Multimodal machine learning methods for pattern analysis in smart cities and transportation

Ifigeneia Drosouli

UNIVERSITE DE LIMOGES

ECOLE DOCTORALE SCIENCE - TECHNOLOGIE

FACULTE DES SCIENCES & TECHNIQUES

Laboratoire XLIM

Thèse de Doctorat

Thèse

pour obtenir le grade de

DOCTEUR DE L' UNIVERSITÉ DE LIMOGES

Discipline / Spécialité : Informatique

présentée et soutenue par

Ifigenia DROSOULI

le [06 2024]

Multimodal Machine Learning methods for pattern analysis in smart cities and transportation

Thèse dirigée par le Professeur Athanasios VOULODIMOS

et le Professeur Djamchid GHAZANFARPOUR

Président du jury: Professeur Nikolaos VASILAS

Université de Limoges

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
UNIVERSITY OF WEST ATTICA

Acknowledgements

I want to express my gratitude to my supervisor, Assistant Professor Athanasios Voulodimos, for dedicating his valuable time and providing unwavering support during my years of PhD and the entire research and writing process for this thesis. His assistance proved indispensable and significantly contributed to the successful completion of my project.

I am also thankful to my supervisor, Professor Djamchid Ghazanfarpour for his essential support and guidance over these years.

Special appreciation goes to Professor George Miaoulis for his consistent support, trust, and guidance, as well as his belief in my capabilities.

I must extend my sincere thanks to Vanessa A. for being a profound inspiration, motivating me, and encouraging me to embark on and complete my PhD journey. Her belief in me has been a driving force throughout these years.

Additionally, I want to convey my special thanks and love to my best friend and colleague, Georgia, for her invaluable assistance, patience, and positive attitude.

Lastly, I want to express my heartfelt thanks to my loving family—Pantelis, Anna, and Vasilis—for their constant moral and practical support throughout my PhD and life in general. This achievement would not have been possible without their unwavering encouragement.

Abstract

In the context of modern, densely populated urban environments, the effective management of transportation and the structure of Intelligent Transportation Systems (ITSs) are paramount. The public transportation sector is currently undergoing a significant expansion and transformation with the objective of enhancing accessibility, accommodating larger passenger volumes without compromising travel quality, and embracing environmentally conscious and sustainable practices. Technological advancements, particularly in Artificial Intelligence (AI), Big Data Analytics (BDA), and Advanced Sensors (AS), have played a pivotal role in achieving these goals and contributing to the development, enhancement, and expansion of Intelligent Transportation Systems.

This thesis addresses two critical challenges within the realm of smart cities, specifically focusing on the identification of transportation modes utilized by citizens at any given moment and the estimation and prediction of transportation flow within diverse transportation systems.

In the context of the first challenge, two distinct approaches have been developed for Transportation Mode Detection. Firstly, a deep learning approach for the identification of eight transportation media is proposed, utilizing multimodal sensor data collected from user smartphones. This approach is based on a Long Short-Term Memory (LSTM) network and Bayesian optimization of model's parameters. Through extensive experimental evaluation, the proposed approach demonstrates remarkably high recognition rates compared to a variety of machine learning approaches, including state-of-the-art methods. The thesis also delves into issues related to feature correlation and the impact of dimensionality reduction.

The second approach involves a transformer-based model for transportation mode detection named TMD-BERT. This model processes the entire sequence of data, comprehends the importance of each part of the input sequence, and assigns weights accordingly using attention mechanisms to grasp global dependencies in the sequence. Experimental evaluations showcase the model's exceptional performance compared to state-of-the-art methods, highlighting its high prediction accuracy.

In addressing the challenge of transportation flow estimation, a Spatial-Temporal Graph Convolutional Recurrent Network is proposed. This network learns from both the spatial stations network data and time-series of historical mobility changes to predict urban metro and bike sharing flow at a future time. The model combines Graph Convolutional Networks (GCN) and Long Short-Term Memory (LSTM) Networks to enhance estimation accuracy. Extensive experiments conducted on real-world datasets from the Hangzhou metro system and the NY City bike sharing system validate the effectiveness of the proposed model, showcasing its ability to identify dynamic spatial correlations between stations and make accurate long-term forecasts.

Résumé

Dans le contexte des environnements urbains modernes et densément peuplés, la gestion efficace des transports et la structure des Systèmes de Transport Intelligents (STI) sont primordiales. Le secteur des transports publics connaît actuellement une expansion et une transformation significatives dans le but d'améliorer l'accessibilité, d'accommoder des volumes de passagers plus importants sans compromettre la qualité des déplacements, et d'adopter des pratiques respectueuses de l'environnement et durables. Les avancées technologiques, notamment dans l'Intelligence Artificielle (IA), l'Analyse de Données Massives (BDA), et les Capteurs Avancés (CA), ont joué un rôle essentiel dans la réalisation de ces objectifs et ont contribué au développement, à l'amélioration et à l'expansion des Systèmes de Transport Intelligents.

Cette thèse aborde deux défis critiques dans le domaine des villes intelligentes, se concentrant spécifiquement sur l'identification des modes de transport utilisés par les citoyens à un moment donné et sur l'estimation et la prédiction du flux de transport au sein de divers systèmes de transport.

Dans le contexte du premier défi, deux approches distinctes ont été développées pour la Détection des Modes de Transport. Tout d'abord, une approche d'apprentissage approfondi pour l'identification de huit médias de transport est proposée, utilisant des données de capteurs multimodaux collectées à partir des smartphones des utilisateurs. Cette approche est basée sur un réseau Long Short-Term Memory (LSTM) et une optimisation bayésienne des paramètres du modèle. À travers une évaluation expérimentale approfondie, l'approche proposée démontre des taux de reconnaissance remarquablement élevés par rapport à diverses approches d'apprentissage automatique, y compris des méthodes de pointe. La thèse aborde également des problèmes liés à la corrélation des caractéristiques et à l'impact de la réduction de la dimensionnalité.

La deuxième approche implique un modèle basé sur un transformateur pour la détection des modes de transport appelé TMD-BERT. Ce modèle traite l'ensemble de la séquence de données, comprend l'importance de chaque partie de la séquence d'entrée, et attribue des poids en conséquence en utilisant des mécanismes d'attention pour saisir les dépendances globales dans la séquence. Les évaluations expérimentales mettent en évidence les performances exceptionnelles du modèle par rapport aux méthodes de pointe, soulignant sa haute précision de prédiction.

Pour relever le défi de l'estimation du flux de transport, un Réseau Convolutif Temporel et Spatial (ST-GCN) est proposé. Ce réseau apprend à la fois des données spatiales du réseau de stations et des séries temporelles des changements de mobilité historiques pour prédire le flux de métro urbain et le partage de vélos à un moment futur. Le modèle combine des Réseaux Convolutifs Graphiques (GCN) et des Réseaux Long Short-Term Memory (LSTM) pour améliorer la précision de l'estimation. Des expériences approfondies menées sur des ensembles de données du monde réel du système de métro de Hangzhou et du système de partage de vélos de la ville de New York valident l'efficacité du modèle proposé, démontrant sa capacité à identifier des corrélations spatiales dynamiques entre les stations et à faire des prévisions précises à long terme.

Contents

List of Figures

List of Tables

Glossary of Acronyms

AI - Artificial Intelligence

AS - Advanced Sensors

BDA - Big Data Analytics

BERT - Bidirectional Encoder Representations from Transformers

Bi-LSTM - Bidirectional Long Short-Term Memory

CART - Classification and Regression Tree

CKS - Cohen Kappa Score

CNN - Convolutional Neural Networks

DL - Deep Learning

DT - Decision Trees

FFNN - Feed-Forward Neural Networks

IoT - Internet of Things

ITS - Intelligent Transportation Systems

KDE - Kernel Density Estimation

KNN - K-Nearest Neighbors

LDA - Linear Discriminant Analysis

LR - Logistic Regression

LSTM - Long Short-Term Memory

MAE - Average Absolute Error

MAPE - Mean Absolute Percentage Error

MCC - Matthews's correlation coefficient

ML - Machine Learning

NB - Naive Bayes

NFC - Near Field Communication

NLP - Natural Language Processing

PC - Principal Components

PCA - Principal Component Analysis

RF - Random Forest

RMSE - Root Mean Square Error

RNN - Recurrent Neural Networks

SHL - Sussex-Huawei Locomotion

ST-GCRN - Spatial-Temporal Graph Convolutional Recurrent Network

SVM - Support Vector Machine

TFE - Transportation Flow Estimation

TMD - Transportation Mode Detection

WSN - Wireless Sensor Networks

Chapter 1

Introduction

1.1 General Introduction

The increasing urbanization and population in recent years have made transportation a crucial and major element that significantly influences the quality of life for individuals residing in large cities. Transportation impacts various aspects of urban life and is vital for the functioning and development of big cities [1].

Mobility and Accessibility are crucial in urban life [2], providing essential means for people to move around the city and facilitating access to employment, education, healthcare, and other services [3]. Moreover, efficient transportation systems are fundamental for economic growth, supporting the movement of goods and services and connecting businesses with suppliers, customers, and markets, thereby fostering economic development [4]. Additionally, well-designed transportation networks enhance productivity by reducing travel time and congestion, allowing timely access to workplaces and smoothing business operations [5]. Reliable transportation positively impacts also quality of life for city residents [6], providing the freedom to participate in activities and improving overall livability [7]. Moreover, addressing environmental sustainability involves developing sustainable transportation systems [8], reducing the carbon footprint, and enhancing air quality [9], whereas congestion management is critical in urban planning, and safety measures are essential for reducing accidents and injuries [10]. Lastly, social equity is promoted through accessible and affordable transportation, ensuring equal opportunities for all residents [11].

The insights obtained by studying human mobility patterns in various transportation environments can greatly assist in effectively addressing these concerns. Citizens' transportation choices involve multiple modes of transportation that evolve over time based on users' requirements [12].

Two critical issues of modern transportation management and Intelligent Transportation Systems (ITSs) structure arise from this: transportation mode recognition and passengers flow estimation [13]. They both contribute to the planning of travel routes, the evaluation of travel demand, the effective and efficient operation of public transport and consequently, to the efficient operation of the whole city and the alleviation of transportation-related problems [14].

In contemporary densely populated cities, the public transportation industry is undergoing expansion and transformation. This sector encompasses not only traditional modes of transport such as road vehicles and underground railways but also newer forms of transportation that have emerged in recent years, such as bike-

sharing systems, ride-hailing services and e-scooters [15]. The aim is to make public transport more accessible, capable of accommodating larger passenger volumes without compromising on travel quality and to adopt a more environmentally conscious and sustainable approach.

Technological advancements have played a significant role in attaining this objective. Artificial Intelligence (AI), along with Big Data Analytics (BDA) and Advanced Sensors (AS), has played a crucial role in the development, advancement, and expansion of Intelligent Transportation Systems (ITS) [16]. These technologies, among others like Internet of Things, Global Positioning System, Mobile Connectivity, and Cloud Computing, have collectively contributed to the growth of modern transportation systems.

AI technologies, such as Machine Learning (ML) and Deep Learning (DL), are used in ITS to analyze large amounts of data, predict traffic patterns, optimize routes, and make decisions for traffic flow [17]. Furthermore, the ability to collect and analyze vast amounts of data from various sources, such as sensors, cameras, and social media, has greatly influenced intelligent transportation. Big data analytics helps in detecting traffic patterns, predicting congestion, and optimizing transportation systems [18]. Likewise, sensors such as radar, LIDAR [19], and cameras are used to collect real-time data on traffic flow, vehicle speed and other parameters. All this data is crucial for various applications such as traffic and passengers flow management, incident detection and adaptive signal control systems. Connected vehicles, autonomous vehicles, intelligent traffic management, smart infrastructure such as smart traffic signals or smart parking, are some notable areas of progress in IT [20].

Mobile devices have undergone significant advancements [21], becoming powerful tools capable of multitasking, running demanding apps, and performing complex tasks. Storage capacity has also increased, allowing for the storage of large amounts of data. Connectivity options like Wi-Fi, Bluetooth, Near Field Communication (NFC), and 5G enable seamless communication and fast data transfer. Additionally, the inclusion of in-built sensors in mobile phones enhances user experience by providing various functionalities. Sensors like GPS, accelerometer and gyroscope gather information about user movements and interactions with transportation modes [22]. This data can be used to analyze travel patterns, understand route preferences, estimate traffic congestion, and identify transportation bottlenecks, making the user experience more immersive and personalized [23].

The utilization of mobile phones and their in-built sensors in the field of transportation, enables the collection of extensive data and can lead in a more comprehensive and precise understanding of transportation behavior.

1.2 Thesis objective

The objective of this thesis is to tackle challenges within smart cities concerning the identification of transportation modes used by citizens at any given moment and to estimate and predict the flow of transportation within diverse transportation systems.

To address these challenges, the ongoing research focuses on leveraging diverse data sources, including information from mobile phone sensors, metro card swiping data, and transaction records from a bike-sharing system which encompass a range of technologies such as GPS trackers and other sensors, mobile applications, and RFID systems, as well as sensors integrated into docking stations.

This thesis objective is:

- To create a resilient and optimized Transportation Mode Detection (TMD) system utilizing sequential data from multiple smartphone sensors, exhibiting improved detection accuracy compared to existing methods.
- To develop a Spatial-Temporal Graph Convolutional Recurrent Network for transportation flow estimation. This method can identify dynamic spatial correlations between stations and is capable of long-term forecasting.

1.3 Contributions

In our current study, we aim to address two significant challenges in the field of transportation: Transportation Mode Detection (TMD) and Transportation Flow Estimation (TFE). While various approaches have been explored in this domain, several major challenges still exist:

Diverse Sensors: Numerous methods for detecting transportation modes and estimating transportation flow rely on data collected from a diverse combination of sensors, including cameras [24], sound [25], smart cards swiping [26,27], ultrasonic sensors [28,29] and ticketing systems [30]. This data fusion results in intricate high-dimensional feature sets encompassing diverse data types, demanding distinct methods for analysis, processing, and evaluation. Utilizing such data leads to intricate models that demand significant computational resources. High-dimensional data can make it difficult to find meaningful patterns, require substantial computational resources, and lead to overfitting if not properly handled.

Multimodal Transportation: In a bustling contemporary city, where a dynamic urban environment occurs, commuters have a plethora of transportation choices, each with unique traits and specific features. Furthermore, these preferences and modes of transport are continually evolving as users exhibit diverse and unpredictable behaviors during transportation. To accurately identify the mode used by commuters on each occasion and estimate transportation flow, it is crucial to consider these variations in traffic variability, movement patterns, speeds, and activity sequences. Variability in user behavior can make it challenging to establish

universal models that accurately represent all users, requiring adaptive algorithms capable of handling diverse behaviors.

Data Quality: Reliable data is essential for accurate predictions. Incomplete, outdated, or biased data can lead to inaccurate forecasts. Sensors, especially those in wearable devices, can introduce noise and inaccuracies in the collected data due to calibration errors, device limitations, or environmental interferences. Noisy sensor data with errors can lead to misclassification and reduced detection accuracy.

Long-Term Estimation: To enhance the accuracy of estimations and ensure their practical applicability, it is crucial to forecast not only in the short term but also in the long term. As the duration of the estimation period increases, the impact of uncertain factors results in a decrease in the accuracy of predictions. Additionally, the dynamic variability of transportation flow further elevates the uncertainty of estimations. Generally, long-range predictions are more demanding than short-range ones, but their practical significance is greater. Thus, it is a challenge to attain a long-term estimation of transportation flow.

Changeable station networks and mobility patterns: When the stations network does not have a fixed structure but its structure is dynamic, the relationship between the stations also changes. The spatial dependencies depend not only on the physical connections of stations, but on the dynamics of the system i.e., the mobility of flow-makers (passengers or bikes) which also depends on various external factors (weather, peak hours, personal choices etc.). Therefore, it's a challenge to capture and take into consideration the dynamic spatial dependency relations between stations in order to make an accurate estimation of transportation flow.

Based on the aforementioned challenges, the research questions that arise and need to be addressed are as follows:

1. How can innovative methods are devised to enhance transportation mode detection and flow estimation, balancing optimal model performance with considerations of accuracy, efficiency, computational resources, and the management of high-dimensional data?
2. What methods can effectively clean and preprocess sensor data from wearable devices to mitigate errors and calibration inaccuracies, ensuring the reliability and accuracy of transportation mode detection?
3. What strategies and forecasting methods can be employed to improve the accuracy of long-term transportation flow predictions, considering the impact of dynamic flow variability over extended periods?
4. How can dynamic spatial dependency relations between stations, influenced by changing station networks and mobility patterns, be effectively captured and incorporated into transportation flow estimation models?

The attempt to address these research questions led to the following contributions:

- We have developed novel techniques to enhance Transportation Mode Detection (TMD) by establishing a robust and efficient TMD system (TMD-Bert model) that enhances detection accuracy compared to current methods (greater than 99.7% accuracy) and that shows exceptional performance in all classes.
  In Transportation Flow Estimation (TFE), we have managed to optimize model (ST-GCRN model) performance while ensuring superior accuracy (error decrease by 98% in the metro system and 63% in the bike sharing system compared to the current state-of-the-art baselines). The proposed methods have been validated on real-world data and compared with a wide range of machine and deep learning techniques. These methods have the potential to redefine the analysis and utilization of transportation data in intricate urban environments.

- We have handled initial sensor (in TMD) and smart cards swiping data (in TFE) effectively by cleaning and preprocessing them (imputation, data normalization, feature extraction), attempting to address errors, missing data, and high-dimensionality issues. These techniques aim to enhance the reliability and accuracy of transportation mode detection and transportation flow estimation, ensuring the availability of high-quality data for analysis.

- We have achieved back-long-term transportation mode detection (by one or more prior times) and transportation flow estimation for different estimation horizons (30 minutes for Metro and 60 minutes for Bike sharing system) with minimized error. This method accounts for the dynamic variability of flow over extended periods, leading to more accurate and reliable predictions. This advancement proves valuable for long-term urban planning and the development of transportation infrastructure.

- We have captured dynamic spatial dependency relations between stations, influenced by evolving station networks and mobility patterns (two types of transportation flow datasets- Hangzhou metro railway system and the NY bike sharing system- with different network structures and changing spatial connections between stations). These models offer valuable insights into the changing relationships among stations in urban environments, facilitating precise transportation flow estimations. This contribution is essential for adapting transportation systems to the dynamic nature of urban landscapes.

Overall, these contributions would significantly advance the field of transportation analysis, providing practical and innovative solutions to complex challenges faced in urban planning, mobility management, and data-driven decision-making processes.

1.4 Thesis Organization

The thesis is organized in two parts. Part I deals with Transportation Mode Detection (TMD) and Part II addresses Transportation Flow Estimation.

Part I comprises four chapters. Chapter 2 delves into the Transportation Mode Detection (TMD) problem, exploring the motivations behind this research. Additionally, it provides an in-depth presentation of related works in the field, an analysis of sensors and their limitations, and an overview of publicly available common datasets.

Chapter 3 addresses data preprocessing, concentrating primarily on the issues and methods examined and utilized in this research including Dimensionality reduction which aims at streamlining and optimizing data utilization efficiency. In Chapters 4 and 5, two deep learning models based on LSTM and BERT-Transformers are respectively presented. The dataset used, its analysis, and the thorough description of experiments results, along with their evaluation, is provided.

Part II comprises two chapters. Chapter 6 addresses the transportation flow forecasting problem, discussing related works in the field and introducing the two datasets of metro and bike-sharing bikes used in the research experiments. In Chapter 7, Graphic Neural Networks are introduced in general, and the proposed framework is extensively described, along with the implemented experiments and their corresponding results. Finally, in Chapter 8, the limitations, overall conclusions and future challenges in the field are presented.

Part I: Transportation mode detection

Chapter 2

Transportation mode detection: Background

2.1 Motivation

Transportation Mode Detection (TMD) refers to the process of identifying and classifying the specific mode of transportation that an individual is using, based on data collected from various sensors on their smartphone or other wearable devices. It can be categorized as an activity recognition task. In this task, TMD algorithms utilize sensor data from GPS, accelerometer, gyroscope, and magnetometer readings to determine the particular transportation mode being used by individuals. This can include activities such as walking, running, driving, taking public transportation (bus, train, subway), or cycling. By analyzing the sensor data, TMD algorithms can accurately identify and classify the specific transportation mode employed by individuals [31,32,33].

The motivation for transportation mode detection (TMD), as illustrated in Figure 2.1, stems from several important reasons. Transportation Mode Detection (TMD) holds immense significance in various domains, profoundly impacting urban planning, transportation efficiency, environmental sustainability, health and safety, personalized services, and more.

In the realm of urban planning and infrastructure development, TMD offers invaluable insights into the transportation modes preferred by individuals within a specific area. This knowledge is pivotal for authorities as it enables optimization of resources and facilitates improvements in transportation systems. By accurately identifying transportation modes, TMD aids in analyzing traffic patterns, pinpointing congestion areas, and implementing strategic measures to alleviate traffic issues. Consequently, this leads to reduced travel times, enhanced traffic flow, and an overall improvement in transportation efficiency.

Beyond its impact on efficiency, TMD plays a vital role in assessing the environmental implications of different transportation modes. By discerning the usage of sustainable options like walking, cycling, and public transportation, TMD contributes to the reduction of carbon emissions and promotes eco-friendly transportation alternatives. Moreover, TMD has direct implications for public health and safety. Through the detection of transportation modes, it becomes possible to analyze pedestrian and cyclist behavior, identify accident-prone areas, and implement targeted safety measures. Additionally, TMD supports the promotion of active transportation modes, such as walking and cycling, fostering personal well-being and health [34].

On a more analytical level, TMD revolutionizes data collection for urban transportation planning. Traditionally, gathering such data was costly, error-prone, and limited in scope [35]. However, with smartphones, TMD facilitates the collection of vast and real-time data, offering accurate and up-to-date information essential for effective urban transportation planning. Furthermore, TMD enables targeted advertisements, enhancing customer experience and benefiting advertisers by optimizing their campaigns. Context-aware applications, empowered by TMD, deliver valuable real-time information to users, aiding in effective journey planning and behavior adjustment.

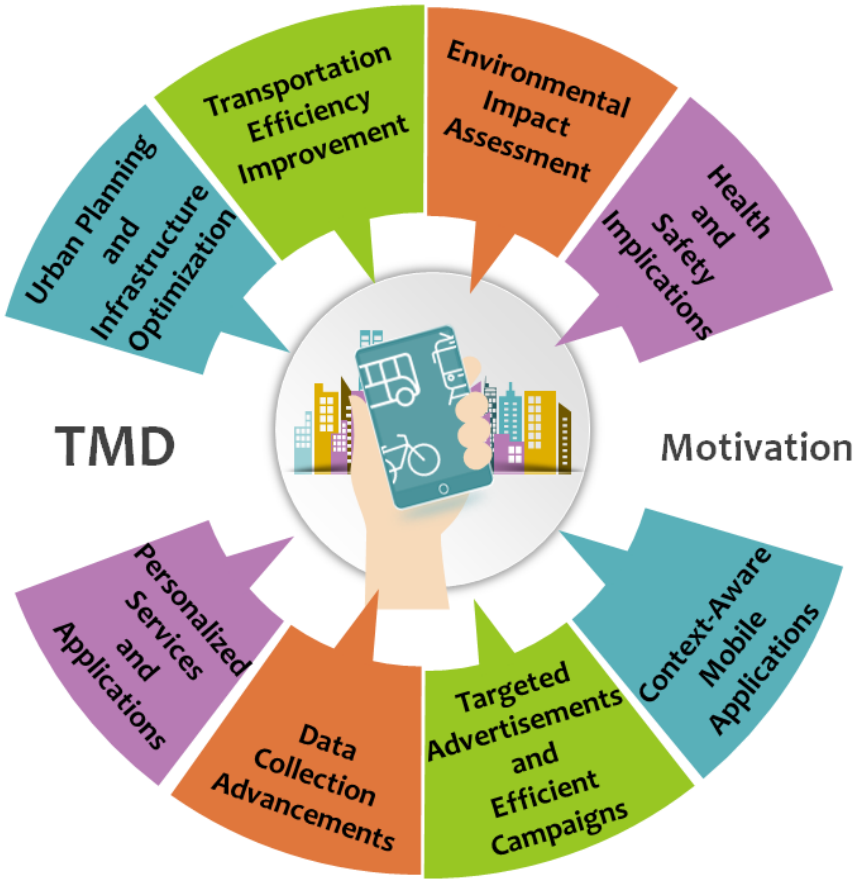


Figure 2.1 Motivation for Transportation Mode Detection.

In the realm of urban planning and traffic management, TMD's insights into daily transportation modes, prove invaluable. This knowledge informs the evaluation of current urban planning effectiveness and estimates traffic patterns, enabling governments to enhance routes, schedules, and public transportation services. Moreover, TMD supports emergency services by swiftly detecting accidents and facilitating timely responses.

