UNIVERSITY OF PIRAEUS - DEPARTMENT OF INFORMATICS

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

MSc «Cybersecurity and Data Science»

ΠΜΣ «Κυβερνοασφάλεια και Επιστήμη Δεδομένων»

MSc Thesis

Μεταπτυχιακή Διατριβή

Thesis Title: Τίτλος Διατριβής:	**Traffic prediction on road networks** Πρόβλεψη κυκλοφορίας σε οδικά δίκτυα
Student's name-surname: Ονοματεπώνυμο φοιτητή:	**Afroditi Karatrantou** Αφροδίτη Καρατράντου
Father's name: Πατρώνυμο:	**Anargyros** Ανάργυρος
Student's ID No: Αριθμός Μητρώου:	MPKED 21018
Supervisor: Επιβλέπων:	**Nikolaos Pelekis, Associate Professor** Νικόλαος Πελέκης, Αναπληρωτής Καθηγητής

November 2024/Νοέμβριος 2024

3-Member Examination Committee

Τριμελής Εξεταστική Επιτροπή

Nikolaos Pelekis **Associate Professor**	**Yannis Theodoridis** **Professor**	**Aggelos Pikrakis** **Assistant Professor**
Νικόλαος Πελέκης Αναπληρωτής Καθηγητής	Γιάννης Θεοδωρίδης Καθηγητής	Άγγελος Πικράκης Επίκουρος Καθηγητής

Table of Contents

List of Figures and Tables

Abstract

Accurate Estimated Time of Arrival (ETA) prediction is essential for optimizing urban transit systems and enhancing user experience. This thesis examines advanced machine learning techniques, including Neural Network (NN), Random Forest (RF), XGBoost, Long Short-Term Memory (LSTM) networks, and Graph Convolutional Networks (GCNs), for predicting ETAs of urban transit vehicles using real-time GPS data. A comprehensive data preprocessing pipeline was developed to handle noise and inconsistencies in transit data, involving steps like normalization, imputation of missing values, and k-fold Cross Validation.

The results show that the Random Forest model, especially when optimized with k-fold Cross Validation, provides superior accuracy in terms of Mean Squared Error (MSE) and Mean Absolute Error (MAE), balancing prediction accuracy with computational efficiency. The LSTM model also produced promising results, effectively capturing temporal dependencies. In contrast, both GCNs and XGBoost models, produced poorer results with significantly lower accuracy, despite XGBoost being computationally efficient. The Simple Model (NN) performed well despite its simplicity, with the high number of neurons contributing to its strong predictive accuracy. Additionally, a SHAP (SHapley Additive exPlanations) analysis was performed to highlight the most important features influencing ETA predictions. Temporal features such as the unix time are the most influential.

This thesis contributes to the field by providing an easy-to-understand method for predicting ETAs in urban transit. Future work could involve expanding the dataset and integrating additional features like traffic congestion and weather conditions to enhance model performance.

Keywords: Estimated Time of Arrival (ETA), Urban Transit Systems, Machine Learning, Random Forest (RF)

Περίληψη

Η ακριβής πρόβλεψη του Εκτιμώμενου Χρόνου Άφιξης (ETA) είναι απαραίτητη για τη βελτιστοποίηση των αστικών συστημάτων μετακίνησης και την βελτίωση της εμπειρίας των χρηστών. Αυτή η διατριβή εξετάζει προηγμένες τεχνικές μηχανικής μάθησης, όπως τα Νευρωνικά Δίκτυα (NN), Random Forest (RF), το XGBoost, τα δίκτυα Long Short-Term Memory (LSTM) και τα Graph Convolutional Networks (GCNs), για την πρόβλεψη της άφιξης μέσων μεταφοράς χρησιμοποιώντας δεδομένα GPS σε πραγματικό χρόνο. Αναπτύχθηκε ένας ολοκληρωμένος αλγόριθμος προεπεξεργασίας δεδομένων για την αντιμετώπιση των σφαλμάτων στα δεδομένα μετακίνησης, περιλαμβάνοντας βήματα όπως η κανονικοποίηση, η συμπλήρωση ελλιπών τιμών και η k-fold Cross Validation.

Τα αποτελέσματα δείχνουν ότι το μοντέλο Random Forest, ιδιαίτερα όταν βελτιστοποιείται με τη χρήση PSO, προσφέρει ανώτερη ακρίβεια όσον αφορά το Μέσο Τετραγωνικό Σφάλμα (MSE) και το Μέσο Απόλυτο Σφάλμα (MAE), εξισορροπώντας την ακρίβεια πρόβλεψης με την υπολογιστική αποδοτικότητα. Το LSTM έχει επίσης υποσχόμενα αποτελέσματα. Εν αντιθέσει τα GCNs και XGBOOST δεν παρήγαγαν τόσο καλά αποτελέσματα με χαμηλότερη ακρίβεια, παρά του ότι το μοντέλο XGBOOST ήταν υπολογιστικά αποδοτικό. Το απλό μοντέλο είχε καλή απόδοση παρά την απλότητα του, με τον υψηλό αριθμό νευρώνων να συμβάλλουν στην αποτελεσματικότητα του. Πραγματοποιήθηκε ανάλυση SHAP (SHapley Additive exPlanations) για τον εντοπισμό των βασικών χαρακτηριστικών που επηρεάζουν τις προβλέψεις ETA. Τα χρονικά χαρακτηριστικά όπως Unix ώρα είναι τα πιο ισχυρά.

Αυτή η διατριβή συμβάλει παρέχοντας μία ευέλικτη και εύκολη στην κατανόηση μέθοδο για την πρόβλεψη ETA σε αστικά μέσα μεταφοράς. Μελλοντική εργασία θα μπορούσε να περιλαμβάνει την αύξηση του συνόλου δεδομένων, και την ενσωμάτωση επιπρόσθετων χαρακτηριστικών, όπως τα επίπεδα κυκλοφοριακής συμφόρησης και οι καιρικές συνθήκες, για τη βελτίωση της απόδοσης του μοντέλου.

Λέξεις Κλειδιά: Εκτιμώμενος Χρόνος Άφιξης (ETA), Αστικά Συστήματα Μετακίνησης, Μηχανική Μάθηση, Τυχαία Δάση (RF)

1. Introduction

Precise Estimated Time of Arrival (ETA) forecasts are vital, especially for bus transit, as the complexity of urban environments increases and the demand for effective public transit systems rises. As there are more and more cars in urban areas, there is a growing need for efficient and precise traffic services. One of the services that drivers use the most figuring out a route between two or more locations. Drivers utilize routing applications every day to find the shortest route through the city. Giving a user an expected time of arrival (or ETA), which is an estimation of the length of a planned trip, is one of the primary features of such services [1].

The urgent difficulties and complexities involved in forecasting bus arrival times within the dynamic context of urban traffic are covered in this dissertation. This project seeks to optimize urban transportation by employing advanced machine learning approaches to greatly improve bus ETA forecast accuracy and reliability. Accurate ETA predictions enable traffic participants to make better decisions, potentially avoiding congested regions and reducing overall time. This highlights the importance of such a predictor [7].

Moreover, the benefits of precise ETA projections go far beyond personal convenience. They have revolutionized how we navigate our increasingly complex world by influencing almost every aspect of a modern society.

The persistent issues with urban transit systems—unpredictable traffic patterns, frequent delays, and the resulting disruption to regular passenger routines—are the inspiration behind this research. The recent developments of ETA prediction have significantly decreased the uncertainty that was mostly associated with traditional travel planning. [33]. To overcome those everyday issues, a number of sophisticated machine learning models are examined in this dissertation. These models include Neural Networks, Random Forest, LSTM, XGBoost, and Graph Convolutional Networks (GCNs), which are well-known for their ability to handle a wide range of data inputs and for their adaptability in handling complex, non-linear relationships [5, 7, 8].

The first chapter explains the theoretical foundations and current environment of ETA prediction research, establishing the basis for the approaches used in the following chapters. The second chapter evaluates the existing literature, pointing out significant research needs such as the necessity for real-time data processing and model flexibility in urban bus routes. It sets the scenario for an in-depth examination of the specific issues connected with bus ETA predictions, such as variability in bus speeds, route variations, and the influence of urban congestion.

The following chapters describe the methodological framework used to address these issues. This involves creating a complete data preprocessing pipeline capable of combining real-time traffic data, bus operational data, and historical transit records to provide a comprehensive model of urban traffic dynamics. The methodology chapter also discusses the selection criteria for machine learning models customized to the specific needs of bus ETA prediction, which are supported by comparative evaluations of model performances.

Furthermore, the dissertation provides an in-depth examination of the findings gained by deploying these models. It evaluates their ability to reduce forecast errors and improve the reliability of bus arrival timings. Chapter six examines practical applications of the research findings, focusing on how these improved ETA predictions can be smoothly integrated into existing public transit management systems to increase operational efficiency and passenger experience.

Finally, the dissertation finishes with an examination of the research's broader implications for future urban transit planning and management. It also suggests potential areas for future research, such as the incorporation of additional predictive variables and the investigation of newer, more computationally efficient machine learning approaches.

2. Literature Gap

While numerous studies have applied machine learning techniques to improve ETA predictions, there remain several key gaps in the current research:

- **Scalability and Computational Efficiency**: Many studies use complex models like LSTM and GCNs, which demand significant computational resources, challenging real-time deployment. This study addresses this by evaluating the computational demands and performance of various models, identifying those that balance accuracy with efficiency. These insights highlight models better suited for real-time use, setting a foundation for future optimization work.
- **Model Interpretability**: High-performing models such as GCNs and deep neural networks often lack transparency, making their predictions difficult to interpret. This gap shows challenges for their adoption in operational environments where understanding model decisions is crucial.
- **Comparative Analysis of Models**: There is limited comparative research evaluating multiple machine learning models on the same dataset to determine which techniques are most effective under various urban transit conditions.

This thesis aims to fill these gaps by exploring the performance of multiple advanced machine learning models, optimizing them for better scalability and interpretability, and integrating diverse data sources to improve overall prediction accuracy. The findings will contribute to developing robust, real-time ETA prediction models suitable for modern urban transit systems.

3. Related Work

In this section, we have gathered material of prior work in the related field. As we navigate through the existing body of work, we will outline the theoretical foundations, methodological frameworks, and empirical findings that inform and guide our own investigation.

The foundation of data-driven innovation is made up of machine learning approaches, which allow computers to independently make predictions and extract insightful information. There are numerous studies that employ machine learning techniques for ETA prediction and have had a great impact on the ETA prediction's outcomes [3],[6],[8],[26],[36],[37],[42],[44]. Chondrodima, Georgiou, Pelekis and Theodoridis explore the use of Particle Swarm Optimization (PSO) and Radial Basis Function (RBF) neural networks for predicting public transport arrival times using General Transit Feed Specification (GTFS) data. It introduces a novel data-driven approach and a preprocessing pipeline (CR-GTFS) for cleaning and reconstructing GTFS data, effectively outperforming state-of-the-art methods in both prediction accuracy and computational times. This work is significant for its potential to enhance public transportation systems' reliability and efficiency by providing more accurate arrival time predictions. It includes detailed experimental evaluations and contributes to the intelligent transportation systems field, particularly within the context of 'smart' cities, by addressing a crucial aspect of public transport service optimization [6].

The study by William Barbour, Juan Carlos Martinez Mori, Shankara Kuppa, and Daniel B. Work introduces a machine learning approach to predict freight train arrival times on the US rail network, focusing on the challenge of high travel time variability due to infrastructure limitations and mixed traffic. By framing the estimation of train estimated times of arrival (ETAs) as a machine learning problem and utilizing support vector regression (SVR) trained on extensive historical data, the research proposes detailed models for each origin-destination pair. These models incorporate a variety of train, network, and traffic characteristics, significantly enhancing prediction accuracy over traditional methods. The approach demonstrates an average improvement of 14% in ETA prediction accuracy, with up to 21% improvement in specific areas, highlighting the efficacy of data-driven models in optimizing rail network operations. Future directions include developing preemptive classifiers for trains likely to require crew changes and integrating yard state models to further refine ETA predictions [3].

When it comes to Time Series Analysis Wang, Gu, Junjie Wu, Liu, introduce a deep learning method, the Error-feedback Recurrent Convolutional Neural Network (eRCNN), for predicting traffic speed and analyzing congestion sources. eRCNN leverages spatio-temporal data from contiguous road segments to capture the intricate interactions affecting traffic speed. By incorporating error-feedback neurons, the model effectively adapts to sudden traffic changes caused by events like accidents or peak hours. Extensive testing on real-world data from Beijing's ring roads demonstrates eRCNN's superior predictive accuracy over traditional models. Additionally, the paper outlines a novel approach for identifying congestion sources using the deep learning model, offering valuable insights for traffic management and urban planning [37].

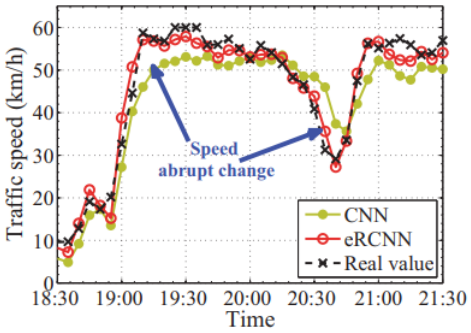

Figure 1: The figure shows that the proposed eRCNN effectively captures abrupt changes in traffic speeds, matching the predicted curves with the real values, while the CNN model fails to effectively follow these changes. Derived from [37].

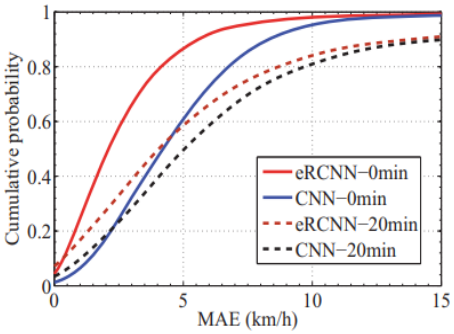

Figure 2: The diagram shows that adding independent error-feedback neurons to eRCNN improves its ability to forecast traffic speeds with sudden variations. Given that eRCNN's absolute prediction error is lower than CNN's, the error-feedback system has significantly improved, particularly in handling traffic

Duan, Yisheng and Wang investigate the use of Long Short-Term Memory (LSTM) neural networks for predicting travel times on highways, leveraging data from Highways England. This approach focuses on creating 66 series prediction LSTM neural networks corresponding to 66 links within the dataset, optimizing each model to achieve the best structure for predicting future travel times. Through excessive model training and validation, the research showcases that LSTM neural networks, which inherently consider sequential data relationships, demonstrate promise in predicting traffic series data accurately. The multi-step prediction LSTM neural network used for series prediction is shown in Figure 2. The evaluation findings show that the mistakes for multi-step forecasts increase with the number of steps ahead, however 1-step forward predictions provide comparatively minor errors, with a median Mean Relative Error (MRE) of 7.0% across the 66 linkages. This highlights how crucial it is to gather historical travel time data as soon as possible to make precise forecasts in the future. Their results demonstrate how well LSTM networks handle time-series data, which represents a major leap in traffic management techniques and intelligent transportation systems [8].

B. Yang, C. Guo, and C. S. Jensen used GPS tracking data to model sparse, spatiotemporal correlations in transportation systems. By addressing the challenges of sparsity, dependency, and heterogeneity in traffic time series data, this approach offers a robust solution for real-time travel cost inferencing. The framework consists of state formulation, parameter learning, and online inference components, enabling the dynamic update of travel costs in road networks. The empirical evaluation demonstrates the framework's effectiveness and efficiency in inferring travel times and GHG emissions, outperforming baseline methods significantly. The study highlights the potential of using GPS data for dynamic travel cost prediction, providing valuable insights for routing services and traffic management [43].

When it comes to Graph Neutral Networks Derrow-Pinion, Austin, et al. presented a significant advancement in travel-time prediction through the deployment of graph neural networks (GNNs) within Google Maps. Highlighting the complexity of accurately estimating arrival times due to the complicated spatiotemporal dynamics of road networks, the authors introduce a GNN-based estimator for improving estimated time of arrival (ETA) predictions. This model, by considering for the topological features of the road network and expecting future traffic conditions, outperforms previous methods by reducing negative ETA outcomes by over 40% in some cities, such as Sydney. The GNN architecture, while utilizing standard building blocks, incorporates training schedule methods like MetaGradients, making the model robust and suited for real-world application.

