times line g501, direction 1
End
Start
GRONINGEN
35
30
25
20
15
10
5
MAE
(C) OpenStreetMap contributors (C) CARTO

Figure 4 – MAE of travel time of different links along the g501 bus route. This figure displays that the most difficult regions to predict in urban areas are the busy regions around the city centre.

The MDI has been analysed for all 12 directional RF models. Notably, in 5 out of the 12 models, a time feature emerged as one of the most influential features. This suggests that while the travel and dwell times of past journeys are important, the journey departure time of the predicted journey is even more critical. For these 5 lines and directions, an hourly time feature consistently played a prominent role in decision-making. This aligns with the findings in the EDA, which indicate that the time of day significantly impacts the length of travel and dwell times. Figure 5 illustrates this, showing the high influence of time features in these RF models.

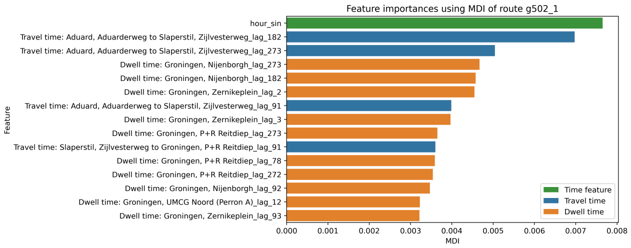


Figure 5 – Feature importance analysis of line g502, direction 1. The time feature hour_sin is influential for the prediction of future travel and dwell times.

For the LSTM model, Figure 6 illustrates the training process of the LSTM model on the data of line g504 in direction 1. At the end of training, there is a significant difference between the loss functions, which indicates overfitting. Additionally, the accuracy of the validation dataset stayed constant during training. This means that the model did not pick up on any patterns and it could not improve its initial guess. This same behaviour was observed for g501 in direction 2, g503 in both directions and g506 in direction 1. This suggests that the model's performance is suboptimal, and its predictive accuracy on unseen data might be limited.

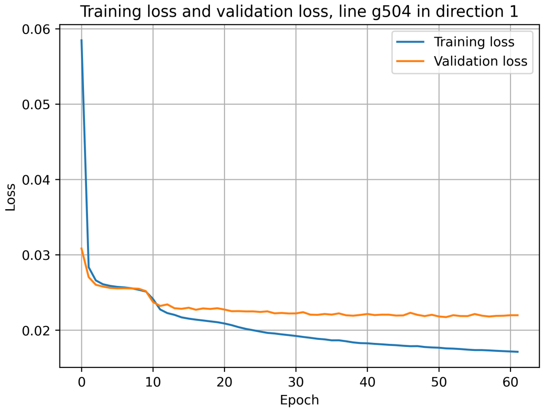


Figure 6 – Training set loss and validation set loss plotted during model training for each epoch. Training set accuracy and validation set accuracy during model training for each epoch. This is shown for the LSTM model of line g504 in direction 1.

The baseline HA model demonstrated solid predictions for travel and dwell times, with MAE values consistently ranging between 9.652 and 14.102 across all directions. In contrast, the VAR models performed poorly on the test dataset for lines g502, g503, and g504, failing to outperform the baseline model. Notably, the MAE for line g502 in direction 2 rose to 65.516, indicating that a short-term linear approach to predicting travel and dwell times is insufficient.

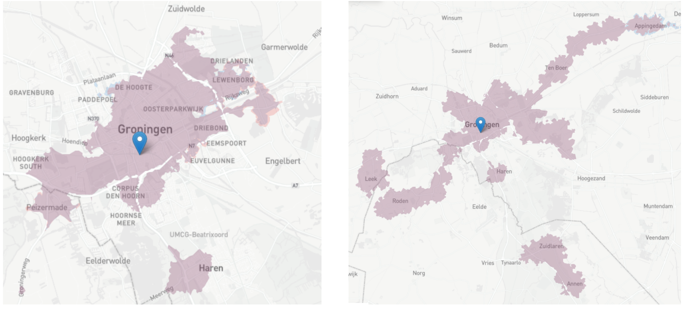
From Table 1, it is evident that complex models are more effective for predicting travel and dwell times. The LSTM model achieved the best results for all lines and directions, significantly improving upon the baseline HA model. This suggests that the relationship between travel and dwell times and their past values is non-linear. The time-dependent HA model significantly improved prediction results by incorporating time features, reflecting daily and weekly cycles. These features provided additional context for complex models like RF and LSTM, which showed good results, particularly with hourly features.

The complex ML models can capture details but are also prone to overfitting, especially with smaller datasets, as observed in some LSTM models. This means these models fixate more on the noise or random fluctuations in the travel and dwell times rather than the underlying patterns. Another drawback of the well-performing RF and LSTM models is model interpretability. Understanding the inner workings of these ML models is challenging. While feature importance analysis of the RF model provides some insights, it remains difficult to diagnose the decisive patterns for travel and dwell times.

As concluded in the EDA, the data contains a lot of noise. The prediction models set out to investigate whether meaningful patterns could be found. Even though the MAE results on the test dataset seem to have improved. The indecisive feature importance analysis of RF and the overfitting of the LSTM model suggest that the data does not contain strong patterns. The erraticness shown by the analysis of individual travel and dwell times of the VAR model does not show any convincing patterns that could be predicted.

5.3 Reachability analysis

Figure 7 show two figures for the same reachability analysis with different cutoff times. Figure 7a is cutoff at 40 minutes and Figure 7b is cutoff at 80 minutes.



(a) Cutoff time 40 minutes.

(b) Cutoff time 80 minutes.

Figure 7 – Reachability analysis. Starting point: Groningen Station. Saturday between 12:00 and 14:00. In these figures, the purple isochrone indicates that the predicted and original schedules overlap.

Based on the predicted travel and dwell times of the six bus lines, slower buses may cause missed transfers to other buses. This issue was observed in the reachability analysis conducted for Monday, October 24, 2024, between 07:00 and 09:00, starting from Groningen station.

In Figure 8, a reachability difference between the schedule and the LSTM prediction can be observed towards the northwest of the City of Groningen. This area, serviced by line 564, was not included in the travel and dwell time prediction, meaning its data is based on the ordinary schedule. The larger blue portion indicates that the ordinary schedule suggests you can travel farther in the same amount of time compared to the predicted schedule.

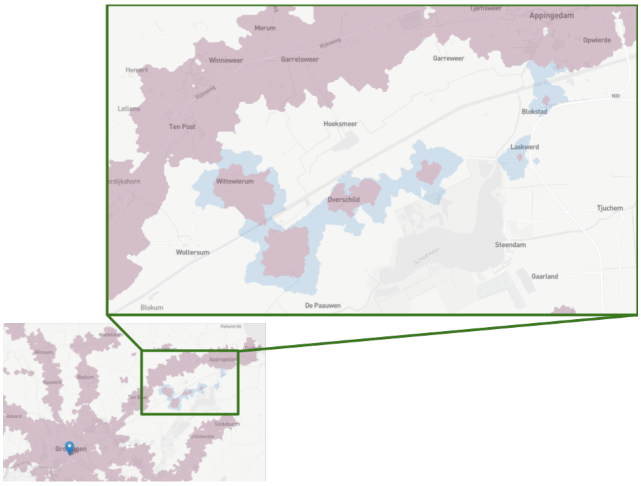


Figure 8 – Reachability difference caused by a missed transfer from line g506 to 564. The blue isochrone is the original schedule, and the purple isochrone is the predicted schedule.

Another example is shown in Figure 9, which shows that the predicted schedule is faster than the original schedule, leading to increased reachability. It can be seen that the red predicted isochrone around Zuidhorn is larger than the purple original schedule isochrone, due to bus g502 being predicted to be faster.

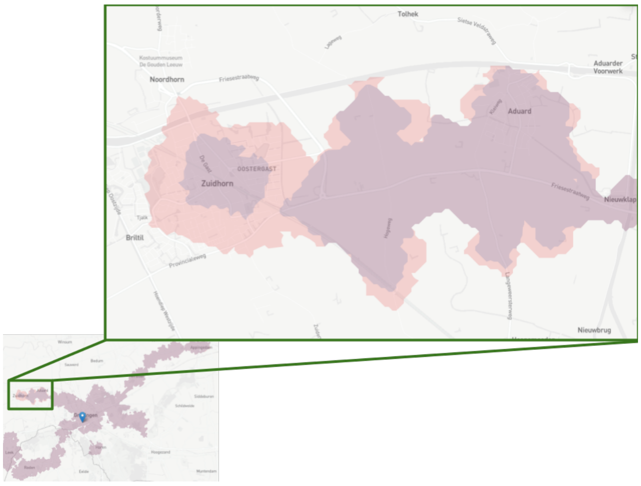


Figure 9 – Increased reachability example for an analysis run on a Wednesday with a starting point Groningen station between 15:00 and 16:00. Here, the larger red region is the predicted schedule by the LSTM model.

While incorporating dwell and travel time prediction enhances the accuracy of reachability analysis, its practical implications for PT engineers may be limited. Figure 8 shows that the difference in reachability regions is relatively small, ranging from approximately 200 to 1000 metres. This corresponds to 1 to 5 minutes of walking. This research proposes a more accurate representation but does not assess its practical applications for PT engineers.

For writing to the GTFS schedule the assumption is made that every bus journey leaves the first stop according to schedule. The predicted travel and dwell times are calculated from this. However, in real life, a bus may leave early or late. A consequence of this is that transfers can not be made, impacting the reachability. This behaviour is not currently captured by the implemented reachability analysis in Conveyal.