TMD doesn't just stop at infrastructure planning; it profoundly influences physical and mental health. Utilizing smartphone sensor data, TMD assesses health status and provides personalized advice. It monitors physical activity, aiding in managing health conditions and promoting overall well-being. Additionally, TMD supports the

promotion of soft transportation modes, aligning with environmental objectives and enhancing accessibility and social equity.

Lastly, TMD informs bicycle usage planning, helping governments and organizations optimize bicycle stations and routes. This knowledge ensures the creation of efficient infrastructure, encouraging and supporting sustainable transportation choices. Overall, TMD's multifaceted impact underscores its pivotal role in shaping modern transportation systems and urban landscapes.

2.2 Mathematical Problem Formulation

In transportation mode detection (TMD), the problem can be mathematically formulated as a classification task.

Let:

- X be the feature matrix representing the sensor data, where each row corresponds to a sample and each column represents a different sensor feature.
- Y be the target variable representing the transportation mode labels, where each element in Y corresponds to the transportation mode of the corresponding sample in X.

Given a set of sensor data readings, including GPS coordinates, accelerometer measurements, gyroscope readings and magnetometer data, the goal is to determine the transportation mode or activity being performed by an individual at each point in time.

Let's denote the sensor data as $X = \{x_1, x_2, \dots, x_n\}$, where x_i represents the sensor readings at each time step i. The sensor data can include features such as GPS latitude and longitude, acceleration values in different axes, angular velocity, and magnetic field strength. We define a set of transportation modes or activities as $Y = \{y_1, y_2, \dots, y_n\}$, where each mode y_i represents a specific transportation mode (e.g., walking, running, driving a car, cycling). We can represent this as a supervised learning problem, where our objective is to learn a function $f(X)$ that maps the sensor data to the corresponding transportation mode labels, such that $f(X) = y$, where y is the transportation mode inferred from the sensor data.

Mathematically, we can express this as:

$$f(X) = y \tag{1}$$

where $f: X \rightarrow Y$ is a mapping function that assigns the transportation mode y to the given sensor data X.

2.3 Related work

There have been several related works in the field of Transportation Mode Detection (TMD).

Machine Learning: Various traditional machine learning methods on multimodal sensor data have been used in the field of TMD. In [36], the sensor data were divided into frames with a sliding window size of 5.12 s with half overlap, and in each frame computed seven features from the magnitude of the three motion sensors. For the accelerometer, the mean, standard deviation, index of the highest FFT (fast Fourier transform value), and ratio between the first- and second-highest FFT values were computed. For the gyroscope, mean and standard deviation were computed, while for the magnetometer only standard deviation was computed. A decision tree algorithm was employed to train a transportation mode classification model, achieving a 71% F1-score. In the 2020 Research Challenge [37] created by the University of Sussex and Huawei Ltd., among the ML classifiers used, XGBoost achieved the highest F1-score (77.9%), followed by random forest (69.1%) and multilayer perceptron (MLP) (52.8%).

Different machine learning classifiers (K-nearest neighbor, support vector machines, tree-based methods, etc.) were developed in [38] to identify different transportation modes, including bike, car, walk, run, and bus, with random forest producing the best overall performance of 95.1%. Tree-based ensemble models (random forest, gradient-boosting decision tree, and XGBoost) were used in [39] to classify the different transportation modes using Global Positioning System (GPS) data. The experimental results showed that the XGBoost model produced the best performance, with a classification accuracy of 90.77%. In [40], three datasets were created and used for TMD, each with different types of sensors. For each of these datasets, four classification machine learning algorithms were used: decision trees (DT), random forest (RF), support vector machines (SVM), and neural network (NN). For all datasets, random forest had the highest accuracy (81–93%).

Numerous experiments were carried out in [41] to compare the impact of different feature sets (e.g., time-domain features, frequency-domain features, Hjorth features), as well as the impact of various classification algorithms (e.g., random forest, naive Bayes, decision tree, K-nearest neighbor, support vector machine), on the prediction accuracy. This system achieved an average accuracy of 98.33% in detecting the vehicle modes when using the random forest classifier.

Deep learning: Deep learning techniques, which attract significant interest in the machine learning community, were applied in addition to traditional ML algorithms to the transportation mode recognition task, in order to improve the models' performance. A unified framework (CL-TRANSMODE) composed of Convolutional Neural Network (CNN) and Long Short-Term memory (LSTM) was proposed in [42], using the Sussex-Huawei Locomotion (SHL) dataset, and managed to outperform Deep Neural Networks (DNN), CNN, Recurrent Neural Networks (RNN), LSTM,

decision tree, random forest, AdaBoost, and XGBoost for identifying eight transportation modes with a 98.1% accuracy. In [43], a model to detect transportation modes based on a partially observed sequence was presented, and CNN, LSTM, and DNN were used for comparison; the proposed model outperformed them, with an accuracy of 92%. In [44], ML and DL techniques were used on the same dataset as in [40]. Random forest had the best performance among all methods (87%), while CNN and LSTM had F1-score of 80% and 76%, respectively. A series of machine Learning approaches for real-time transportation mode recognition were presented in [45], built on both statistical feature extraction and raw data, and a comparison was made for these approaches using Random Forest (RF), Support Vector Machines (SVMs), Feed-Forward Neural Networks (FFNNs), multilayer LSTM Recurrent Neural Networks, RNNs, and CNNs. RNNs obtained the best performance (88%) using statistical features, while CNNs obtained 98.6% via the analysis of the seven raw data measures without any pre-processing. In [46], a DNN-based approach was proposed to efficiently recognize five transportation modes (still, walk, run, bike and vehicle) from accelerometer, magnetometer, and gyroscope measurements. DNN achieved approximately 95% classification accuracy, and outperformed four machine learning methods, i.e., AdaBoost, Decision Trees (DT), K-Nearest Neighbors (KNN), and SVM. In [47], a novel input set consisting of extracted features, rather than raw data, was fed to an LSTM model, for 10 different transportation modes, achieving 96.82% performance. A CNN model built on the one-dimensional acceleration data was used to determine the transportation mode in [48].

Transformers: Transformers [49] adopt an attention-mechanism which examines an input sequence and decides at each step the importance of the other parts of the sequence. Since their introduction they have been gradually shown to outperform LSTM which was the previous state-of-the-art model in sequential data analysis. Attention was initially designed in the context of Neural Machine Translation using Seq2Seq Models in the Natural Language Processing (NLP) field. In NLP, both attention-mechanism and Transformers have been used effectively in a variety of tasks such as reading comprehension, abstractive summarization, word completion, and others [50,51,52].

Before Transformers appear, most state-of-the-art NLP models were based on RNN [53], LSTM [54] or CNN [55], however these methods presented certain limitations. RNN [56,57] processed data sequentially but firstly this was not very efficient in handling long sequences and secondly, it was difficult to take full advantage of modern fast computing devices such as TPUs and GPUs. Even though LSTM offered a slight improvement over conventional RNN concerning long dependency issues, there were still particular constraints such as the need for sequential processing that not allowed parallel training and the difficulty in handling long sequences and long-range dependencies. Convolutional Neural Networks (CNNs) [58], widely used in NLP field, trained quite fast and were efficient with short texts, but the number of different kernels required to capture dependencies between all possible

combinations of words in a sentence, would be huge and impractical. On the other hand, the reason Transformers outperform all other architectures is the fact that they completely avoid recursion. That is because, thanks to multi-head attention mechanisms and positional embeddings, they process sentences as a whole and they learn relationships between words.

The additional feature of training parallelization allows training on larger datasets than was once possible. This feature led to the development of pretrained systems such as BERT (Bidirectional Encoder Representations from Transformers) [59] which was trained with large language datasets, such as the Wikipedia Corpus and Common Crawl. Thus, although Transformers were used primarily in the fields of natural language processing (NLP), the development of pretrained systems such as BERT sparked a wave of research in other domains, too. In Computer Vision domain, models which can be well adapted to different image processing tasks were developed. In [60], ImageGPT a sequence Transformer is trained to auto-regressively predict pixels, without built-in knowledge of the 2D input structure. In [61], a pre-trained model with transformer architecture is developed for image processing, and in [62], Facebook AI researchers presented DETR showing that transformers can be used for object detection and segmentation, with very competitive results. In Biology and Chemistry research field, AlphaFold2 model [63] was developed, an equivariant structure prediction module for protein structure prediction, as well as SE(3)-Transformers [64] a variant of the self-attention feature for 3D point clouds and graphs, which is equivalent under continuous 3D rotational translations. In Transportation domain, spatial transformer [65], a new variant of graph neural networks has been developed. This model uses directed spatial dependencies with self-attention mechanism and manages to capture and predict traffic flows in real-time. In [66] TrafficBERT is proposed, a model for traffic flow prediction which is based on pre-trained BERT and can be used on a variety of roads as it is pretrained with a large traffic dataset.

Data Fusion: Data fusion entails amalgamating data acquired from various mobile and wearable sensor devices. This amalgamation enhances the dependability, resilience, and overall effectiveness of recognition systems. The primary objective is to diminish uncertainty and overcome the challenges presented by indirect capture, which are hard to tackle using data from a solitary sensor source [67,68].

While the use of Data Fusion techniques to model complex systems is not new [69,70], there is a growing interest in applying them to transportation systems. Specifically, road traffic is an area where Data Fusion techniques are expected to bring significant advantages. The prediction of traffic patterns plays a crucial role in intelligent transportation systems. Accurate traffic predictions are vital for improving routing, dispatching, and congestion management strategies. Recent research in this area has resulted in effective solutions for forecasting traffic flow. Authors in [71] propose a deep learning urban traffic prediction model that combines information extracted from tweet messages with traffic and weather

information. In [72] an attribute channel fusion module is built to fuse local motion states and capture the spatial dependencies so as to perform accurate transportation mode identification. In [73] a transportation mode recognition methodology is introduced, which utilizes a multi-scale fusion technique to integrate motion sensor and GNSS data, achieving a more detailed and precise analysis. A data fusion strategy for sensor networks is suggested in [74], aiming to identify the type of vehicles in traffic flow detection. In [75] authors use fusing multimodal data from wearable sensors (motion, sound and vision) for transportation mode recognition.

Although sensor fusion offers numerous advantages, it also presents certain challenges and limitations in its application.

- **Data quality**: The reliability and accuracy of sensor data used in data fusion can be compromised due to errors, such as outliers, missing values, or incorrect readings. Poor data quality can lead to inaccurate decision-making and adversely impact the effectiveness of data fusion in transportation.
- **Data heterogeneity**: Transportation systems often rely on data from various sources, such as different types of sensors, GPS devices, or traffic cameras. Integrating and fusing heterogeneous data from these diverse sources can be challenging due to differences in data formats, resolutions, or sampling rates.
- **Scalability**: As transportation systems generate large volumes of data, the scalability of data fusion becomes a concern. Processing and fusing data from numerous sensors in real-time can require significant computational resources and may pose challenges in terms of latency and system performance.
- **Privacy concerns**: Data fusion involves aggregating and analyzing data from different sources, which can raise privacy concerns. Ensuring the protection of sensitive information and complying with privacy regulations can be challenging, especially when dealing with personal data in transportation systems.
- **System complexity**: Implementing data fusion in transportation systems requires integrating multiple technologies, algorithms, and data sources. The complexity of designing and managing such systems can be a limitation, as it requires expertise in various domains and coordination among different stakeholders.
- **Cost**: Deploying and maintaining data fusion systems in transportation can involve significant costs. The need for sophisticated sensors, infrastructure, computational resources, and ongoing maintenance can be a limitation, especially for smaller transportation organizations or regions with limited budgets.
- **Data sharing and interoperability**: Effective data fusion in transportation often relies on sharing data among different entities, such as transportation agencies, service providers, or vehicles. Ensuring data sharing agreements, establishing interoperability standards, and addressing data ownership issues can be complex and pose limitations to seamless data fusion.
- **Data fusion algorithms and techniques**: Selecting appropriate data fusion algorithms and techniques that are suitable for transportation applications can be challenging. Different situations, such as traffic congestion, accidents, or

weather conditions, may require specific algorithms or approaches, and choosing the right ones can be a limitation without proper expertise and research.

Overall, while data fusion offers significant potential in transportation, these limitations need to be addressed to ensure its effective implementation and utilization.

2.4 Sensors

Numerous research efforts have been conducted to study activity detection and classification using either inertial or multimodal sensors [76]. Inertial sensors, as presented in Figure 2.2, include accelerometers, gyroscopes, and magnetometers and provide dynamic information through direct measurement. These sensors have become popular due to their small size, cost-effectiveness, and integration into smartphones and wearable devices. Accelerometer-based systems have been extensively studied in various applications concerning posture and movement recognition [77]. Due to the improved processing power of mobile phones and the availability of diverse sensors enabling data collection and transmission via Wi-Fi or Bluetooth interfaces, mobile devices have integrated activity recognition, enabling efficient real-time monitoring [78].

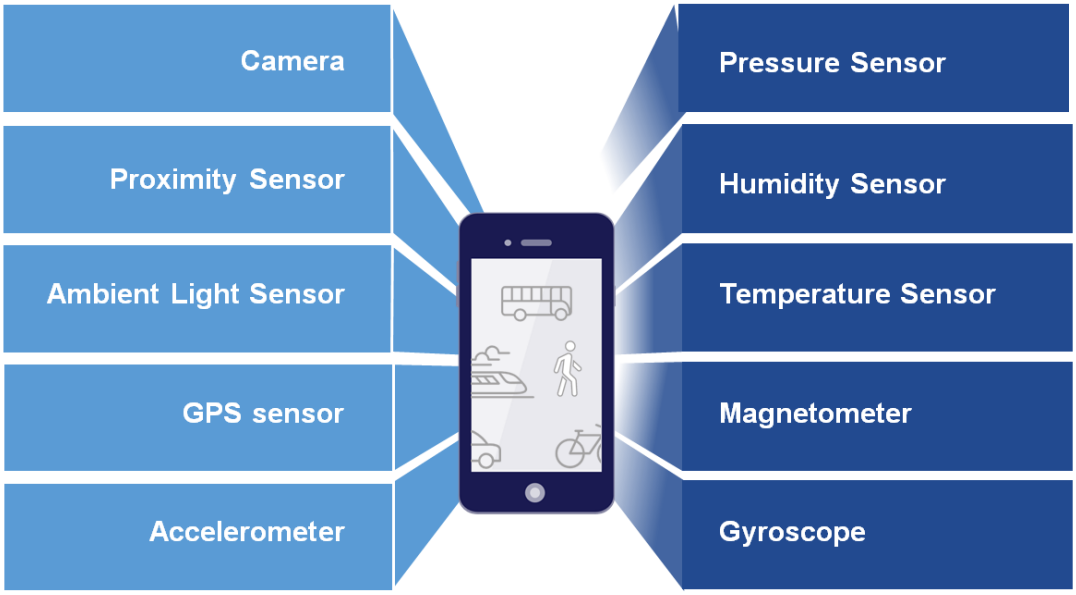


Figure 2.2 Smart phone sensors.

However, as a single accelerometer is affected by the positioning of the sensor and the sensor drift, along with its insufficiency to recognizing complex movements, researchers have proposed using gyroscopes or multiple accelerometers placed on different body parts to improve recognition rates [79,80]. Some studies have explored the placement of multiple accelerometers to differentiate various activities, but this approach can be burdensome, especially for elderly patients.

Sensor fusion techniques are currently being developed to integrate multiple inertial sensors and multimodal sensors for the purpose of human activity recognition and monitoring [81,82,83,84]. The fusion of these sensors provides complementary advantages and improves orientation detection and response time. Combining gyroscopes with accelerometers and magnetometers corrects errors and improves the accuracy of rotation output. Magnetic fields also help generate accurate acceleration readings independent of the smartphone's orientation during motion. These sensor fusions enable the deduction of context-aware monitoring, transportation mode analysis, and real-time detection [85] using smartphones.

2.5 Relevant Datasets

2.5.1 Geolife trajectory dataset

This GPS trajectory dataset (2007- 2012) [86] was collected in (Microsoft Research Asia) Geolife project by 182 users in a period of over three years (from April 2007 to August 2012). A GPS trajectory of this dataset was represented by a sequence of time-stamped points, each of which contained the information of latitude, longitude and altitude. This dataset contained 17,621 trajectories with a total distance of about 1.2 million kilometers and a total duration of 48,000+ hours. These trajectories were recorded by different GPS loggers and GPS-phones. This dataset recoded a broad range of users' outdoor movements, including not only life routines like go home and go to work but also some entertainments and sports activities, such as shopping, sightseeing, dining, hiking, and cycling. As far as TMD is concerned, the possible transportation modes are: walk, bike, bus, car, subway, train, airplane, boat, run and motorcycle.

Every single folder of this dataset stores a user's GPS log file:

- Latitude in decimal degrees.
- Longitude in decimal degrees.
- All set to 0 for this dataset.
- Altitude in feet (-777 if not valid).
- Date - number of days (with fractional part) that have passed since 12/30/1899.
- Date as a string.
- Time as a string.

This trajectory dataset can be used in many research fields, such as mobility pattern mining, user activity recognition, location-based social networks, location privacy, and location recommendation [87,88,89].

2.5.2 TMD-Kaggle

The TMD-Kaggle Dataset (2017) [90] has been developed at the University of Bologna. It is based on thirteen users who collected the data during their daily activities. The dataset contains 5893 samples, includes all sensors available in phones and distinguishes five transportation modes: being on a car, on a bus, on a train, standing still and walking.

Based on meaning reasons, the following sensors were excluded from data used for training model:

- light
- pressure
- magnetic field
- magnetic field uncalibrated
- gravity
- proximity

From remaining nine possibly relevant sensors three different sensors set were created. The first set contains only three sensors whose are base sensors and have almost complete or complete users support (accelerometer, gyroscope, and sound). The second set contains sensors from first set plus all the other sensor excluded speed, that is a GPS-based sensor, for its high consumption of the battery. Lastly, the third set contains all sensors that defined as relevant for the task.

The dataset was used to build different models, with different classification algorithms (Decision Tree, Random Forest, Support Vector Machine and Neural Network) reaching a maximum level of accuracy of 96% [91].

2.5.3 Sussex-Huawei Locomotion (SHL)

The SHL dataset (2017) [92] from University of Sussex, which is considered as one of the biggest dataset in the research community, was collected primarily to investigate the recognition of users' modes of locomotion and transportation from the sensors in mobile phone based on using machine learning methods.

The SHL dataset was recorded over a period of 7 months in 2017 by 3 participants in the United Kingdom. Each participant carried four smartphones simultaneously at four body locations (in the hand, at the torso, in the hip pocket, in a backpack or handbag), which results in 4 × 703 = 2812 hours of annotated data. Each smartphone was logging the data of the 15 sensors available in the smartphone (e.g., inertial sensors, GPS, ambient pressure sensor, ambient humidity). Beside the smartphones, the participants also wore a front-facing camera, to verify and correct

annotations and to introduce additional post collection annotations. This resulted in 28 total annotation types, including 8 modes of transportation (e.g., Still, Walk, Run, Bike, Car, Bus, Train, and Subway), the participant's posture, inside/outside location, road conditions, presence in tunnels, social interaction, and having meals.

The original SHL dataset contains the data from all sensors of the phones that are used. The following sensor modalities were recorded:

- Accelerometer: x, y, z in m/s2
- Gyroscope: x, y, z in rad/s
- Magnetometer: x, y, z in µT
- Orientation: quaternions in the form of w, x, y, z vector
- Gravity: x, y, z in m/s2
- Linear acceleration: x, y, z in m/s2
- Ambient pressure in hPa
- Google's activity recognition API output: 0-100% of confidence for each class ("in vehicle", "on bicycle", "on foot", "running", "still", "tilting", "unknown", "walking")
- Ambient light in lx
- Battery level (0-100%) and temperature (in °C)
- Satellite reception: ID, SNR, azimuth and elevation of each visible satellite
- Wi-Fi reception including SSID, RSSI, frequency and capabilities (i.e. encryption type)
- Mobile phone cell reception including network type (e.g. GSM, LTE), CID, Location Area Code (LAC), Mobile Country Code (MCC), Mobile Network Code (MNS), signal strength
- Location obtained from satellites (latitude, longitude, altitude, accuracy)
- Audio

A body worn-camera recorded a time-lapse video with one frame taken every 30 seconds. The camera was worn on the chest or backpack straps, generally facing in the forward direction [93]. Table 2.1 shows the basic characteristics of the common datasets comparatively.

Dataset	Devices/user	Sensors	Particip.	Labels	Hours	Samples
Geolife	GPS logger and GPS phone	GPS	182	walk, bike, bus, car, subway, train, airplane, boat, run and motorcycle	48000+	18670 (28 trajectories per user- 800 points per trajectory)
TMD	1 smartphone	9	13	car, bus,	31	5893 per

		(Accelometer, Sound, Gyroscope, Orientation, Linear Acceleration, Rotation vector, Game Rotation vector, Gyroscope uncalibrated, Speed)		train, still and walking		dataset
SHL	4 smartphones + 1 camera	15 (Accelometer, Gyroscope, Magnetometer, Orientation, Gravity, Linear Acceleration, Ambient Pressure, Google's activity recognition API, Ambient light, Battery level and temperature, Satellite reception, Wi-Fi reception, Mobile phone cell reception, Location obtained from satellites, Audio)	3 users	Still, walk, run, bicycle, car, bus, train, subway	3000	More than 1.ooo.ooo per day

Table 2.1 An overview of the common datasets from the literature.

Geolife, which solely consisted of GPS measurements, was excluded from consideration among the three datasets. When comparing TDM and SHL, both of which included measurements from multiple sensors, SHL was ultimately selected due to the following reasons:

- More labels: SHL provided a greater number of labeled instances, allowing for a more comprehensive analysis and evaluation of TMD algorithms.
- More samples: SHL offered a larger dataset size, providing a broader range of data points for training and testing TMD models.
- More hours: SHL encompassed a longer duration of data collection, allowing for a more extensive examination of transportation patterns and behavior across different time periods.
- More sensors: SHL incorporated data from a wider range of sensors, enabling the exploration of various features and their impact on TMD accuracy.

Considering these factors, SHL emerged as the preferred dataset TMD research and experimentation.

The SHL dataset was accessible in two versions for download:

1. SHL Preview: This version consisted of data from 3 users, spanning 3 days per user, and included information from 4 phone locations and a time-lapse camera.

2. SHL Complete User 1 – Hips phone: The second version focused on a single user and covered all the recording days, approximately 7 months. It specifically included data from the phone placed at the pocket (Hips) and the time-lapse camera.

For this research, the second version was selected as it encompassed a greater number of recording dates. This provided a larger sample size, offering more opportunities for experimentation and analysis.

Chapter 3

Data Preprocessing and Manipulation for Transportation Mode Detection

Due to the rise of the Internet of Things (IoT) and Wireless Sensor Networks (WSNs), IoT applications can now encompass hundreds or even thousands of sensors, generating extensive datasets. However, this data becomes ineffective if it contains errors. Poor sensor data quality, resulting from these errors, can lead to inaccurate decision-making, rendering the data unreliable.

More specifically, sensors data errors may include various points that impact the reliability and accuracy of the collected data [94]. One issue is noise, which arises from random variations in the measured quantity. This noise can distort the actual signal, leading to inaccuracies in measurements. Calibration errors are another concern. Sensors must be calibrated to ensure precise measurements. Inaccurate calibrations can result in unreliable readings, compromising the integrity of the data. Drift is a phenomenon where sensor readings deviate from the true value over time. This can be caused by factors such as aging or environmental changes, introducing inaccuracies into the data. Interference from external sources, like electromagnetic interference or cross-talk between sensors, can also introduce errors, leading to incorrect readings. Additionally, missing data can pose a challenge when sensors fail to capture data points at specific times, creating gaps in the dataset. These gaps can disrupt the continuity and completeness of the information, affecting the overall analysis. Outliers, or readings significantly different from the rest of the data points, can distort the analysis if not properly identified and handled. Managing outliers is essential for accurate data interpretation. Data inconsistencies, including issues with formats, units, or timestamps, can complicate the integration and analysis of sensor data. Incompatible or inconsistent data formats can hinder the seamless processing of information, making it challenging to draw meaningful insights. Addressing these challenges is crucial to ensuring the reliability and usefulness of sensor-generated data in various applications.

3.1 Data Preprocessing with Feature Transformers

Data preprocessing in machine learning is essential because it cleans, transforms, and organizes data in a way that maximizes the effectiveness and accuracy of machine learning algorithms. Proper preprocessing ensures that models are built on reliable, high-quality data, leading to more meaningful and reliable insights.

Preprocessing in machine learning is a vital step that addresses various aspects of data preparation to enhance the performance and reliability of algorithms. One key aspect is Data Quality Assurance, where preprocessing methods are employed to

clean and manage noisy, missing, or inconsistent data, ensuring the dataset used for training models is of high quality and dependable.