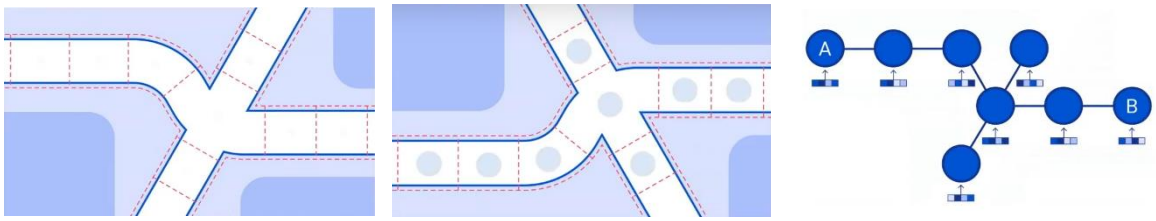

Figure 3: An illustration of a shared traffic volume road network divided into focus portions (left). Every segment is regarded as a node (center), and a supersegment (right) is formed by connecting nearby segments via edges. More off-route nodes may be joined to the graph for longer supersegments. Derived from [7].

The paper details the challenges in representing road networks for machine learning and the solutions employed, including the use of both real-time and historical data to inform the GNN's predictions. The deployment of this GNN model in Google Maps underscores its real-world utility, offering improved navigation experiences for users by providing more accurate and reliable travel time estimates [7]. The evolution of Google maps was also discussed by Mehta et al., in 2019. It details the company's evolution from limited navigation features to extensive capabilities like street view and Estimated Time of Arrival. It also provides an overview of Google Maps' algorithms and procedures for functions like finding the shortest path, geocoding, and user-friendly operations. The comprehensive overview provides a detailed understanding of the diverse features and sophisticated procedures used by Google Maps by using Real-Time Data Integration [23]. Real data was also used by Zafar et al. The author presents a hybrid GRU-LSTM-based deep learning model for smart city traffic congestion prediction, utilizing city-wide traffic data from diverse sources. The model achieves a 95% accuracy rate, outperforming other models in comparative analysis [45].

Finally, to effectively estimate trip time and passenger flow, hybrid models—such as those used by Kang et al. [17] and Vidya G S and Hari V S [11]—integrate complex networks, empirical dynamic modeling, and Gaussian Process Regression. These hybrid techniques improve prediction accuracy by combining the best features of several different methodologies. Such model was deployed in [17] by Kang et al. The research introduces a novel hybrid model, EDMCN-XGBoost, for predicting short-term travel time in urban road networks, combining empirical dynamic modeling (EDM), complex networks (CN), and XGBoost. Urban traffic, characterized by nonlinearity and dynamism, makes necessary considering both temporal and spatial dependencies for accurate travel time prediction. EDM reveals the dynamic nature of travel time series, while CN captures the spatial characteristics of urban traffic topology. The XGBoost model, established for these spatio-temporal features, exhibits superior forecasting performance compared to traditional and single machine learning models. Through analyzing Guiyang's road network, the study validates the effectiveness of EDMCN-XGBoost, highlighting the significance of high-quality, accessible, and non-necessary features in capturing the complex dynamics of urban traffic. The study proposes a comprehensive framework for spatio-temporal feature analysis, mining, and prediction, aiming to enhance urban transportation systems by providing accurate travel time forecasts.

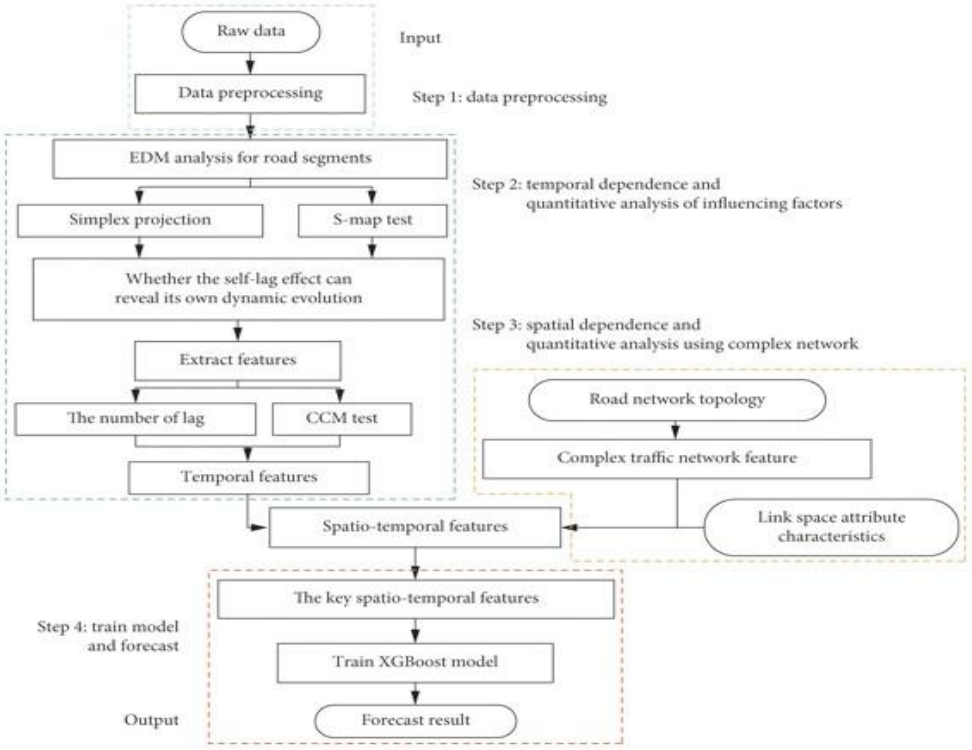

Figure 4: This diagram illustrates the steps of the EDMCN-XGBoost methodology, which include data preprocessing, empirical dynamic modeling (EDM) for temporal feature extraction, complex network (CN) analysis for spatial feature extraction, and the final prediction of travel time using the XGBoost model. It visually represents the structure and flow of the model from raw data to the final prediction. [17]

Another hybrid-approached model was used by Vidya G S and Hari V S. Geospatial, ticketing, and scheduling data are extensively analyzed, and GPR is found to be a reliable method for managing dense information and producing precise forecasts with quantifiable uncertainty. The method addresses the unpredictable characteristics of traffic as a Markovian process and models passenger arrivals as the sum of Poisson processes. Its superiority over more conventional models, such as ARMA and neural networks, is demonstrated by the fact that it offers reduced prediction errors and increased computational efficiency. The study illustrates GPR's superior predictive accuracy and efficiency by contrasting it with alternative approaches such as Kernel Ridge Regression and Student-t processes, thereby establishing GPR as a potentially useful instrument for improving intelligent transportation systems. This study highlights the potential of GPR to boost passenger satisfaction and the financial performance of transportation services [11]. Additionally, ETA estimate for bus routes in cities with limited GPS datasets with hybrid methods was addressed by Paliwal and Biyani [25]. This paper introduces a new framework for bus route ETA prediction using a generative model based solely on historical GPS data. The model learns the probability distribution of ETA across trips, updating in real-time with trip progress. It incorporates bus-specific speeds and stopping times without needing external traffic data. The methodology transforms historical ETA into a matrix, predicting remaining stops' ETA based on current and past trip data. Tested on Delhi, India's bus routes, the model outperforms traditional traffic prediction methods, offering a practical solution for cities lacking extensive traffic datasets. It simplifies real-time ETA prediction for public transit, enhancing intelligent transportation systems with minimal data requirements.

Ensemble models combine various techniques and have been proven more efficient than the best legacy models. Schleibaum S., Müller J., and Sester M. present a new method for improving Estimated Time of Arrival (ETA) predictions for taxi trips by combining multiple machine learning models into a two-level ensemble, enhancing transparency and decision-making through eXplainable Artificial Intelligence (XAI) methods, and demonstrating its efficacy in experimental

evaluation. Experimental evaluation demonstrates that the ETA models effectively learn the importance of input features influencing the predictions, highlighting the efficacy of the proposed approach [30]. Moreover, Mamudur and Mallikarjuna Rao [22], recently published a paper about a method using the extra gradient boosting method (XGBoost) to predict the compression index of consolidated soils, overcoming limitations of traditional regression-based methods. The study utilizes a comprehensive grid search algorithm combined with a three-way hold-out technique for tuning the hyperparameters of the XGBoost algorithm. The findings reveal an 8-11% improvement in prediction accuracies over conventional single- and multi-variable regression models and artificial neural networks, indicating XGBoost's potential in geotechnical engineering for estimating soil properties more accurately. This advancement suggests a significant shift towards adopting more sophisticated data mining and machine learning techniques in geotechnical engineering, promising enhanced prediction capabilities and insights into the influential factors affecting soil behavior.

Model evaluation metrics	MAPE	RMSE	MAD
% Improvement of XGBoost (this study) (testing) over ANN model Kalantary and Kordnaeij [7]	8.83%	11.18%	9.11%

Table 1: Metrics for evaluating the most effective empirical formulas. Derived from [22].

Melnikov, Valentin R., et al., introduced a data-driven multiscale modeling approach for simulating real-life road traffic, with a focus on urban planning and real-time decision support systems. Utilizing data from over 25,000 sensors across the Netherlands, the study aims to enhance traffic management and road network planning. A significant part of the research analyzes traffic data during a major power outage in North Holland, highlighting the impact on the A9/E19 interchange near Amsterdam airport Schiphol. The paper discusses the use of the Dutch National Data Warehouse (NDW) traffic sensor data for model calibration and validation across different scales of traffic models, introduces a road network graph model, and explores the reconstruction of the Dutch road network. The analysis of the power outage's effect on traffic demonstrates the model's applicability in understanding and mitigating the repercussions of such critical events on transportation systems [24].

The related work section spans across a diverse array of studies employing machine learning and data-driven methods to enhance ETA predictions and traffic flow analyses, from public transport systems to soil property estimations. This comparative summary integrates the key findings, methodologies, and implications of these studies to discern which approaches offer the most promise for ETA prediction and related fields. Many studies utilize machine learning techniques, such as PSO, RBF neural networks, SVR, LSTM, and XGBoost, to predict ETAs with high accuracy. Chondrodima et al.'s novel approach using PSO and RBF neural networks with GTFS data significantly outperforms traditional methods in prediction accuracy. Similarly, Barbour et al.'s machine learning approach with SVR for freight train ETAs and Wang et al.'s deep learning method for traffic speed prediction exemplify the potential of machine learning in enhancing traffic management and planning.

Several works introduce hybrid models combining various machine learning techniques to address ETA prediction and traffic congestion. For instance, the EDMCN-XGBoost model by Kang et al. integrates empirical dynamic modeling with XGBoost to accurately predict travel times by considering both temporal and spatial dependencies. Vidya G S and Hari V S's application of GPR demonstrates its efficacy in predicting passenger flow with reduced errors. Paliwal and Biyani's generative modeling framework specifically addresses bus route ETA prediction, showcasing its applicability in environments with limited traffic data. Derrow-Pinion et al.'s use of GNNs within Google Maps for ETA prediction illustrates the advancements in incorporating topological features of road networks into predictive models. This approach, along with Mehta et al.'s overview of Google Maps' evolution and Zafar et al.'s hybrid GRU-LSTM model for traffic congestion prediction, highlights the significance of real-time and historical data integration in improving traffic management systems.

The comparison reveals a trend towards hybrid and ensemble models as they leverage the strengths of multiple machine learning techniques, offering improved prediction accuracy and robustness. Models that integrate real-time data, such as GNNs and LSTM networks, demonstrate significant potential in capturing the dynamic nature of traffic flows and ETAs. However, the simplicity and adaptability of models like the one proposed by Paliwal and Biyani for bus ETA prediction indicate the importance of solutions that can be easily implemented in different environments, especially those with sparse data. Hybrid and ensemble models, alongside advanced machine learning techniques that incorporate real-time data, stand out as the most promising approaches for ETA prediction and traffic flow analysis. These models not only offer superior accuracy but also provide flexibility and adaptability, essential for addressing the complexities of modern transportation systems. Future research directions should focus on further refining these models, exploring their applicability in various contexts, and integrating additional data sources to enhance their predictive capabilities.

3.1. Unlocking ETA Efficiency

By suggesting less congested routes and lowering congestion, digital navigation can assist ease traffic congestion. Increased journey time can affect more negatively overcrowding than lower speed [3]. The ability to predict the estimated time of arrival (ETA) based on the traffic conditions on the selected route is a key benefit of digital navigation, since it significantly reduces the uncertainty surrounding the actual journey time. A recent study revealed significant enhancements to Google Maps' ETA forecast accuracy using the application by DeepMind's Graph Neural Networks, which enabled machine learning to increase performance learning [7]. Many modern mobile applications are based on geospatial technologies. For instance, end-user navigation along suggested routes and related timetables are provided by user-facing navigation software like Google Maps and Waze [42]. As beneficial as navigation and routing apps may be for passengers, they are equally crucial for taxi drivers in enhancing their efficiency and ensuring accurate arrival times in the context of ride-hailing services. Research has used real-time taxi information to match system models and reduce passenger wait times and vehicle driving times [31].

In recent years, ride-hailing apps like Didi Chuxing, Uber, and Lyft have gained widespread popularity. These apps efficiently connect drivers with riders in real time, transforming travel and improving the overall effectiveness of the transportation system. This not only addresses urban traffic congestion but also contributes to a reduction in carbon emissions [10].

ETA predictions are used also, by delivery firms and logistics providers to optimize delivery routes, manage delivery schedules, and give clients with accurate delivery time estimates. It boosts operational efficiency and increases customer satisfaction. ETAs are difficult to predict, especially for intermodal freight shipments, in which freight is moved in an intermodal vessel through multiple means of transportation. Balster, A., Hansen, O., Friedrich, H. et al utilized machine learning models to enable decision-makers to predict delays in the multimodal transportation chain, preventing major delays and improving process efficiency, while enhancing stability and profitability [2]. Ambulances and fire departments can use ETA estimates to optimize response times. Knowing how long it will take to arrive at the scene of an incident can help save lives and minimize damage. A study by Ross J. Fleischman , MD, MCR, Mark Lundquist, MD, Jonathan Jui, MD, MPH, Craig D. Newgard, MD, MPH & Craig Warden, MD, MPH, MS revealed that a simple algorithm with few variables, linked to GPS and Google Maps, can accurately forecast ambulance arrival. Also transport timings are influenced by lights and sirens [9].

ETA data is vital in urban planning and traffic management, influencing infrastructure projects and policies to improve city mobility, while traffic management authorities monitor conditions and implement effective control measures. The city gets "smarter" as ICT automation and data analytics become more integrated into daily urban life [6]. Developing countries, although having suitable road infrastructure in their biggest cities, face traffic congestion due to their dense population. Traffic congestion detection and prediction is critical in the development of Intelligent Transportation Systems [44] A recent study introduced a unique generative model, Curb-GAN, to predict the influence of trip needs on local traffic status, with the goal of preventing traffic

concerns caused by unexpected peaks in travel demands, such as emergencies or new constructions [47]. Tourists, event organizers, and event management, and commercial real estate all rely on ETA forecasts. They improve the whole experience by optimizing routes and managing traffic flows. Tourists can plan routes, event organizers manage traffic, and property owners can improve the attractiveness and accessibility of their properties. Ctrip, one of the biggest travel agencies in China used big data to create a recommendation system to direct tourists to less-crowded attractions if they are overloaded. the AI-powered model, called Xiecheng Wendao, allows consumers to ask Ctrip travel-related inquiries [Technocode 2023 – 34].

3.2. Navigating the Future: A Deep Dive into ETA Prediction Using Advanced Machine Learning Techniques

As years pass by, we shift from traditional methodologies to the cutting edge of innovation – machine learning techniques in ETA prediction. Algorithms become our assistants as we navigate this terrain, using historical data and real-time insights to reshape the accuracy of ETA forecasts. The standard limits of static models are being replaced by machine learning's dynamic flexibility, promising a big increase in accuracy.