6 Conclusion

This paper aimed to enhance reachability analysis by predicting travel and dwell times using historical travel time data. Among the four models tested (HA, VAR, RF, and LSTM), the LSTM deep learning model showed the best accuracy, providing a more realistic representation of bus performance than the schedule. This led to variations in reachability and missed transfers.

The literature review identified the four prediction models, which improved understanding of travel time patterns. The case study framework efficiently manipulated the KV6 and GTFS datasets for reachability analysis, revealing expected discrepancies between original and predicted schedules.

Future research should include delayed departure predictions and advanced models like ensemble, gated recurring units, and transformer-based models, combined with comprehensive datasets (e.g., weather and traffic data). These improvements will enhance the design of sustainable and efficient public transportation networks, ultimately improving reachability.

References

[1] J. Beaudoin, Y. H. Farzin, and C.-Y. C. Lin Lawell, "Public transit investment and sustainable transportation: A review of studies of transit's impact on traffic congestion and air quality," *Research in Transportation Economics*, vol. 52, pp. 15–22, 2015, Sustainable Transportation, ISSN: 0739-8859. DOI: https://doi.org/10.1016/j.retrec.2015.10.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0739885915000487.

[2] I. Makarova, A. Pashkevich, and K. Shubenkova, "Ensuring sustainability of public transport system through rational management," *Procedia Engineering*, vol. 178, pp. 137–146, 2017, RelStat-2016: Proceedings of the 16th International Scientific Conference Reliability and Statistics in Transportation and Communication October 19-22, 2016. Transport and Telecommunication Institute, Riga, Latvia, ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2017.01.078. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705817300784.

[3] B. Yao, P. Hu, X. Lu, J. Gao, and M. Zhang, "Transit network design based on travel time reliability," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 233–248, 2014, Special Issue on "Nature-Inspired Optimization Techniques in Transportation Planning and Operation", ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2013.12.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X13002647.

[4] Stichting OpenGeo, *Dutch real-time transit data*, 2013-2019. [Online]. Available: https://ndovloket.nl/.

[5] E.-H. Chung and A. Shalaby, "Expected time of arrival model for school bus transit using real-time global positioning system-based automatic vehicle location data," *Journal of Intelligent Transportation Systems*, vol. 11, no. 4, pp. 157–167, 2007. DOI: 10.1080/15472450701649398. eprint: https://doi.org/10.1080/15472450701649398. [Online]. Available: https://doi.org/10.1080/15472450701649398.

[6] Z. Ma, H. N. Koutsopoulos, L. Ferreira, and M. Mesbah, "Estimation of trip travel time distribution using a generalized markov chain approach," *Transportation Research Part C: Emerging Technologies*, vol. 74, pp. 1–21, 2017, ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2016.11.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X16302248.

[7] B. A. Kumar, L. Vanajakshi, and S. C. Subramanian, "Bus travel time prediction using a time-space discretization approach," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 308–332, 2017, ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2017.04.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X17301080.

[8] C. García-Mauriño, P. J. Zufiria, and A. Jarabo-Peñas, “Improving bus arrival time predictors using only public transport api data,” *Transportation Letters*, vol. 16, no. 8, pp. 804–813, 2024. DOI: 10.1080/19427867.2023.2245994. eprint: https://doi.org/10.1080/19427867.2023.2245994. [Online]. Available: https://doi.org/10.1080/19427867.2023.2245994.

[9] J. Greenfeld, “Automatic vehicle location (avl) for transit operation,” in *2000 10th Mediterranean Electrotechnical Conference. Information Technology and Electrotechnology for the Mediterranean Countries. Proceedings. MeleCon 2000 (Cat. No.00CH37099)*, vol. 2, 2000, 656–659 vol.2. DOI: 10.1109/MELCON.2000.880019.

[10] B. Predic, D. Rančic, D. Stojanovic, and A. Milosavljevic, “Automatic vehicle location in public bus transportation system,” *Annual Conference on Computers*, pp. 675–680, Jul. 2007. [Online]. Available: https://www.researchgate.net/profile/Aleksandar_Milosavljevic3/publication/228996934_Automatic_vehicle_location_in_public_bus_transportation_system/links/0046353cd723a76cee000000.pdf.

[11] Y. Yan, Z. Liu, and Y. Bie, “Performance evaluation of bus routes using automatic vehicle location data,” *Journal of Transportation Engineering*, vol. 142, no. 8, p. 04 016 029, 2016. DOI: 10.1061/(ASCE)TE.1943-5436.0000857. eprint: https://ascelibrary.org/doi/pdf/10.1061/%28ASCE%29TE.1943-5436.0000857. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29TE.1943-5436.0000857.

[12] L. D’Acierno, A. Cartenì, and B. Montella, “Estimation of urban traffic conditions using an automatic vehicle location (avl) system,” *European Journal of Operational Research*, vol. 196, no. 2, pp. 719–736, 2009, ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2007.12.053. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221708003482.

[13] S. Maiti, A. Pal, A. Pal, T. Chattopadhyay, and A. Mukherjee, “Historical data based real time prediction of vehicle arrival time,” in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, pp. 1837–1842. DOI: 10.1109/ITSC.2014.6957960.

[14] Suwardo, N. Madzlan, and K. Ibrahim, “ARIMA models for bus travel time prediction,” *No journal*, Jun. 2010. [Online]. Available: http://dspace.unimap.edu.my:80/handle/123456789/13714.

[15] B. Yu, H. Wang, W. Shan, and B. Yao, “Prediction of Bus Travel Time Using Random Forests Based on Near Neighbors,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 4, pp. 333–350, Nov. 2017. DOI: 10.1111/mice.12315. [Online]. Available: https://doi.org/10.1111/mice.12315.

[16] J. Ma, J. Chan, G. Ristanoski, S. Rajasegarar, and C. Leckie, “Bus travel time prediction with real-time traffic information,” *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 536–549, 2019, ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2019.06.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X18309082.

[17] E. Chondrodima, H. Georgiou, N. Pelekis, and Y. Theodoridis, “Public transport arrival time prediction based on gtfs data,” in *Machine Learning, Optimization, and Data Science*, Cham: Springer International Publishing, 2022, pp. 481–495, ISBN: 978-3-030-95470-3.

[18] J. Pang, J. Huang, Y. Du, H. Yu, Q. Huang, and B. Yin, “Learning to predict bus arrival time from heterogeneous measurements via recurrent neural network,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3283–3293, 2019. DOI: 10.1109/TITS.2018.2873747.

[19] P. He, G. Jiang, S.-K. Lam, and D. Tang, “Travel-time prediction of bus journey with multiple bus trips,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 11, pp. 4192–4205, 2019. DOI: 10.1109/TITS.2018.2883342.

[20] H. Liu, H. Xu, Y. Yan, Z. Cai, T. Sun, and W. Li, “Bus arrival time prediction based on lstm and spatial-temporal feature vector,” *IEEE Access*, vol. 8, pp. 11 917–11 929, 2020. DOI: 10.1109/ACCESS.2020.2965094.

[21] O. Alam, A. Kush, A. Emami, and P. Pouladzadeh, “Predicting irregularities in arrival times for transit buses with recurrent neural networks using gps coordinates and weather data,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 7, pp. 7813–7826, Sep. 2020. DOI: 10.1007/s12652-020-02507-9.

[22] Q. Han, K. Liu, L. Zeng, G. He, L. Ye, and F. Li, “A bus arrival time prediction method based on position calibration and lstm,” *IEEE Access*, vol. 8, pp. 42 372–42 383, 2020. DOI: 10.1109/ACCESS.2020.2976574.

[23] X. Zhang, L. Lauber, H. Liu, J. Shi, M. Xie, and Y. Pan, “Travel time prediction of urban public transportation based on detection of single routes,” *PLOS ONE*, vol. 17, pp. 1–23, Jan. 2022. DOI: 10.1371/journal.pone.0262535. [Online]. Available: https://doi.org/10.1371/journal.pone.0262535.

[24] H. Lütkepohl, *New Introduction to Multiple Time Series Analysis*. Jan. 2005. DOI: 10.1007/978-3-540-27752-1. [Online]. Available: https://doi.org/10.1007/978-3-540-27752-1.

[25] E. Lewinson, *Three Approaches to Encoding Time Information as Features for ML Models | NVIDIA Technical Blog*, Aug. 2022. [Online]. Available: https://developer.nvidia.com/blog/three-approaches-to-encoding-time-information-as-features-for-ml-models/.

[26] S. Hochreiter and J. Schmidhuber, “Long Short-Term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. DOI: 10.1162/neco.1997.9.8.1735. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[27] Conveyal, *Conveyal - Evaluate changes to your public transportation system*, 2024. [Online]. Available: https://conveyal.com/.

A List of symbols

Table 2 – Symbol definitions used in the methodology.