Feature transformation techniques combine the original set of features to obtain a new set of less-redundant variables [95].

Handling Missing Values is another critical function of preprocessing. Techniques can either fill in missing values or remove instances with missing data, preventing biased analysis and significantly improving the accuracy of models.

Feature Scaling is essential because features often have different scales, which can impact the performance of certain machine learning algorithms. Preprocessing methods such as normalization or standardization ensure that all features contribute equally to the model. This prevents a single feature from dominating due to its larger scale, ensuring a more balanced and accurate analysis.

When dealing with categorical data, preprocessing techniques like one-hot encoding or label encoding are applied. These methods transform categorical variables into a format compatible with algorithms that typically work with numerical data, enabling the inclusion of categorical information in the analysis.

Another critical preprocessing technique is Dimensionality Reduction. High-dimensional data can pose challenges for algorithms. Techniques like Principal Component Analysis (PCA) are utilized to reduce the number of features, making the dataset more manageable. This helps prevent the curse of dimensionality, a phenomenon where high-dimensional data can lead models to overfit the data.

Preprocessing also facilitates Feature Engineering, allowing the creation of new features based on existing ones. These engineered features can capture intricate relationships in the data, enhancing the performance of machine learning models by providing them with more relevant and valuable input.

Additionally, preprocessing methods aid in Outlier Detection and Removal. Identifying and handling outliers ensures that extreme values do not unduly influence the model, leading to more robust and accurate predictions.

Data Normalization is yet another crucial preprocessing step. Normalizing data to a standard scale ensures that the model is not swayed by the magnitude of the values, making it simpler to compare and analyze different features on an equal footing.

3.2 Imputation of missing data

Imputation [96] refers to the process of filling in missing values in datasets with incomplete information.

Fixed Value Imputation is a form of imputation where missing values in a dataset are replaced by specific fixed values. These fixed values can be constants (such as 0) or other predetermined values. This approach assumes that the missing values are missing at random and do not significantly affect the overall distribution.

Fixed Value Imputation, as the name suggests, involves filling in the missing entries with specific, predetermined constants. These constants could be zero, a negative value, or any other chosen numerical value. Unlike more complex imputation methods, Fixed Value Imputation simplifies the process by uniformly assigning a constant to all missing data points.

While Fixed Value Imputation may seem simplistic, it finds applications in situations where the missing values do not carry significant information or where the focus lies on retaining the dataset's structure rather than the exact values. This method is particularly useful when dealing with categorical variables, ordinal data, or when the missing data is random and doesn't carry a specific pattern. Despite its simplicity, Fixed Value Imputation can provide a quick solution in scenarios where maintaining dataset completeness is crucial for subsequent analyses. It is quick and straightforward, making it useful for rapidly filling missing values in large datasets without introducing significant computational overhead.

3.3 Data normalization

Data normalization [97,98], also known as data scaling or feature scaling, is a preprocessing technique used in statistics and machine learning to standardize the range of independent variables or features in a dataset. The goal of data normalization is to bring all features to a similar scale, ensuring that no single feature dominates the others. Normalization is particularly important when features have different units of measurement or varying scales, as it allows algorithms to converge faster and produce more accurate results during the training process.

There are different methods for data normalization, but two common techniques are:

- Z-score Normalization (Standardization): Subtracts the mean from each data point and then divides the result by the standard deviation. This results in a dataset with a mean of 0 and a standard deviation of 1.

$$z = \frac{x-\mu}{\sigma} \tag{2}$$

Where:

- z is the standardized value
- x is the original value
- μ is the average value of the feature (mean)
- σ is the standard deviation of the feature

With mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{3}$$

And standard deviation:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \quad (4)$$

- Min-Max Scaling (Normalization): The original data undergoes a linear transformation. The process involves extracting the minimum and maximum values from the dataset and each data point is then replaced based on the following formula. Data is scaled to a specific range $[min, max]$. By default all values will be scaled to $[0, 1]$ interval.

  Min – Max Scaling:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (5)$$

  Where, X_{max} and X_{min} are the maximum and the minimum values of the feature, respectively.

  When the value of X is the minimum value in the column, the numerator will be 0, and hence X_{norm} is 0. On the other hand, when the value of X is the maximum value in the column, the numerator is equal to the denominator, and thus the value of X_{norm} is 1. If the value of X is between the minimum and the maximum value, then the value of X_{norm} is between 0 and 1.

Both Z-score Normalization and Min-max Scaling are valuable methods in data preprocessing and the choice between them depends on the specific requirements and the context of the application. For instance, in clustering analyses, standardization is crucial for comparing similarities between features using specific distance measures. Another notable case is Principal Component Analysis (PCA), where standardization is generally favored over Min-Max scaling. This decision is influenced by the objective, particularly when the goal is to maximize variance in the components. The choice also hinges on whether PCA computes components via the correlation matrix or the covariance matrix.

While normalization has certain drawbacks, such as potential loss of information if the original scale of input features is crucial, increased difficulty in detecting outliers since they are scaled along with other data, and added computational expenses, its advantages outweigh these limitations. Normalization can enhance the performance of ML algorithms by standardizing input features to a uniform scale, mitigating the influence of outliers, and facilitating the interpretation of machine learning model results due to the consistent scale of inputs.

3.4 High-dimensional data

High-dimensional data refers to datasets that have a large number of features or variables. In Machine Learning, this can pose challenges because as the number of features increases, the complexity of the problem also increases.

In the realm of high-dimensional data, several challenges emerge. Firstly, the curse of dimensionality becomes apparent as the number of dimensions grows, leading to an exponential expansion of the space. Consequently, the data becomes sparse in this high-dimensional space, posing difficulties in identifying meaningful patterns due to this sparsity issue.

Secondly, the abundance of features in high-dimensional datasets raises concerns about overfitting [99]. Machine learning models, when confronted with a large number of features, might become excessively intricate, capturing noise in the data rather than the underlying patterns. To mitigate this, employing regularization techniques is imperative, serving as a safeguard against overfitting in the realm of high-dimensional data analysis.

Lastly, the computational complexity escalates significantly with the rise in dimensions. Many machine learning algorithms demand substantial computational resources when operating on high-dimensional data, making tasks such as training and evaluating models a computationally expensive endeavor. These computational challenges add another layer of complexity to the analysis of high-dimensional datasets.

Navigating the complexities of high-dimensional data involves addressing several pivotal challenges. Firstly, selecting pertinent features proves to be of paramount importance in this domain. Employing feature selection methods becomes crucial as they aid in pinpointing the most informative features, thus reducing the dataset's dimensionality effectively. Additionally, employing dimensionality reduction techniques such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) becomes imperative. These methods enable the reduction of data dimensionality while preserving its fundamental characteristics, enhancing the manageability of the dataset. To combat the risk of overfitting in high-dimensional data, regularization techniques like Lasso and Ridge [100] regression come into play. These methods penalize large coefficients, discouraging an over-reliance on any single feature and ensuring a more balanced model.

Ensemble methods like Random Forest and Gradient Boosting offer another avenue for effectively handling high-dimensional data. By amalgamating predictions from multiple models, these techniques enhance the robustness and accuracy of the analysis. Delving into the realm of deep learning, architectures like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) present a compelling solution for deciphering hierarchical features from high-dimensional data. However, it's crucial to note that these methods necessitate substantial

amounts of data and computational resources for optimal performance. Lastly, proper implementation of cross-validation techniques [101] proves indispensable in evaluating model performance within high-dimensional settings. These techniques play a pivotal role in averting overfitting during model training, ensuring the reliability and accuracy of the analysis conducted on high-dimensional datasets.

Dealing with high-dimensional data requires careful preprocessing, feature engineering, and model selection. It's essential to strike a balance between capturing the complexity of the data and preventing overfitting. By employing appropriate techniques, researchers and practitioners can extract meaningful insights and build accurate predictive models from high-dimensional datasets.

3.4.1 Feature importance and Correlation

Feature importance and correlation are two essential concepts in machine learning, each serving distinct roles in the analysis and understanding of data. Here's how they differ and how they relate to one another:

Feature importance in machine learning refers to the process of evaluating and quantifying the impact of each feature (variable or attribute) in a dataset on the outcome or prediction made by a machine learning model. Understanding feature importance is essential because it helps identify which features have the most significant influence on the target variable. By recognizing these important features, data scientists and analysts can prioritize them for further analysis, focus on relevant aspects of the data, and potentially improve the model's performance. Feature importance helps in identifying which features have the most substantial impact on the model's outcomes. It aids in feature selection, allowing data scientists to focus on relevant variables and potentially improve the model's accuracy and efficiency. Feature importance can be determined through techniques such as decision trees [102], random forests [103], permutation importance [104], LASSO regression [105], and gradient boosting machines [106]. These methods provide scores or rankings indicating the importance of each feature in the model.

This paper introduces a method for calculating feature importance based on permutation tests, which has become a popular technique.

Correlation measures the statistical relationship between two variables. In the context of machine learning, it is often used to quantify the degree to which two features change together. These features provide redundant information and affect the performance of the model. Correlation values range from -1 to 1, where 1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation and 0 indicates no correlation. Correlation helps in understanding how variables interact with each other. High correlation between features might indicate redundancy, suggesting that only one of the correlated features needs to be considered for the model. Common correlation measures include Pearson

correlation coefficient (for linear relationships) [107] and Spearman rank correlation coefficient (for monotonic relationships) [108].

If two features are negatively correlated, it means they move in opposite directions. In some cases, a ML model might assign high importance to one feature and low importance to the other, effectively capturing the information from both features. If two features are positively correlated, they move in the same direction. A machine learning model might assign similar importance to both features. However, if one feature is substantially more important, it might overshadow the importance of the correlated feature. Finally, if features are uncorrelated, each feature's importance is determined independently based on its contribution to the target variable.

Correlation is useful because it can help in predicting one attribute from another, it can indicate the presence of a causal relationship and it is used as a basic quantity for many modeling techniques

Understanding both feature importance and correlation is crucial. Feature importance guides feature selection and model improvement, while correlation analysis helps in identifying relationships between features, avoiding redundancy, and gaining insights into the dataset's structure. Data scientists often utilize both concepts in conjunction to build more effective and interpretable machine learning models.

3.5 Dimensionality Reduction

Dimensionality Reduction is a preprocessing method in machine learning. It is a crucial preprocessing step that prepares the data for subsequent analysis and modeling, leading to more accurate, efficient, and interpretable machine learning models.

Central to contemporary AI initiatives are Machine Learning systems that rely on data to harness their predictive capabilities. While Machine Learning algorithms can handle large datasets [109], their efficiency decreases as the number of dimensions rises [110].One of the main data issues that are crucial for building accurate, reliable and effective machine learning models apart from the poor quality of data and the limited data for specific problems, is the high dimensionality of data. As the number of features or dimensions in a dataset increases, the amount of data required to cover the feature space adequately grows exponentially. This can lead to sparse data, making it difficult for machine learning algorithms to generalize well.

Dimensionality reduction mitigates the curse of dimensionality [111, 112] by reducing the number of features, making the data more manageable and improving the performance of machine learning algorithms. Feature dimensionality reduction utilizes current feature parameters to establish a low-dimensional feature space, effectively addressing redundant or irrelevant information. This process f ensures the essential information from the original features $X = [x_1, x_2, \dots, x_m]$ is condensed into a smaller feature set $Y = [y_1, y_2, \dots, y_n]$ where $n < m$.

In practical scenarios, reducing the dimensionality significantly impacts classification performance, making complex problems more manageable. This reduction in the number of features can transform a problem that seemed impossible to solve into one that is feasible. This reduction can be accomplished through either feature selection or feature extraction methods [113].

Feature selection, as shown in Figure 3.1, is the process of selecting a subset of relevant features for model construction, thus reducing training times, simplifying the models (to make interpretation easier) and improving the chances of generalization, avoiding overfitting [114].

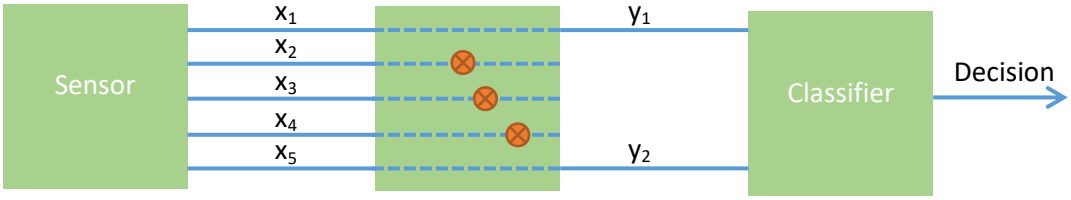


Figure 3.1 Feature Selection where a subset of relevant features for model construction is selected.

Feature extraction [115,116], as shown in Figure 3.2, is the transformation of original data to a data set with a reduced number of variables, which contains the most discriminatory information. This reduces the data dimensionality, removes redundant or irrelevant information, and transforms it to a form more appropriate for subsequent classification.

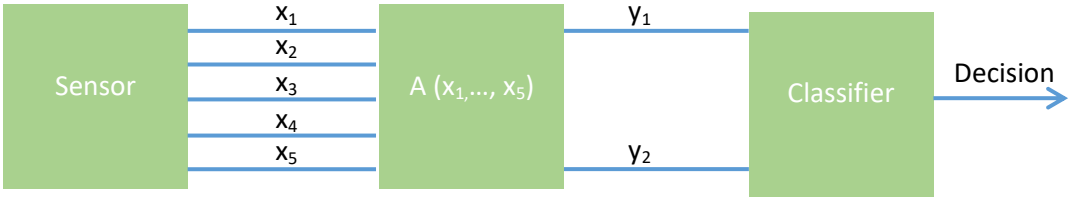


Figure 3.2 Feature Extraction where original data is transformed to a dataset with a reduced number of variables.

3.5.1 Dimensionality Reduction with PCA

Principal Component Analysis (PCA) was first introduced by Karl Pearson in [117] in 1901. In this work, Pearson presented the fundamental concepts and mathematical principles behind PCA, making it one of the earliest works on the topic.

PCA [118] is a method used to reduce the dimensionality of datasets containing numerous features or dimensions. Employing principles from linear algebra, PCA identifies the most significant features within a dataset. Once these crucial features are pinpointed, they can be utilized to train a ML model, enhancing computational efficiency without compromising accuracy. PCA is a common approach in exploratory data analysis, feature extraction, and dimension reduction [119, 120].

When dealing with a dataset comprising multivariate observations, the objective is to decrease dimensionality and enhance the interpretability of raw data while minimizing information loss. By calculating principal components (PCs) and utilizing them to establish a new basis for the data, a reduced set of variables with minimal redundancy can be obtained. These PCs represent observed signals as a linear combination of orthogonal components.

In the process of PCA, several steps are taken to transform the original high-dimensional dataset into a reduced, lower-dimensional form. Firstly, the dataset is standardized, ensuring that all features have a mean of 0 and a standard deviation of 1, which guarantees equal contribution of all features to the analysis. Subsequently, PCA calculates the covariance matrix of the standardized data, representing the relationships between different features. The computation then involves deriving the eigenvectors and eigenvalues from this covariance matrix. Eigenvectors serve as the principal components, capturing the directions of maximum variance, while eigenvalues signify the magnitude of variance along each principal component. To select the most significant components, the eigenvectors are ranked based on their corresponding eigenvalues, with the top k eigenvectors (where k represents the desired number of dimensions for the reduced dataset) being chosen. Finally, the original high-dimensional data is projected onto these selected principal components, forming a new lower-dimensional dataset. This process ensures a more manageable and interpretable representation of the data.

Figure 3.3 illustrates the fundamental concept and schematic process of PCA with 2 principal components, $PC1$ and $PC2$. Given a dataset X with n variables in PCA, the first principal component $PC1$ which retains the maximum variance, can be calculated as a linear combination:

$$PC1 = \omega_{11}X_1 + \omega_{12}X_2 + \cdots + \omega_{1n}X_n \quad (6)$$

It establishes the direction with the highest variability within the data. The first component captures the most significant information, and no other component can exceed its variability. The first principal component forms a line that best fits the data, minimizing the sum of squared distances between data points and the line.

Similarly, the second principal component can also be computed. $PC2$ is another linear combination of the original predictors, capturing the remaining variance in the dataset and being uncorrelated with $PC1$. In simpler terms, there should be zero correlation between the first and second components. It can be expressed as:

$$PC2 = \omega_{12}X_1 + \omega_{22}X_2 + \cdots + \omega_{n2}X_n \quad (7)$$

Where ω are the weights (loadings) assigned to each corresponding original variable $x_1, x_2, \ldots, x_n$ in each principal component.

These weights ω indicate the contribution of each original variable to the construction of the second principal component. The values of ω are determined

during the PCA process, specifically in the calculation of the eigenvectors and eigenvalues of the covariance or correlation matrix of the original data. Each weight represents the importance of the corresponding variable in the formation of the principal component.

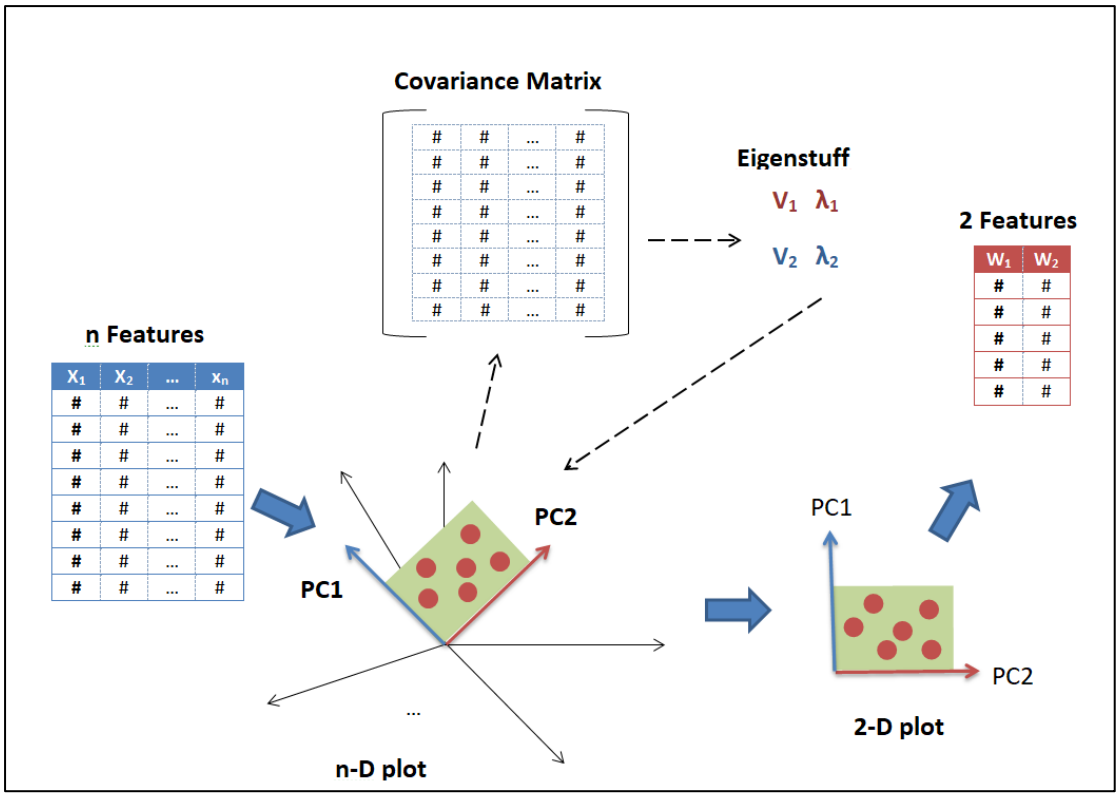


Figure 3.3 The schematic process of PCA with 2 principal components, PC1 and PC2

3.5.2 Dimensionality Reduction with LDA

Linear Discriminant Analysis (LDA) [121] is a supervised dimensionality reduction technique used in machine learning and pattern recognition. Unlike PCA, which is an unsupervised method, LDA takes into account the class labels of the data points during the reduction process. LDA aims to find a lower-dimensional space that maximizes the separation between different classes in the dataset. It does this by projecting the data points onto a new subspace, where the classes are well-separated. The key idea is to maximize the between-class scatter while minimizing the within-class scatter.

The initial step requires computing the mean vectors for each class in the dataset. Let's consider a dataset with N samples, D features and C classes. The mean vector (often denoted as μ_c) for each class c is calculated as follows: For a single feature, the mean of class c is calculated as the average of all samples belonging to class c.

$$\mu_c = \frac{1}{N_c}\sum_{i=1}^{N_c} x_i \tag{8}$$

Where N_c is the number of samples in class c_i and x_i represents the i-th sample in class c.

These mean vectors $\mu_c$are essential in LDA as they help determine the optimal linear discriminant functions that maximize the separability between classes.

LDA evaluates the spread of data within each class by calculating the within-class scatter matrix. It measures how far individual data points within a class are from the class mean. Simultaneously, LDA calculates the between-class scatter matrix, which assesses the spread between different class means. It measures how distinct the classes are from each other in the feature space. LDA then performs eigenvalue decomposition on the inverse of the within-class scatter matrix multiplied by the between-class scatter matrix. This step results in eigenvalues and corresponding eigenvectors. LDA selects the top k eigenvectors corresponding to the k largest eigenvalues, where k represents the number of desired dimensions in the reduced feature space. These eigenvectors serve as transformation vectors. By projecting the original high-dimensional data onto these vectors, a new set of feature vectors is obtained.

$$y = v_k^T x \tag{9}$$

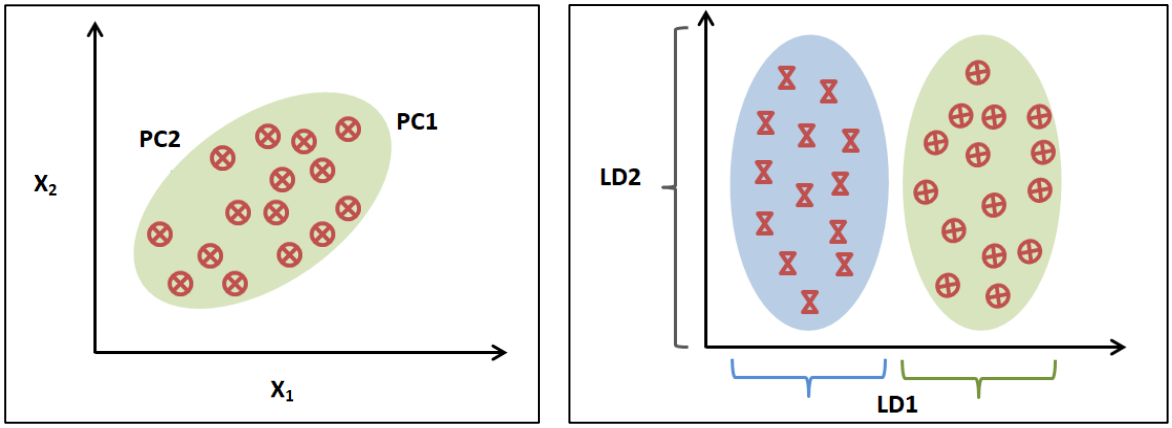
Where v_k represents the matrix of the k selected eigenvectors.

PCA and LDA are both popular techniques for dimensionality reduction in machine learning and data analysis. However, they have different objectives and work under different assumptions.

PCA is an unsupervised dimensionality reduction technique. Its primary goal is to find the orthogonal axes (principal components) along which the variance of the data is maximized. PCA does not take class labels into account during the reduction process; it focuses on capturing the overall variance within the dataset. It is useful for exploratory data analysis and visualization.

LDA, on the other hand, is a supervised dimensionality reduction technique. It considers class labels and aims to find a subspace where the classes are well-separated. LDA's objective is to maximize the between-class scatter and minimize the within-class scatter. Unlike PCA, LDA is specifically designed for improving class separability, making it beneficial for classification tasks [122, 123].

We can picture PCA as a technique that finds the directions of maximal variance.



(a) (b)

Figure 3.4 (a) PCA focuses on capturing the overall variance within the dataset; (b) LDA aims to find a subspace where the classes are well-separated.

In contrast to PCA in Figure 3.4 (a), LDA attempts to find a feature subspace that maximizes class separability. LDA makes assumptions about normally distributed classes, classes that have identical co-variance matrices and features statistically independent of each other. LDA as shown in Figure 3.4 (b), on the x-axis separates the two normally distributed classes well. Although the LD2 captures a lot of variance in the dataset, it would fail as good linear discriminant since it doesn't capture any of the class discriminatory characteristics.