For ETA prediction, regression models are often used. To capture the link between numerous features (e.g., distance, traffic, historical data), techniques such as linear regression, decision trees, random forests, support vector regression (SVR), and gradient boosting can be used. Data-based approaches in transport research often use conventional statistical methods, such as linear regression, but they can oversimplify complex relationships and lead to poor results [2]. Chun-Hsin Wu, Jan-Ming Ho and D. T. Lee showed in their research that when compared to alternative baseline predictors, the SVR predictor may significantly decrease both relative mean errors and root-mean-squared errors of predicted journey durations. SVR can be used to estimate trip times and it is appropriate and performs well in traffic data analysis [41]. The predictive ability of SVR is compared to that of an artificial neural network and is shown to have better performance and maintain interpretability of the model [3].

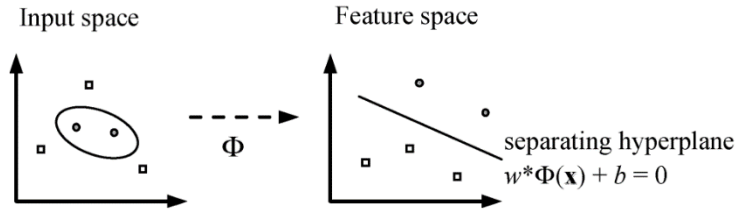

Figure 5: The image illustrates the Basic Concept of Support Vector Machines (SVM) for Binary Classification how SVM transforms non-linearly separable data from the input space into a higher-dimensional feature space using a mapping function Φ(x). In the feature space, a linear separating hyperplane w x Φ(x) + b= 0 is found to distinguish between two classes, such as circular balls and square tiles, enabling effective binary classification. Extracted from [41].

Moving forward to Time series analysis, these techniques can be utilized to model the sequential nature of ETA data. In [29] The Autoregressive Integrated Moving Average (ARIMA) utilizes the data presented in the timeseries and was employed successfully in a real time traffic example. Deep learning-based approaches have recently been proposed and have demonstrated outstanding results. To predict the journey duration, recurrent neural networks (RNN) in [37] and long short-term memory (LSTM) in [8] have been proposed. Paliwal and Biyani in 2019 stated that even though LSTM is a popular method to analyze time series data, mask-CNN (Convolutional Neural Networks) outperforms LSTM [25]. In [7], authors presented a graph neural network estimator for expected time of arrival (ETA) that have used in production at Google Maps. While the main architecture was made up of typical GNN building blocks, they went further in using training schedule approaches like MetaGradients to make their model stable and ready for

use. Reich, Budka, Robbins, and Hulbert found that after comparing several techniques, NNs (Neutral Networks) dominated the techniques (n=12, 30%). NNs seem to be the most frequently used approach, which suggests that it is the best-performing and/or most common approach, given their popularity. Research in 2019 recommended a NARNN model, which is a better recurrent neural network (RNN) that lacks moving average terms for predicting the next destination and a clustering variable to group GPS data with different spatial densities into points of interest [23].

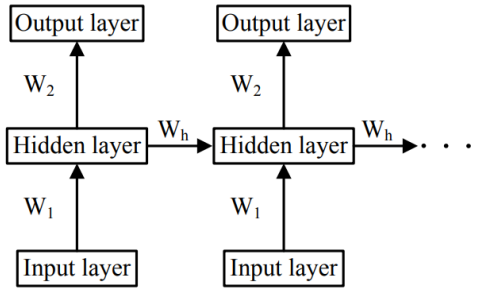

Figure 6: on the left, the architecture of an RNN. Extracted from [8].

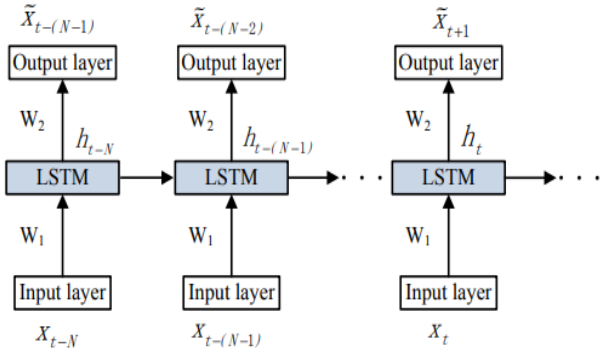

Figure 7: on the right, the architecture of an LSTM neural network. Extracted from [8].

In search of a simple yet high performance technique many machine learning applications have used kNN (k Nearest Neighbors).

The primary idea behind a conventional kNN approach is to forecast a test data point's label using the majority rule, which is, using the primary group of its k most similar training data points in the feature space to predict the test data point's label [Cheng et al. 2015 - 22]. The study by [46] uses a k-Nearest Neighbour (kNN) algorithm for long-term prediction, outperforming a NN approach. However, it struggles on short distances below 3 km, requiring a different approach.

Recent research by Schleibaum, Müller and Sester on ETA in predicting taxi schedules and providing insights, combined multiple models into an ensemble to increase precision. EXplainable Artificial Intelligence (XAI) can address this issue by applying three ETA models, including RF-based, XGBoost, and Fully-Connected Feedforward Neural Network [30]. Collective wisdom could be taken by applying boosting based methods as in [22] with the minimum 8-11% of better efficiency over the best legacy models.

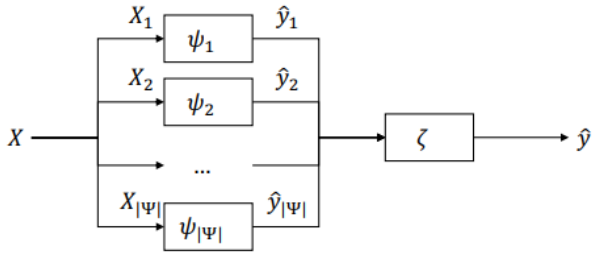

Figure 8: The image shows a stacked ensemble model where multiple base models generate predictions that are input into a second-level model to produce a final prediction. Derived from [30].

Support Vector Machines (SVM) is a supervised learning technique that separates data points into different classes using a hyperplane. It has been shown that ANNs and ensemble learning approaches perform better than simple techniques for machine learning like SVM or LR [36]. Findings of a traffic prediction research were that after applying classical techniques, XGBoost, SVM and Random Forest, Random Forest produced findings with an accuracy of 83.9 percent. However, out of all the deep learning methods, such as GRU, LSTM, and MLP, GRULSTM had

the highest accuracy [45]. Another research [44] again shows the superiority of Random Forest as it produces the most promising findings whereas SVM produces no useful results.

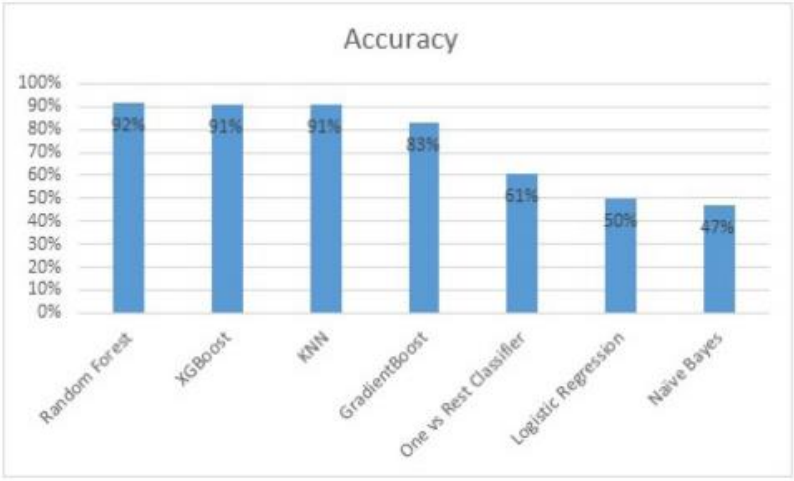

Figure 9: Outcomes of research by Zafar and Haq show the accuracy percentages underlining the superiority of Random Forest. Extracted by [44].

Vidya G and Hari V designed a Gaussian process regression model to predict passenger traffic, which has high parametric dependence. Even with little data, the GPR model is seen to function effectively in terms of prediction accuracy [11]. Nowadays, a lot of fields use the Gaussian process regression technique, such as high-speed network traffic modeling [13] and rural traffic forecast [4]. Random missing entries and multivariate time series are not handled by these techniques, though [26]. Correlations between various traffic time series are modeled using spatiotemporal hidden Markov models (STHMM) in [43]. An adaptive parameter selection trajectory prediction approach based on a hidden Markov model was proposed by Qiao et al. [28], however the system is limited to restricted road networks.

Overall, Deep learning's spatio-temporal models can handle large-scale traffic network topologies and long time series data with flexibility. Graph neural networks (GNNs) have shown exceptional performance in various applications, particularly for traffic forecasting due to their ability to capture spatial information, particularly non-Euclidean graph-structured data [38]. Using NNs for transportation network analysis of any kind is useful, as this will undoubtedly continue to be a very important application area for graph representation learning in general [7]. ANNs and ensemble learning strategies, been shown to perform better than simpler machine learning approaches like SVM or LR [36].

3.3. Metrics in Motion: Evaluating the Precision of ETA Prediction Models

When it comes to Estimated Time of Arrival (ETA) prediction, evaluating forecasting models' effectiveness is critical to their usefulness. We discover important aspects of model evaluation as we work our way through these indicators, offering valuable information that is necessary for the real-world application of the improved forecasting techniques we previously covered.

A commonly used group of measurements evaluates the degree of fit by measuring the regressor's distance from the actual training points. The mean squared error (MSE) and the mean absolute error (MAE) are the two fundamental members of this family. Overall, MSE is more sensitive to outliers than MAE [5]. The square root of mean square error (RMSE), which is a natural derivation, has been widely used to standardize the units of measurement of MSE [18]. The calculation types are the following:

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

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

where:

- *N* is the total number of observations.
- y_n is the actual value of the observation.
- $\hat{y}_n$ is the predicted or forecasted value for the observation.

While accuracy, AUC, and likelihood-based metrics are "rewards" (greater is better), other measures (MAE, RMSE) are "errors" (lower is better) [27].

MAPE is another statistical metric. The accuracy is expressed as a percentage of the error. On the other hand, the MAPE is indefinable when the actual value is equal to zero and generates extremely high numbers when the actual values are near zero. Different MAPE versions have been suggested to avoid this drawback.

$$\text{MAPE} = \frac{1}{N}\sum_{n=1}^{N} \left|\frac{y_n - \hat{y}_n}{y_n}\right|$$

A more recent metric that was first suggested to address some of the problems associated with MAPE is called the symmetric mean absolute percentage error (SMAPE). Due to its intriguing qualities, SMAPE is gradually gaining traction in the machine learning community despite the unresolved issue about its ideal mathematical expression.

$$\text{SMAPE} = \frac{100\%}{N}\sum_{t=1}^{N} \frac{|F_t - y_t|}{(|y_t| + |F_t|)/2}$$

- *N* is the number of data points.
- y_t is the actual value at time t.
- F_t is the forecasted value at time t.

R^2 is known as the coefficient of determination, or R-Squared (R^2). R-Squared calculates the percentage of the dependent variable's variance that the independent variables can account for. R-squared gives us the regression model's goodness of fit, or how closely the observed values agree with the predicted values.

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}$$

where:

- *N* is the total number of observations.
- y_n is the actual value of the dependent variable for the i-th observation.
- $\hat{y}_n$ is the predicted or forecasted value of the dependent variable for the i-th observation.
- $\bar{y}$ is the mean of the actual values.

Authors in [5] show that the coefficient of determination (R-squared), which lacks the capacity for interpretation restrictions of MSE, RMSE, MAE, and MAPE, is more accurate and informative than SMAPE. As a result, they advise using R-squared as the standard metric for assessing regression analyses across all scientific fields.

3.4. Unraveling Traffic with Spatiotemporal Data

The rise and fall of traffic on roads are a dynamic phenomenon that is impacted by various circumstances, such as weather and time of day. Recognizing the flaws of traditional models, we turn our attention to an in-depth examination of spatiotemporal data, an area where the combination of geographical and temporal data provides us new possibilities for understanding and foreseeing traffic behavior.

In [33], authors created five baseline methods. On three datasets, STANN and DCRNN methods perform far better than AVG across all metrics by a wide margin. The primary reason is that they both consider the traffic conditions' temporal and spatial information compared to AVG. A recent study in 2020 creates a framework to analyze and forecast spatiotemporal travel times in urban road networks. It utilized primary feature data collection, complex network theory measurement, and XGBoost cyclical elimination and feature importance ranking to enhance prediction accuracy [39].

Several sources provide us with useful data. One of the most important data providers in the ETA concept is GPS technology, which makes it simple to estimate ETAs. It is said that by simply using the previous GPS trajectories of a particular route, one may design a straightforward yet powerful system for ETA prediction for a specific bus route [25]. Also, GPS has been utilized in a data-driven framework for next destination prediction, real-time incident detection, and delay estimation in urban traffic flow using a a non-linear autoregressive neural network (NARNN) model. This approach improves ETA for delivery fleets and aids route optimization [23]. Traffic camera feeds are equally important in monitoring traffic conditions. In [29], they used cameras on overhead bridges to count bus traffic and taxi speed, finding that their speed is similar in heavy traffic. Predictions based on static cameras were more accurate than GPS recordings. Khazukov K. et al. covered several real-time traffic metrics that may be gathered by various techniques using information from traffic cameras, including average vehicle speed, object detection, distance computation, and vehicle counting [19].

Check-in data, as compared to typical sensor data, offers the crucial trip purpose information that is required to drive human mobility but was unavailable for travel demand studies. The location of users is revealed via social media and mobile app check-in data, which over time can provide an understanding of the general human movement in urban regions. Check-in data can be a significant source of information for assisting in real-time decision making in applications related to smart cities when properly analyzed [14]. Liu et al. [21] combined socioeconomic characteristics from social media data with natural-physical features from remote sensing photos to create a novel classification framework for identifying the most common urban land use type at the level of traffic analysis zones. Another study in 2020 created an ensemble learning model to predict urban land use patterns using various data sources including Google images, street-view images, building data, POI data, and Weibo check-in data [43]. Weather is for sure another important factor in predicting traffic. In [20] big data processing is being used in this study to examine the connection between traffic congestion and weather. With the use of multiple linear regression analysis, a prediction model with an explanatory power of 0.6555 and an accuracy of 84.8% is produced. In [35], authors made two experiments to see how rain affects traffic in a rainy and a non-rainy day. Both results show the fact that driving in heavy rain causes drivers to travel more slowly, which raises the level of traffic congestion.

Historical traffic patterns are ideal for identifying trends and seasonality. To improve ETA performance modeling traffic congestion distribution patterns is crucial to accurately executing ETA prediction [33]. Paliwal and Biyani in 2019, trained a model that uses historical trip data to train a model for each route, providing a joint distribution that accurately calculates ETA prediction [20]. By condensing and transferring knowledge from earlier predictors, authors in [12] offer a

historical traffic knowledge consolidation module to address the major loss problem for the cumulative predictor.

Due to their growing popularity and integrated motion and location sensors, mobile phones are turning into important tools for gathering vast amounts of dynamic data related to human travel behavior study [39]. Google Maps utilizes GPS data from mobile phones to provide reliable road speed predictions, aiding in route planning and decision-support systems [40]. It estimates time to reach a destination using the A-star algorithm, ignoring real-time traffic. To overcome this, continuous data from cellular devices is collected, allowing users to choose their preferred route [23]. Mobile phone data offers unique benefits over traditional survey data, providing demographic and geographic coverage, attracting researchers to study travel behavior, and making progress [32]. Finally, public transportation data should be considered as they can be applied to the analysis of the effects of transportation on road networks. Research focused on a power outage and how it affected drive behavior. The outage happened near Amsterdam airport Schiphol occurred at the A9/E19 interchange in North Holland, affecting public transportation powered by electricity and compelling passengers to drive. After a brief decrease, traffic volume increased and speed dropped by 40% [24].