Symbol	Definition	Unit
j	Index for journey in the dataset	-
k	Index for stop in a certain journey	-
l	Index for link in a certain journey	-
$\mathbf{k}_j$	Vector of stops in journey j	-
$\mathbf{l}_j$	Vector of links in journey j	-
n_j	Number of stops in journey j	-
N	Number of journeys in dataset	-
at_k	Arrival time at stop k	(yyyy/mm/dd: hh/mm/ss)
dt_k	Departure time from stop k	(yyyy/mm/dd: hh/mm/ss)
y_k	Dwell time at stop k	s
y_l	Travel time of link l	s
$\mathbf{y}_{\mathbf{k},j}$	Vector of dwell times of journey j	s
$\mathbf{y}_{\mathbf{l},j}$	Vector of travel times of journey j	s
$\hat{y}_k$	Predicted dwell time at stop k	s
$\hat{y}_l$	Predicted travel time of link l	s
$\hat{\mathbf{y}}_{\mathbf{k},j}$	Vector of predicted dwell times in journey j	s
$\hat{\mathbf{y}}_{\mathbf{l},j}$	Vector of predicted travel times in journey j	s

B Data format example

Table 3 – This is an example of one direction of a single bus line presented in a tabular format. Dwell time y_{k_1} is the difference between the arrival time at_{k_1} at the first stop and the departure time dt_{k_1} from the first stop. This is done for all the stops in $\mathbf{k}_j$ in a journey j. Travel time y_{l_1} is the difference between the departure time dt_{k_1} at the first stop and the arrival time at_{k_2} at the second stop. This is, also, done for all links in $\mathbf{l}_j$ in a journey j. The table is indexed by the departure time dt_{k_1} from the first stop.

Journey	Index	y_{k_1}	y_{l_1}	y_{k_2}	y_{l_2}	$\cdots$	$y_{l_{n-1}}$	y_{k_n}
$j = 1$	$t_0(1)$	30	80	0	65	$\cdots$	125	20
$j = 2$	$t_0(2)$	45	70	50	90	$\cdots$	135	10
$j = 3$	$t_0(3)$	0	95	40	85	$\cdots$	140	15
$j = 4$	$t_0(4)$	45	75	0	75	$\cdots$	130	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$j = N$	$t_0(N)$	35	90	0	70	$\cdots$	140	0

C Bus network of Groningen overview

Table 4 – QBUZZ lines in Groningen. Outer stop 1 to outer stop 2 is indicated as direction 1 and outer stop 2 to outer stop 1 is indicated as direction 2.

Line	Outer stop 1	Outer stop 2	# of stops	Length (km)
g501	Groningen, P+R Reitdiep	Groningen, Hoofdstation	19	7.73
g502	Zuidhorn, Station	Groningen, Station Europapark	26	17.90
g503	Groningen, Ruischerbrug	Leek, Oostindie	39	29.64
g504	Groningen, Wibenaheerd	Roden, Kastelenlaan	39	26.70
g505	Annen, Zuid	Scharmer, Goldberweg	33	33.80
g506	Delfzijl, Station	Groningen, Hoofdstation	48	36.65

B Additional figures

This appendix contains figures of lines and directions which are not presented in the main body of the report.

B.1 Historical Average travel time evaluation maps

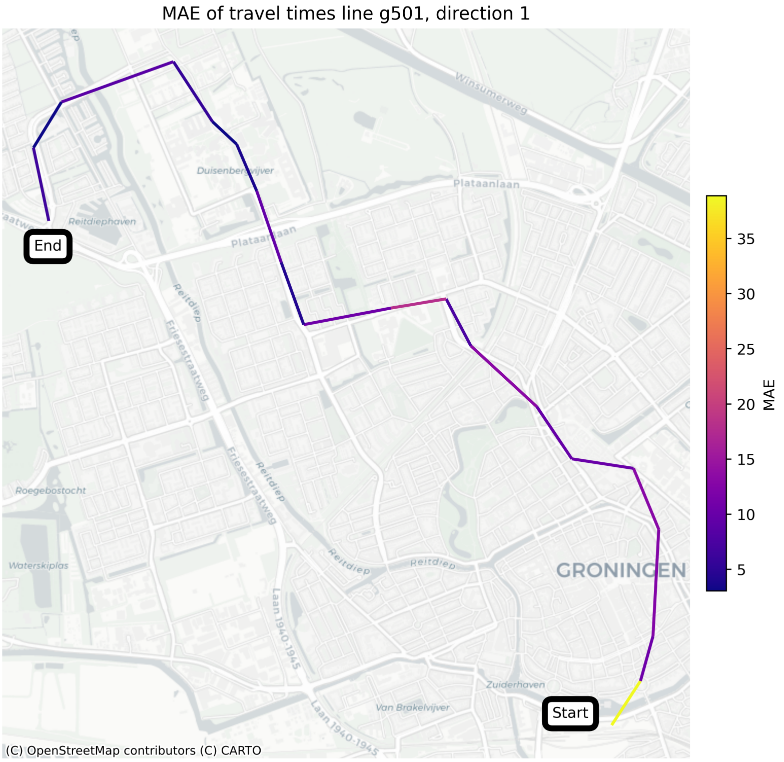


Figure B.1 – MAE of travel times of HA model of line g501 in direction 1.

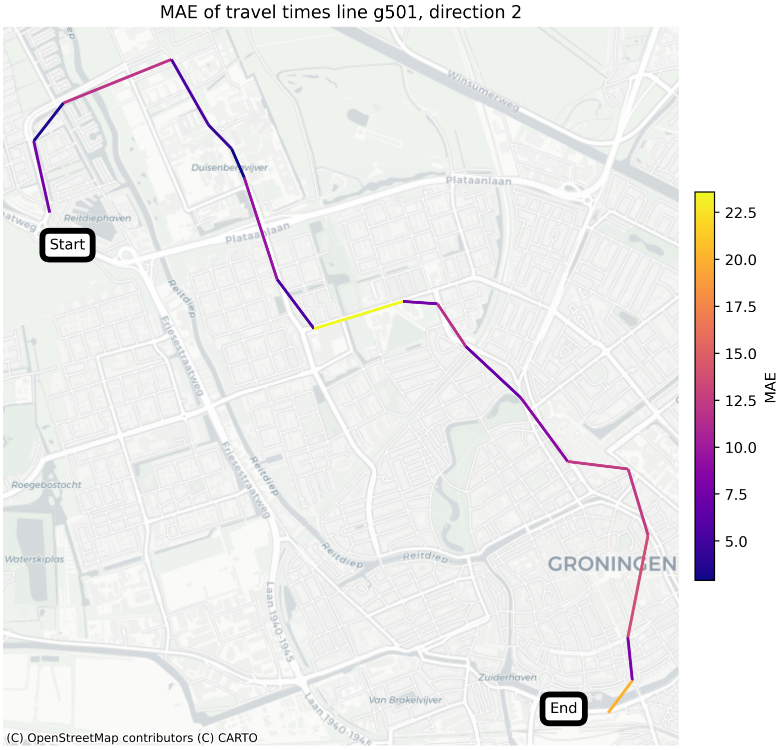


Figure B.2 – MAE of travel times of HA model of line g501 in direction 2.

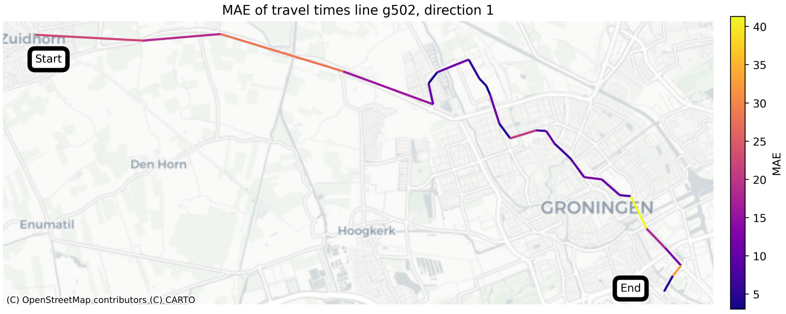

MAE of travel times line g502, direction 1
Zuidhorn
Start
Den Horn
Enumatil
Hoogkerk
GRONINGEN
End
(C) OpenStreetMap contributors (C) CARTO
40
35
30
25
20
15
10
5
MAE

Figure B.3 – MAE of travel times of HA model of line g502 in direction 1.

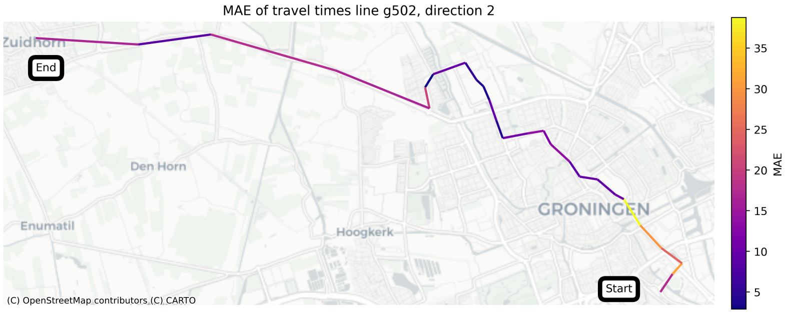

MAE of travel times line g502, direction 2
Zuidhorn
End
Den Horn
Enumatil
Hoogkerk
GRONINGEN
Start
(C) OpenStreetMap contributors (C) CARTO
MAE
35
30
25
20
15
10
5

Figure B.4 – MAE of travel times of HA model of line g502 in direction 2.

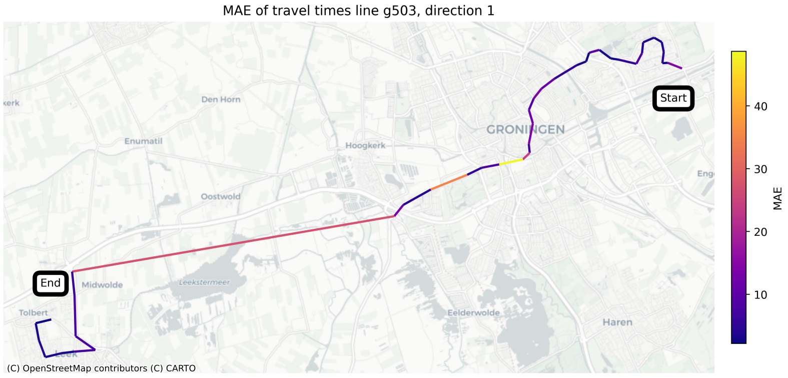


Figure B.5 – MAE of travel times of HA model of line g503 in direction 1.