Chapter 4

Bayesian-Optimized Long Short-Term Recurrent Modeling for Transportation Mode Detection

4.1 LSTM for Recognition of Transportation Mode

Recurrent Neural Networks (RNNs) are artificial neural networks that specialize in processing sequential or time series data. They are commonly used for tasks involving time-series data, such as language translation, natural language processing (NLP), speech recognition, and image captioning. As depicted in Figure 4.1, their Memory State, allows them to consider previous inputs when processing current input and generating output. Unlike traditional deep neural networks that assume input-output independence, RNNs rely on prior elements in the sequence to determine their output. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

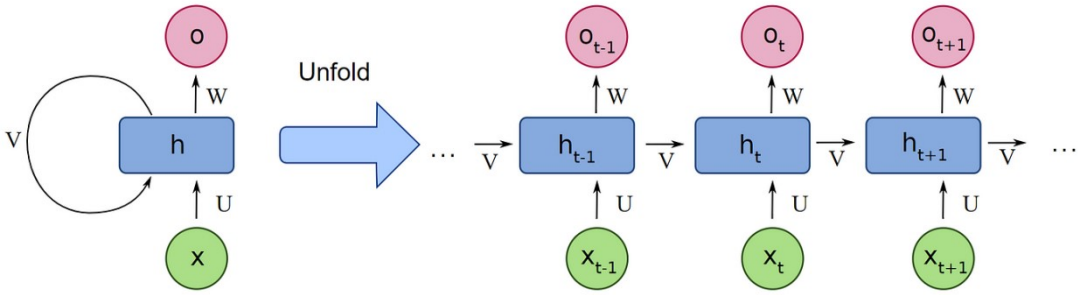


Figure 4.1 The folded and unfolded representations of the network. X is the input; O is the output; h is the main block of the RNN which contains the weights and the activation functions of the network; v represents the communication from one time-step to the other. The h block sends its output back to itself. It keeps doing that until it is told to stop.

The recognition of transportation modes presents an inherent recurrent challenge, making the utilization of RNNs a natural choice. One significant advantage of RNNs is their capacity to incorporate contextual information when mapping input and output sequences. However, a common issue arises with RNNs known as the "vanishing gradient problem" [124], where the sensitivity of initial inputs decreases over time as they traverse the network's recurrent connections, causing the network to forget crucial information. Early attempts in the 1990s aimed to address this problem, and one successful approach is the LSTM architecture [125].

LSTM is a specific type of RNN that enables the network to retain long-term dependencies from previous timesteps. It comprises interconnected memory cells,

forming recurrent subnets, which facilitate the storage and retrieval of information over extended periods. By combining straightforward deep neural network architectures with intelligent mechanisms, LSTM can effectively learn which historical information to remember and which to forget. The ability of LSTM to capture patterns in data across lengthy sequences makes it particularly suitable for time-series forecasting tasks.

As depicted in Figure 4.2, the memory cell within an LSTM network consists of three distinct components [126], the Gates which control the flow of information:

1. *The Forget Gate* $\mathrm{F(n)}$: determines which information should be retained in the memory cell by filtering out unnecessary information.
2. *The Input Node* $\mathrm{H(n)}$ and Input Gate $\mathrm{I(n)}$: the Input Node activates the respective state based on the output from the tanh, activation function. The Input Gate controls the significance of the hidden state for accurate transportation mode estimation by deciding which new information to store in the cell state.
3. *The Output Gate* $\mathrm{O(n)}$: regulates whether the current memory cell's response is significant enough to contribute to the next cell.

The Cell State denoted as C_t acts as a memory unit within the LSTM. It can retain information over long sequences, allowing LSTMs to capture long-term dependencies. The cell state can be updated or modified through a series of operations, allowing the network to learn which information to keep, forget, or add.

The Hidden State, denoted as H_t is the output of the LSTM cell at a specific time step t. It carries information about the current input, the cell state, and the operations performed by the LSTM cell. The hidden state is used to make predictions or can be passed to subsequent LSTM cells in a sequence.

Each gate operation involves a weighted sum of the input X_t and the previous hidden state H_{t-1}. These inputs are processed through activation functions (sigmoid for gates, tanh for input candidates) to produce gate outputs between 0 and 1. These outputs control the flow of information into and out of the cell state.

The updated cell state is then passed through the output gate, and an $\tanh$ activation function is applied to squash the values between -1 and 1. The resulting values are the hidden state H_t for the current time step.

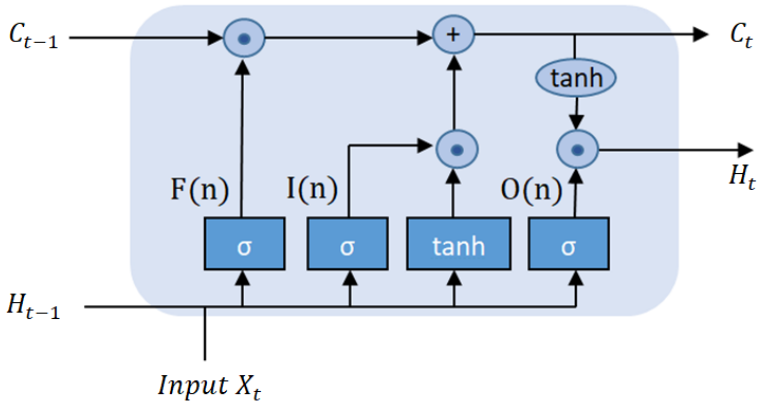


Figure 4.2 The memory cell within an LSTM network.

A limitation of the conventional LSTM memory cell is that it only processes previous state information, making it insufficient for modeling non-causal phenomena. Causality refers to the dependence of the system output (in this case, the current transportation mode) solely on past and present inputs, excluding future inputs, which may not always hold true. In such cases, bidirectional LSTM offers a promising alternative as it processes data in both forward and backward directions, capturing potential dependencies from "future" instances [127].

4.2 Bayesian Optimization

Regarding the selection of model parameters, in this research a probabilistic Bayesian framework was employed, through which the model configuration parameters were optimally tuned.

Assuming that a certain number of configuration parameters are available, such as the number of memory cells, the learning rates, etc., denoted as π_i , if we construct a set of Q different configurations - i.e., $D_{1:Q} = \{\pi_1 \dots \pi_Q\}$ - then we can evaluate the error $E(p, d, \pi)$ that the network gives when (1) it receives as inputs the data p, (2) the network output is compared against the desired (target) outputs d, and (3) a given π model configuration. In this context, we have omitted index n, since we refer to any time instance. Let us denote as E_{min} the minimum across all Q configurations. Then, an improvement function is given:

$$I(p, d, \pi) = max\{0, E_{min} - E(p, d, \pi)\} \quad (10)$$

In a probabilistic framework, we estimate:

$$Expect(I(p, d, \pi)) = Expect(max\{0, E_{min} - E(p, d, \pi)\}) \quad (11)$$

Equation (11) can be solved only by knowing the probability distribution of the error function given a set of configurations, i.e., $P(E|D_{1:Q})$. Based on the Bayesian rule, we have:

$$P(E|D_{1:Q}) \propto P(D_{1:Q}|E)P(E) \quad (12)$$

$P(E)$ generally follows a Gaussian distribution, and $P(E|D_{1:Q})$ is then expressed as a Gaussian process of mean $\mu(\pi)$ and standard deviation Σ [128].

$$\Sigma = \begin{bmatrix} k(\pi_1,\pi_1) & \cdots & k(\pi_1,\pi_Q) \\ \vdots & \ddots & \vdots \\ k(\pi_Q,\pi_1) & \cdots & k(\pi_Q,\pi_Q) \end{bmatrix} \quad (13)$$

In (13) $\mathrm{k}(\cdot)$ is a kernel function. The goal of the optimization is to find a new configuration $\pi^* \equiv \pi_{Q+1}$, which decreases the MSE or equivalently increases the improvement $I(p,d,\pi^*)$. For the augmented set $D_{1:Q+1}$ containing $\pi^* \equiv \pi_{Q+1}$, $P(D_{1:Q+1}|E)$ will again be a Gaussian process of standard deviation:

$$\begin{bmatrix} \Sigma & b \\ b^T & k(\pi_{Q+1},\pi_{Q+1}) \end{bmatrix} \quad (14)$$

where $b = [k(\pi_{Q+1},\pi_1) \dots k(\pi_{Q+1},\pi_Q)]$.

It can be proven [129] that $P(E_{Q+1}|D_{1:Q},\pi_{Q+1})$ is also Gaussian, with a mean value and standard deviation related to previous variables. Equation (2) can be used to compute the new configuration π^*, as the integral of $I(\cdot)$ and $P(E_{Q+1}|D_{1:Q},\pi_{Q+1})$, i.e., the probability that $I(\cdot)$ follows.

4.3 Experimental Setup

This section presents the performance analysis of the proposed Bayesian-optimized LSTM-based model. Firstly, a brief description of the datasets is provided, followed by data preparation and the analysis of the experimental results.

4.3.1 Dataset Description

The SHL dataset used in the experiments is a subset of the original Sussex-Huawei Locomotion–Transportation (SHL) dataset, which contains the data recorded from one participant's phone placed in their front trouser pocket, and includes a period from 1 March 2017 to 5 July 2017. For the analysis, eight main activities are considered: still, walk, run, bike, car, bus, train, and subway. SHL is a multivariate time-series dataset that contains 22 features representing measurements from 6 smartphone sensors: accelerometer, gyroscope, magnetometer, pressure sensor, GPS (altitude metrics) and temperature. Even though the number of participants was limited, the focus was on the quality of the collected and annotated data and on

collecting real-life data over a long period (2812 h of labeled data and 17,562 km of traveled distance collected over 7 months).

The data were used to frame a forecasting problem where, given the sensor measurements and mode of transport used previously, the mode of transport at the next time could be predicted.

4.3.2 Preliminary Data Analysis

The dataset was already a supervised learning problem with input and output variables. The first column represents the timestamp of the sample, in milliseconds, while the rest of the columns represent the x, y, and z measurements of the accelerometer (m/s^2), gyroscope (rad/s), magnetometer (μT), gravity (m/s^2), and linear acceleration (m/s^2), as well as the w, x, y, and z measurements of orientation, and the ambient pressure (hPa), altitude, and temperature. The mean values of features per class are shown in Figure 4.3.

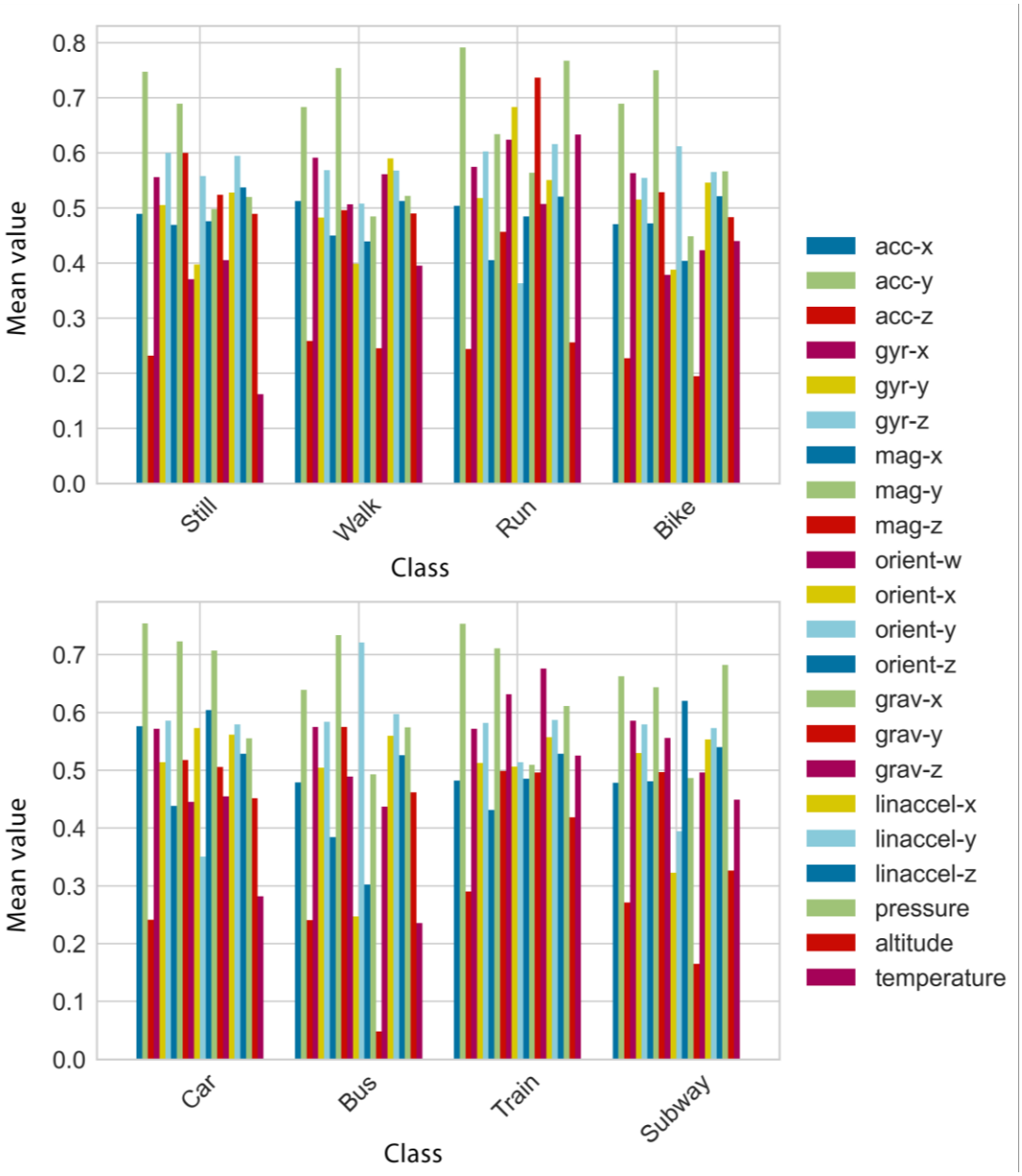


Figure 4.3 The mean values of features per class.

The correlation between and among the continuous variables, so as to better understand the underlying relationships in the data, is shown in a heat map that visualizes the correlation matrix, in Figure 4.4.

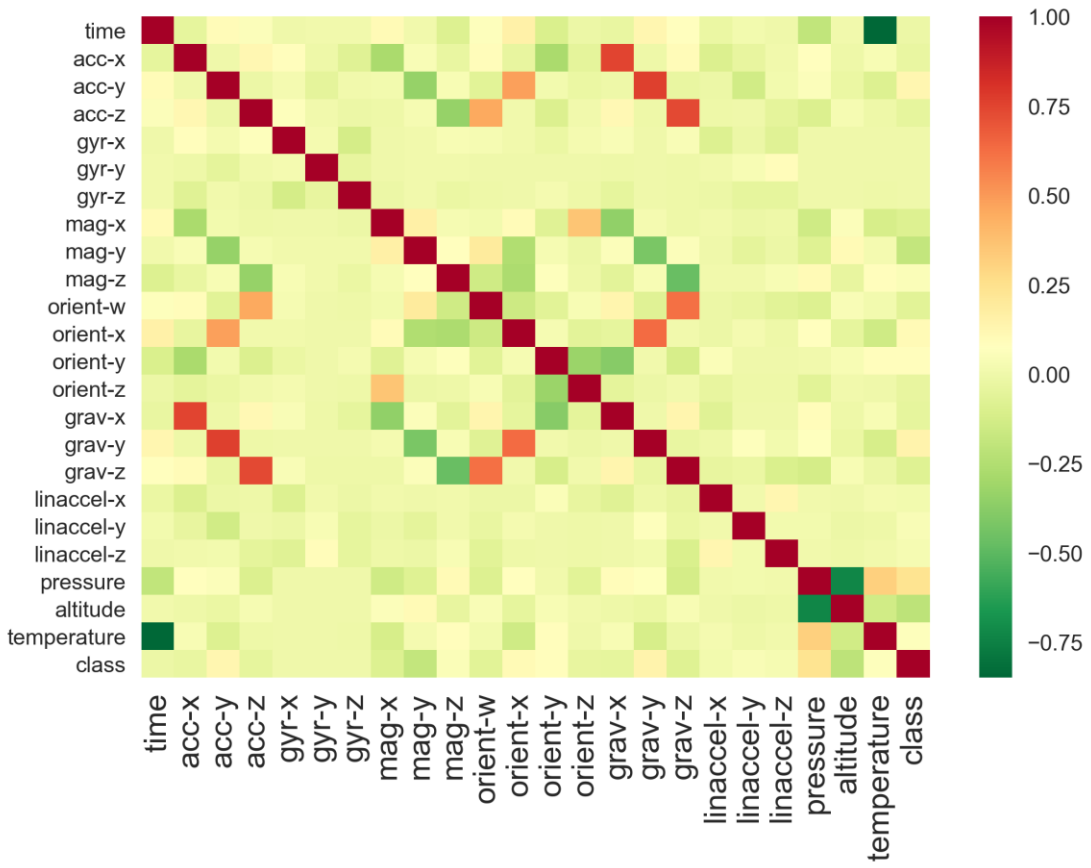


Figure 4.4 A heat map that visualizes the correlation matrix and shows the correlations between and among the continuous variables.

The highest positively correlated features are:

• Acceleration with gravity, which seems reasonable, because the accelerometer out-puts the acceleration of the device in three axes by measuring consequent forces applied to the device, and the force of gravity influences this measurement

• Orientation with gravity, because both are typically derived from the accelerometer.

The highest negatively correlated features are:

• Altitude with pressure

• Gravity with magnitude.

To get insight—not only on the distribution of a single machine learning variable, but also on the interrelationship between sensor parameters and their impact on each class—the dataset was represented with pair plots using the Seaborn visualization library.

The Kernel Density Estimation (KDE) diagrams on the diagonal allows to see the distribution of a single variable, while the scatterplots on the upper and lower triangles show the relationship (or lack thereof) between two variables. As depicted

in Figure 4.5, the pair plots show the relationships between all pairs of features per class and, in case of a linear relationship, denote the strength of this relationship. The more the dots scatter from the trend line, the weaker the relationship. Strong linear correlations exist between gravity and acceleration (A) and between altitude and pressure (B). Gravity-y and orientation-x (C), as well as gravity-z and orientation-w (C), have medium linear relationships. Gravity and magnitude (D) have a weak linear relationship. The large amount of overlap between classes in the univariate kernel density plots shows that neither feature of the pair alone is able to classify transportation modes very well. The classes are not well separated into clusters based on the sensor measurements. Thus, it becomes very difficult to distinguish one class from another.

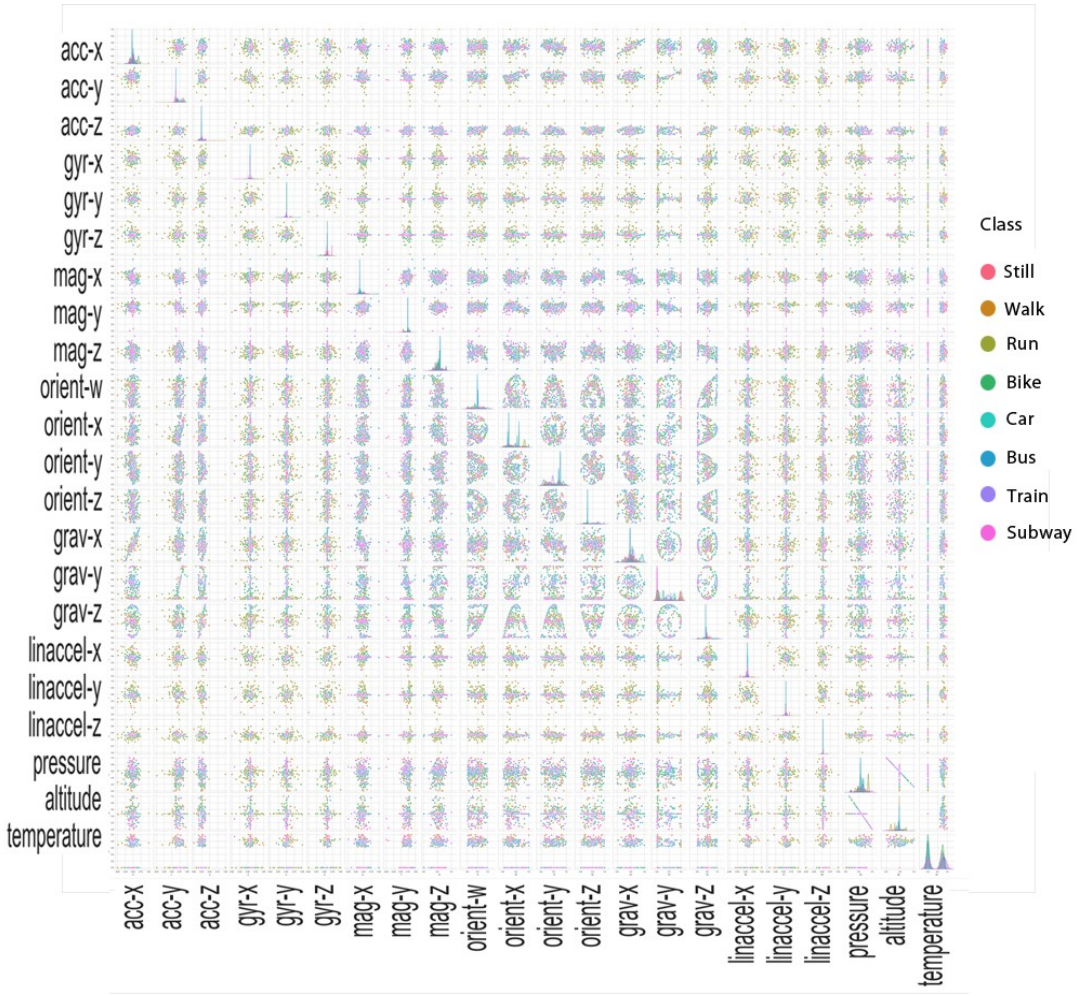


Figure 4.5 The pair plots show the relationships between all pairs of features per class and in case of a linear relationship, denote the strength of this relationship.

4.3.3 Data Preparation

The dataset was already a supervised learning problem with input and output variables. The number of samples is presented in Table 4.1.

In Figure 4.6 (I, II, III), the data preparation process is depicted. For a better performance, two data preprocessing techniques were employed in this work. The first step was to drop rows with class 0 (null class) and sort the column of time values so that the data were sorted by date. The original dataset described the daily sensor measurements for 7 months at various times per day. The dataset was resampled so as to be available at the same frequency that we wanted to make predictions. Down sampling decreased the frequency of the samples from milliseconds to minutes. This makes sense, as switching between transportation modes can be quite frequent.

The goal was to establish a multivariate LSTM prediction model. The problem was framed so that multiple, recent timesteps could be used to make the prediction for the next timestep. For example, given the current time (t), we wanted to predict the value at the next time in the sequence $(t+1)$, so we used the current time (t), as well as the one or more prior times$(t-1, t-2, \dots, t-n)$, as input variables. The input features were normalized, as there were differing scales in input values.

Class	Samples
Still	19,085
Walk	46,987
Run	39,814
Bike	43,988
Car	26,268
Bus	3861
Train	623
Subway	693

Table 4.1 Samples per class.

4.4 Proposed LSTM Model

First, the dataset was split into training and test sets (80% for training, 20% for testing), and then the training and test sets were split into input and output variables. As depicted in Figure 4.6 (IV-a), two approaches were made for constructing the final prediction model. Firstly, the data were fed to LSTM as a whole set of the original 22 features, and then a dimensionality reduction algorithm was used before applying LSTM to the data so as to reduce the dimensions of the training data and positively affect the performance of the model. The inputs were reshaped into the 3D format expected by LSTM. The LSTM model selected through the Bayesian optimization process was defined with 64 neurons in the first hidden

layer, and an output layer with a softmax activation function and 9 output values. Only one layer of LSTM was used, but different numbers of LSTM cells—such as 128, 256, 512, and 1024, which are the most used numbers in the literature—were tested. Categorical cross-entropy was used as a method for error calculation and, in order to update the weights of our neural network, the Adam optimizer was used, with a learning rate of 0.001. Additionally, dropout was implemented to randomly drop 20% of units from the network. To reduce underfitting and improve model performance, the network was fitted for a different number of epochs and various batch sizes, while a 10-fold cross-validation was used to overcome the limitation of dataset size and prevent overfitting.

Each experimental scenario was run 10 times with the same parameters so as to obtain an average outcome. Various tests were conducted concerning resampling, LSTM model (number of cells, dropout), epochs, and batch size variances.

The model learned better after each epoch. However, it started to memorize with the increasing number of epochs. As the batch size value decreased, the complication of the model increased and gave better results, but after a certain time there was no significant improvement.