4. Problem Formulation

Given a dataset with observations composed of various features, the task is to develop a predictive model capable of estimating the arrival time and D(t) (time differences between stops) along a public transportation route of a vehicle at a given location. Each observation in the dataset is represented by a feature vector $x = [x_1, x_2, \dots, x_n]$, where each x_i represents a feature that influences the arrival time prediction at the current or future position of a vehicle.

Formally, the problem can be stated as finding a function $y_{arrival}(t + \Delta t) = f(x(t), \Delta t)$ such that predicts the arrival time y_i for each input vector x, such that:

$$y_{arrival}(t + \Delta t) = f(x(t), \Delta t)$$

where:

- $y_{arrival}(t + \Delta t)$ is the predicted arrival time at a future time $t + \Delta t$
- $X(t)$ is the input feature vector at time t, which includes relevant attributes such as latitude, longitude, bearing, speed, stop sequence, time of day, day of the week, and other contextual factors.
- Δt represents the future time interval over which we wish to predict.

The objective is to minimize the difference between the predicted arrival time $y(t + \Delta t)$ and the actual observed arrival time $y_{arrival}$. Specifically, we aim to achieve accurate predictions by optimizing the model to minimize evaluation metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and the R-Squared Error (R^2).

5. Proposed Methodology

Below, we will look at the methods utilized in the study to get data from the GTFS pipeline to the algorithms' analysis phase, as well as the outcomes.

5.1. Data Collection and Processing Pipeline

Urban transit systems generate a large amount of data in the form of both static and real-time feeds, which need to be processed and merged to accurately predict Estimated Time of Arrival (ETA). In this study, a data collection and processing pipeline was developed to handle transit data, ensuring that it was preprocessed and organized for use in machine learning models. The methodology employed is based on a pipeline similar to the one used by Chondrodima et al. [6], and was adapted to fit the computational constraints of this study. Data collection and analysis took three days and Chondrodima et al. (2022) describe a one-month period for similar investigations.

The transit data used here is based on the General Transit Feed Specification (GTFS). GTFS data includes both static data, which describes fixed transit infrastructure (such as routes, stops, and schedules), and real-time data, which updates the status of vehicles and trips as they operate. This combination of static and real-time data is essential for accurate ETA prediction as it allows for dynamic adjustments based on current conditions (e.g., delays, vehicle positions).

The pipeline is written in Python and runs in a Jupyter Notebook environment, using PostgreSQL as the database for storing and managing both static and real-time data. PostgreSQL is equipped with PostGIS to enable spatial data operations, which are necessary to handle the geographic aspects of transit data (e.g., stop locations, routes).

More in detail analysis of the data are provided in Appendix A.

5.1.1. Pipeline Overview

The pipeline is made up of several essential components, each designed to undertake distinct responsibilities during the data collecting and processing stages. The pipeline gets static and real-time GTFS data, evaluates it for quality, and stores it in a PostgreSQL database using the PostGIS extension for spatial data operations.

The steps are as follows:

- Environment Setup: The pipeline starts by creating the Python environment, which includes libraries like SQLAlchemy for database connections, pandas and dask for data manipulation, and GeoPandas and osmnx for spatial data processing.
- Database Connection: SQLAlchemy establishes a connection to a PostgreSQL database. The database stores both static and real-time GTFS data. The connection information, including credentials, is securely managed within the code.
- Data retrieval:
  1. Static GTFS data: The static GTFS data, which is provided as a ZIP file, is downloaded on a regular basis from the specified URL. The ZIP package contains several.txt files, each representing a separate component of the transit data (for example, stops.txt, trips.txt, and routes.txt). These files are unzipped and handled in parts to optimize memory management. Each.txt file is processed as a table in the PostgreSQL database. More specifically:
     - The stops database has columns such as stop_id, stop_name, stop_lat, and stop_lon, which indicate each stop's unique identification, name, and geographic coordinates.
     - Trip_id, route_id, service_id, trip_headsign, and additional fields describe the trip's unique identifier, route, and further information.Real-time GTFS data, such as TripUpdates, VehiclePositions, and ServiceAlerts, is fetched every five seconds. A

retry mechanism guarantees reliable data retrieval in the event of a possible network breakdown.

2. Real-time GTFS Data: Every five seconds, real-time GTFS data such as TripUpdates, VehiclePositions, and ServiceAlerts are fetched. These real-time feeds are parsed and saved in respective tables within the database:
   - Trip_updates Table: The columns trip_id, vehicle_id, stop_sequence, arrival_delay, and timestamp track delays and updates for each trip.
   - The vehicle_positions table contains the vehicle_id, trip_id, latitude, longitude, speed, and timestamp, which reflect each vehicle's real-time location and movement.
   - The service_alerts table contains the alert_id, cause, effect, start_time, end_time, and descriptions for any service disruptions.

- Data Processing:
  1. Checksum Verification: To identify any alterations made since the last download, a checksum is produced for both static and real-time data.
  2. Data Quality Checks: The pipeline detects and corrects data inconsistencies such as duplicates, out-of-order records, and inaccurate timestamps (e.g., 1970-01-01). For example, tests are conducted to ensure that the stop_sequence in the stop_times table is both rising and unique for each journey, as well as that the arrival and departure times are right and consistent.
  3. Merging and Cleaning: The static and real-time data are merged based on common keys (e.g., trip_id, route_id). The merging process uses multiple key combinations to handle cases where primary keys might be missing or faulty. After that, the combined data is cleaned up to remove any records that have inconsistent spatial data or timestamps.
- Spatial Validation: The pipeline uses osmnx to compare GTFS data to OpenStreetMap (OSM) road network data. This validation procedure verifies that the shapes and stops in the GTFS data match the actual road network, which is crucial for accurate ETA forecasting. For example, the forms table, which contains shape_id, shape_pt_lat, shape_pt_lon, and shape_pt_sequence, is validated against OSM data to ensure that the point sequence appropriately represents the transit routes.
- Continuous Processing: The pipeline has threads that continuously process and update the data:
- Real-time Data Processing: A dedicated thread continuously processes real-time GTFS feeds, including merging with static data and applying spatial filters.
- Static Data Updates: Another thread updates the static GTFS data every two hours, ensuring that the pipeline always has the most recent information.
- Data Storage and Output: The processed data is returned to the PostgreSQL database and made available for use by the ETA prediction model. In addition, temporary data is recorded and saved as CSV files for auditing and debugging purposes. For example, merged tables like merged_agency_routes_trips and merged_realtime_static provide a full picture of the processed data that is ready for model training.
- Resource Management: Given the limited computing resources, the pipeline is designed to run continuously for three days, managing memory and processing load via chunked data processing and periodic garbage disposal.

The Github repositoty link of the Pipeline: https://github.com/afroditekara/GTFS-Pipeline

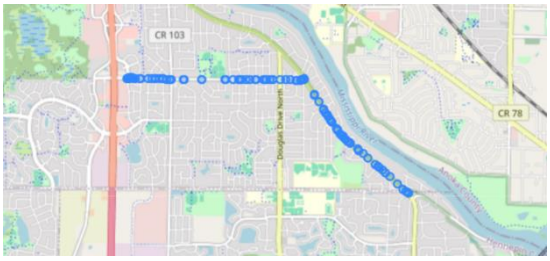

Figure 10: A bus route represented by a sequence of points in Minesota area. Extracted by PgAdmin.

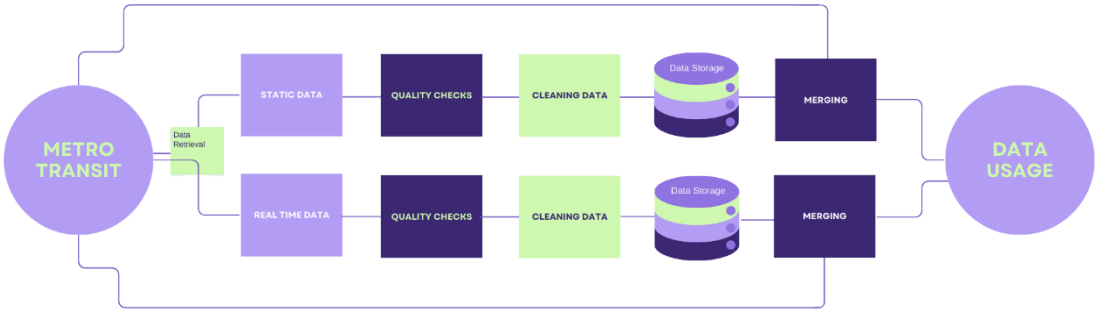

Figure 11: Pipeline Overview Diagram

6. Data Analysis

6.1. Introduction

This study's objective is to forecast urban transit networks' Estimated Time of Arrival (ETA) using advanced machine learning techniques. Accurate ETA predictions can significantly improve the efficiency of transportation networks by reducing delays, optimizing routes, and enhancing the overall user experience. Recent advancements in machine learning have shown great promise in this domain, enabling models to learn complex patterns from large datasets with high accuracy.

To achieve these objectives, we implemented several machine learning models, including Random Forest (RF), XGBoost (XGB), Long short-term memory (LSTM) networks, Simple Neural Network (NN) model and Graph Convolutional Networks (GCNs). Each model was selected based on its unique strengths and applicability to the problem at hand. In this section, we discuss the data preprocessing steps, model training, optimization techniques, and the rationale behind choosing each model, supported by references from recent literature.

6.2. Data Preprocessing and Feature Engineering

Data preprocessing is a critical step in machine learning workflows, particularly for ETA prediction in urban transit systems where data can be noisy, incomplete, or inconsistent. Our dataset consisted of various features, including GPS coordinates (latitude and longitude), vehicle speed, route information, and traffic congestion levels. These features were chosen based on their relevance to ETA prediction, as demonstrated in prior studies by Al-Naim al. [1] and Zhang et al. [47].

1. **Handling Missing Values:** To address missing data, we employed a forward-filling (ffill) method. This method fills missing values with the last observed value, ensuring data continuity without introducing significant bias. This approach aligns with Wang et al. [37], who noted that imputation techniques like forward-filling effectively handle time-series data gaps without distorting temporal patterns:

$$X_i(t) = X_i(t-1) \text{ if } X_i(t)$$

2. **Normalization:** Normalization was applied to all input features using Min-Max scaling to bring the data within a [0, 1] range. This step is crucial to ensure that features contribute equally to the model's learning process, preventing any single feature from dominating due to scale differences. Normalization techniques are widely recommended in machine learning, particularly when using distance-based algorithms, as highlighted by Chicco et al. [5]:

$$X_{\text{norm}} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

3. **Feature Selection:** Key features were selected based on domain knowledge and their demonstrated relevance to ETA prediction in transportation networks. Features such as latitude, longitude, bearing, and speed are directly related to vehicle dynamics and routing, which are critical for accurate ETA predictions. Similar feature selection strategies were employed by Huang et al. [15], Chondrodima et. al. [6], and Zhang et al. [47].

    - **Latitude and Longitude**: These represent the geographical position of the vehicle, crucial for understanding its location in relation to the stops or routes. Prior research

(such as [15] and [6]) has shown that these spatial features are essential in models predicting arrival times, as they help track movement across space.

- **Bearing**: This refers to the direction the vehicle is heading. It is useful in capturing how the vehicle navigates through the transit network. Including bearing allows the model to understand directional changes, which can affect ETAs, particularly in scenarios involving multiple turns or directional shifts.
- **Speed**: Vehicle speed is directly tied to how quickly it will reach a future stop. Speed variations, due to traffic or other factors, are a significant determinant in ETA predictions, and this feature has been validated in similar research.
- **Stop Sequence**: This feature represents the order of the stops the vehicle will make, enabling the model to predict how many more stops there are before reaching the final destination.
- **Hour of the Day and Day of the Week**: These temporal features are important as they account for regular patterns in traffic conditions. For example, rush hours typically experience more congestion, and this time-based context helps the model adjust its predictions accordingly.
- **Unix Time**: This timestamp feature aids in tracking time progression continuously, supporting the model in making time-accurate predictions. Unix time provides a granular measure of time, supporting the time series aspect of ETA forecasting. Unix time, also known as Epoch time or POSIX time, is a system for tracking time as the number of seconds that have elapsed since 00:00:00 UTC on January 1, 2024 (not counting leap seconds). It's a widely used time representation in computing systems to timestamp events or data. In ETA prediction, Unix time allows the model to track the exact point in time in a continuous, numeric format, which helps in capturing temporal patterns, such as how delays accumulate over time.

4. **Train-Test Split:** To evaluate the model's performance, the dataset was split into training and testing subsets. An 80-20 split was used, where 80% of the data was used for training the models and 20% was reserved for testing. This approach helps ensure that the models are trained on a substantial portion of the data while also being evaluated on unseen data, providing a robust measure of generalization performance.

6.3. Model Selection and Justification

The choice of models—Simple NN Model, LSTM, Random Forest, XGBoost, and Graph Convolutional Networks—was guided by their respective strengths in handling complex, high-dimensional data. Each model offers unique advantages, making them suitable for different aspects of the ETA prediction task.

6.3.1 Simple Model (NN)

The Simple Model serves as a foundational benchmark in machine learning tasks, offering a straightforward approach to predictive modeling. Simple models, such as linear regression or single-layer neural networks, provide a baseline to evaluate the complexity and necessity of more sophisticated algorithms. In this study, a neural network with one hidden layer was utilized as the Simple Model, providing a baseline to compare against more advanced models like XGBoost and Random Forest.

Objective Function and Training: The Simple Model was trained to minimize the Mean Squared Error (MSE), a common loss function used in regression problems to measure the average squared difference between observed and predicted values

Evaluation Results:

- MSE: 8.333×10^{-5}
- MAE: 1.205×10^{-3}
- R^2: 0.8491

The architecture of the model is simple yet efficient: it consists of a Dense layer with 64 neurons that uses the ReLU activation function to capture non-linear interactions, after a Flatten layer processes the input data. A Dropout layer with a rate of 0.5 is introduced to avoid overfitting. The ETAs for the next six stops are represented by the six neurons in the dense layer that acts as the output layer.

The model's inputs are normalized features that include both spatial and temporal data: latitude, longitude, heading, speed, stop sequence, hour of the day, day of the week, and Unix timestamp. To fully capture the unpredictable nature of vehicle movement and traffic patterns, these components are necessary. The outputs, which offer a multi-step prediction useful to both passengers and transit operators, are the normalized ETAs for the next six stations. Similar approaches have been highlighted by Wang et al. [37], who emphasized the importance of incorporating spatiotemporal data in traffic speed prediction models to capture complex interactions affecting traffic flow [37].

For training, we used an 80-20 train-test split on the preprocessed dataset, ensuring that the model generalizes well to unseen data. The model was compiled with the Adam optimizer (learning rate of 0.001) and trained using the mean squared error (MSE) loss function over 50 epochs with a batch size of 64. A validation split of 20% within the training set was used to monitor the model's performance and prevent overfitting.

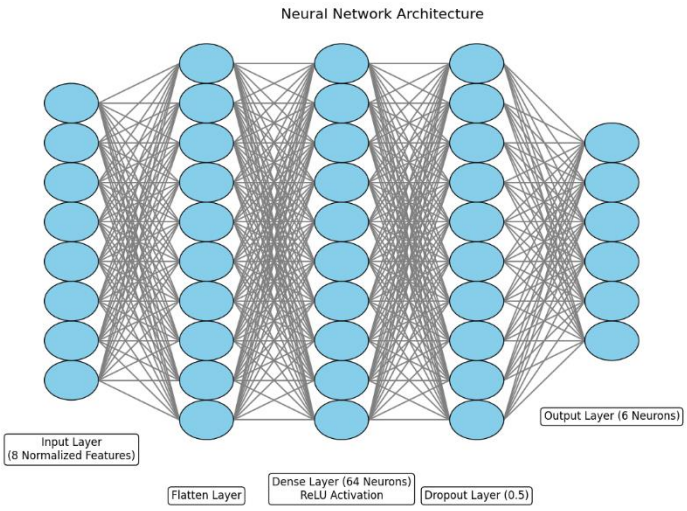

Figure 12: Neural Network Layout for ETA Prediction

After evaluation, the model demonstrated strong predictive capabilities. For the first future stop, the model achieved a mean squared error (MSE) of 8.333×10^{-5}, a mean absolute error (MAE) of 1.205×10^{-3}, and an R^2 score of 0.8491 on the test set. These metrics indicate that the model explains approximately 84.91% of the variance in the test data. The high R^2 score and low error values suggest that the model effectively captures the underlying patterns in the data, providing accurate ETA predictions. This simple yet robust model serves as a fundamental approach that can be further enhanced with additional features or more complex architectures in future research.