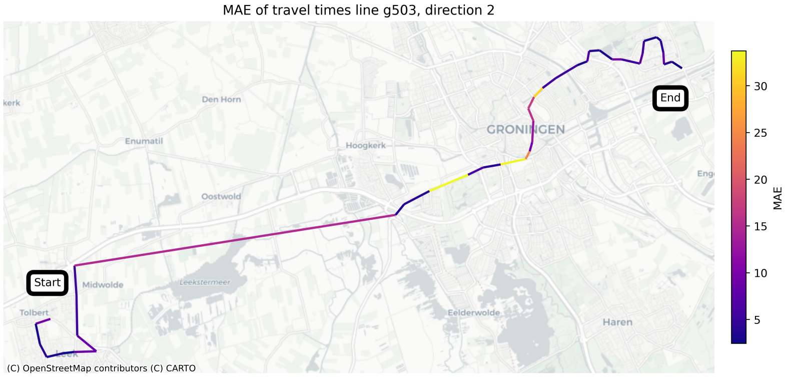


Figure B.6 – MAE of travel times of HA model of line g503 in direction 2.

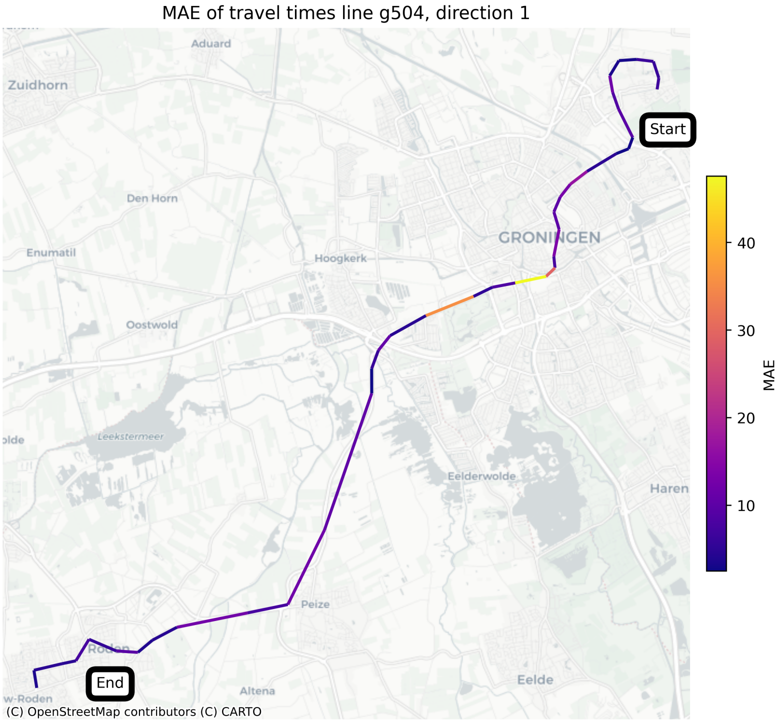


Figure B.7 – MAE of travel times of HA model of line g504 in direction 1.

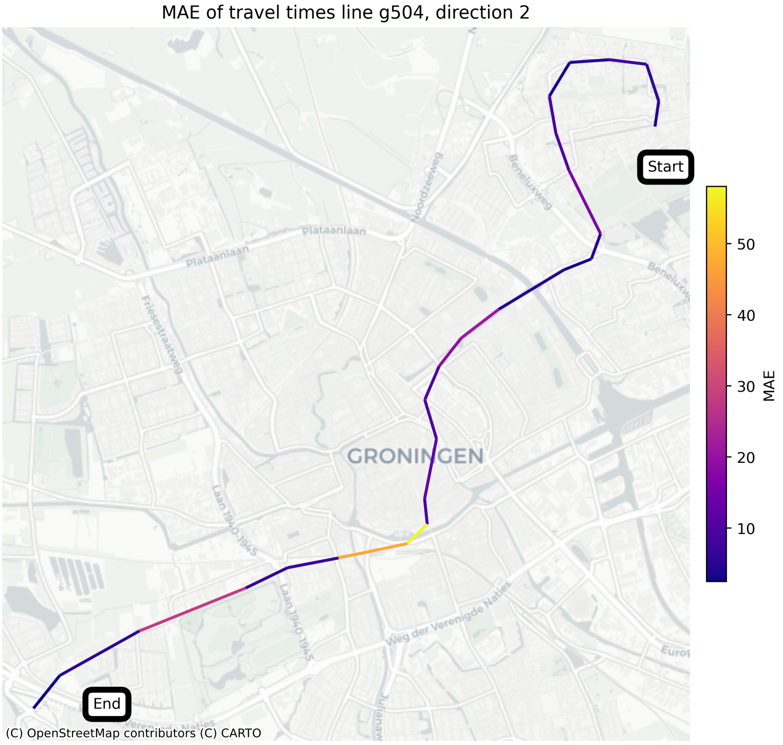


Figure B.8 – MAE of travel times of HA model of line g504 in direction 2.

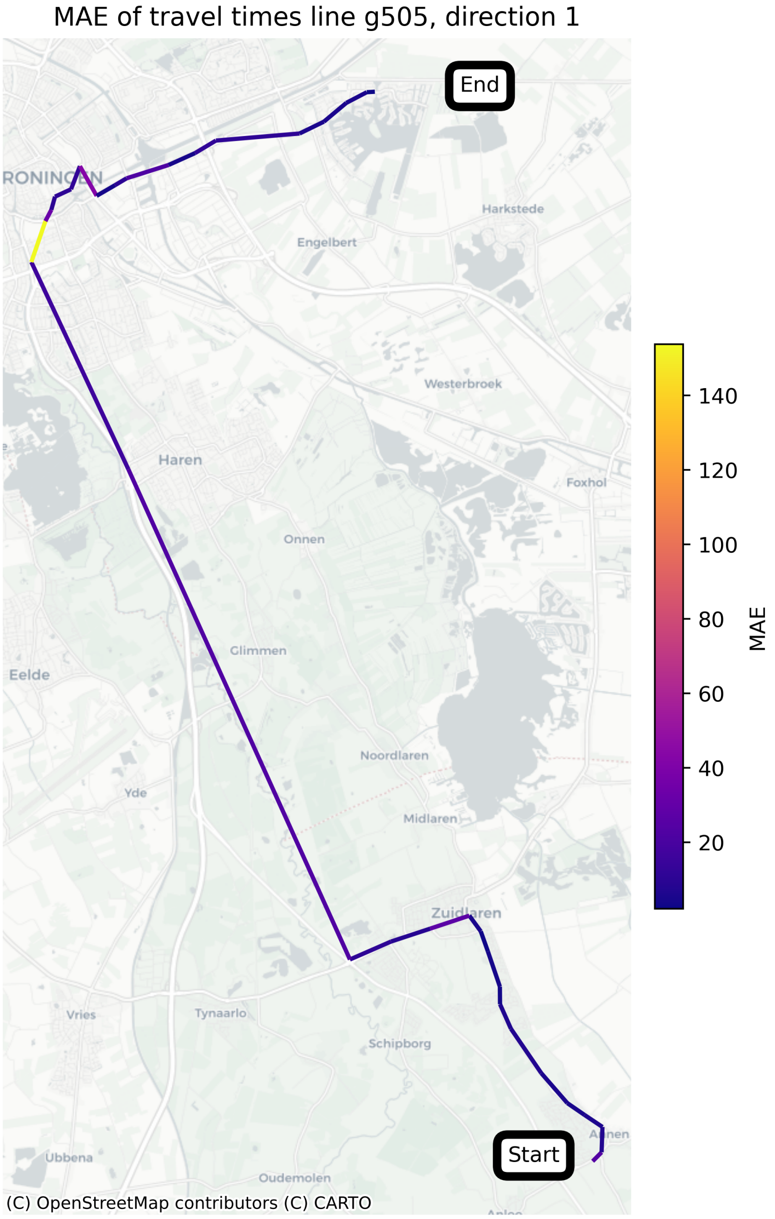


Figure B.9 – MAE of travel times of HA model of line g505 in direction 1.

MAE of travel times line g505, direction 2

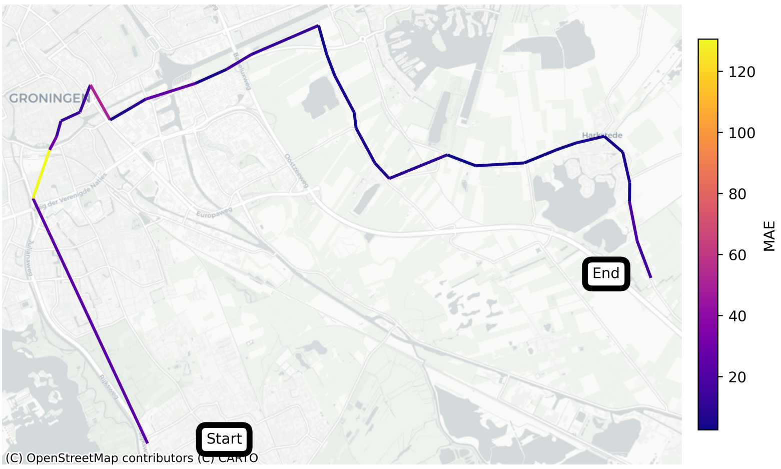


Figure B.10 – MAE of travel times of HA model of line g505 in direction 2.