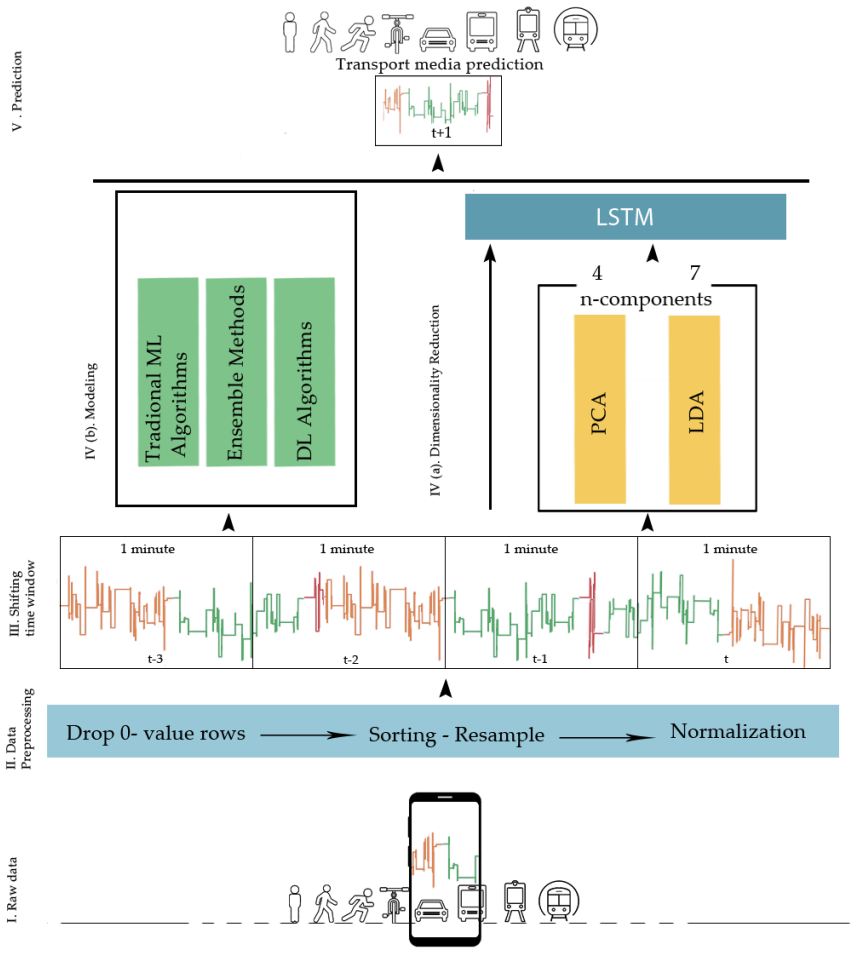


Figure 4.6 The proposed LSTM model structure.

4.5 Experimental Results

In this subsection, we evaluate the performance of the proposed optimized LSTM model. Several traditional machine learning algorithms, ensemble methods, and deep learning models are used to determine transportation modes as a benchmark to the LSTM model, as shown in Figure 4.6 (IV-b):

- Traditional machine learning algorithms: Logistic Regression (LR), Linear Discriminant Analysis (LDA), k-Nearest Neighbor (kNN), Classification and Regression Tree (CART), Naive Bayes (NB), Multilayer Perceptrons (MLPs);
- Ensemble algorithms: Random Forest (RF, bagging algorithm), AdaBoost, XGBoost, (boosting algorithms), Bagging (bootstrap aggregating), Extra Trees (extremely randomized trees), Voting (hard or soft voting);
- Deep learning algorithms: Convolutional Neural Network (CNN), Bidirectional Long Short-Term Memory networks (Bi-LSTM).

The hyperparameters of all models were optimized, and are summarized in Table 4.2, Table 4.3 and Table 4.4.

	LR	LDA	k-NN	NN	RF	CART
Multiclass	One-vs-rest	-	-	-	-	-
K value			5	-	-	-
Classifier	-	-	-	MLP	-	-
Layers				100 hidden		
n_estimators	-	-	-	-	20	-
Max depth	-	-	-	-	5	-
Max tree depth	-	-	-	-	-	None
Solver	-	Singular value decomposition (svd)	-	-	-	-

Table 4.2 The parameters of traditional ML algorithms.

	Ada-Boost	Bagging	Extra Trees	XG Boosting	Voting
n_estimators	1000	150	150		-
Base estimators	Random forest classifier	Decision tree classifier	-		RF, SVC, DT
Number of trees	-	500	-		-
Max_features	-	-	5		-
RF n-estimators	-	-	-		50
SVC parameters	-	-	-		default

Table 4.3 The parameters of ensemble methods.

	CNN	Bi-LSTM	LSTM
Conv1D layer	2	-	-
Activation	Relu	-	-
Layers	MaxPooling Flatten	-	-
Number of neurons in	8	8	64

the first layer			
Output values	8	8	8
Optimizer	Adam	Adam	Adam
Dropout	0.1	0.1	0.2
Learning rate	0.001	0.001	0.001
Error calculation	Categorical cross-entropy	Categorical cross-entropy	Categorical cross-entropy
Dense layer	2 (50,8)	1(8)	1 (8)
Activation	ReLU, softmax	Softmax	Softmax

Table 4.4 The parameters of DL methods.

In order to evaluate the performance of different classification models, among several metrics used in this study - i.e., accuracy, precision, recall, F1-score, and confusion matrix, weighted F1-score was selected to be the most representative, as it computes the F1-score for each label, and returns the average, considering the proportion of each label in the dataset.

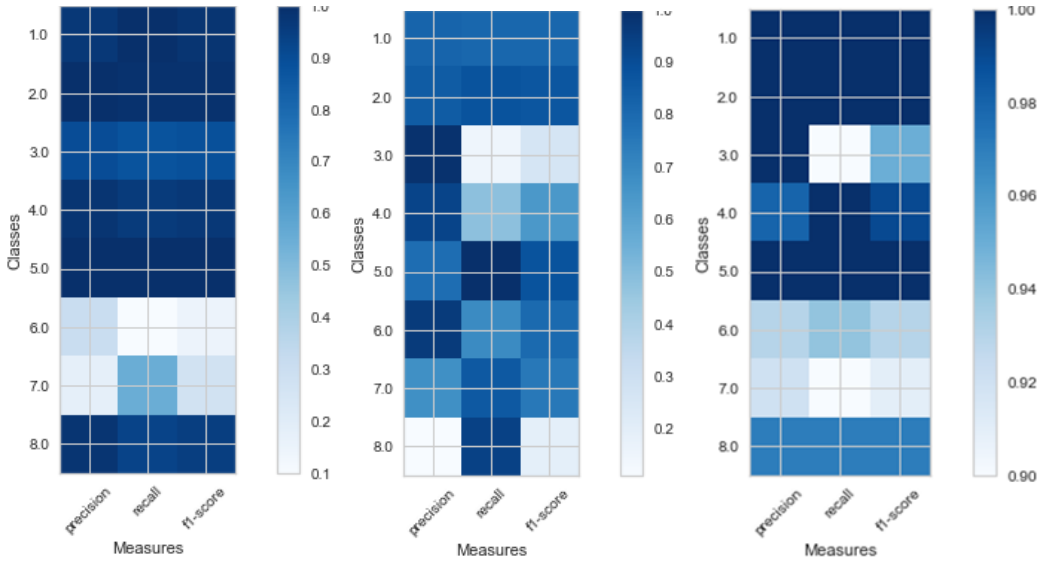
For forecasting, three different cases were used, i.e., 1-before, 2-before, and 3-before, meaning that 1, 2, and 3 min before were taken into consideration, respectively, so as to predict the transportation media used in the next minute.

As presented in Table 4.5, all traditional ML algorithms had similar F1-score value in all three cases, apart from ANN, which performed better when prediction took 3 previous minutes into consideration. K-NN performed best, achieving 98% in all three cases.

	LR	LDA	k-NN	CART	NB	ANN	RF
1-before	70	87	98	92	92	79	87
2-before	68	87	98	91	92	86	88
3-before	68	87	98	92	89	98	87

Table 4.5 F1-scores for cases 1-, 2-, and 3-before of applying traditional ML algorithms.

As Figure 4.7 shows, k-NN classified all classes the best, whereas ANN showed a better prediction in the case of 3-before for all classes, except for run and subway.



(a) (b) (c)

Figure 4.7 Classification reports for (a) k-NN in the 1-before case, (b) ANN in the 3-before case and (c) ANN in the 1-before case.

The ensemble methods performed better than the traditional ML algorithms, as presented in Table 4.6. AdaBoost in the case of 3-before, and hard voting in the 1-before and 2-before cases, reached 98% and 99% F1-score, respectively.

	AdaBoost (RF)	Bagging	Extra Trees	XG Boosting	Hard Voting
1-before	62	92	53	98	98
2-before	69	92	50	98	99
3-before	99	92	82	98	95

Table 4.6 F1-score for cases 1-, 2-, and 3-before of applying traditional ensemble methods.

The ensemble methods that achieved less than 70% F1-score had difficulty in accurately predicting the car and bike classes, as presented for example in Figure 4.8 (a, b), where AdaBoost and extra trees in the 1-before case prediction per class are presented. In Figure 4.8 (c, d), the same results are shown for the two cases that achieved the best results—hard voting in the 2-before case, and AdaBoost in the 3-before case—where the prediction was almost perfect. DL methods performed as shown in Figure 4.9. Bi-LSTM performed well, achieving a 98% F1-score only in the 1-before case, whereas LSTM in both the 1-before and 3-before cases reached 99%. In Figure 4.10, the LSTM F1-scores for each class are presented. All classes were predicted better in the 1-before case and, in general, among all classes, the motorized media were more difficult to predict accurately—especially in the 2-before case.

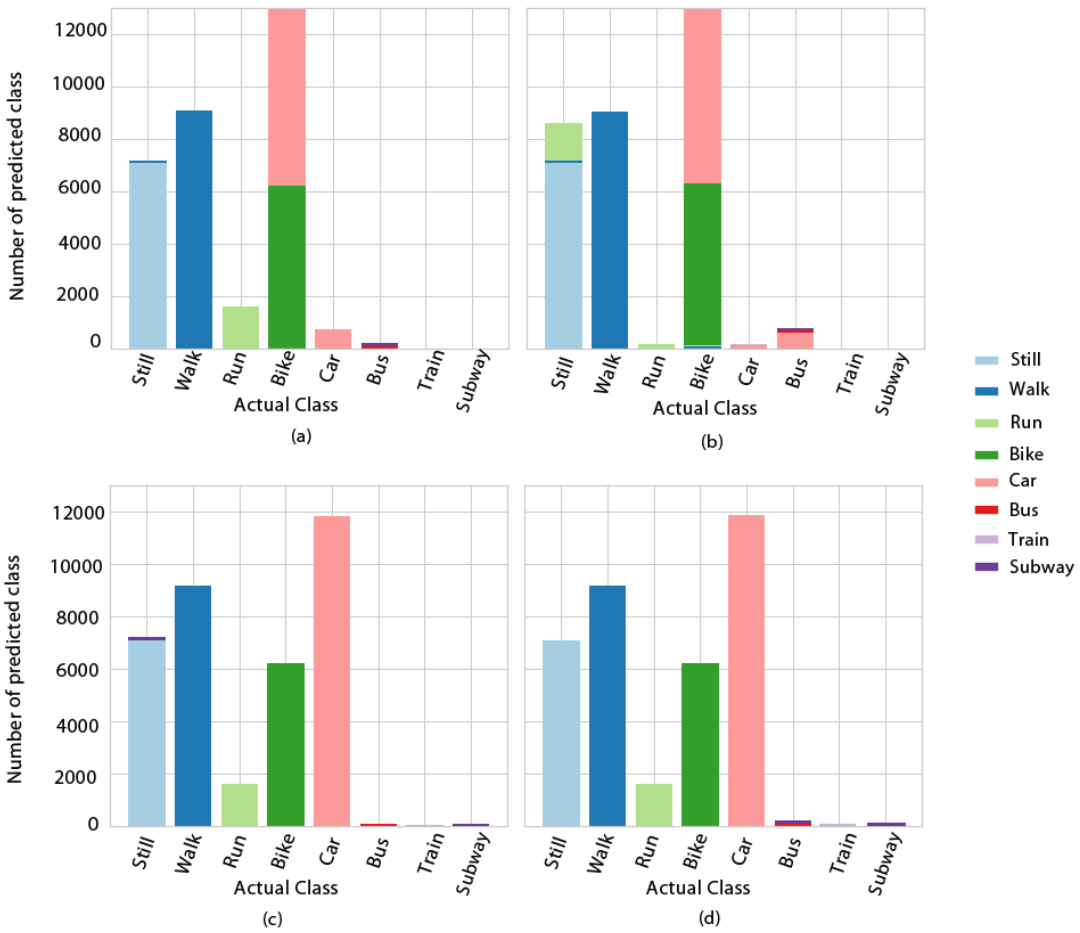


Figure 4.8 Class prediction error for (a) AdaBoost (random forest) in the 1-before case; (b) extra trees in the 1-before case; (c) hard voting in the 2-before case; and (d) AdaBoost (random forest) in the 3-before case.

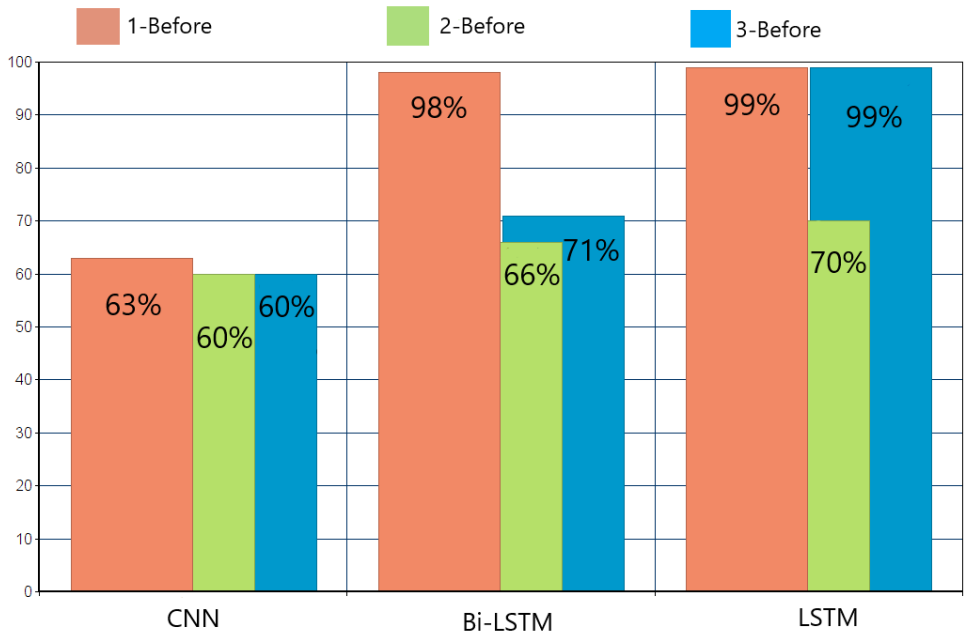


Figure 4.9 Deep learning algorithms' F1-scores in the 1-, 2-, and 3-before cases.

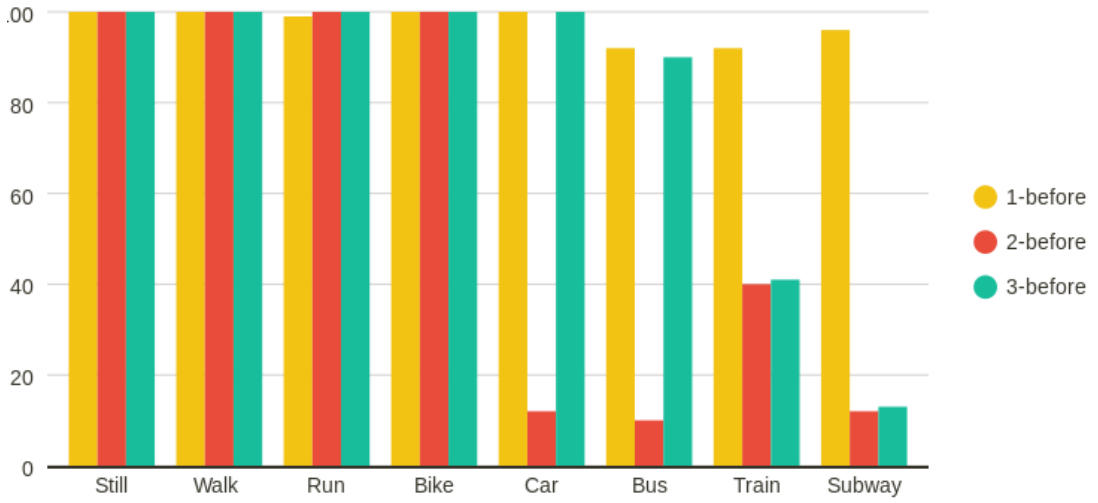
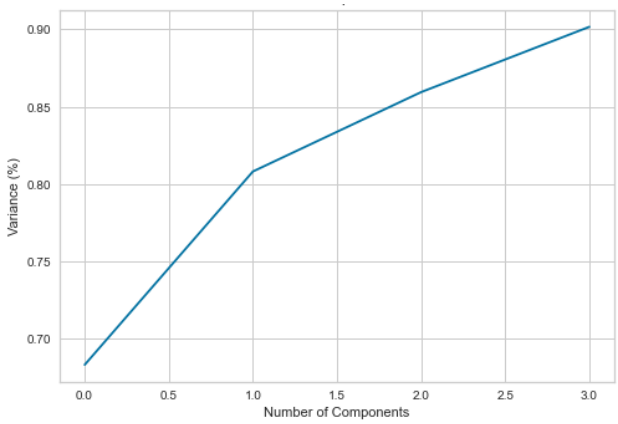


Figure 4.10 LSTM's F1-score for each class in the 1-, 2-, and 3-before cases.

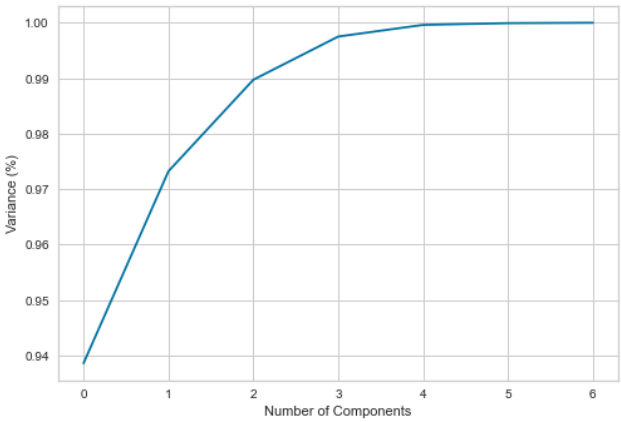
The training times for each algorithm achieving an F1-score of more than 95% are presented in Table 4.7. XGBoost was the fastest, followed by the deep learning algorithms—namely, Bi-LSTM and LSTM—which, despite their complex structures, performed faster than the others in all three cases.

Since in our research, the features obtained from mobile phone sensors have no physical meaning and a tangible significance so that there is a potential loss of vital information, Feature extraction was chosen for dimensionality reduction. To reduce the number of input features, two dimensionality reduction techniques were applied prior to the LSTM model: Principal Component Analysis (PCA), and Linear Discriminant Analysis (LDA). Several n-components values were tested to accomplish

the best performance of the model. According to Figure 4.11, for the implementation of PCA, four principal components were selected so as to preserve over 90% of the total variance of the data, and seven principal components were finally selected for LDA.



(a)



(b)

Figure 4.11 Number of principal components for the (a) PCA algorithm and (b) LDA algorithm.

	1-Before	2-Before	3-Before
kNN	128	323	346
ANN			2264
AdaBoost (RF)			172
Bagging	230	453	704
XGBoost	33	53	78
Hard voting	433	761	
Bi-LSTM	108		
LSTM	107		500

Table 4.7 Training time (in seconds) for the algorithms achieving F1-score of more than 95% in the 1-, 2-, and 3-before cases.

In Table 4.8, the results of these two techniques are presented concerning F1-score and accuracy metrics. It is clear that PCA had better results than LDA, and made LSTM perform better in all three cases. PCA worked best in that direction, achieving an increase in F1-score from 99.5 to 99.8%. In the 3D scatterplot in Figure 4.12, we see that PCA's three components hold more information than LDA's, especially for specific classes—i.e., still, walk, run, car—but clearly not enough to set all of them apart.

	PCA-LSTM	LDA-LSTM
1-before	99.8/99.5 (4 n-components)	70/68 (7 n-components)
2-before	99.7/99.7 (5 n-components)	97/99 (7 n-components)
3-before	99.5/99.5 (15 n-components)	97/99.7 (7 n-components)

Table 4.8 F1-score/accuracy metrics and number of n-components in the 1-, 2-, and 3-before cases for LSTM after applying PCA and LDA algorithms.

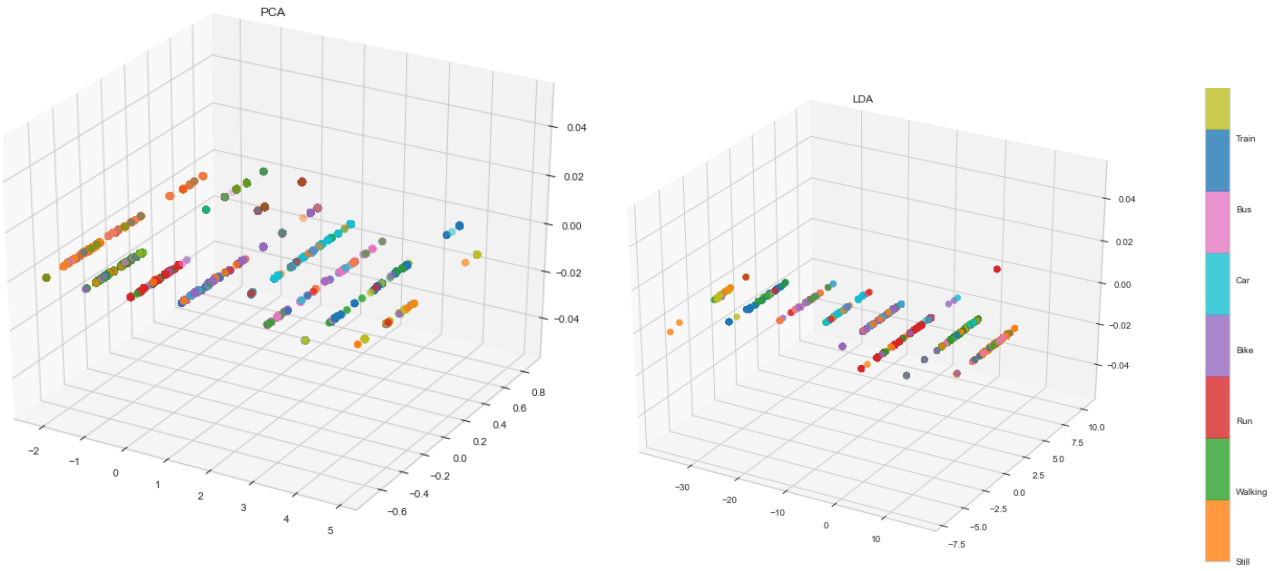
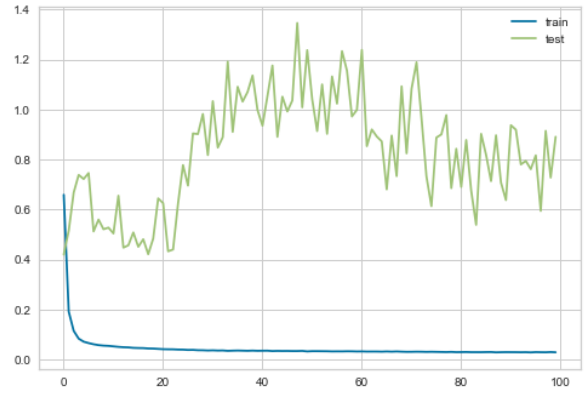
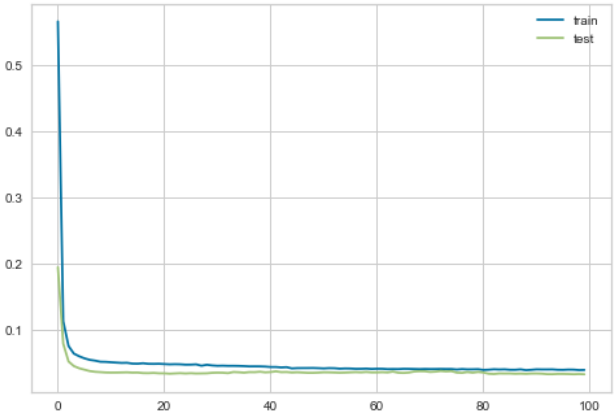


Figure 4.12 Visualization of the first three principal components of PCA in the 1-before case.

A plot of learning curves is shown in Figure 4.13, indicating loss for the 2-before case, both before and after applying PCA. With PCA, the training and validation loss decreased to a point of stability, with a minimal generalization gap between the two final loss values, showing a well-fitted model.



(a)



(b)

Figure 4.13 Learning curves indicating loss for the 2-before case, (a) before and (b) after applying PCA.

Figure 4.14 depicts LSTM's performance before and after reducing the dimensionality of the data. With PCA, the performance was improved in the 2-before and 3-before cases, but in 1-before it decreased by ~0.3%.