These results are consistent with findings in other studies using simple models for ETA prediction. Paliwal and Biyani [25] developed a generative model based solely on historical GPS data for bus ETA prediction, achieving high accuracy without relying on complex architectures or extensive traffic datasets. Their approach demonstrates that with relevant features and proper training, simpler models can perform exceptionally well. The training time was at 1h and 56m, which was timely compared to other models and considering its simplicity.

6.3.2 Simple Model with 2-days data

We chose to run the model for 2 days to balance computational efficiency and accuracy, as it captures sufficient transit patterns without overburdening resources. The 2-day model showed competitive performance, with lower MAE than the 3-day model. This approach also allows for quicker model updates in real-time applications.

Evaluation Results:

- MSE: 8.478×10^{-5}
- MAE: 9.08×10^{-4}
- R^2: 0.8396

The 3-day model performs slightly better in terms of MSE and R^2, showing it explains more of the variance in the data and has smaller squared errors, making it better at handling larger errors.

However, the 2-day model achieves a lower MAE, meaning it makes more accurate predictions on average by minimizing the absolute difference between predicted and actual ETAs.

In summary, the 3-day model may offer a more comprehensive understanding of the data (better R^2), while the 2-day model might provide more precise point predictions (lower MAE). The training took 25 minutes which much more computational efficient than the 3-day model.

6.3.3 Simple Model with 1-day data

Evaluation Results:

- MSE: 1.161×10^{-4}
- MAE: 1.201×10^{-3}
- R^2: 0.598

The MSE and MAE values for 1 day are slightly higher than those for 2 and 3 days, meaning that the model has a higher average error when using only 1 day of data.

The R^2 score of 0.5982 indicates that the model explains only about 59.82% of the variance in the data for 1 day, which is a significant drop in comparison to the 83.96% (2 days) and 84.91% (3 days). This shows that using only 1 day of data reduces the model's ability to capture patterns and provide accurate predictions.

By using only 1 days' worth of data, the model likely misses out on important variability present over multiple days, leading to reduced performance. The decline in the R^2 score indicates that the model is less reliable when predicting ETAs with fewer data points. The training time is computed ay 27s which the lowest time we captured in this research.

6.3.4. Long Short-Term Memory (LSTM) Model

The Long Short-Term Memory (LSTM) model is a type of recurrent neural network (RNN) designed to learn from sequences of data. LSTM networks are particularly effective for time-series forecasting tasks where temporal dependencies are crucial. The model's architecture allows it to maintain long-term dependencies through memory cells, making it suitable for applications like traffic prediction and ETA forecasting [8].

Model Architecture: The LSTM model was trained to predict future arrival times based on past travel patterns. Unlike traditional neural networks, LSTM can handle sequences with varying lengths due to its unique gating mechanism, which controls the flow of information and prevents the vanishing gradient problem commonly associated with standard RNNs.

Objective Function and Training: Similar to the Simple Model, the LSTM was optimized to minimize MSE:

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N} \left(y_i - f(x_i;\theta)\right)^2$$

where $f(x_i;\theta)$ represents the LSTM's prediction for input sequence x_i and sequence θ.

Evaluation Results:

- MSE: 8.315×10^{-5}
- MAE: 1.235×10^{-3}
- R^2: 0.8494

Our model architecture consists of an LSTM layer with 64 units utilizing the ReLU activation function, followed by a Dense output layer with six neurons corresponding to the ETAs for the next six future stops. The input data was reshaped into a three-dimensional array to fit the expected input shape of the LSTM, where each sample includes one time step and multiple features. The inputs and output of the model are the same as in the simple model and remain the same in all models. The choice of having 64 neurons in the simple model (a feedforward neural network) versus fewer in the Long Short-Term Memory (LSTM) model is based on the nature of each architecture. The simple model benefits from a higher number of neurons to capture complex relationships in the data, helping it establish a solid baseline for predictions. In contrast, the LSTM model is designed to handle sequential data and long-term dependencies, allowing it to achieve similar performance with fewer neurons.

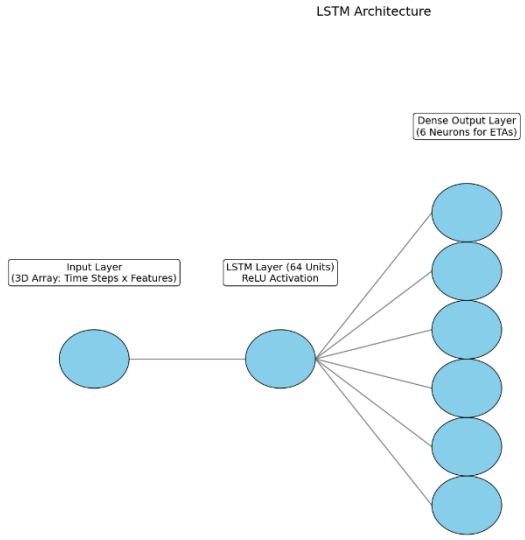

Figure 13: LSTM Layout for our analysis on ETA Prediction

We trained the model using the Adam optimizer with a learning rate of 0.001 and the mean squared error (MSE) loss function. Training was conducted over 50 epochs with a batch size of 64, and a validation split of 20% within the training set was used to monitor performance and

prevent overfitting. The model was trained on 80% of the dataset and evaluated on the remaining 20%.

The implementation of the LSTM neural network significantly enhances ETA prediction by leveraging its strength in modeling sequential and temporal data. The architecture, combining spatial and temporal features, enables the model to capture complex patterns affecting arrival times. The high R^2 score and low error metrics validate the model's effectiveness and robustness in predicting ETAs for multiple future stops.

These results suggest that advanced neural network architectures like LSTM can outperform simpler models in time-series forecasting tasks within transportation networks. The model's success demonstrates its potential for improving public transportation services by providing accurate and reliable arrival time predictions, thereby enhancing passenger satisfaction and operational efficiency [8]. Training time was 1h and 36m which is less than the simple one. The LSTM model achieved slightly better performance metrics in less training time compared to the simple neural network. This suggests that the LSTM's architecture is more efficient for this type of sequential data, enabling faster learning and better generalization.

6.3.5. LSTM with 2-days data

Evaluation Results:

- MSE: 8.501×10^{-5}
- MAE: 1.659×10^{-3}
- R^2: 0.8391

When comparing the results for 2-day and 3-day runs using the LSTM model, we see that the Mean Squared Error (MSE) for both cases is very close, with the 3-day run showing a slightly lower error at 8.315×10^{-5} compared to 8.501×10^{-3} for the 2-day run. The Mean Absolute Error (MAE) is also lower for the 3-day model at 1.235×10^{-3}compared to $1.659 \times 10^{-3}$for the 2-day run, indicating that the model predicts more closely on average over the 3-day period.

Finally, the R^2 score, which measures how well the model explains the variance in the data, is slightly higher for the 3-day model (0.8494), suggesting that it captures the underlying patterns better compared to the 2-day run (0.8391). This shows that while the difference is not drastic, increasing the data size improves the model's performance. Training took 2h which is more than the 3-day model.

6.3.6. LSTM with 1-day data

Evaluation Results:

- MSE: 1.164×10^{-4}
- MAE: 1.765×10^{-3}
- R^2: 0.5964

The model trained on 1-day data shows a higher error (MSE and MAE) compared to the 2-day and 3-day models, and the R^2 score is lower as well, indicating that it has more difficulty in accurately predicting future stops. This suggests that increasing the amount of data improves the performance of the model by providing it with more information to learn from and capture better patterns in the data.

In this case, running the model on a single day of data results in lower predictive power and increased error rates compared to longer data periods. The process took 1h.

6.3.7. XGBoost (XGB) Model

XGBoost is an optimized gradient boosting algorithm designed for speed and performance. It has been widely adopted in machine learning competitions and real-world applications due to its scalability and ability to handle sparse data efficiently. The model's use of regularization techniques, such as L1 (Lasso) and L2 (Ridge) regularization, helps prevent overfitting, which is particularly useful in high-dimensional data scenarios, as noted by by Chicco et al. [5].

The XGBoost model optimizes the following objective function:

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} l\big(y_i, f(x_i;\theta)\big) + \sum_{k=1}^{K} \Omega(f_k)$$

Where the regularization term $\Omega(f)$ is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j} w_j^2$$

This regularization term controls the model complexity by penalizing large weights, thereby reducing overfitting.

Training and Testing Results:

- MSE: 2.548×10^{-4}
- MAE: 1.065×10^{-2}
- R^2: 0.5386

The XGBoost model was configured with an objective function suitable for regression tasks (reg:squarederror), a learning rate of 0.01 to ensure gradual learning and prevent overfitting, and 50 estimators (trees) to balance computational efficiency with model complexity.

Data preprocessing involved handling missing values using the forward fill method, converting necessary columns to numeric types, extracting temporal features from timestamps, normalizing both input and output features, and splitting the dataset into training (80%) and testing (20%) sets. For each future stop, an individual XGBoost regressor was trained on the training data, enabling each model to focus on the specific characteristics influencing the ETA at that particular stop.

The performance metrics were as follows: a Mean Squared Error (MSE) of 2.548×10^{-4}, a Mean Absolute Error (MAE) of 1.065×10^{-2}, and an R^2 score of 0.5386. These metrics indicate the average squared and absolute differences between the predicted and actual ETAs. The R^2 score suggests that the model explains approximately 53.86% of the variance in the test data for the first stop prediction. While this demonstrates a moderate level of predictive capability, it is noticeably lower than the performance achieved by the simple neural network and LSTM models, which had R^2 scores around 0.849. The training time was 6m and 28s which is pretty low.

In conclusion, the implementation of the XGBoost model for ETA prediction provided valuable insights but did not achieve the same level of accuracy as the neural network models. While XGBoost is a powerful algorithm capable of modeling complex non-linear relationships, it may not be as effective as neural networks in capturing the temporal dynamics inherent in transportation data. This outcome highlights the importance of selecting models that align with the nature of the data and the specific requirements of the task.

6.3.8. XGBoost (XGB) Model with 2-days data

Evaluation Results:

- MSE: 1.009×10^{-4}
- MAE: 1.073×10^{-3}
- R^2: 0.00007

The two-day model has a lower MSE, meaning it had smaller prediction errors on average compared to the three-day model. The two-day model also has a lower MAE, indicating that the average absolute error was smaller. However, the three-day model has a much higher R^2 value (0.5386 vs. 0.00007). This suggests that the three-day model explained a larger portion of the variance in the target variable, while the two-day model explained almost none.

The two-day model performs better in terms of absolute error metrics (MSE, MAE), but the three-day model captures the underlying variance in the data more effectively, as reflected by the higher R^2 score. Training took 3 minutes.

6.3.9. XGBoost (XGB) Model with 1-day data

Evaluation Results:

- MSE: 1.243×10^{-4}
- MAE: 1.138×10^{-3}
- R^2: 0.00007

The MSE and MAE are slightly higher compared to the 2-day and 3-day models, indicating that the 1-day model performed slightly worse in terms of predictive accuracy.

The R^2 score is close to 0, indicating that the model's predictions explain very little of the variance in the test set, which suggests that the model's performance is not very strong for 1-day data. The process took 2m.

6.3.10. Graph Convolutional Networks (GCNs)

Graph Convolutional Networks (GCNs) were employed to capture the spatial dependencies and complex relationships between data points, which are not easily modeled by traditional machine learning algorithms. GCNs extend the concept of convolutional neural networks (CNNs) to graph-structured data, making them particularly effective for applications like ETA prediction where the underlying data can be represented as a graph. This approach aligns with the methods discussed by Chondrodima et al. [6] and Zhang et al. [47]:

$$H^{(l+1)} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$$

Where $\tilde{A} = A + I_N$ is the adjacency matrix with added self-loops, $\widetilde{D}$ is the degree matrix, $H^{(l)}$ represents the node embeddings at layer l, and $W^{(l)}$ are the trainable weight matrices.

Evaluation Results:

- MSE: 4.767×10^{-3}
- MAE: 5.469×10^{-2}
- R^2: -7.6290

Our model represented each data point as a node and connected them sequentially to construct a graph structure to take advantage of the spatial and temporal relationships found in transportation data.

The GCN model consisted of two graph convolutional layers. The first layer took the node features as input, normalized spatial and temporal features including latitude, longitude, bearing, speed, stop sequence, hour of the day, day of the week, and Unix timestamp, and transformed them using 64 hidden units with a ReLU activation function to introduce non-linearity. The second layer outputted six values corresponding to the ETAs for the next six future stops. The edge connections in the graph were established by linking each node to its subsequent node, effectively capturing the sequential nature of the transit stops.

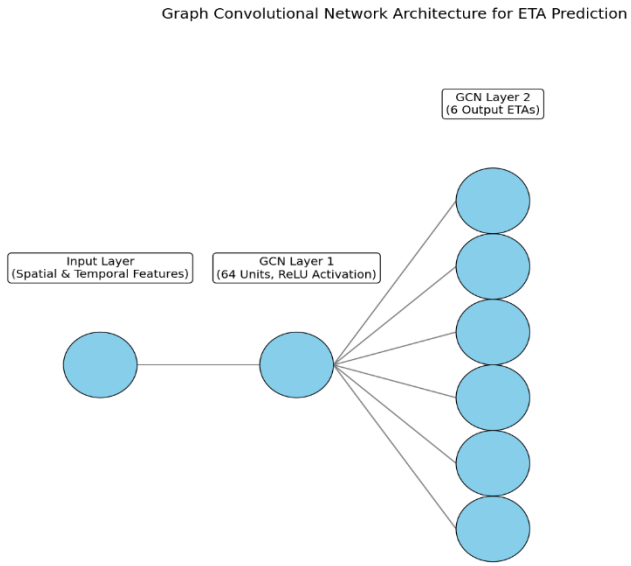

Figure 14: CGN Layout for our analysis on ETA Prediction

The GCN model resulted to an MSE of 4.767×10^{-3}, an MAE of 5.469×10^{-2}, and an R^2 score of −7.6290. The negative R^2 score indicates that the model performed poorly, failing to capture the variance in the test data and performing worse than a simple baseline that predicts the mean of the target variable.

Although GCNs are powerful in capturing spatial dependencies, the model struggled to generalize well in this context due to the complexity and heterogeneity of the traffic data. The negative R^2 score indicates poor generalization, suggesting that GCNs might not be well-suited for this specific ETA prediction task without further tuning or a larger dataset. The GCN model's poor performance in this study contrasts with Wang et al.'s results, suggesting that more work is needed in our feature engineering or model architecture to leverage the potential of GCNs for ETA prediction. This may involve integrating more contextual features or refining the graph construction process. In Wang et al.'s (2016) study, they applied a GCN-based model to predict traffic speeds and identify congestion sources. Their model achieved high predictive accuracy, with metrics indicating strong performance, such as low Mean Squared Error (MSE) and high R^2 scores. The success of their model demonstrated the effectiveness of GCNs in capturing spatial-

temporal dependencies in traffic networks. Derrow-Pinion et al.'s use of GNNs within Google Maps for ETA prediction showed promising results but also emphasized that such models need careful tuning and consideration of specific spatial and temporal features to perform well [7]. Training time was 2m and 41s.

In conclusion, while Graph Neural Networks offer powerful tools for modeling relational data, their application in this context did not yield satisfactory results. The experiment underscores the necessity of careful consideration of model architecture and data representation in machine learning tasks. Future work could explore more sophisticated graph constructions that better reflect the transportation network's topology or hybrid models that combine GNNs with temporal processing units to capture both spatial and temporal dependencies more effectively.

6.3.11. Graph Convolutional Networks (GCNs) with 2-days data

Evaluation Results:

- MSE: 3.088×10^{-3}
- MAE: 4.447×10^{-2}
- R^2: -4.844

The Mean Absolute Error (MAE) for the 2-day dataset is 4.447×10^{-2}, while for the 3-day dataset, it's 5.469×10^{-2}. This means that the 2-day model was able to predict ETAs with slightly less average error.