MAE of travel times line g506, direction 1

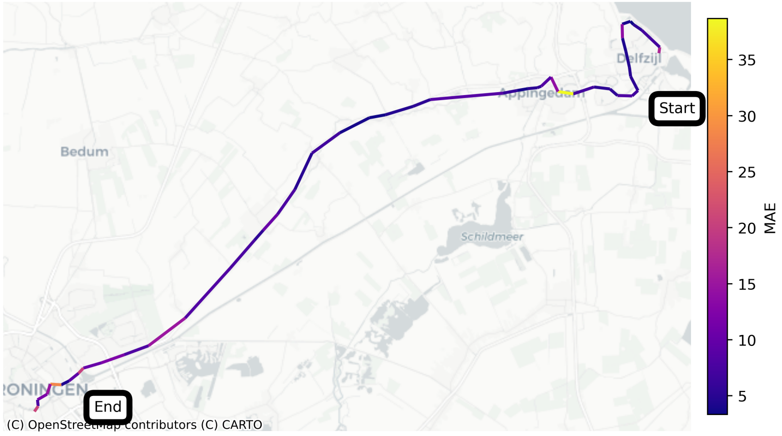


Figure B.11 – MAE of travel times of HA model of line g506 in direction 1.

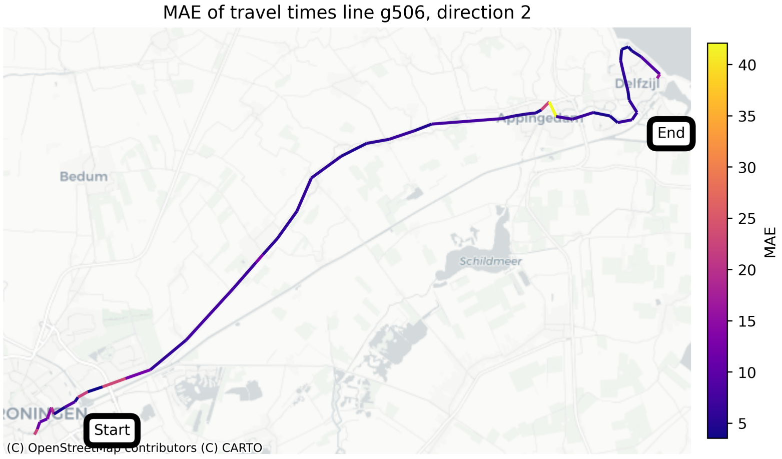

MAE of travel times line g506, direction 2
Bedum
Schildmeer
Appingedam
Delfzijl
GRONINGEN
Start
End
(C) OpenStreetMap contributors (C) CARTO
MAE
40
35
30
25
20
15
10
5

Figure B.12 – MAE of travel times of HA model of line g506 in direction 2.

B.2 Random Forest feature importance evaluation

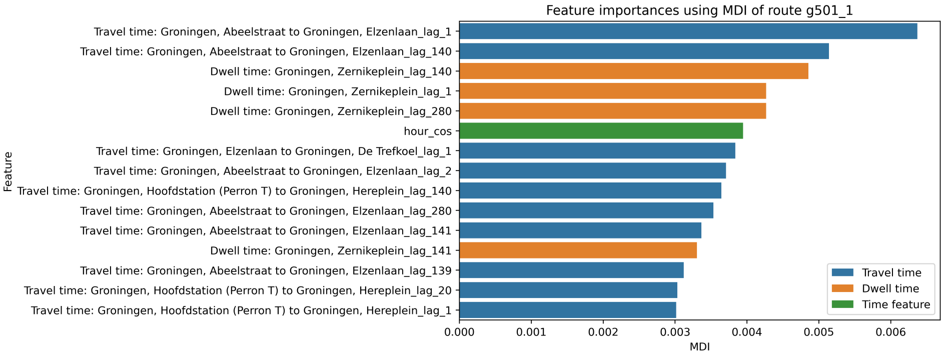


Figure B.13 – Feature importance evaluation of RF model for line g501 in direction 1.

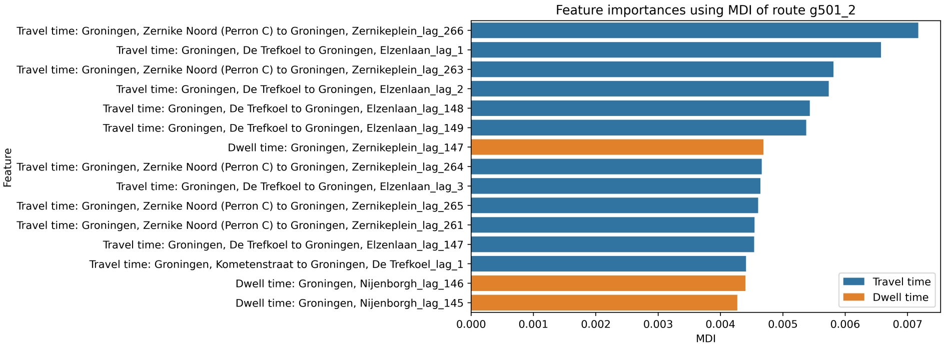


Figure B.14 – Feature importance evaluation of RF model for line g501 in direction 2.

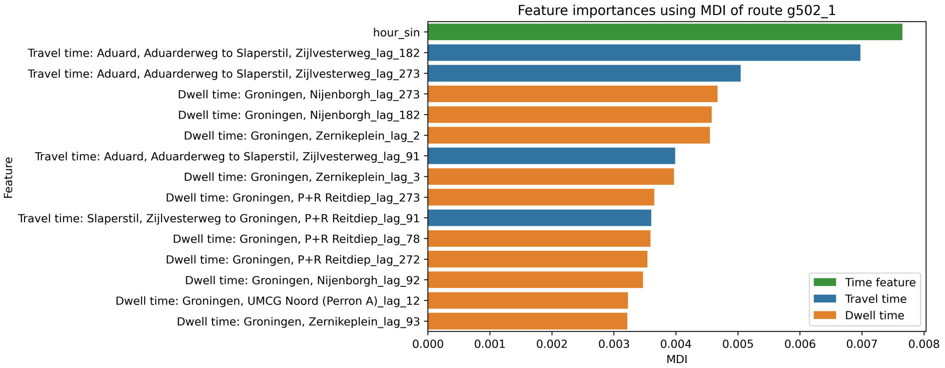


Figure B.15 – Feature importance evaluation of RF model for line g502 in direction 1.

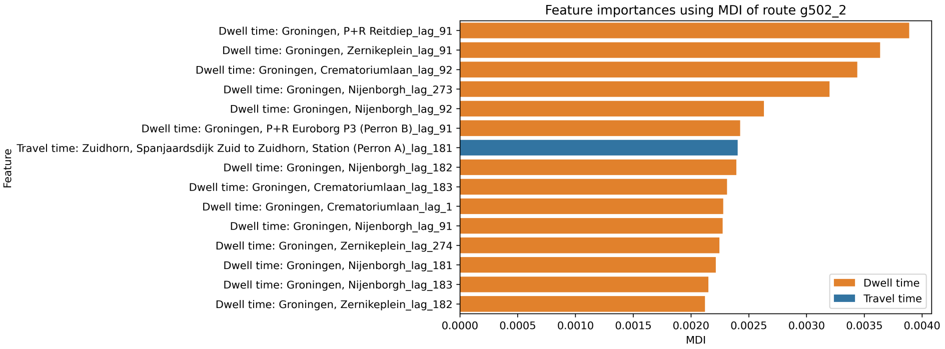


Figure B.16 – Feature importance evaluation of RF model for line g502 in direction 2.

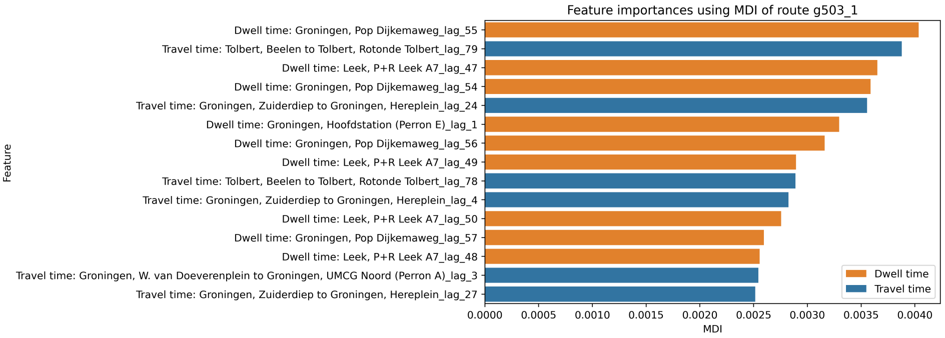


Figure B.17 – Feature importance evaluation of RF model for line g503 in direction 1.

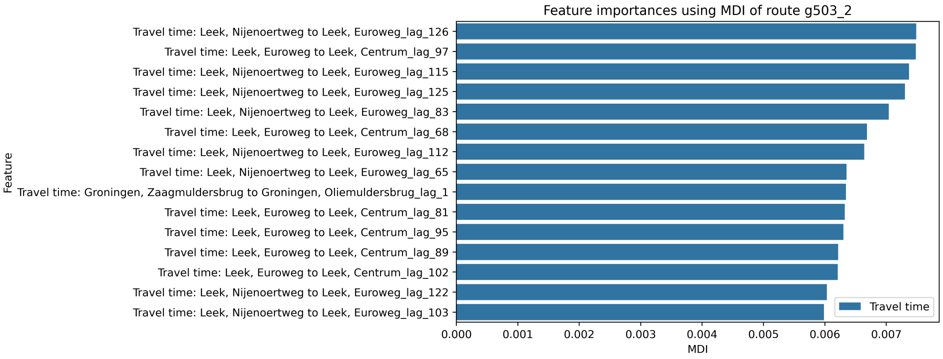


Figure B.18 – Feature importance evaluation of RF model for line g503 in direction 2.