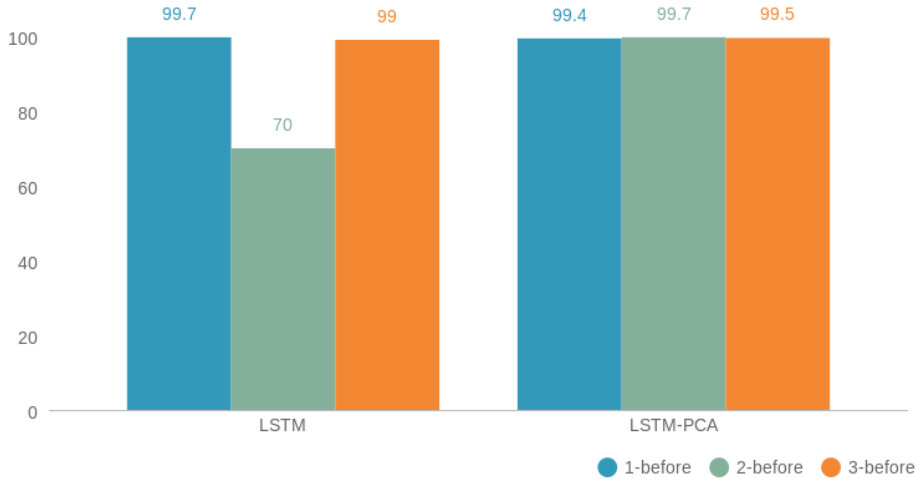


Figure 4.14 LSTM's performance before and after applying PCA for the 1-, 2-, and 3-before cases.

In Figure 4.15, the blue line represents the average values of the actual transportation media used for all 1-, 2-, and 3-before cases, while the yellow line represents the predicted values. It is clear that LSTM was able to capture the overall trend.

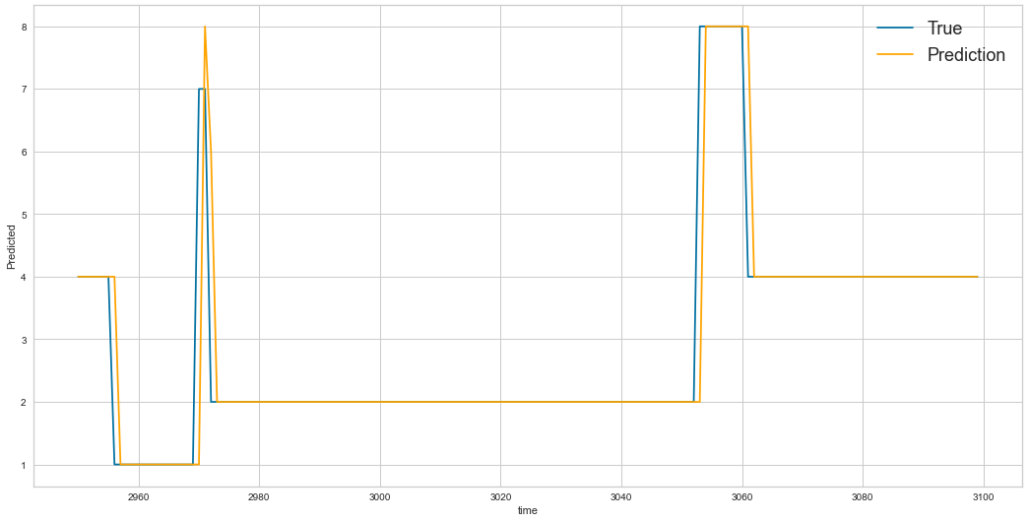


Figure 4.15 Predicted and actual average values of all 1-, 2-, and 3-before cases for LSTM after applying the PCA algorithm.

The classes predicted before and after applying PCA are presented in Figure 4.16. In the 1-before case, the train and subway classes were predicted worse after dimensionality reduction, whereas with PCA all classes were predicted better in the other two cases.

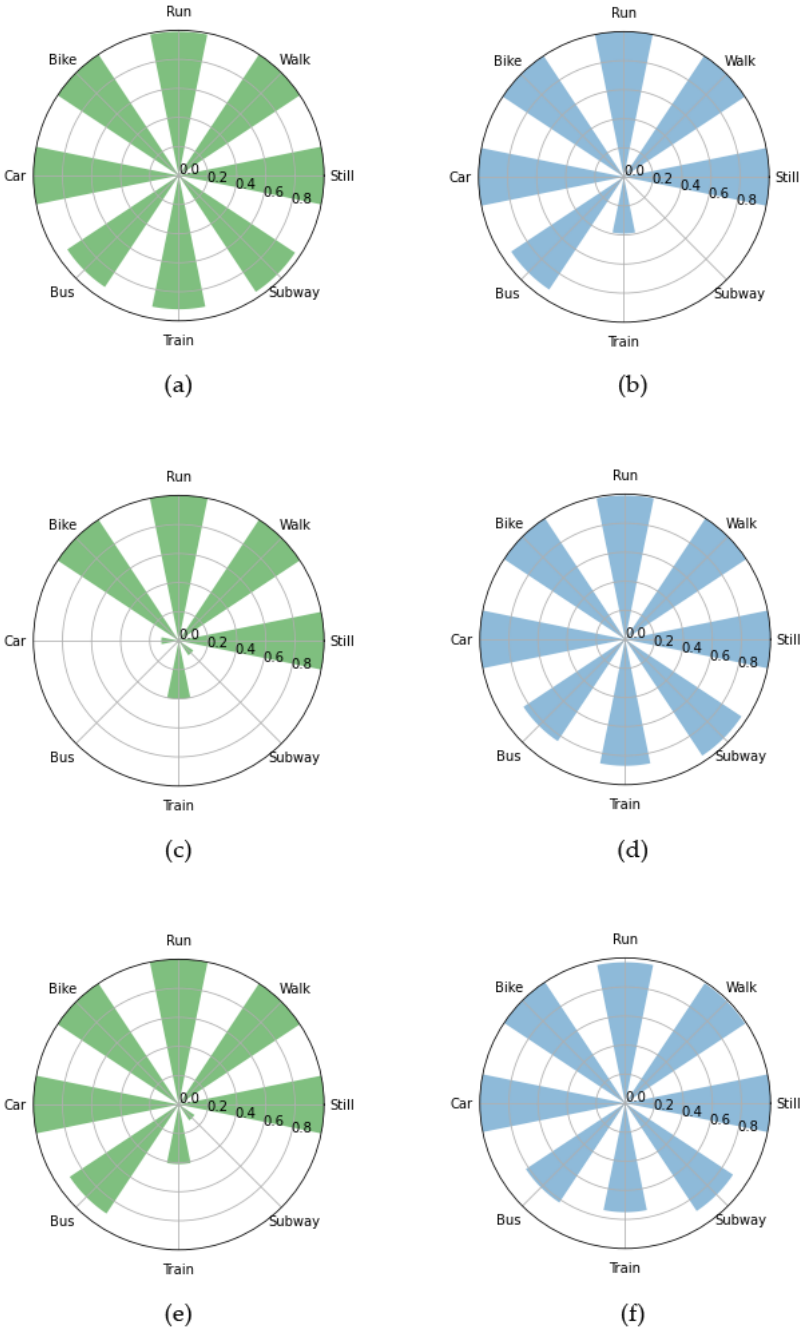
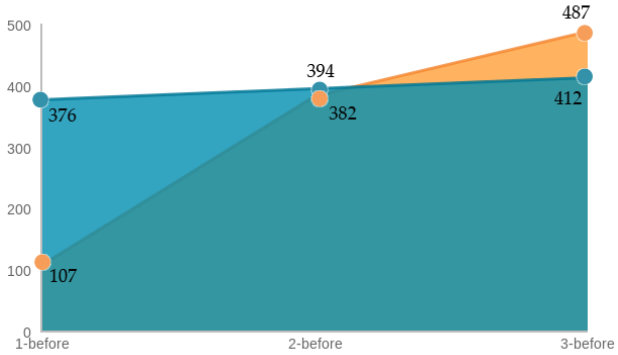
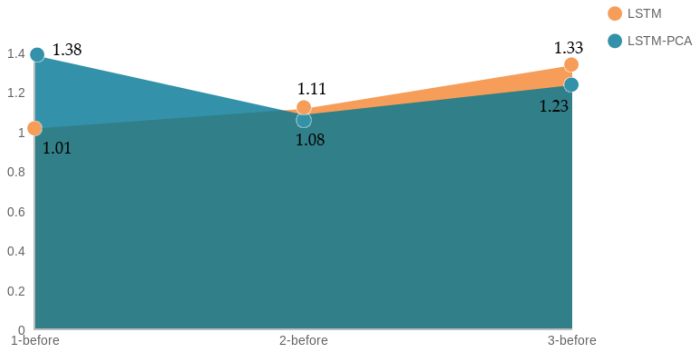


Figure 4.16 The classes predicted before (a, c, e) and after (b, d, f) applying PCA for the 1-, 2-, and 3-before cases.

In terms of time, according to Figure 4.17, LSTM–PCA had an average training time of 394 s in all three cases, while LSTM without dimensionality reduction had an average training time of 325 s. Both models' testing times had similar average values, at 1.23 and 1.15 s, respectively.



(a)



(b)

Figure 4.17 (a) Training time and (b) prediction time before and after applying PCA to LSTM.

4.6 Conclusion

In this work, the effectiveness of several machine learning methods used for Transportation Mode Detection based on multimodal smartphone sensor data was scrutinized. The proposed Bayesian-optimized LSTM model can recognize eight transportation modes with 99.7% accuracy and 99.8% F1-Score, and significantly outperforms other state-of-the-art methods. Despite the good results achieved in this study, there were also some limitations. Firstly, the limited number of annotated data—especially for specific classes— resulted in the model finding it difficult to distinguish certain classes, such as bus, train and subway even in the 1-before case, as illustrated in Figure 4.10.

Chapter 5

Transformer-based model for Transportation Mode Detection

5.1 Attention and Transformers

The basic transformer network model comprises a multi-head attention module and a position-wise feed-forward network. The normalized input sequence enters i the multi-head attention layer, which computes attention scores. These are then passed on to a position-wise feed-forward layer.

Attention mechanism in transformers is affected as a Query-Key-Value (QKV) model. Attention is composed of a series of linear transformations which analyze input sequences in an order-invariant fashion, calculating and allocating importance weights to each spot in the sequence. Single-head dot-product attention mechanism therefore applies linear transformations to the input signal to form query (Q), key (K) and value (V) matrices.

If $x \in \mathbb{R}^{s_b \times s_l}$ is the input sequence, where s_b is the batch size and s_l the input length, then the linear transformations can be represented by the matrices: $W_q \in \mathbb{R}^{s_l \times s_q}$, $W_k \in \mathbb{R}^{s_l \times s_k}$ and $W_v \in \mathbb{R}^{s_l \times s_v}$. Then $Q = W_q^T x$, $K = W_k^T x$, $V = W_v^T x$. Allowing weight matrices to have the same dimensions for easier matrix computations, $s_q = s_k = s_v$, the single-head dot product attention A is a matrix multiplication of Q, K and V following a scaling and softmax operation.

$$A(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{s_k}}\right)V \tag{15}$$

The first term in the above equation can be seen as the weighting of values at all locations of the sequence. Attention thus can inherently comprehend which parts of the sequence are significant to estimate the output and overlook parts that are not significant. This attribute is especially useful in the cases of imbalanced datasets since the respective weight for negative samples can automatically be set to a small value.

Futher, transformers deploy a multi-head attention mechanism, rather than simply applying a single attention function. Multi-Head Attention is computed by extending the single-head attention mechanism to *h* dimensions (multiple heads) by appending the single-head attention outputs, followed by a linear layer.

$$\text{MultiHead Attention} = \text{Concat}\left(A(Q_i, K_i, V_i)\right), \quad i = 1, \dots, h \tag{16}$$

The normalized attention scores are then fed to a Position-wise Feed-Forward layer (PFFN) that performs linear transformations with Gaussian Error Linear Units (GELU) activation function. The linear transformations are applied to each position

individually and identically; in other words, the transformations utilize the same parameters for all positions of a sequence and differentiated parameters across layers.

5.2 TMD-BERT model

In this work we used BERT to train a text classifier and built a model that was fine-tuned on our data in order to produce state of the art predictions. Specifically, an untrained layer of neurons was added to the end of the pre-trained BERT model and then the new model was trained for our classification task. The selection of a pre-trained model rather than developing a specific deep learning model (LSTM, CNN, etc.) that is suitable for the specific task is justified by the fact that developing such complex models requires high-performance computing resources so by using a pre-trained model as base, it makes possible to develop high-performance models and solve complicated problems efficiently.

An overview of TMD-BERT model is presented in Figure 5.1. More specifically, the implementation of the model consisted of four phases.

I. **Time data from sensors**: the dataset, a series of time data sensors, was already a supervised learning problem with input and output variables.
II. **Data Preprocessing**: The data preparation process included (a) Resampling so as the data would be available at the same wanted to make predictions frequency, (b) down sampling in order to decrease the frequency of the samples from milliseconds to minutes and (c) normalization, by rescaling real-valued numeric attributes into a 0 to 1 range to make model training less sensitive to the scale of features. The goal was to establish a multivariate prediction model so that multiple, recent timesteps could be used to make the prediction for the next timestep. The preprocessing phase completed in two schemas, one with and one without Dimensionality Reduction with Principal Component Analysis (PCA), in order to investigate its effect on the model performance.
III. **Bert model**: Because the labels were imbalanced, we splitted the data set using for each class 85% for training and 15% for testing. The inputs were tokenized using BERT tokenizer by instantiating a pre-trained BERT model configuration to encode the train and test data.
IV. **Prediction**: An output of increased detection and prediction accuracy was predicted.

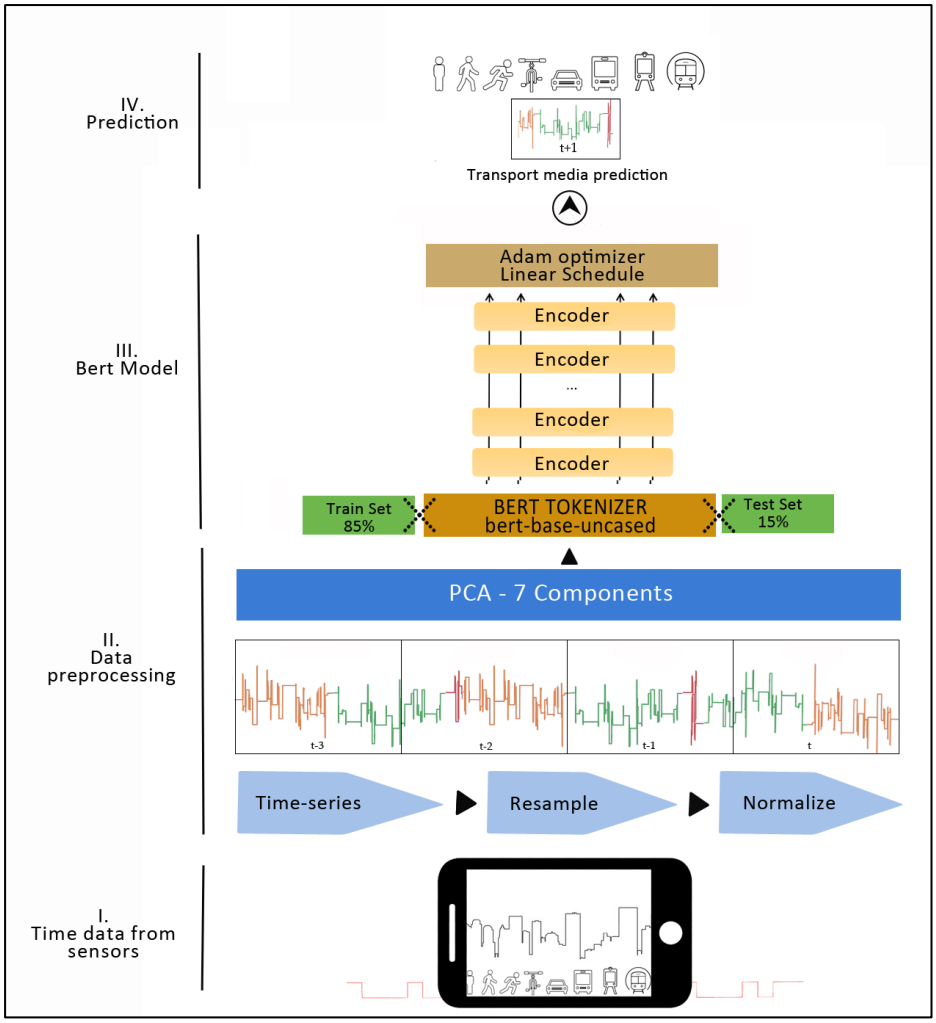


Figure 5.1 The model structure.

More analytically, the maximum length of the sentences was 256. There are many variants of pretrained BERT models. In this work, bert-base-uncased was used. The BERTBase model uses 12 layers of transformers block with 768-hidden layers, 12 self-attention heads and around 110M trainable parameters. It is trained on lower-cased English text. We used BERT architecture with also adding a linear layer for predicting output. The last hidden layer which corresponds to the [CLS] token was imported to the output layer of size num_classes which is eight here. Hence, we set the flag do_lower_case to true in BertTokenizer. The input to the BERT model was the set of measurements for each timestamp that was treated as a sequence so that one sequence would be classified to one of the 8 labels. BertForSequenceClassification is the Bert Model transformer with a sequence classification/ regression head on top (a linear layer on top of the pooled output). The model was tested for various batch sizes and different number of epochs so as to avoid any possible underfitting and improve the overall model performance. One epoch was finally chosen among several numbers of epochs tested, as there was not significant performance improvement in comparison with time execution. We used batch size 1 as larger batch size resulted in CPU memory overflow. We used a

Random sampler just for the training set, so as the model is more representative of the overall data distribution. For validation set, balancing batches is not an issue, so Sequential sampler was used. We also used the Adam optimizer. According to [130], Adam is "computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/parameters". The learning rate was initially set to $1e{-}5$ and epsilon is a very small number to prevent any division by zero in the implementation. Get_linear_scheduler_with_warmup creates a schedule with a learning rate that after a warmup period during which it increases linearly from 0 to the learning rate set in the optimizer, it then starts to decrease linearly to 0. The hyper-parameters of the model are summarized in Table 5.1.

	TMD-BERT model
DataLoader	RandomSampler (for train data) SequentialSampler (for test data)
Batch size	1
epochs	1
Optimizer	Adam
Learning rate	1e-5
epsilon	1e-8
Scheduler	get_linear_scheduler_with_warmup

Table 5.1 The parameters of TMD-BERT model.

For the reason that BERT requires inputs in a specific format, for each tokenized input sentence, there were created:

- Input ids which are a sequence of integers that represent each input token in the vocabulary of BERT tokenizer.
- Attention mask which is a sequence of 0 for padding and 1 for input tokens.
- Labels which are a sequence of 0 and 1 that represent the 8 labels of transportation modes.

In Figure 5.2 the fine-tuning procedure for sequence classification tasks is visualized. The final hidden state of the [CLS] token which is taken as the input sequence, is imported to the classification layer. That has a dimension of 8 (number of labels) x H (size of the hidden state). In the Figure 5.2, E represents the input embedding, Ti

represents the contextual representation of token I and [CLS] is a special symbol that represents the classification output.

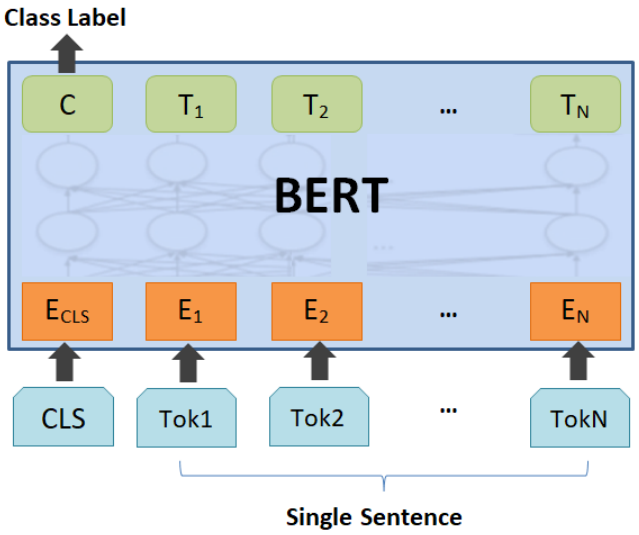
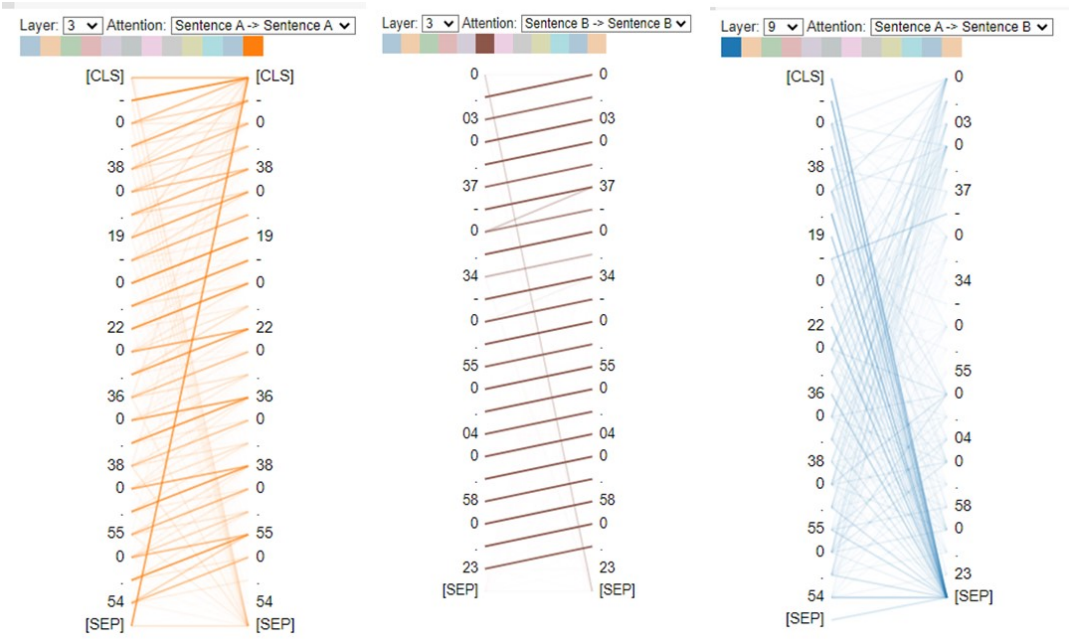


Figure 5.2 The fine-tuning procedure for sequence classification tasks.

As mentioned before, the attention mechanism allows the model to comprehend the relation between words taking into consideration the content of the sentence. Attention provides us with a way in which we can see how BERT forms complex representations to understand language. Using BertViz [131], an interactive tool developed to visualize attention in BERT from multiple perspectives, Figure 5.3 depicts the attention resulting from an input text from our dataset, "-0.38 0.19 -0.22 0.36 0.38 0.55 0.54" and "0.03 0.37 -0.34 -0.55 0.04 0.58 0.23", with the values representing the seven PCA principal components on particular timestamps. Self-attention is represented by colored lines depending on the color that corresponds to each attention head. These lines connect the tokens that are attending (left) with the tokens being attended to (right). The line weight reflects the attention score. For example, in Figure 5.3, for different layers and attention heads, in (a) each word seems to attend to more than one previous words with attention distributed in a slightly uneven way across previous words in the sentence. In contrast, in (b) each word seems to attend strongly to the immediately preceding word in the sentence. Finally in (c) there is a between-sentence connection without a specific pattern for this sentences case. The fact that the attention head focuses on the [SEP] token indicates that it cannot find anything else in the input sentence to focus on.



(a) (b) (c)

Figure 5.3 (a) Attention for the input text "-0.38 0.19 -0.22 0.36 0.38 0.55 0.54" with attention head 11 (orange) and layer 3 selected; (b) Attention for the input text "0.03 0.37 -0.34 -0.55 0.04 0.58 0.23" with attention head 5 (brown) and layer 3 selected; (c) Attention for the input text "-0.38 0.19 -0.22 0.36 0.38 0.55 0.54" and "0.03 0.37 -0.34 -0.55 0.04 0.58 0.23" with attention head 0 (blue), layer 9 and Sentence A → Sentence B selected.

The above visualizations show a mechanism of attention within the model, but BERT learns multiple mechanisms of attention, i.e. heads, that operate alongside with each other. In contrast with single-attention-mechanism, multi-head attention has the benefit that the model can capture a more extensive range of word relations. Figure 5.4 shows the attention in all the heads at once time for the same input text as before (i.e., "-0.38 0.19 -0.22 0.36 0.38 0.55 0.54" and "0.03 0.37 -0.34 -0.55 0.04 0.58 0.23"). Each cell shows the attention pattern for a specific head (column) in a particular layer (row). From this visualization, we can see that BERT produces a rich array of attention patterns.