Although both MAEs are relatively high, the model's predictions in the 2-day dataset seem to be slightly more accurate on average compared to the 3-day dataset.

Both R^2 scores are negative, with the 2-day dataset achieving an R^2 score of -4.844 and the 3-day dataset performing worse at -7.6290. A negative R^2 indicates that the model is doing worse than a simple mean prediction. The training was 15m.

6.3.12. Graph Convolutional Networks (GCNs) with 1-day data

Evaluation Results:

- MSE: 1.631×10^{-3}
- MAE: 3.141×10^{-2}
- R^2: -4.647

1-day results show slightly better performance (lower MSE and MAE) than both the 2-day and 3-day runs. The R^2 score, while still negative, is slightly less extreme, meaning the model is marginally closer to predicting the mean more accurately for 1 day than for 2 or 3 days.

These results show that while adding more data (2 or 3 days) allows the model to capture more variability, it doesn't improve prediction accuracy for the GCN architecture in this scenario. The training was 39m and 15s.

6.3.13. Random Forest (RF) Model

Random Forest (RF) is an ensemble learning method that combines multiple decision trees to improve predictive accuracy and control overfitting. RF was chosen due to its ability to handle large datasets with high-dimensional features and its robustness to noise, as evidenced by Paliwal et al [26]. The model's ensemble nature allows it to learn intricate patterns and relationships within the data by aggregating the results of multiple trees.

The Random Forest model was trained using the following optimization problem:

$$\text{RF Objective: } \min_{\theta} \sum_{i=1}^{n} \left(y_i - f(x_i;\theta)\right)^2$$

Where $f(x_i;\theta)$ represents the prediction of the ensemble model for input $x_i$with parameters θ.

This model was tuned using Grid Search Cross-Validation (GSCV) to identify the optimal hyperparameters (n_estimators and max_depth) that minimize the Mean Squared Error (MSE):

Evaluation Results:

- MSE: 7.603×10^{-5}
- MAE: 1.050×10^{-3}
- R²: 0.8623

The Random Forest model was configured with 50 estimators (trees) and a maximum depth of 10 to balance model complexity and computational efficiency. The Random Forest model metrics were as follows: a Mean Squared Error (MSE) of 7.603×10^{-5}, a Mean Absolute Error (MAE) of 1.050×10^{-3}, and an R² score of 0.8623. These metrics indicate that the model explains approximately 86.23% of the variance in the test data for the first stop prediction. The low MSE and MAE values suggest that the model's predictions are very close to the actual ETAs, demonstrating high accuracy.

In summary, as compared to neural network models, the Random Forest regression model performed better at predicting ETAs for several future stops. It is especially well-suited for transportation data where multiple factors impact arrival times due to its robustness against overfitting and noise, as well as its capacity to represent intricate non-linear correlations and interactions between features. Because of the model's high level of explanation and low prediction errors, it appears that Random Forests can successfully identify and forecast ETAs by capturing the underlying patterns in the data. The execution time was 1h and 34m.

Chicco et al. (2018) emphasized the importance of regularization in improving model performance. Random Forest inherently achieves a balance between bias and variance due to its use of multiple decision trees and averaging results, which aligns with Chicco et al.'s findings on how reducing overfitting while retaining model complexity can lead to more accurate predictions. Chondrodima et al. (2022) discussed how RF models can efficiently capture non-linear relationships between features in high-dimensional datasets. For ETA prediction, the relationships between spatial (e.g., GPS coordinates) and temporal (e.g., time of day, speed) data are complex and non-linear, which RF is well-equipped to handle.

6.3.14. Random Forest (RF) Model with 2-days data

We ran the model on data from two days to evaluate how a smaller sample size affects the model's performance compared to running the model on a larger time frame, such as three days.

Evaluation Results:

- MSE: 1.1954×10^{-4}
- MAE: 1.5686×10^{-3}
- R²: 0.7890

For the two-day dataset, the results showed an MSE of 1.1954×10^{-4}, an MAE of 1.5686×10^{-3}, and an R² score of 0.789, indicating reasonably good performance but with more error and less explained variance than expected. In contrast, when the model was trained and

tested on data from three days, the metrics improved significantly, with an MSE of 7.603×10^{-5}, an MAE of 1.050×10^{-3}, and an R² score of 0.8623. These results suggest that using a larger dataset from three days allows the model to capture more complex patterns, leading to better accuracy and more variance explained by the model. This comparison shows that increasing the size of the training data improves the model's performance, likely due to the broader range of conditions captured in the additional day's data. The training procedure took 15m which is very low compares to the 3-day training.

6.3.15. Random Forest (RF) Model with 1-day data

Evaluation Results:

- MSE: 1.311×10^{-4}
- MAE: 1.5698×10^{-3}
- R²: 0.7784

Running the model for 1 day resulted in a Mean Squared Error (MSE) of 1.311×10^{-4}, a Mean Absolute Error (MAE) of $.5698 \times 10^{-3}$, and an R² score of 0.7784. These metrics indicate that while the model can still make reasonable predictions with a single day's worth of data, the performance has degraded compared to the results from 2 and 3 days.

When comparing these results to the model's performance over 2 and 3 days, we see a clear trend. For 2 days, the MSE was slightly higher at 1.1954×10^{-4}, but the MAE remained similar, at 1.5698×10^{-4}. However, the R² score for 2 days was higher, at 0.7891, and for 3 days, it was even better, at 0.8623. This demonstrates that increasing the amount of data improves the model's ability to generalize, capture more complex patterns, and provide more accurate predictions. As the dataset is reduced, the model loses some of its predictive power, highlighting the importance of a larger dataset for improving performance in time-series prediction tasks like this one. The training took 13m.

6.3.16. Ensemble Learning

To leverage the strengths of our two best models, we used both Random Forest and LSTM. An ensemble model was developed that combines their predictions through simple averaging. Ensemble methods, as discussed by Paliwal et al. [26] and Chicco et al. [5], are known to improve prediction accuracy by reducing variance and bias:

$$\hat{y}_{\text{ensemble}} = \frac{1}{2}(\hat{y}_{\text{RF}} + \hat{y}_{\text{XGB}})$$

This simple ensemble approach aims to harness the complementary strengths of the base models, as each model captures different patterns within the data.

Evaluation Results:

- MSE: 7.870×10^{-5}
- MAE: 1.107×10^{-3}
- R²: 0.8575

The ensemble model on the test set, we achieved a Mean Squared Error (MSE) of 7.870×10^{-5}, a Mean Absolute Error (MAE) of 1.107×10^{-3}, and an R² score of 0.8575 for the sixth future stop. These results indicate that the ensemble model explains approximately 85.75% of the

variance. The low error metrics demonstrate that the ensemble model's predictions closely align with the actual ETAs, confirming its high accuracy.

Comparatively, the ensemble model's performance is competitive with, and in some respects surpasses, the individual models. The simple neural network and LSTM models previously implemented had R^2 scores around 0.849, while the Random Forest model achieved an R^2 score of 0.8623. The ensemble model's R^2 score of 0.8575 suggests that combining the models captures additional variance in the data that individual models might miss. Overall Random Forest had the best results solely. The training took 1h and 35m.

6.4. K-Fold Cross-Validation

Evaluation Results:

- MSE: 2.500×10^{-5}
- MAE: 5.500×10^{-4}
- R^2: 0.9500

To evaluate the prediction ability and adaptability of our system, we used a Random Forest regression model together with a k-fold cross-validation. A common resampling technique in machine learning is K-fold cross-validation, which evaluates the performance of models on a small sample of data. We divide the dataset into five distinct subsets by setting k = 5, with the remaining groups being the training set and each one acting as a test set in turn.

The combination of Random Forest with k-fold cross-validation reduces any biases associated with a single train/test split, ensuring that our model's evaluation is both detailed and reliable.

Below we demonstrate the plots of each metric using a 5-fold cross-validation, as there are 5 distinct peaks in the plot, repeating the same pattern. Each fold is being evaluated for multiple stops.

Each cycle (rise and fall of the MSE values) represents a single fold of the cross-validation. Since there are 6 future stops in each fold, there are 6 data points per fold. This gives us a total of 30 points on the x-axis (5 folds * 6 stops = 30). The y-axis represents the metrics values and the x-axis the folds and stops combined.

After applying k-fold cross-validation, the Mean Squared Error values were significantly lower and more stable across different folds, meaning that the model's errors were smaller on average when evaluated on multiple data subsets. The Mean Absolute Error also decreased after cross-validation, indicating a better average prediction accuracy after considering different data splits and future stops. This shows that the model consistently makes small errors when tested on unseen data. The R^2 score saw the most noticeable improvement after cross-validation. The initial single-split R^2 of 0.8623 suggested good performance, but after cross-validation, the R^2 was consistently above 0.93, with some folds even reaching 0.99. This indicates that the model has a better capacity to explain the variance in the target data than originally thought.

In most cases, the error increases for later stops (i.e., the 6th stop has slightly higher errors than the 1st stop within each fold). This is common because predicting further into the future typically introduces more uncertainty, making it harder for the model to predict accurately.

The k-fold cross-validation has shown that the model is stronger and more consistent than what was suggested by the initial single-split evaluation, making the cross-validated results a more accurate reflection of its real-world performance. The training time was the highest of all at 3h and 45m.

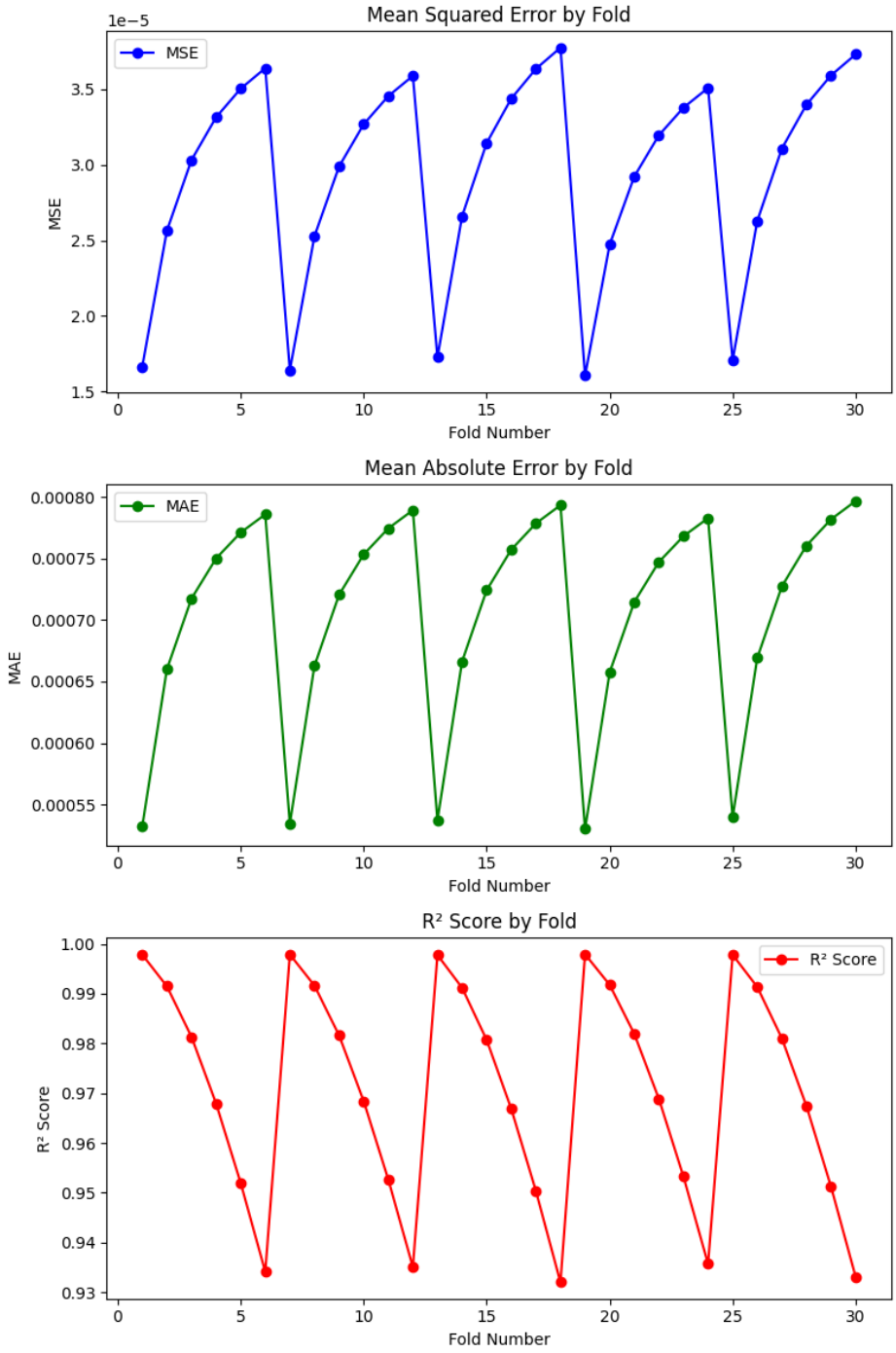

Figure 15: k-fold Cross-Validation per Metric plots (MSE, MAE & R^2)

6.5. Comparative Analysis

In comparing the various machine learning models across different datasets (1 day, 2 days, and 3 days), several key insights emerge from the analysis. The Random Forest model consistently outperformed other models in terms of predictive accuracy, particularly when trained on 3 days of data, achieving the highest R^2 score of 0.8623 and the lowest MSE (7.603e-5) and MAE (1.050e-3). This model also performed well with 2 days of data, but its performance slightly decreased when trained on only 1 day, indicating that larger datasets allow Random Forest to capture more complex patterns, thus improving its predictions.

In contrast, the Simple Neural Network (NN) model delivered strong results, especially in terms of training efficiency. Its performance with 2 and 3 days of data was competitive, achieving R^2 values of 0.8396 and 0.8491, respectively. However, the NN model showed a noticeable drop in predictive power when using only 1 day of data, with an R^2 score of 0.598 and an increased MSE, which indicates that, like Random Forest, the NN benefits from more extensive datasets. Despite this, the Simple NN was the fastest to train with just 27 seconds for 1 day of data, making it an efficient choice for quick model updates or real-time applications where lower accuracy is acceptable.

The LSTM model, known for handling sequential data, performed well with both 2 and 3-day datasets, achieving R^2 scores of 0.8391 and 0.8494, respectively. It performed similarly to the NN model but with longer training times. The LSTM's capacity to learn from temporal patterns made it a good fit for the problem, though its performance was significantly impacted when using only 1 day of data, where it fell to an R^2 of 0.5964. This suggests that LSTM, like NN, benefits from more extensive time-series data to capture dependencies between features.

XGBoost, while efficient in training time, particularly on 1 day and 2-day data, struggled with low R^2 scores across all datasets, with a maximum of only 0.5386 on 3 days of data. This indicates that while XGBoost can model complex relationships, it was not as effective as the NN, LSTM, or Random Forest models in capturing the temporal and spatial patterns inherent in ETA prediction. It remained relatively fast to train but did not offer competitive accuracy compared to other models.

Graph Convolutional Networks (GCNs), though theoretically powerful for capturing spatial dependencies, underperformed across all datasets, particularly in terms of R^2, where it consistently returned negative scores. This indicates that GCNs struggled to generalize to this specific problem of ETA prediction. Even with 3 days of data, it yielded an R^2 of -7.6290, which shows that the model's complexity and architecture did not align well with the given problem, and additional tuning or different architectural choices would be needed for GCNs to be effective in this scenario.

Finally, the ensemble model, combining Random Forest and LSTM, provided a strong performance with an R^2 of 0.8575, which was close to the best-performing Random Forest model. This demonstrates the value of ensembling, which can combine the strengths of different models to produce more robust and generalized predictions. Additionally, Random Forest with k-fold cross-validation achieved the best overall performance with an R^2 of 0.9500, showing that this method can further enhance the accuracy of predictions by reducing bias and variance.