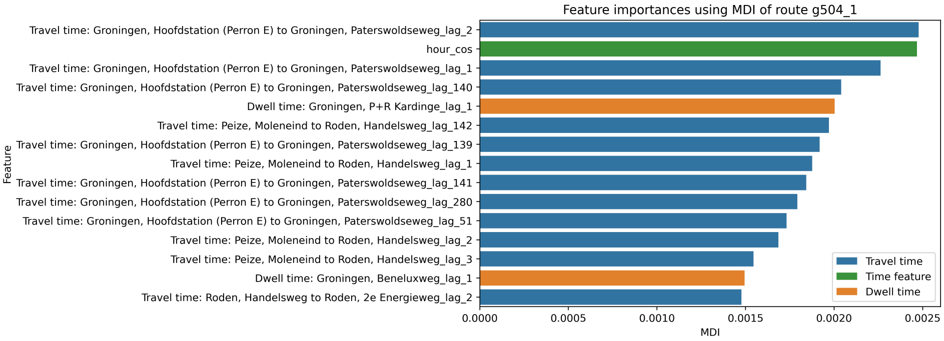


Figure B.19 – Feature importance evaluation of RF model for line g504 in direction 1.

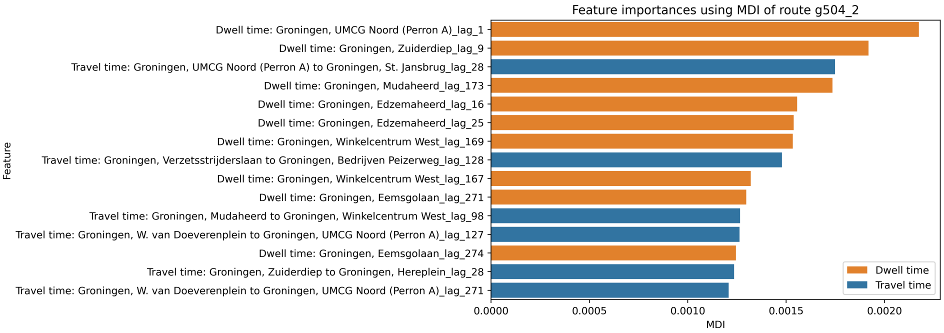


Figure B.20 – Feature importance evaluation of RF model for line g504 in direction 2.

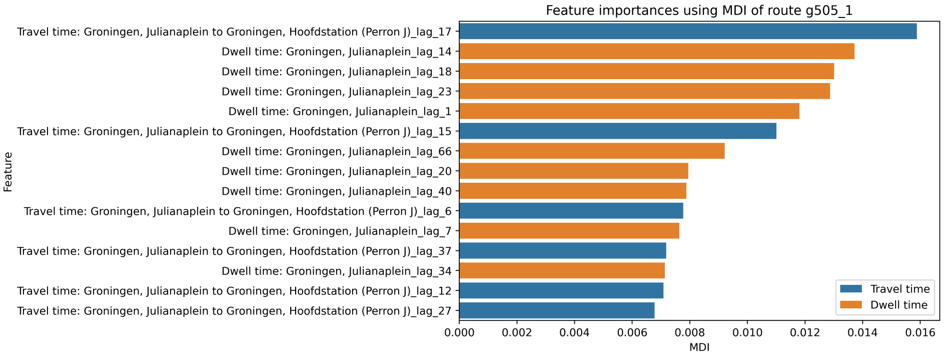


Figure B.21 – Feature importance evaluation of RF model for line g505 in direction 1.

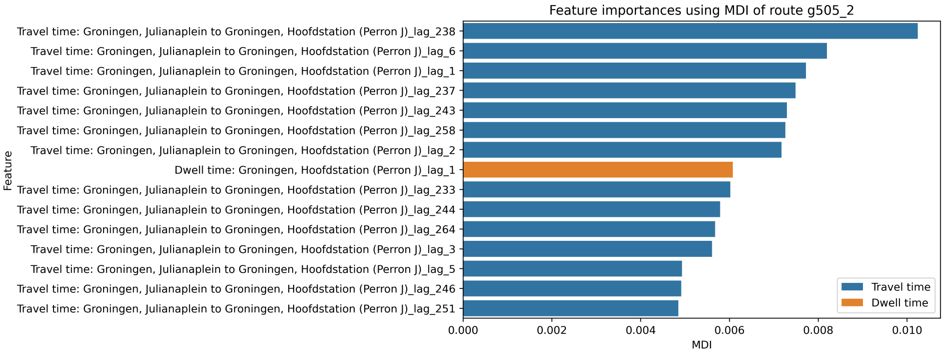


Figure B.22 – Feature importance evaluation of RF model for line g505 in direction 2.

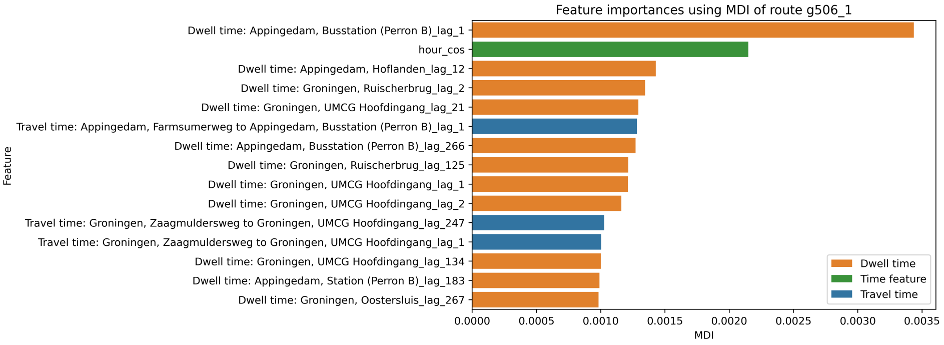


Figure B.23 – Feature importance evaluation of RF model for line g506 in direction 1.

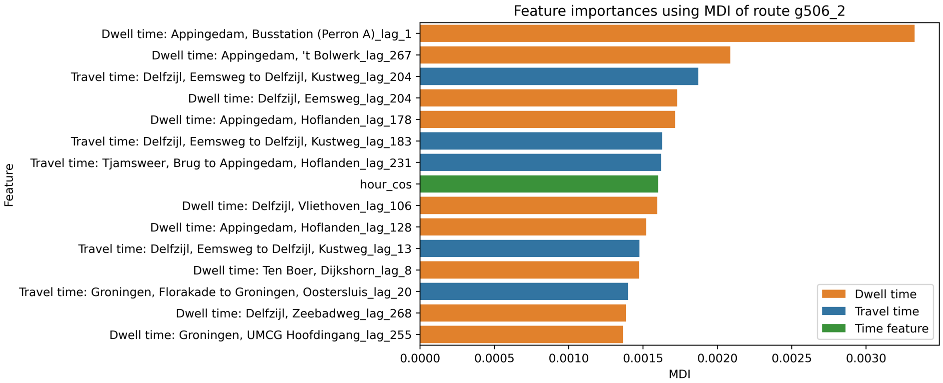


Figure B.24 – Feature importance evaluation of RF model for line g506 in direction 2.

B.3 Long Short-Term Memory loss analysis

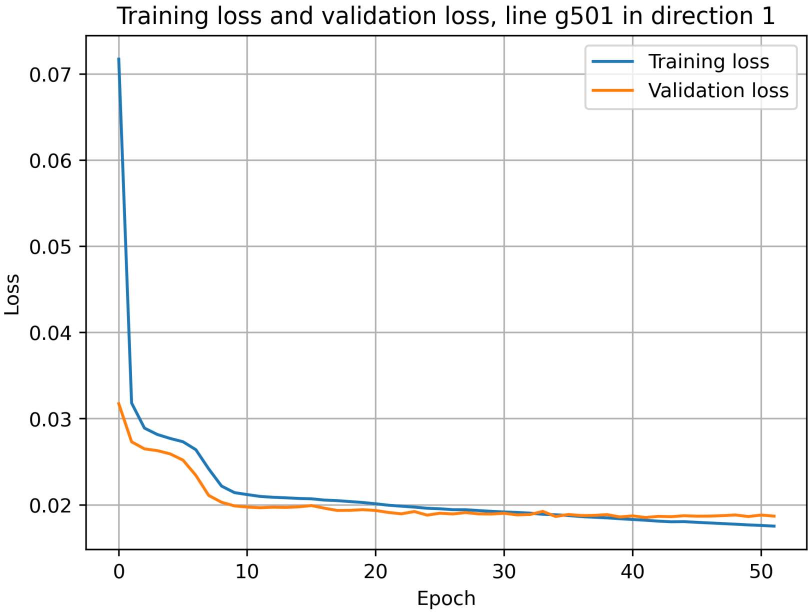


Figure B.25 – Validation dataset and training dataset loss during training of LSTM model for line g501 in direction 1. The x-axis displays epochs.