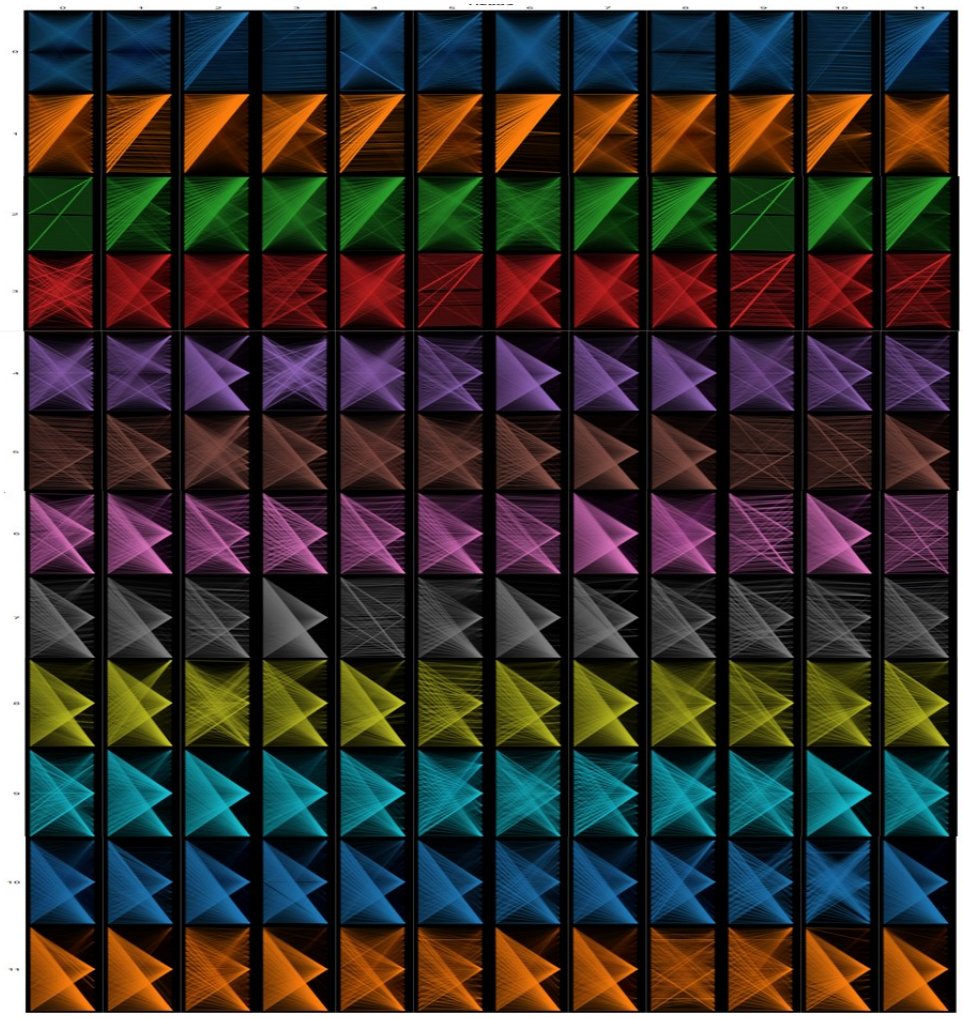


Figure 5.4 The attention in all the heads for the input text "-0.38 0.19 -0.22 0.36 0.38 0.55 0.54" and "0.03 0.37 -0.34 -0.55 0.04 0.58 0.23".

5.3 Experimental setup

In this section, the performance analysis of the proposed TMD-BERT model is presented. At first, there is a brief description of the dataset, then the data preparation is presented and finally the analysis of the results follows.

5.3.1 Dataset Description

The dataset exploited in the experiments of this work [92], is a part of the initial Sussex-Huawei Locomotion-Transportation (SHL) dataset. It comprises 68 days of recording during a period from 1/3/2017 to 5/7/2017 and it was collected from one participant's phone sensors. The phone was in his trousers front pocket. The participant used a variety of 8 transportation modes: Still, Walk, Run, Bike, Car, Bus, Train, and Subway. The measurements are derived from 4 smartphone sensors: magnetometer, accelerometer, gyroscope and pressure sensor which measures

temperature, pressure and altitude. The data set consists of 181319 timestamps. The goal was to frame a forecasting problem for predicting the transportation mode that will be used at the next timestep, given the sensor measurements and transportation mode used in past, recent timesteps.

5.3.2 Preliminary Data Analysis

The dataset used in the experiments was a supervised learning problem that had input and output variables (labels). The features values per class are shown in

Table 5.2.

Class	Samples
Still	19085
Walk	46987
Run	39814
Bike	43988
Car	26268
Bus	3861
Train	623
Subway	693

Table 5.2 Samples per class.

The fact that certain class categories comprise a larger proportion of the dataset than others, is not an issue in this work firstly because none of the transportation modes is more significant than others, secondly Bert seems to handle imbalanced classification well, thus removing the need to use standard data augmentation methods so as to limit this problem of imbalance [132] and thirdly, as it appears from the results all classes are predicted regardless of the samples number for each class.

Figure 5.5 shows the distribution of all sensor measurements, including the mean value of the 3 or 4 axis of acceleration, magnitude, orientation, gravity and linear acceleration, as long as the temperature, pressure and altitude values. The same features were used to calculate correlation in pairs.

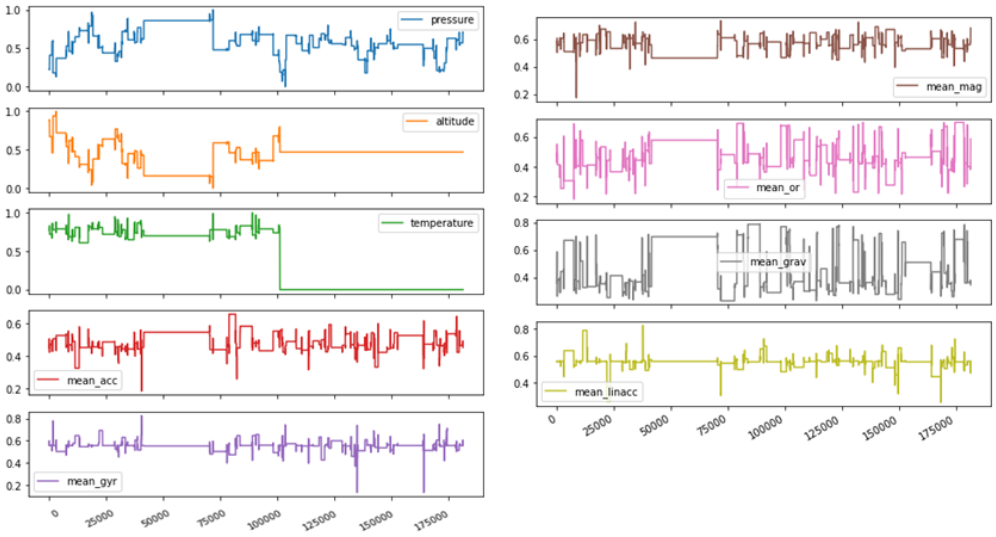


Figure 5.5 The distribution of sensor measurements.

The results of the correlation analysis were calculated using Pearson's Correlation Coefficient (PCC) [133] and are shown in Figure 5.6. The PCC appears above the scatterplot for every pair of features. It shows the linear relationship between two sets of data by returning a value of between -1 and +1. The highest positively correlated pairs of features are acceleration – gravity (ρ=0.87) and the highest negatively ones are altitude and pressure (ρ=-0.85), as long as gravity and magnitude (ρ=-0.77). Most of the other pairs of features have a low percentage of interaction. In order to additionally visualize the insights of the dataset, not only on the interrelationship between sensor parameters but on the single machine learning variable distribution, too, the diagonal of the pair plot diagram shows the distribution of each single variable whereas the lower triangle depicts the probability distribution of one with respect to the other values.

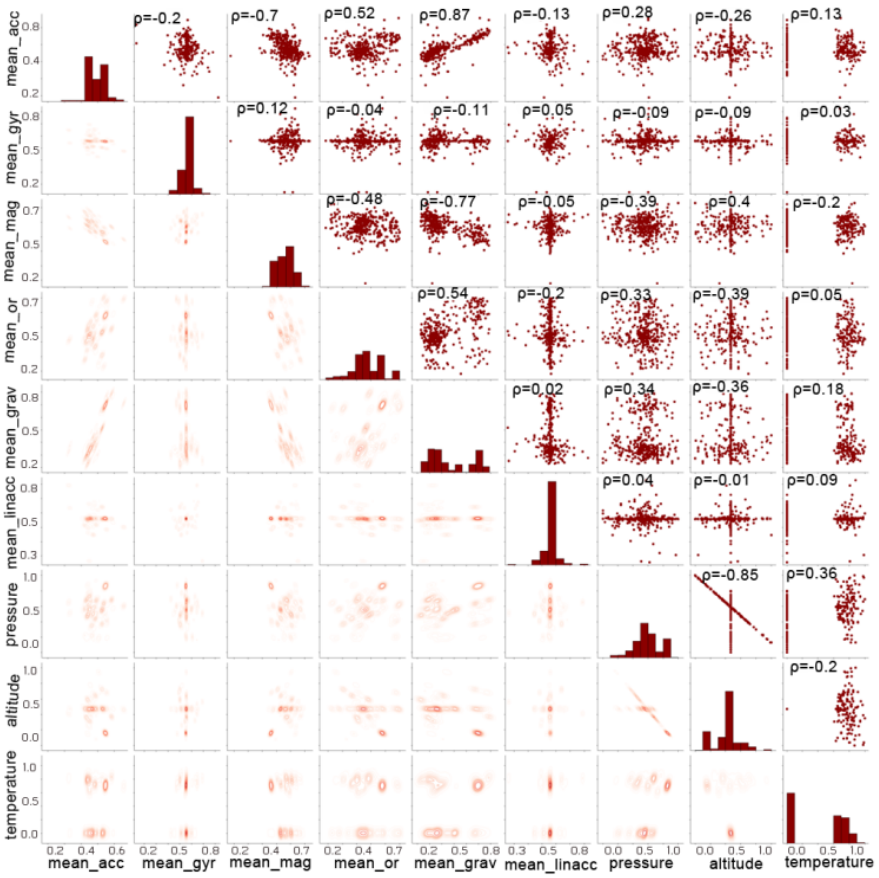


Figure 5.6 A correlation analysis using Pearson's Correlation Coefficient (PCC).

5.3.3 Data preparation

The dataset, having input and output variables, was already a supervised learning problem. For a better performance, data was processed by dropping rows with class 0 (null class) and by sorting time values by date. Taking into consideration the fact that switching between modes of transport can be quite frequent, the dataset was resampled at a 1 minute time-window frequency, by making a down sampling from milliseconds to minutes. The problem was framed so that recent time steps could be used to predict the transport mode for the next time step. Also, the input features were normalized because of the different scales in input values.

Two different approaches were tested before feeding the input data to the model. Firstly, the data was used as the initial set of 22 features. After that, before applying BERT, a dimensionality reduction algorithm was applied, so as to reduce the dimension of the training data and examine whether this will affect and to what extent the performance of the model. Because this study assumes the sensor values for each timestamp as sequences and each measurement as a word, the values were rounded to 2 decimal places so to have a more distinct and clear vocabulary. The technique used for dimensionality reduction was Principal Component Analysis (PCA). A number of n-components values were tested to fulfill the best model performance. As presented in Figure 5.7, in order to implement PCA, seven principal

components was the best choice so as to keep over 90% of the total variance of the data.

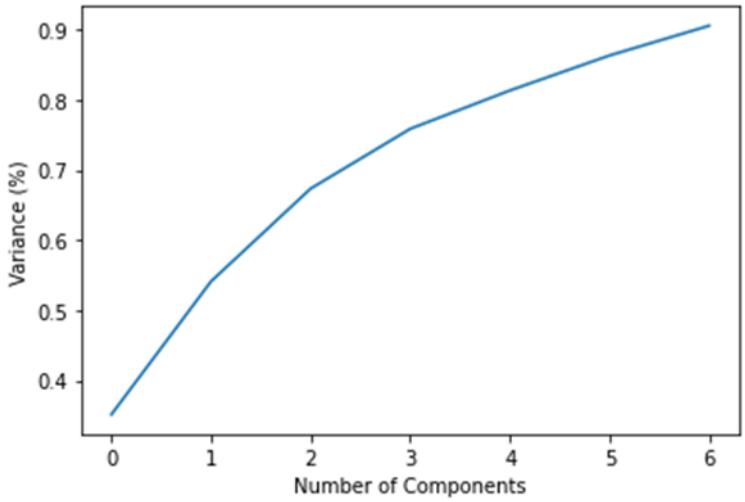


Figure 5.7 Number of principal components for PCA algorithm.

5.4 Experimental Results

In this subsection, the performance of the proposed optimized BERT model is evaluated. As a benchmark to the BERT model in order to determine transportation modes, LSTM was used, since it is the best performing model up to the present.

The LSTM model was defined having 64 neurons in the first hidden layer, a Softmax activation function which is popularly used for multiclass classification problems and 8 output values in the output layer. Categorical crossentropy was used as a method to calculate errors and Adam optimizer with a learning rate of 0.001 in order to update the weights of the neural network. According to [130], Adam is "computationally efficient, has little memory requirement and is well suited for problems that are large in terms of data/parameters". Additionally, dropout was implemented to randomly drop 20% of units from the network. Various tests were performed on the number of epochs and batch sizes until the appropriate values were selected for optimal results in model performance. In summary, the LSTM model hyperparameters are presented in Table 5.3.

LSTM hyperparameters	Values
Num of neurons in the first layer	64
Output values	8
Optimizer	Adam
Dropout	0.2
Learning rate	0.001
Error calculation	Categorical crossentropy
Dense Layer	1
Activation function	Softmax

Table 5.3 The parameters of LSTM.

The performance evaluation of various classification models, was implemented by calculating several metrics, i.e., accuracy, precision, recall, F1-score, confusion matrix, Matthews correlation coefficient (MCC) and Cohen Kappa Score (CKS). Among these, weighted F1-score was selected as the most representative metric, because it corresponds to the average F1-score value, taking into consideration the proportion of each class in the dataset. For forecasting, 1 minute before was taken into consideration, so as to predict the media of transport used in the next minute.

The experimental results indicated that TMD-BERT outperformed LSTM implementation. Figure 5.8 shows F1 Score for TMD-BERT compared with LSTM before and after dimensionality reduction with PCA, for both techniques.

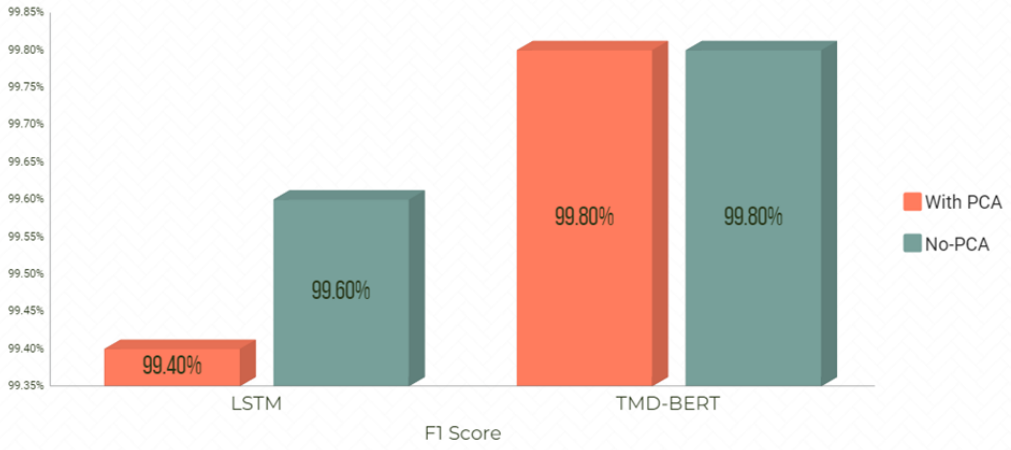


Figure 5.8 F1 Score for TMD-BERT compared with LSTM, before and after dimensionality reduction with PCA.

PCA has not affected the TMD-BERT performance concerning F1 Score. Precision and Recall values remain unchanged, too, as depicted in Figure 5.9.

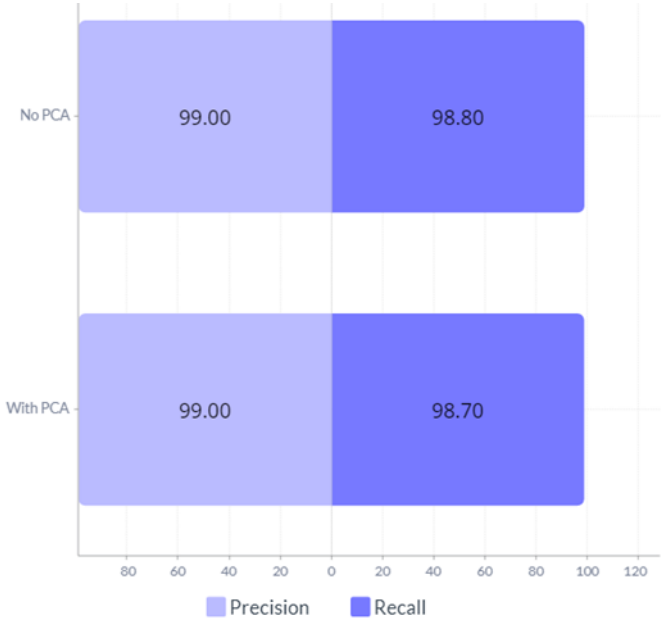


Figure 5.9 Precision and Recall for TMD-BERT before and after dimensionality reduction with PCA.

Figure 5.10 shows the average values of the actual transport media used, indicated by the blue line and the predicted values indicated by the orange line. It is clear that the TMD-BERT model was able to capture the overall trend as the orange and blue lines coincide for the most part.

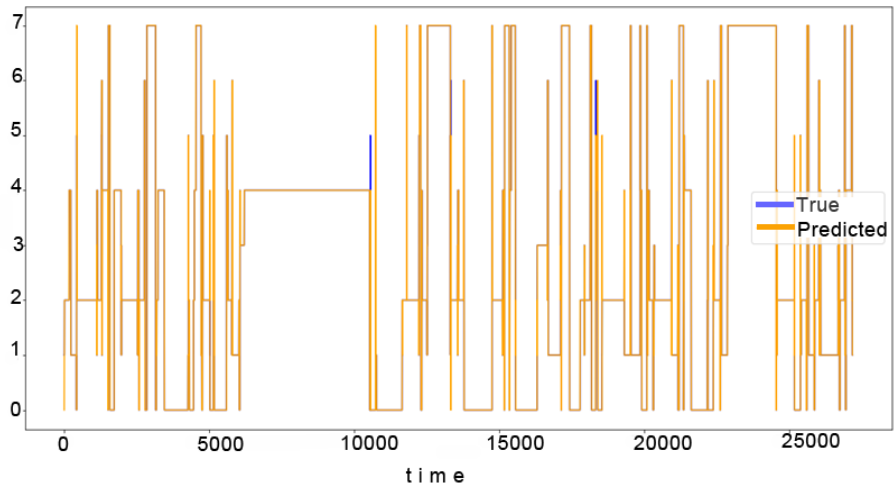


Figure 5.10 The average values of the actual transport media used, indicated by the blue line and the predicted values indicated by the orange line.

Similarly, the actual and predicted values per class are shown in Figure 5.11. The model does seem to provide an adequate fit in all classes.

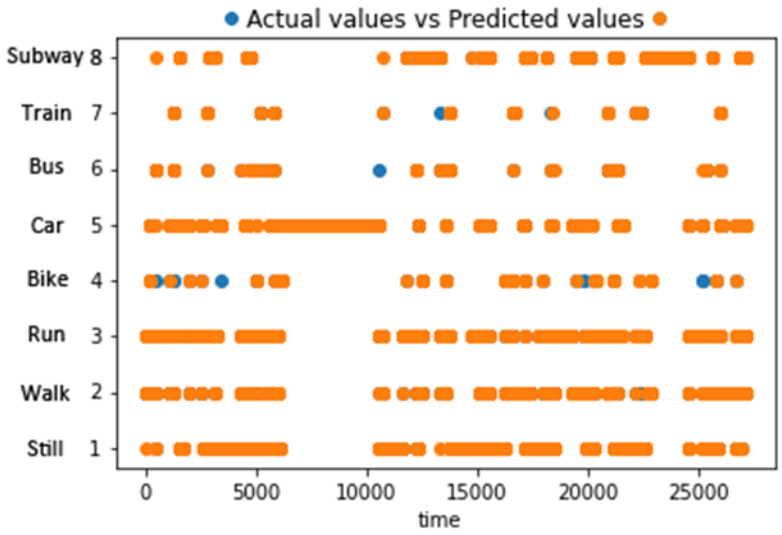


Figure 5.11 The actual and predicted values per class without dimensionality reduction.

Figure 5.12 presents the accuracy per class before and after applying PCA algorithm to our model. The classes Still, Train and Subway were predicted more accurately without dimensionality reduction and only Bus achieved an increase in accuracy after applying PCA. The prediction for the other classes (Walk, Run, Bike and Car) was almost perfect in both cases.

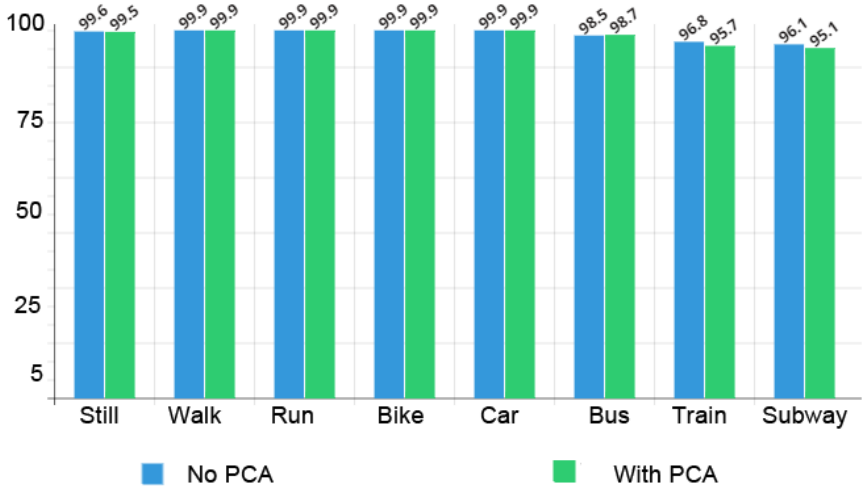
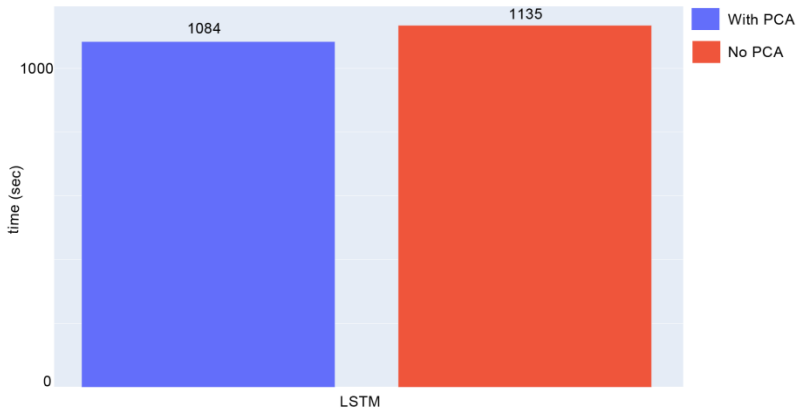
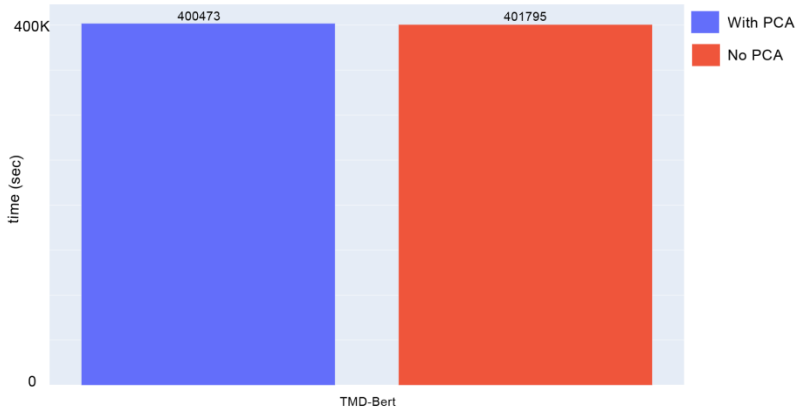


Figure 5.12 The accuracy per class before and after applying PCA.