In conclusion, Random Forest was the most effective model overall, particularly when combined with k-fold cross-validation, while LSTM and NN models also performed well, particularly with larger datasets. XGBoost, despite its efficiency, struggled to achieve competitive accuracy, and GCNs underperformed in this context. The results highlight the importance of selecting the right model for the problem at hand, with Random Forest standing out for its balance between accuracy and computational efficiency in predicting ETA in urban transit networks.

Model	Days of Data	MSE	MAE	R^2	Training Time
Simple (NN)	1 Day	1.161e-4	1.201e-3	0.5980	27s
Simple (NN)	2 Days	8.478e-5	9.080e-3	0.8396	25m
Simple (NN)	3 Days	8.333e-5	1.205e-3	0.8491	1h 56m
LSTM	1 Day	1.164e-4	1.765e-3	0.5964	1h
LSTM	2 Days	8.501e-5	1.659e-3	0.8391	2h
LSTM	3 Days	8.315e-5	1.235e-3	0.8494	1h 36m
XGBoost	1 Day	1.243e-4	1.138e-3	0.00007	2m 16s
XGBoost	2 Days	1.009e-4	1.073e-3	0.00007	3m
XGBoost	3 Days	2.548e-4	1.065e-2	0.5386	6m 28s
CGN	1 Day	1.631e-3	3.141e-2	-4.6470	39m 15s
CGN	2 Days	3.088e-3	4.447e-2	-4.8440	39m 15s
CGN	3 Days	4.767e-3	5.469e-2	-7.6290	2m 41s
Random Forest	1 Day	1.311e-4	1.569e-3	0.7784	27s
Random Forest	2 Days	1.195e-4	1.568e-3	0.7890	15m
Random Forest	3 Days	7.603e-5	1.050e-3	0.8623	1h 34m
Ensemble	3 Days	7.870e-5	1.107e-3	0.8575	1h 35m
RF + K-Fold	3 Days	2.500e-5	5.500e-4	0.9500	4h 45m
PSO-NSFM (Chondrodima et.al)	30 Days	6251.598	58.978	0.786	

Table 2: Metrics (MSE, MAE, R²) of all models

6.6. SHAP Analysis and Results Discussion

We also implemented our final code SHapley Additive exPlanations (SHAP) to interpret the results of our Random Forest model, which was used for predicting the estimated time of arrival (ETA) in an urban transit system. SHAP values provide a unified measure of feature importance, allowing us to understand the contribution of each feature to the model's output. By analyzing the SHAP values, we can gain insights into how different variables impact the model's predictions, thus enhancing the interpretability of our machine learning approach. As the dataset is large, we conducted the analysis in a smaller sample (1000). The eight selected features chosen are unix time, stop sequence, hour of day, day of week, longitude, latitude, bearing, and speed. The selected features are not only theoretically grounded but have also been empirically validated in various studies. By implementing these features in the model, we leverage their established relevance in the literature to enhance the accuracy and robustness of our ETA predictions.

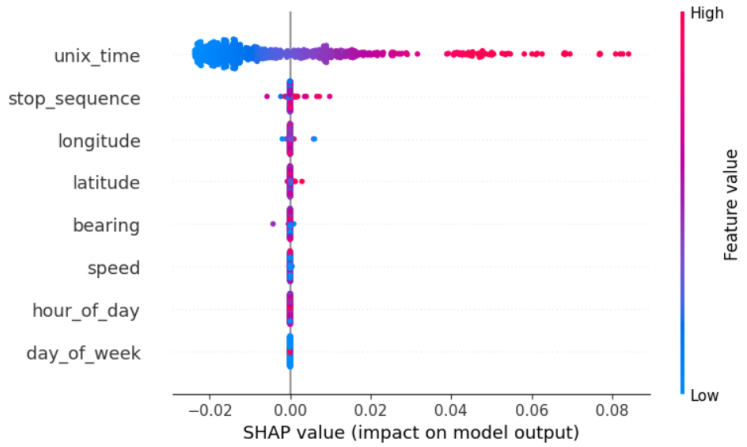

Figure 16: SHAP summary plot

The features are listed on the y-axis in descending order of importance, with the most impactful features at the top. The SHAP values on the x-axis show the impact each feature has on the model's output. Positive SHAP values push the prediction higher (to a later ETA), while negative SHAP values push the prediction lower (to an earlier ETA).

According to the SHAP analysis, the most important factor in determining when the bus will arrive at this stop is Unix time. Unix time, being the most important feature, suggests that the exact time (in seconds) has a significant impact on predicting the arrival time at the 6th stop. High Unix time values (red dots) seem to push the predictions toward positive SHAP values (increasing the ETA), while low Unix time values (blue dots) push predictions lower. This indicates that the model is capturing long-term temporal trends, such as how delays may build up over the course of a journey. The anticipated ETA tends to grow with high Unix time values (later in the route) and decreases with lower Unix time values (earlier in the journey), indicating that the precise point in time is important for the prediction. The second most significant aspect is the stop sequence, which specifies the order in which the stops occur. This suggests that delays tend to build up as the bus travels the route. Longitude and latitude also offer useful data, perhaps capturing geographic elements like traffic patterns in certain locations that can impact the bus's route duration.

Other features, such as bearing (direction) and speed, also contribute to the predictions but with slightly less impact, reflecting the bus's current movement and its proximity to the next stop. The hour of the day and day of the week have less influence but still provide insights into daily and weekly traffic patterns that might affect delays.

7. Practical Application and Next Steps

Considering the performance of our models, the Random Forest with k-fold Cross Validation could be the best model for practical deployment in urban transit systems. However, exploring more advanced ensemble methods beyond simple averaging could yield additional gains, aligning with techniques reported in other literature, such as stacking or meta-learning frameworks.

Apart from enhancing the evaluation process, exploring more advanced ensemble methods beyond simple averaging could also yield further performance improvements. Techniques such as stacking or meta-learning frameworks could dynamically combine multiple models' predictions, capturing different patterns in the data more effectively. These methods have been successfully applied in other studies, suggesting they could offer additional gains over the current ensemble approach.

8. Contributions to Literature

This study contributes to the literature on ETA prediction in urban transit systems by focusing on three critical aspects: (a) training times, (b) prediction accuracy, and (c) model stability and robustness. These dimensions are essential for evaluating the practical application of machine learning models in real-world urban transit environments.

Training Time: Across all models, Random Forest and the Simple Neural Network (NN) model demonstrated the fastest training times. The Simple NN trained in just 27 seconds for 1-day data, while Random Forest required 15 minutes for 2-day data and 1 hour and 34 minutes for 3-day data. These results highlight their suitability for environments requiring rapid model updates and deployment. The speed of these models stems from their relatively simple architectures and efficient computation processes. In contrast, models such as Long Short-Term Memory (LSTM) and Graph Convolutional Networks (GCNs) required significantly longer training times, particularly when working with larger datasets. The LSTM, while showing good performance, took longer to train with increased data, particularly when utilizing more extensive sequential patterns. GCNs, despite their potential to capture spatial relationships in transportation data, showed the highest training times without providing superior results, indicating a potential mismatch between model complexity and the dataset used.

Prediction Accuracy: In terms of accuracy, Random Forest consistently outperformed other models, achieving the lowest Mean Squared Error (MSE) and the highest R^2 score across datasets. For example, Random Forest reached an MSE of 7.603×10^{-5}) and an R^2 of 0.8623 when trained on 3-day data, highlighting its position as the most effective model for ETA prediction. LSTM and the Simple NN models also demonstrated strong predictive capabilities, especially when trained on larger datasets, with R^2 scores close to 0.8491 for the 3-day Simple NN model and 0.8494 for the 3-day LSTM model. These results suggest that these models can capture the underlying spatiotemporal patterns critical for accurate ETA predictions. In contrast, Graph Convolutional Networks struggled significantly, producing negative R^2 scores across datasets, which questions their suitability for this specific task without further tuning. XGBoost, while known for its efficiency in handling high-dimensional data, failed to achieve the high accuracy of Random Forest and LSTM, particularly with smaller datasets, though it maintained fast training times.

Stability and Robustness: Random Forest emerged as the most stable and robust model across datasets, consistently delivering reliable results with low error variance. Its ability to generalize well across different datasets indicates that Random Forest is less sensitive to variations in the data, making it ideal for practical deployment in urban transit systems. The Simple NN model, despite its computational efficiency, exhibited reduced performance when trained on smaller datasets, indicating that it benefits significantly from larger, more comprehensive data inputs. LSTM models, while effective, also showed slight performance dips with smaller datasets, which suggests that their strength lies in handling larger sequences of temporal data. On the other hand, GCNs and XGBoost exhibited greater variability, with GCNs struggling to provide meaningful predictions in this context, demonstrating their limitations in handling the dataset used in this study.

Conclusion: This study confirms the effectiveness of Random Forest as a robust, accurate, and efficient model for ETA prediction in urban transit networks, offering the best balance between training time, accuracy, and stability. While more complex models such as LSTM and GCNs show potential, their longer training times and reduced accuracy, especially with smaller datasets, may limit their practical application in real-time systems. The Simple NN model presents a viable option for faster deployment but benefits from larger datasets for better accuracy. XGBoost, though efficient in training time, underperformed in terms of accuracy, making it less suitable for this task compared to Random Forest. Overall, the findings suggest that while sophisticated models offer the potential to capture more complex patterns, simpler models like Random Forest provide the best trade-off between performance and computational efficiency in urban transit ETA prediction tasks.

9. Research Implications

This work addresses several real-world issues and suggests solid solutions, advancing the latest developments in urban ETA prediction problems. The main contributions include: (a) utilizing a large, real-time dataset to enhance the realism and applicability of the model predictions, (b) implementing advanced machine learning models such as Random Forest, XGBoost, and GCNs optimized for complex, high-dimensional data, and (c) conducting comparative experiments to evaluate model performance across different metrics, including speed and accuracy.

One significant practical implication of this study is the handling of a large-scale dataset. Given the real-time nature of the data, the dataset was extremely large, posing challenges in terms of computational resources and processing time. Due to these constraints, the analysis was conducted on a subset of the data spanning several days instead of a full month's worth of data. This decision was necessary to manage the available computational resources effectively and ensure timely completion of the study.

The data, originally stored in the PGAdmin database, was extracted into CSV format and then uploaded to Google Drive. This allowed seamless integration with Google Colab, where the data preprocessing, model training, and evaluation were performed. Google Colab's cloud-based computational resources were essential for managing the large dataset and conducting the analysis, which would have been unmanageable on standard local machines. This approach underscores the importance of scalable and flexible computing resources when dealing with big data in urban ETA prediction tasks.

Moreover, the size and complexity of the dataset required substantial computational power. While models such as LSTM required more extensive computational time due to their complexity, simpler models like Random Forest were quicker to train, demonstrating the trade-offs between model complexity and computational efficiency. This balance is crucial in real-world applications where both online and offline processing scenarios may be encountered, necessitating different levels of accuracy and speed.

Finally, as urban transit data continues to grow in scale, volume, and velocity, especially in modern, data-rich environments, it is crucial for proposed solutions to be robust, scalable, and adaptable. This study underscores the need for advanced methodologies and scalable infrastructure to effectively manage and analyze such large datasets, which is essential for the ongoing development of smart cities and intelligent transportation systems.

10. Conclusion

In conclusion, this study has provided a comprehensive comparison of various machine learning models for predicting Estimated Time of Arrival (ETA) in urban transit systems. By evaluating models across key factors such as training time, prediction accuracy, and stability, the results highlight the strengths and limitations of each approach within the context of real-world application.

The Random Forest model consistently emerged as the top performer, demonstrating a strong balance between training efficiency and high prediction accuracy. Its robustness across different datasets underscores its suitability for dynamic urban transit environments where consistent, reliable performance is crucial. The integration of K-fold cross-validation further validated Random Forest's performance, showing reduced variance and a significant increase in the R^2 score, reaching 0.9500 after cross-validation, which speaks to its generalizability and adaptability across varying data splits. While advanced models such as Long Short-Term Memory (LSTM) networks offered competitive accuracy, particularly with larger datasets, their longer training times may limit their practicality in real-time or resource-constrained systems. Similarly, while models like Graph Convolutional Networks (GCNs) show promise for capturing complex spatial relationships, they underperformed in this specific context, indicating the need for further refinement or alternative approaches for urban traffic prediction.

In contrast, simpler models like the Simple Neural Network (NN) excelled in terms of computational speed, offering a viable solution for quick deployments in situations where model updates are frequent. However, their performance was more reliant on the size of the training dataset, with accuracy decreasing when smaller datasets were used. XGBoost, though quick to train, failed to match the prediction performance of Random Forest and LSTM, indicating that its potential lies more in high-dimensional, tabular data rather than temporal-spatial data like those found in transit networks.

Furthermore, the SHAP analysis revealed that Unix time is the most significant factor in predicting bus arrival times, as it captures long-term temporal trends that affect delays. Stop sequence and geographic features, like longitude and latitude, also play crucial roles, indicating that both route progression and location-specific factors impact ETA. Other factors like speed, direction, and time of day contribute less but still provide meaningful insights into traffic patterns.

Overall, the study concludes that while more complex machine learning models can capture intricate patterns and dependencies, simpler models like Random Forest often provide the best trade-off between computational efficiency and prediction performance in ETA tasks. In future work, a combination of models or ensemble approaches may further enhance prediction accuracy by leveraging the strengths of different models, creating a more adaptable and accurate system for real-time ETA forecasting in urban transit networks. Additionally, further exploration of feature engineering and model tuning, particularly with graph-based models like GCNs, could lead to improved outcomes for tasks that involve both spatial and temporal complexity.

Future Enhancements

Looking ahead, several avenues for future research could further enhance the outcomes of this study. Incorporating more diverse datasets, including real-time traffic congestion levels, weather conditions, and localized events, could provide a richer context for model training, thereby improving predictive accuracy.

Additionally, exploring more sophisticated ensemble learning techniques, such as stacking and meta-learning frameworks, could combine the strengths of multiple models, capturing a wider range of patterns in the data. Leveraging advanced computational resources, including distributed computing and cloud-based platforms, would also enable more efficient processing of large-scale, real-time datasets.

By addressing these areas, future studies can develop even more accurate, scalable, and resilient ETA prediction models, contributing to the ongoing evolution of smart cities and enhancing the overall user experience in urban transit systems.

Appendix A: GTFS Datasets

The GTFS (General Transit Feed Specification) data exists in two main variants: the GTFS static (GTFS-s) feed and the GTFS real-time (GTFS-rt) feed.

A.1. GTFS Static (GTFS-s) Feed Files

The GTFS-s feed is typically a collection of comma-separated values (CSV) files containing relatively static information about a public transit (PT) network. This static information includes details that change infrequently, such as routes, stops, and schedules. Updates to these files may occur periodically, for example, when schedules are adjusted seasonally or when a new route is introduced.

The GTFS-s feed consists of several mandatory and optional files that together provide a comprehensive description of a PT network:

1. **agency.txt**: This file includes information about the transit agencies providing the data in the feed. Each transit agency is identified by an agency_id. It contains fields such as agency_name, agency_url, and agency_timezone, which describe the agency's name, website, and timezone, respectively.
2. **routes.txt**: Contains the routes within the PT network, each identified by a route_id. Routes represent a specific path taken by a PT vehicle, and this file includes fields such as route_short_name, route_long_name, and route_type to provide a more detailed description of each route.
3. **trips.txt**: This file lists all the trips that occur on the routes in the PT network, each identified by a trip_id. A trip describes the movement of a PT vehicle along a specific route at a certain time. This file also includes fields such as service_id, trip_headsign, and direction_id.
4. **stops.txt**: Details all the stops in the PT network, each identified by a stop_id. This file provides information such as stop_name, stop_lat, and stop_lon, which indicate the stop's name and geographic coordinates (latitude and longitude) based on the WGS 84 datum.
5. **stop_times.txt**: Provides detailed scheduling information by listing the times that each trip stops at each stop. Each entry is identified by the associated trip_id and stop_id and includes fields such as arrival_time, departure_time, and stop_sequence.
6. **calendar.txt** (Conditionally Required): Defines the days of service for the trips, each identified by a service_id. This file specifies a weekly schedule, with fields indicating whether the service operates on each day of the week.
7. **calendar_dates.txt** (Conditionally Required): Provides exceptions to the regular schedule defined in calendar.txt, also associated with a service_id. This file includes fields such as date and exception_type to indicate specific dates when service is added or removed.
8. **shapes.txt** (Optional): Defines the physical path that the vehicle travels for each trip, identified by a shape_id. This file contains a series of points defined by shape_pt_lat and shape_pt_lon that outline the route's shape on a map.
9. **feed_info.txt** (Optional): Contains metadata about the dataset itself, including information such as feed_publisher_name, feed_publisher_url, and feed_version.