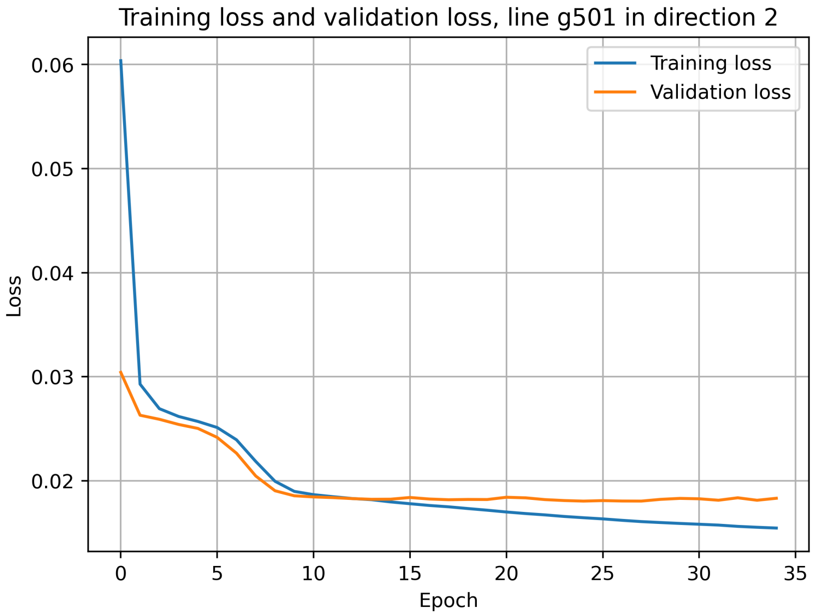


Figure B.26 – Validation dataset and training dataset loss during training of LSTM model for line g501 in direction 2. The x-axis displays epochs.

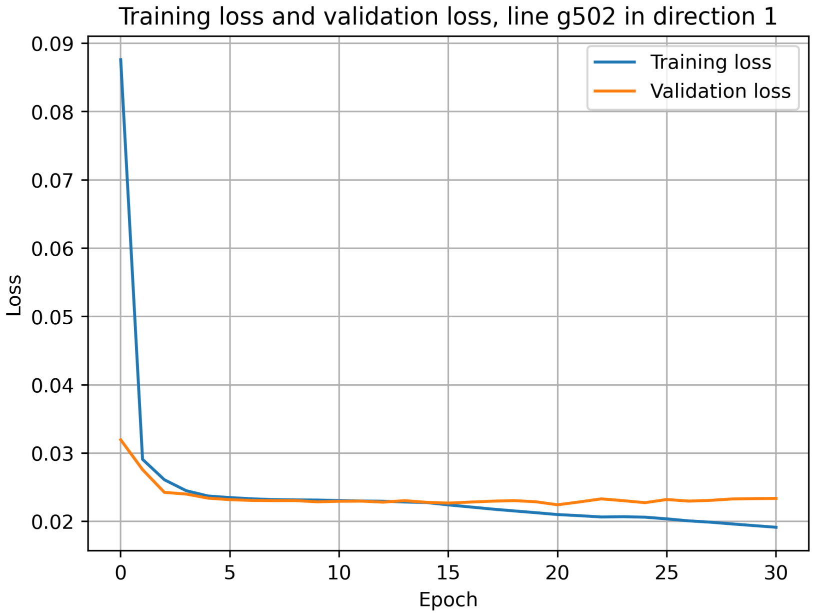


Figure B.27 – Validation dataset and training dataset loss during training of LSTM model for line g502 in direction 1. The x-axis displays epochs.

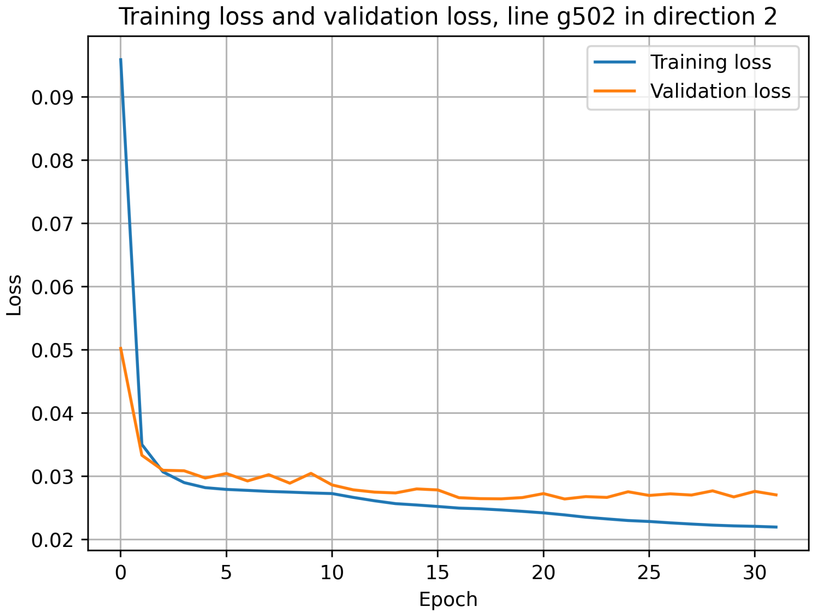


Figure B.28 – Validation dataset and training dataset loss during training of LSTM model for line g502 in direction 2. The x-axis displays epochs.

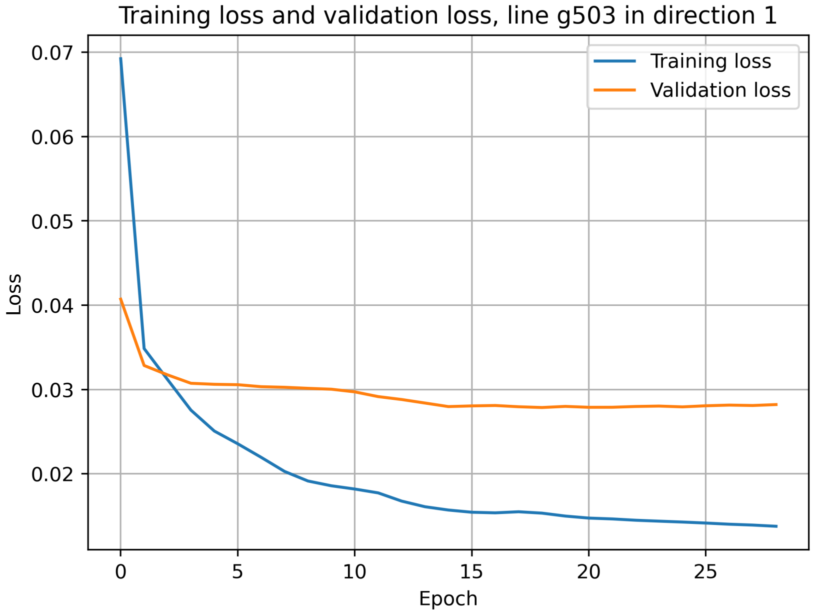


Figure B.29 – Validation dataset and training dataset loss during training of LSTM model for line g503 in direction 1. The x-axis displays epochs.

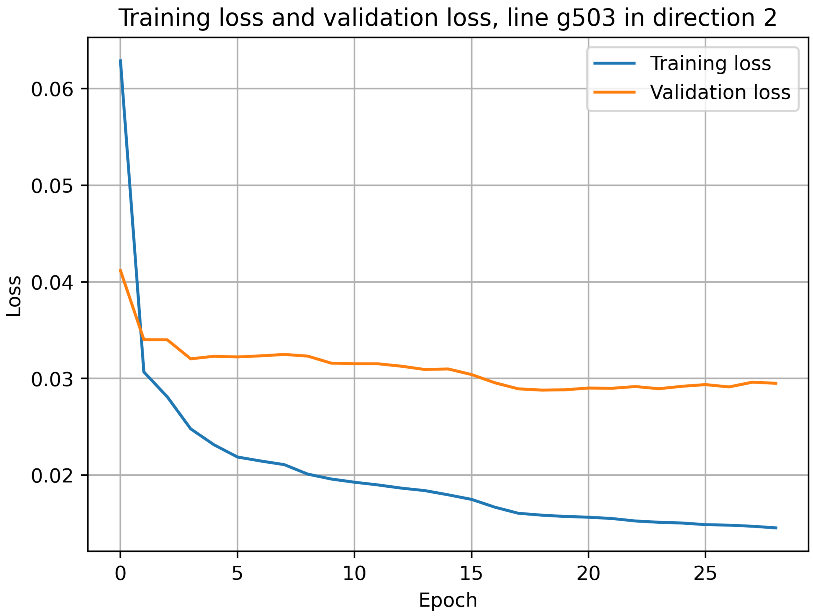


Figure B.30 – Validation dataset and training dataset loss during training of LSTM model for line g503 in direction 2. The x-axis displays epochs.

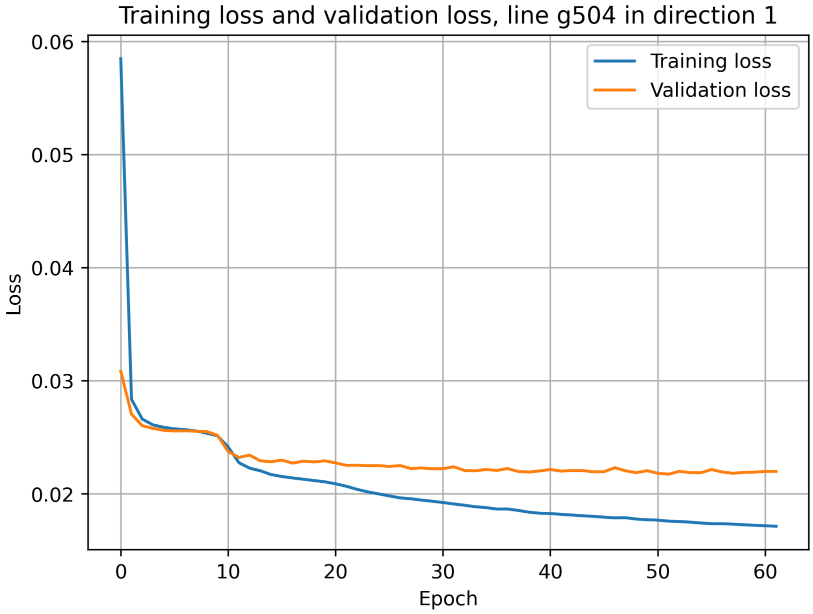


Figure B.31 – Validation dataset and training dataset loss during training of LSTM model for line g504 in direction 1. The x-axis displays epochs.

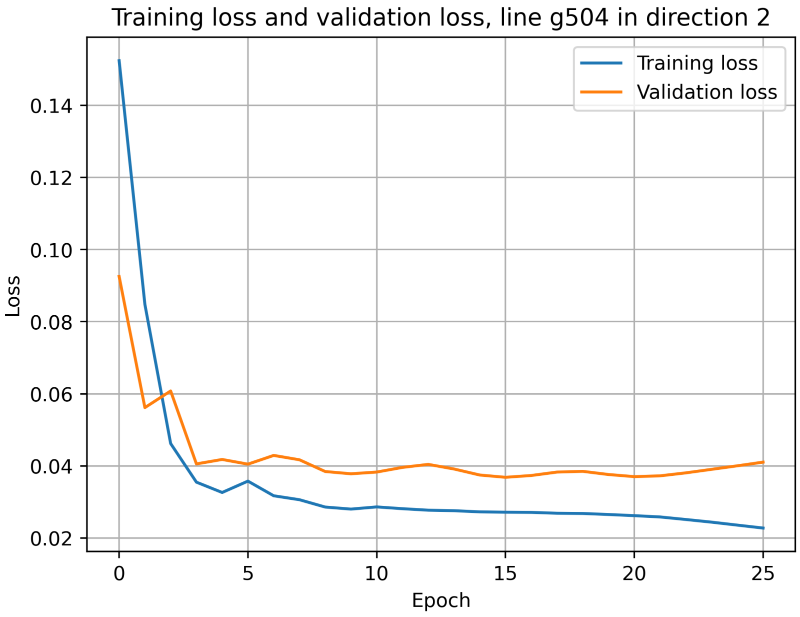


Figure B.32 – Validation dataset and training dataset loss during training of LSTM model for line g504 in direction 2. The x-axis displays epochs.

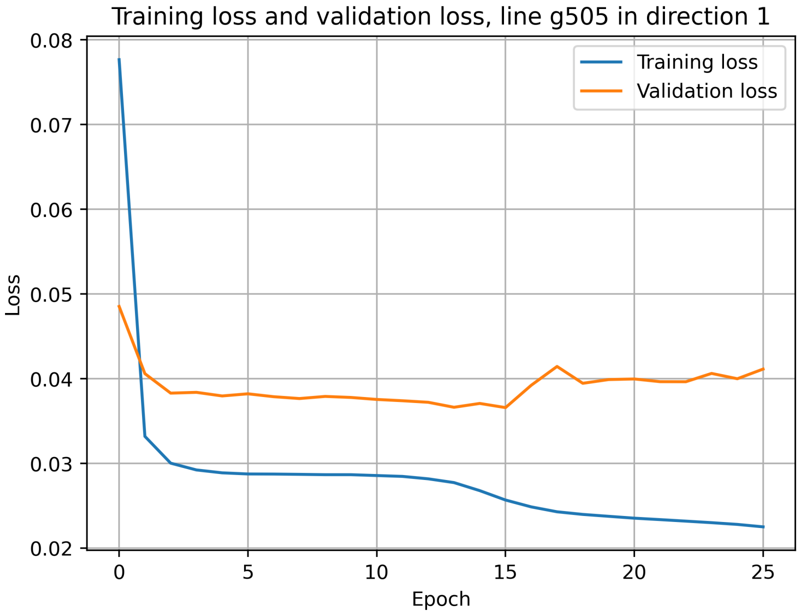


Figure B.33 – Validation dataset and training dataset loss during training of LSTM model for line g505 in direction 1. The x-axis displays epochs.

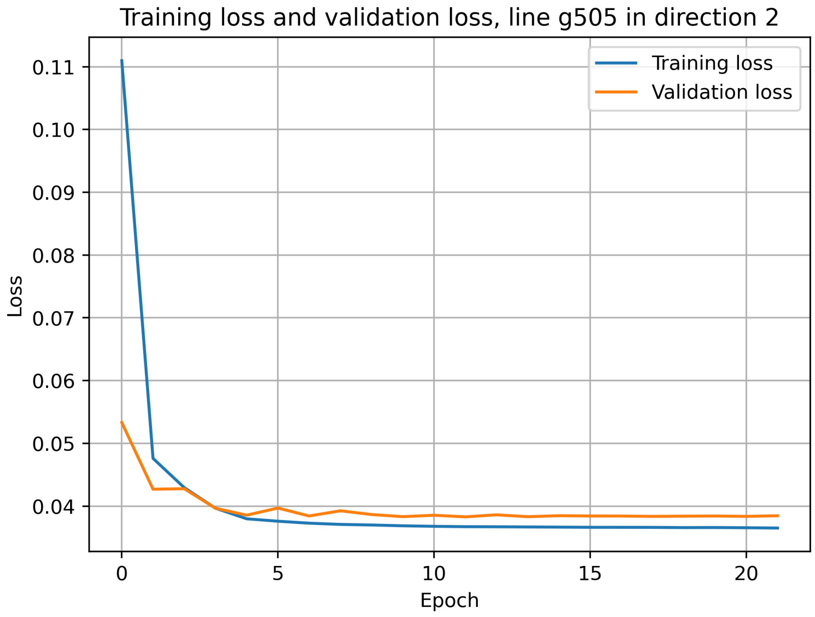


Figure B.34 – Validation dataset and training dataset loss during training of LSTM model for line g505 in direction 2. The x-axis displays epochs.

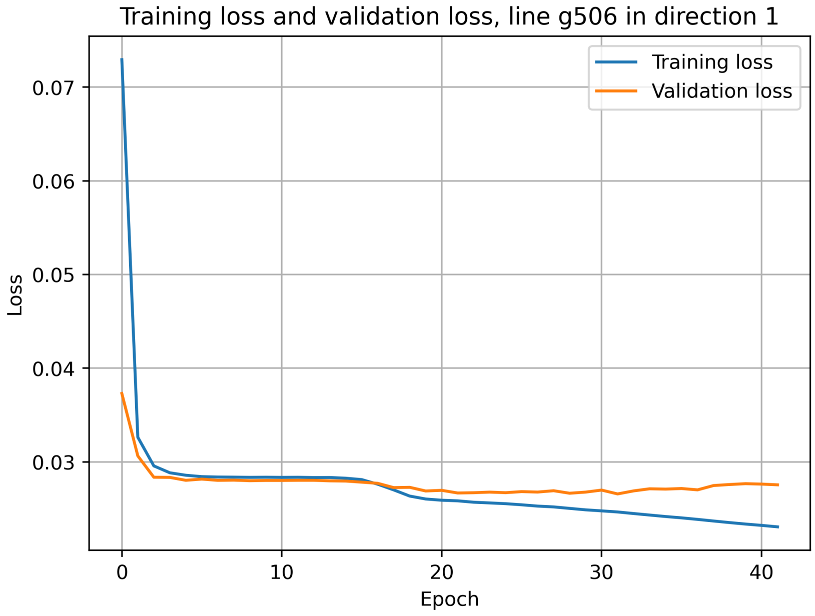


Figure B.35 – Validation dataset and training dataset loss during training of LSTM model for line g506 in direction 1. The x-axis displays epochs.

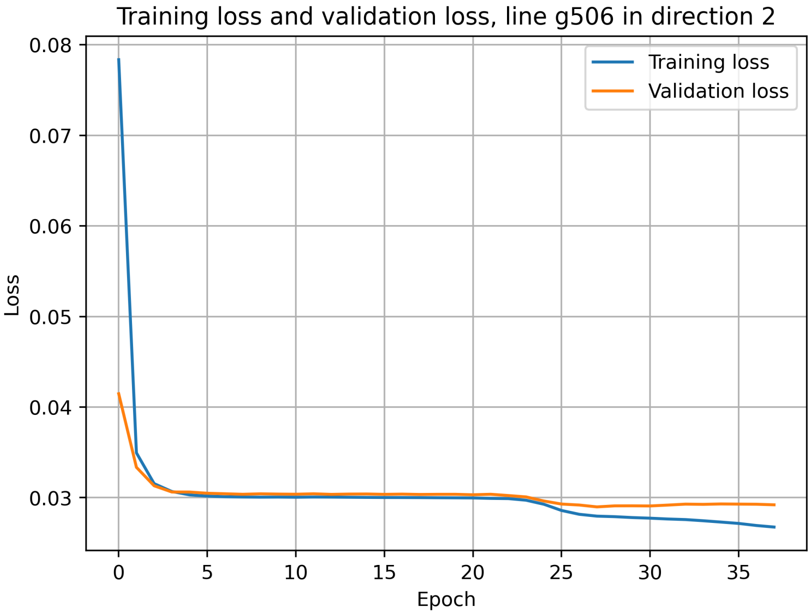


Figure B.36 – Validation dataset and training dataset loss during training of LSTM model for line g506 in direction 2. The x-axis displays epochs.