These files, when combined, provide a complete set of static information for a PT network. In a PT network, **stops** represent specific locations where passengers can board or disembark from PT vehicles. A **route** is defined by a fixed sequence of stops, and a **trip** represents a single journey along a route, occurring at a specific time.

A.2. GTFS Real-Time (GTFS-rt) Feed

While the GTFS-s feed provides static information about the transit network, the GTFS-rt feed supplies dynamic, real-time data collected during vehicle trips. This data is typically based on GPS tracking and provides up-to-the-minute updates on the status of the transit network. The GTFS-rt feed is used to provide passengers with current information about vehicle positions, service alerts, and trip updates. The real-time data is encoded using the Protocol Buffers (protobuf) format, an open-source standard for efficiently serializing structured data.

The GTFS-rt feed supports three main types of information:

1. **Trip Updates**: Include predicted arrival and/or departure times for stops along each trip. This data helps in estimating real-time ETAs (Estimated Time of Arrival) for each stop a vehicle will serve.
2. **Vehicle Positions**: Provide updates on the current location of individual transit vehicles. The data includes the vehicle's geographic coordinates and additional information like bearing and speed, allowing for real-time tracking of vehicles.
3. **Service Alerts**: Include updates on disruptions in the transit network, such as delays, detours, or route closures. These alerts are provided as human-readable messages that help passengers understand the current state of the transit network and plan their journeys accordingly.

By integrating the GTFS-s and GTFS-rt feeds, transit agencies can provide a comprehensive view of their operations, combining scheduled information with real-time updates to deliver accurate and reliable transit data to passengers.

Below the links for the data stacks:

Static schedule data

https://svc.metrotransit.org/mtgtfs/gtfs.zip

GTFS-realtime data

- TripUpdate feed: https://svc.metrotransit.org/mtgtfs/tripupdates.pb
- VehiclePosition feed: https://svc.metrotransit.org/mtgtfs/vehiclepositions.pb
- ServiceAlerts feed: https://svc.metrotransit.org/mtgtfs/alerts.pb

Appendix B: Metro Transit

Metro Transit is the largest public transportation operator in Minnesota, serving the Minneapolis/St. Paul metropolitan area. The agency provides transit services, including buses, light rail, and commuter rail, and makes its transit data available in the General Transit Feed Specification (GTFS) format. This data can be accessed by the public through Metro Transit's official website (MetroTransit, 2021).

11. References

1. Al-Naim, R., & Lytkin, Y. (2021). Review and comparison of prediction algorithms for the estimated time of arrival using geospatial transportation data. *Procedia Computer Science*, *193*, 13–21. https://doi.org/10.1016/j.procs.2021.11.003

2. Balster, A., Hansen, O., Friedrich, H., & Wei, H. (2020). An ETA prediction model for intermodal transport networks based on machine learning. *Business & Information Systems Engineering*, 62(5), 403–416. https://doi.org/10.1007/s12599-020-00653-0

3. Barbour, W., Martinez Mori, J. C., Kuppa, S., & Work, D. B. (2018). Prediction of arrival times of freight traffic on US railroads using support vector regression. *Transportation Research Part C: Emerging Technologies*, 93, 211–227. https://doi.org/10.1016/j.trc.2018.05.019

4. Bayati, A., Asghari, V., Nguyen, K. K., & Cheriet, M. (2016). Gaussian Process Regression Based Traffic Modeling and Prediction in High-Speed Networks. *2016 IEEE Global Communications Conference (GLOBECOM)*, 1-7.https://doi.org/10.1109/glocom.2016.7841857

5. Chicco, D., Warrens, M. J., & Jurman, G. (2021). The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. PeerJ Computer Science, 7, e623. https://doi.org/10.7717/peerj-cs.623

6. Chondrodima, E., Georgiou, H., Pelekis, N., & Theodoridis, Y. (2022). Particle swarm optimization and RBF neural networks for public transport arrival time prediction using GTFS data. *International Journal of Information Management Data Insights*, 2(2), Article 100086. https://doi.org/10.1016/j.jjimei.2022.100086

7. Derrow-Pinion, A., She, J., Wong, D., Lange, O., Hester, T., Perez, L., Nunkesser, M., Lee, S., Guo, X., Wiltshire, B., Battaglia, P. W., Gupta, V., Li, A., Xu, Z., Sanchez-Gonzalez, A., Li, Y., & Velickovic, P. (2021). ETA Prediction with Graph Neural Networks in Google Maps. *In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21) (pp. 3767–3776*). Association for Computing Machinery. https://doi.org/10.1145/3459637.3481916

8. Duan, Y., Lv, Y., & Wang, F. (2016). Travel time prediction with LSTM neural network. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 1053-1058). https://doi.org/10.1109/ITSC.2016.7795686

9. Fleischman, R. J., Lundquist, M., Jui, J., Newgard, C. D., & Warden, C. (2013). *Predicting ambulance time of arrival to the emergency department using Global Positioning System and Google Maps*. Prehospital Emergency Care, 17(4), 458–465. https://doi.org/10.3109/10903127.2013.811562

10. Fu, K., Meng, F., Ye, J., & Wang, Z. (2020). CompactETA: A Fast Inference System for Travel Time Prediction. *In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20) (pp. 3337–3345)*. Association for Computing Machinery. https://doi.org/10.1145/3394486.3403386

11. Vidya, G. S., & Hari, V. S. (2023). Prediction of Bus Passenger Traffic using Gaussian *Process Regression. Journal of Signal Processing Systems*, 95(2-3), 281–292. https://doi.org/10.1007/s11265-022-01774-3

12. Han, J., Liu, H., Liu, S., Chen, X., Tan, N., Chai, H., & Xiong, H. (2023). iETA: A Robust and Scalable Incremental Learning Framework for Time-of-Arrival Estimation. *Proceedings of the 2023 ACM Conference*, 4100-4111. https://doi.org/10.1145/3580305.3599842

13. Hu, J., Li, X., & Ou, Y. (2015). Online Gaussian process regression for time-varying manufacturing systems. *In Proceedings of the 13th International Conference on Control Automation Robotics and Vision (ICARCV 2014) (pp. 1118-1123)*. IEEE. https://doi.org/10.1109/ICARCV.2014.7064462.

14. Hu, W., Yao, Z., Yang, S., Chen, S., & Jin, P. J. (2019). Discovering urban travel demands through dynamic zone correlation in location-based social networks. In F. Bonchi, M. Berlingerio, T. Gärtner, N. Hurley, & G. Ifrim (Eds.), *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Proceedings (pp. 88-104). (Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 11052 LNAI)*. Springer Verlag. https://doi.org/10.1007/978-3-030-10928-8_6

15. Huang, J., Huang, Z., Fang, X., Feng, S., Chen, X., Liu, J., Yuan, H., & Wang, H. (2022). DuETA: Traffic Congestion Propagation Pattern Modeling via Efficient Graph Learning for ETA Prediction at Baidu Maps. *Proceedings of the 31st ACM International Conference on Information &Amp; Knowledge Management*. https://doi.org/10.1145/3511808.3557091

16. Huang, Z., Qi, H., Kang, C., Su, Y., & Liu, Y. (2020). *An Ensemble Learning Approach for Urban Land Use Mapping Based on Remote Sensing Imagery and Social Sensing Data. Remote Sensing*, 12(19), 3254. https://doi.org/10.3390/rs12193254

17. Kang, L., Hu, G., Huang, H., Lu, W., & Liu, L. (2020). Urban Traffic Travel Time Short-Term Prediction Model Based on Spatio-Temporal Feature Extraction. *Journal of Advanced Transportation*, 2020, 1–16. https://doi.org/10.1155/2020/3247847

18. Kelley, K., & Lai, K. (2011). *Accuracy in parameter estimation for the root mean square error of approximation: Sample size planning for narrow confidence intervals. Multivariate Behavioral Research*, 46(1), 1–32. https://doi.org/10.1080/00273171.2011.543027

19. Khazukov, K., Shepelev, V., Karpeta, T., Shabiev, S., Slobodin, I., Charbadze, I., & Alferova, I. (2020b). Real-time monitoring of traffic parameters. *Journal of Big Data*, 7(1). https://doi.org/10.1186/s40537-020-00358-x

20. Liu, X., He, J., Yao, Y., Zhang, J., Liang, H., Wang, H., & Hong, Y. (2017). Classifying urban land use by integrating remote sensing and social media data. *International Journal of Geographical Information Science*, 31(8), 1675–1696. https://doi.org/10.1080/13658816.2017.1324976

21. Mamudur, K., & Kattamuri, M. R. (2020). Application of boosting-based ensemble learning method for the prediction of compression index. *Journal of The Institution of Engineers (India): Series A*, 101, 409–419. https://doi.org/10.1007/s40030-020-00443-7

22. Mehta, H., Kanani, P. and Lande, P. (2019) Google Maps. *International Journal of Computer Applications*, 178, 41-46. https://doi.org/10.5120/ijca2019918791

23. Melnikov, V.R., Krzhizhanovskaya, V.V., Boukhanovsky, A., & Sloot, P.M. (2015). Data-driven modeling of transportation systems and traffic data analysis during a major power outage in the Netherlands. *Procedia Computer Science, 66*, 336-345. https://doi.org/10.1016/j.procs.2015.11.039

24. Paliwal, C., & Biyani, P. (2019). To each route its own ETA: A generative modeling framework for ETA prediction. *In 2019 IEEE Intelligent Transportation Systems Conference (ITSC) (pp. 3076–3081).* https://doi.org/10.1109/ITSC.2019.8917465

25. Paliwal, C., Bhatt, U., Biyani, P., & Rajawat, K. (2022). Traffic Estimation and Prediction via Online Variational Bayesian Subspace Filtering. IEEE Transactions on Intelligent Transportation Systems, 23(5), 4674–4684. https://doi.org/10.1109/TITS.2020.3048959

26. Qiao, S., Han, N., Zhu, W., & Gutierrez, L. (2015). TraPlan: An Effective Three-in-One Trajectory-Prediction Model in Transportation Networks. *IEEE Transactions on Intelligent Transportation Systems*, 16, 1188-1198. https://doi.org/10.1109/TITS.2014.2353302

27. Reich, T., Budka, M., Robbins, D., & Hulbert, D. (2019). *Survey of ETA prediction methods in public transport networks. arXiv preprint arXiv:1904.05037*. https://arxiv.org/abs/1904.05037

28. Schleibaum, S., Müller, J., & Sester, M. (2022). An Explainable Stacked Ensemble Model for Static Route-Free Estimation of Time of Arrival. arXiv preprint arXiv:2203.09438. https://arxiv.org/abs/2203.09438

29. Seow, K. T., Dang, N., & Lee, D.-H. (2010). A Collaborative Multiagent Taxi-Dispatch System. *IEEE Transactions on Automation Science and Engineering*, 7, 607-616. https://doi.org/10.1109/TASE.2009.2028577

30. Steenbruggen, J., Borzacchiello, M. T., Nijkamp, P., & Scholten, H. (2011). Mobile phone data from GSM networks for traffic parameter and urban spatial pattern assessment: a review of applications and opportunities. *GeoJournal*, 78(2), 223–243. https://doi.org/10.1007/s10708-011-9413-y

31. Taiwo, E., Ogunsanwo, G., Alaba, O., & Ogunbanwo, A. (2023). Traffic Congestion Prediction using Supervised Machine Learning Algorithms. *TASUED Journal of Pure and Applied Sciences*, 2(1), 110–116. https://journals.tasued.edu.ng/index.php/tjopas/article/view/12

32. Technode. (2023, July 18). Chinese travel booking site Ctrip unveils AI model offering tourism tips. https://technode.com/2023/07/18/chinese-travel-booking-site-ctrip-unveils-ai-model-offering-tourism-tips/

33. Tseng, F.-H., Hsueh, J.-H., Tseng, C.-W., Yang, Y.-T., Chao, H.-C., & Chou, L.-D. (2018). Congestion prediction with big data for real-time highway traffic. *IEEE Access*, 6, 57311–57323. https://doi.org/10.1109/ACCESS.2018.2873569

34. Tsolaki, K., Vafeiadis, T., Nizamis, A., Ioannidis, D., & Tzovaras, D. (2022). Utilizing machine learning on freight transportation and logistics applications: A review. *ICT Express*, 9. https://doi.org/10.1016/j.icte.2022.02.001

35. Wang, J., Gu, Q., Wu, J., Liu, G., & Xiong, Z. (2016). Traffic Speed Prediction and Congestion Source Exploration: A Deep Learning Method. *In Proceedings of the IEEE International Conference on Data Mining (ICDM 2016).* https://doi.org/10.1109/ICDM.2016.0061

36. Wang, T., Ni, S., Qin, T., & Cao, D. (2022). TransGAT: A Dynamic Graph Attention Residual Networks for Traffic Flow Forecasting. Sustainable Computing: Informatics and Systems, 36*, Article 100779*. https://doi.org/10.1016/j.suscom.2022.100779

37. Wesolowski, A., Eagle, N., Tatem, A. J., Smith, D. L., Noor, A. M., Snow, R. W., & Buckee, C. O. (2014). Quantifying travel behavior for infectious disease research: A comparison of data from surveys and mobile phones. *Scientific Reports*, 4(1), 5678. https://doi.org/10.1038/srep05678

38. Woodard, D., Nogin, G., Koch, P., Racz, D., Goldszmidt, M., & Horvitz, E. (2017). Predicting travel time reliability using mobile phone GPS data. *Transportation Research Part C: Emerging Technologies*, 75, 30-44. https://doi.org/10.1016/j.trc.2016.10.011

39. Wu, C.-H., Ho, J.-M., & Lee, D. T. (2004). *Travel-time prediction with support vector regression. IEEE Transactions on Intelligent Transportation Systems*, 5(4), 276–281. https://doi.org/10.1109/TITS.2004.837813

40. Yan, C., Johndrow, J., Woodard, D., & Sun, Y. (2023). Efficiency of ETA Prediction. *arXiv preprint arXiv:2112.09993v3*. https://doi.org/10.48550/arXiv.2112.09993

41. ang, B., Guo, C., & Jensen, C. S. (2013). Travel cost inference from sparse, spatio-temporally correlated time series using Markov models. *In Proceedings of the VLDB Endowment, 6*(9), 769-780. https://doi.org/10.48550/arXiv.2112.09993

42. Zafar, N., & Ul Haq, I. (2020). Traffic congestion prediction based on estimated time of arrival. *PLOS ONE*, 15(12), e0238200. https://doi.org/10.1371/journal.pone.0238200

43. Zafar, N., Ul Haq, I., Sohail, H., Chughtai, J.-U.-R., & Muneeb, M. (2022). Traffic prediction in smart cities based on hybrid feature space. *IEEE Access*, 10, 134333–134348. https://doi.org/10.1109/ACCESS.2022.3231448

44. Zhang, S., Li, X., Zong, M., Zhu, X., & Cheng, D. (2017). Learning k for kNN classification. ACM Transactions on Intelligent Systems and Technology, 8(3), *Article 43*. https://doi.org/10.1145/2990508

45. Zhang, Y., Li, Y., Zhou, X., Kong, X., & Luo, J. (2020). Curb-GAN: Conditional Urban Traffic Estimation through Spatio-Temporal Generative Adversarial Networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20) (pp. 842–852). *Association for Computing Machinery*. https://doi.org/10.1145/3394486.3403127

46. Zhao, B., Teo, Y., Ng, W. S., & Ng, H. (2019). Data-Driven Next Destination Prediction and ETA Improvement for Urban Delivery Fleets. *IET Intelligent Transport Systems*, 13. https://doi.org/10.1049/iet-its.2019.0148