In terms of time, as depicted in Figure 5.13, TMD-BERT compared to LSTM had a longer training and prediction time, as the model ran on a CPU-system. The training time for both TMD-BERT and LSTM decreased after applying PCA algorithm.



(a)



(b)

Figure 5.13 Training time for (a) LSTM and (b) TMD-BERT with and without PCA.

The Matthews correlation coefficient (MCC) [134] is a metric used widely in NLP field to evaluate performance, especially for imbalanced classes. With +1 being the best score, and -1 the worst score, 0.997-0.998 value indicates a high performance of the model. Cohen Kappa Score (CKS) [135], a measure that can handle efficiently both multi-class and imbalanced class issues, can be defined as a metric used to compare the predicted labels deriving from a model with the actual data labels. The value ranges from -1 (indicating worst performance) to 1 (indicating best performance). A Cohen Kappa value of 0 means that the model is close to random guessing whereas a value of 1 means that the model is perfect [136]. A value of 0.997-0.998 indicates a high performance of the model. In Figure 5.14 (a), F1 Score, Accuracy, Matthews Correlation Coefficient (MCC) and Cohen Kappa Score (CKS) values before and after applying PCA, are depicted. True positives, False negatives, False positives and True negatives values involved in the calculation of the above metrics, are shown in Figure 5.14 (b), being indicative for Bike class.

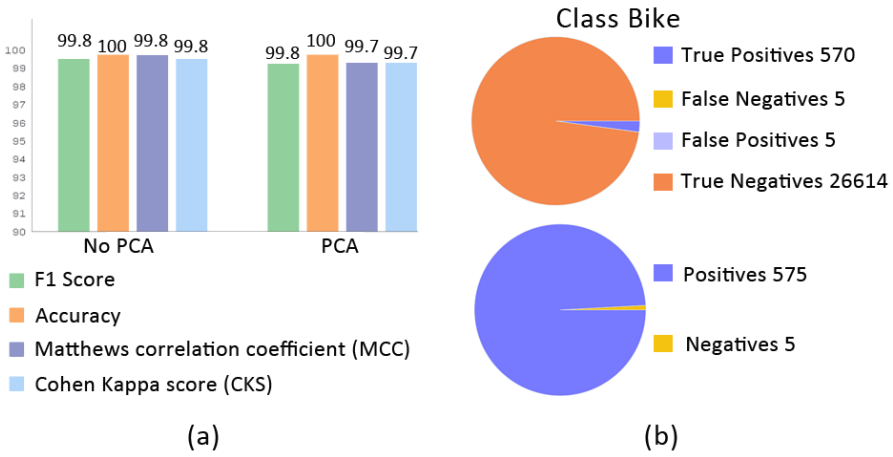


Figure 5.14 (a) F1 Score, Accuracy, Matthews correlation coefficient (MCC) and Cohen Kappa Score (CKS) values before and after applying PCA (b) True positives, False negatives, False positives, True negatives for Bike class.

5.5 Conclusion

To address the challenges derived from LSTM model approach for Transportation Mode Detection described in Chapter 4, a Transformers model (TMD-BERT) was presented in this chapter in order to forecast the transportation media used. One minute before was taken into consideration, so as to predict the media of transport used in the next minute.

The proposed TMD-BERT model introduces the strengths of the bidirectional encoder representations from transformers approach in the TMD problem, handling the entire sensor data sequence as a whole and thus allowing for better long-range dependency modeling. An extensive experimental evaluation shows the high performance of the model (99.8% F1-Score) in comparison to the state of the art.

Part II: Traffic Flow Estimation

Chapter 6

Traffic Flow Estimation: Background

6.1 Motivation

With the rapid progress of urban development, population growth, and technological advancements such as Artificial Intelligence, Internet of Things, and Internet of Vehicles, there is an urgent need for more advanced and efficient transportation systems. These systems require improvements in their structure, organization, and operation.

Traffic flow estimation is a crucial aspect of modern transportation management and Intelligent Transportation Systems (ITS) [13]. It plays a significant role in route planning, evaluating travel demand, optimizing public transport operations, and ultimately addressing transportation-related issues.

The transportation sector in cities is expanding and evolving to become more accessible, accommodating larger numbers of passengers without compromising travel quality, and adopting eco-friendly and sustainable practices. Modern public transport systems now encompass various modes of transportation, including road vehicles, underground railways, bike sharing systems, ride-hailing services, and e-scooters, among others.

The underground transit seems to be the leading force in public transportation. Worldwide investments in underground rail transportation infrastructure are growing fast every year [137]. Most cities without metro are planning to construct one, and the cities that have already developed subways are building new tracks and expanding their subway network due to everyday traffic pressure. The speeds of movement, the accuracy of the transition times as well as an extensive network of stations, are some of the reasons why it excels over other public transportation media.

In addition to underground railway transit, shared bicycles seem also to be one of the best alternatives to address the climate emergency, traffic road congestion and overcrowding. A bike sharing system consists of bicycles which are available to the citizens to rent them for short or longer distances. Thus, several stations are strategically distributed throughout the city, in order to allow people to rent and return the bikes according to their traveling needs. Bike-sharing systems have already been implemented in numerous places throughout the world.

Hence, the transport network has become more complex and difficult to manage. Because of the irregular structure of the transportation network and the fluctuating temporal features of traffic flow, the transportation flow forecasting problem seems to be more challenging than other traditional time series forecasting problems

which have a simpler structure and complexity. The traffic state in a specific location has both spatial and temporal dependency, thus it is important to take both of them into account so as to make an accurate estimation.

6.2 Mathematical Problem formulation

The transportation network in this work, is presented as an undirected graph $G = (V, E, A)$, where V represents the set of nodes (stations), E stands for the set of edges (physical or dynamic connections between stations) and A describes the adjacency matrix between stations. The adjacency matrix element A_{ij} represents the connection relationship between two nodes, v_i and v_j. The element A_{ij} in A equals 1 if node i and j are connected or 0 if otherwise. So adjacency matrix A indicates the neighbors of a node. Each node (station) has a number of features.

The purpose of transportation flow forecasting is to make predictions about future transportation flow based on past transportation flow for a specific period of time. The goal of this paper's work is to construct a function f that takes historical transportation flow data X_t as well as the graph G as inputs and estimates flow of all nodes at the next time, X'_{t+n}:

$$\mathrm{X'_{t+n}} = \mathrm{f(X_t, G)} \tag{17}$$

6.2.1 Metro and Bike

Each system in real-world has its unique traits and specific features, as briefly described in Table 6.1. Hangzhou Metro has a more structured stations network, making it more difficult to reformat the positions of the stations. The traffic patterns in Hangzhou Metro are mostly predetermined, allowing for efficient planning and management. The stations in Hangzhou Metro have physical connections between them, ensuring a smooth flow of passengers. In terms of time intervals, Hangzhou Metro has smaller intervals, resulting in more frequent usage and shorter waiting times for passengers. Additionally, Hangzhou Metro has a smaller number of stations, providing a more compact and streamlined system.

On the other hand, NYCBS (New York City Bike Share) has a more unstructured stations network, allowing for greater flexibility in changing the positions of the stations. The usage of stations in NYCBS is more random, as passengers can choose different stations based on their needs or preferences. The connections between stations in NYCBS are dynamic, adapting to factors such as traffic conditions or demand. In contrast to Hangzhou Metro, NYCBS has bigger time intervals in bike usage. This means that it may take longer for a bike to become available for use, resulting in potentially longer waiting times for users. Additionally, NYCBS has a greater number of stations to accommodate different routes and connections in a larger city like New York.

Therefore, the variations in the structure of each transport system, their adaptability to change, the level of predictability in travel patterns, and the time of usage and switching between modes of transportation result in distinct behaviors of each system. As a consequence, each system demands a unique approach.

Hangzhou Metro	NYCBS
More structured stations network	More unstructured stations network
More difficult to reformat stations position	More flexible to change stations position
Mostly predetermined traffic patterns	More random station usage
Physical connections between stations	Dynamic connections between stations
Smaller time intervals in metro usage	Bigger time intervals in bike usage
Smaller number of stations	Greater number of stations

Table 6.1 Description of datasets specific features each system has in real-world.

6.3 Related work

Transportation flow estimation is a significant research problem in the intelligent transportation field. There have been several methods to predict transportation in metro, bus, bike, train etc. systems.

Statistical methods: Traditional methods are mainly based on mathematical statistical models which use the techniques of time series analysis in statistics. Autoregressive Integrated Moving Average (ARIMA) [138] and its variants [139, 140] are the most effective approaches designed to predict future values in short-term traffic data based on historical observations. In [141], an ARIMA model has been used to predict the passenger flow of Guangzhou metro based on the historical passenger flow data collected by the ticketing automatic system of urban rail transit. In [142] the rule of passenger flow in and out of Beijing subway station is analyzed in accordance with time changes, and the SARIMA model is used for modeling.

Although these methods allow transportation pattern identification, they assume linear relationships between variables and that past pattern will repeat in the future. Thus, modern transportation systems complex multivariate data are not processed effectively and nonlinear relationships between variables is difficult to be captured.

In order to address these issues, researchers eventually switched from statistical approaches to machine learning and deep learning models due to the intricate relationships of transportation flow data.

Traditional machine learning methods: Various Machine Learning methods are used in order to analyze heterogeneous data from various sources [143, 144, 145]. Numerous studies have been conducted to forecast passengers flow in various transportation media. In [146], a hybrid model is used that combines K-means algorithm for clustering the original sample set and Support Vector Machine (SVM) to forecast the public bicycle traffic flow. A method to extract passenger flow of different routes on bus stations is implemented by using an XGBoost model in [147] whereas in [148] a Multi-feature Gradient Boosting Decision Tree (GBDT) model is proposed in order to accurately predict short-term bus passenger flow.

Deep learning methods: Deep Learning methods have demonstrated significantly higher effectiveness in predicting passenger flow in transportation media when compared to ML or statistical techniques [149, 150]. These methods are able to represent complex non-linear spatio-temporal dependencies which are a major characteristic in transportation data.

In the process of passenger flow prediction, Recurrent Neural Networks (RNN) can effectively solve the problem of randomness and nonlinearity which cannot be solved by the existed linear models. In [151], the combination of RNNs and wavelet transform is employed to predict the passenger flow and the results show that the method can effectively improve the prediction accuracy. Regular RNNs drawback is the vanishing gradient issue which means that part of the data from previous layers gets lost. Long short term memory (LSTM) was used frequently to anticipate passenger movement as these models seem to address the vanishing gradient problem. A deep irregular convolutional LSTM network model called DST-ICRL for urban traffic passenger flow prediction was used in [152] whereas in [153] an end-to-end deep learning architecture based on the LSTM, termed as Deep Passenger Flow (DeepPF), managed to forecast the metro inbound and outbound passenger flow. LSTM models primarily take into account the temporal aspects of transportation flow. However they do not take into account the limitations of network topology on transportation data changes.

Convolutional neural networks (CNNs) have proved to extract spatial dependencies of transportation and spatial correlations. For predicting an urban rail transit passenger flow time series and spatiotemporal series, two deep learning neural networks were utilized in [154] a long short-term memory neural network (LSTM) for time series prediction and a convolutional neural network (CNN) for spatiotemporal series prediction.

Although Deep learning models results seem promising to represent the non-linearity of transportation flow prediction, there are still some limitations considering the criticality of missing data [155] as well as the requirement of large amounts of historical data for training the model [156] which may result in over-fitting of the model because of fluctuations in a small time interval transportation flow. Both limitations are crucial for transportation flow forecasting, as data in this field must be accurate so as to lead to an effective performance.

Graph Neural Networks: Due to the recent continuous development of Graph Neural Networks [157, 158] researchers have taken into consideration the graph structure of transportation networks and have started to use GNN-based methods for transportation flow prediction tasks.

With Graph Neural Networks which are special types of neural networks capable of processing graph structured data in non-Euclidean space [159], spatial and temporal dependencies can be learned, making more accurate transportation flow predictions.

As in this study two real-world transportation flow datasets have been utilized, one from Hangzhou metro railway system and one from NY bike sharing system, previous works are focused on these two areas of data.

Specific to bike sharing system flow, the problem of accurate bike demand estimation in bike sharing systems was investigated in [160] for effective station rebalancing by building a spatiotemporal graph neural network (ST-GNN) model to predict the city-wide bike demands. Similarly in [161, 162], a model is built in order to predict flow at station-level by viewing the bike sharing system from the graph perspective and taking into account also external influential factors, such as events [161], weather [162], etc. In [161], the attention-based mechanism is introduced to further improve the performance of a model which predicts the number of available bikes in bike-sharing systems in cities.

Various remarkable works have been performed in metro railway systems, too. In [163], a dynamic spatio-temporal hypergraph neural network is proposed, to forecast passenger flow. Furthermore, hypergraph convolution and spatio-temporal blocks are proposed to extract spatial and temporal features to achieve node-level prediction. The model in [164] integrates the relational graph convolutional network (R-GCN), split-attention mechanism, and long short-term memory (LSTM) to explore the spatiotemporal correlations and dependence between passenger inflow and outflow.

Based on a priori knowledge, [165] creates multi-view graphs to express the static feature similarity of each station in the metro network which are then input to the multi-graph neural network in order to realize the complex spatial dependence of each station's passenger flow by extracting and aggregating features.

In summary, there have been a number of approaches in this field for predicting traffic flow based on conventional statistical, machine learning, and deep learning models. However, a number of major challenges still exist:

1. When the stations network does not have a fixed structure but is dynamic, the relationship between the stations also changes. The spatial dependencies depend not only on the physical connections of stations, but on the dynamics of the system i.e. the mobility of flow-makers (passengers or bikes) which also depends on various external factors (weather, peak hours, personal choices etc.).

Therefore, it's a challenge to capture and take into consideration the dynamic spatial dependency relations between stations in order to make an accurate estimation of transportation flow.

2. The estimation of transportation flow in transport is highly dependent on whether the station network is structured or unstructured. Predicting flow is simpler in a structured network, where travel patterns are mostly predetermined, compared to an unstructured network where station usage is more random and fluctuates significantly over time. Thus, it is crucial to examine and contrast the estimation of transportation flow in both types of systems.

3. To enhance the accuracy of estimations and ensure their practical applicability, it is crucial to forecast not only in the short term but also in the long term. As the duration of the estimation period increases, the impact of uncertain factors results in a decrease in the accuracy of predictions. Additionally, the dynamic variability of transportation flow further elevates the uncertainty of estimations. Generally, long-range predictions are more demanding than short-range ones, but their practical significance is greater. Thus, it is a challenge to attain a long-term estimation of transportation flow.

4. In a smart modern city, there are various modes of transportation, each with its unique traits and specific features. Therefore, it is essential that traffic prediction methods are comprehensive and not restricted to a single type of transportation mode.

To address the aforementioned challenges, we suggest ST-GCRN, a Spatial-Temporal Graph Convolutional Recurrent Network for estimation of transportation flow. This technique can detect the dynamic spatial correlation between these stations and can perform long-term forecasting.

6.4 Relevant Datasets

Two real-world datasets, Hangzhou metro and Bike NYCBS are used to evaluate the performance of our method.

Hangzhou Metro System [166]**:** Hangzhou Metro dataset was published by the Tianchi BigData Competition. It is a metro card swiping dataset which includes 25 days of subway card data files from 2019-01-01 to 2019-01-24, a total of about 70 million records from 81 subway stations on 3 lines. The number of samples, i.e. the outflow/inflow of passengers at a station at the various time intervals, is 2293.

NY City Bike System (NYCBS) [167]**:** The data is taken from the NYC Bike system from January 2021 to December 2022. Citi Bike is NYC's official bike share program, designed to give citizens an alternative to walking, taxis, buses and subways. From the total number of stations, we used the 50 most frequently used. Also, there is a total of 14848 samples, representing the number of bikes leaving or arriving at a station at each given moment.

The details of these two datasets are summarized in Table 6.2.

Dataset	Hangzhou Metro	NYCBS
City	Hangzhou	New York
Number of stations	81	50
Number of samples	2293	14848
Time Interval	15 minutes	60 minutes

Table 6.2 Description of datasets used in the evaluation of ST-GCRN.

6.5 Preliminary Data Analysis

The features of each dataset that were used in our approach are depicted in Table 6.3 and Table 6.4.

Hangzhou Metro	
time	Timestamp
stationID	Id of the station
status	0: outcoming passengers flow 1: incoming passengers flow
userID	Id of the user

Table 6.3 Description of datasets features (columns) in Hangzhou Metro.

NYCBS	
time	Timestamp
bikeID	Id of the bike
StartStationID	The start station of the trip Id
EndStationID	The end station of the trip Id

Table 6.4 Description of datasets features (columns) in NYCBS.

The transportation flow through time in Hangzhou metro system dataset is presented in Figure 6.1. We can see that the three most frequently used stations are Station 4 (S4), Station 9 (S9) and Station 15 (S15). At S4, it is observed that on weekdays there is a peak of traffic in the morning and afternoon while on the weekends the traffic decreases with no peak hours. At S9, the peak hours are early in the morning and late in the afternoon, whereas at the weekends the traffic is gathered during midday hours where there is a peak of traffic. The most used metro station seems to be S15 through all the week. The passengers flow follows a certain

pattern on weekdays. There are morning peak hours except for Fridays where the most traffic is in the afternoon. On Saturdays S15 is used most around 16.00.

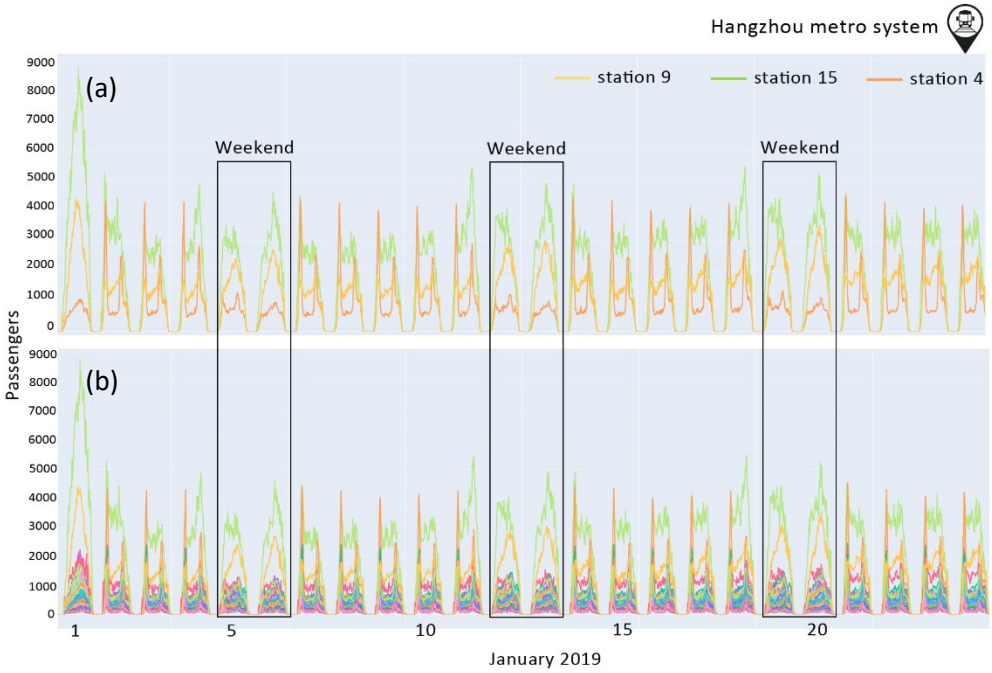


Figure 6.1 The transportation flow through time in Hangzhou metro system dataset (a) for the three most frequently used stations (b) for all stations. The passengers flow follows a certain pattern on weekdays. During weekends, there is a shift in the pattern, with the peak hours occurring in the afternoon.

The transportation flow through time in NYCBS is presented in Figure 6.2. As we can see, there is a year pattern. In the winter months bike demand falls and starts to rise again from May onwards. Peak demand is found in the summer months. For instance, in July 2021, the peak hours for the most frequently used stations (HB101, HB103), are from 17.00 to 20.00 in the afternoon. There are also stations that are not in use through all year. For example, stations HB101, 102, 103, have started their usage in May 2021 whereas HB103 interrupts its operation at certain intervals.

The operation management of the stations, their interruption for certain periods of time or permanently when demand is low, or the creation of new stations due to high traffic in the area, is a matter of critical importance for the organization and the efficient operation of the system. Therefore, knowing the traffic flow of people in a mode of transportation like the metro and of bicycles in a bike rental system is essential to all these issues so that they may be organized and run as efficiently as

possible.

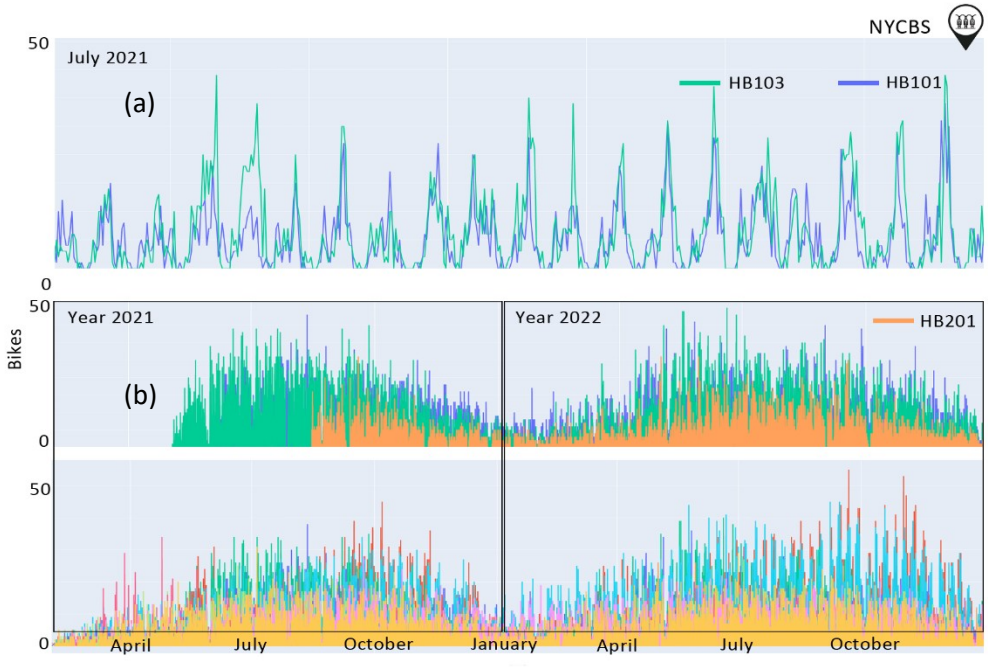


Figure 6.2 The transportation flow through time in NYCBS system dataset (a) for July 2021 (b) for years 2021 and 2022. There is a year pattern. During the winter months, there is a decline in bike demand, which starts to rise again from May onwards. The peak demand is observed during the summer months.

In order to see the spatial dependency between stations, the correlation matrix is calculated and is visualized by a heatmap presented in Figure 6.3 and Figure 6.4 for Hangzhou Metro and NYCBS datasets respectively. The color inside each cell indicates the correlation coefficient of the relationship. As the color indicates, most of the Hangzhou Metro stations are dependent on each other to a great extent, in contrast with NYCBS which are less correlated. This is reasonable, since the metro network is more structured and the transportation flow has more specific patterns.

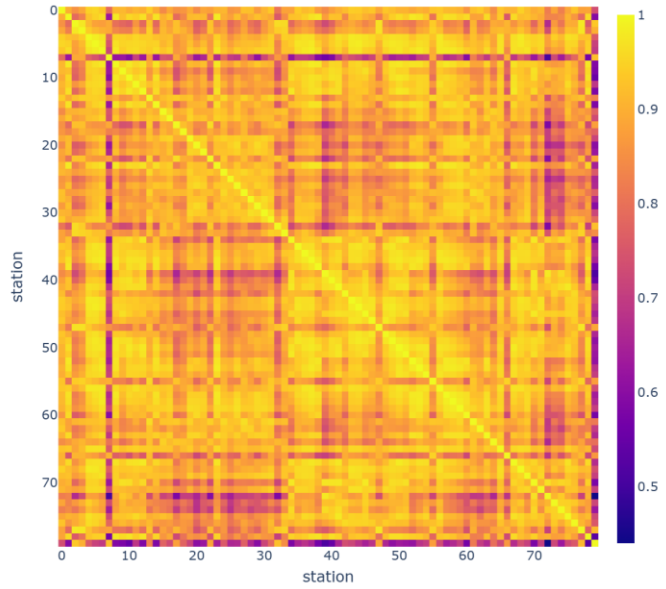


Figure 6.3 Heatmap that visualizes the spatial dependency between stations, in Hangzhou metro system dataset. The majority of Hangzhou Metro stations exhibit a high level of dependency on each other, with correlation values closer to 1.

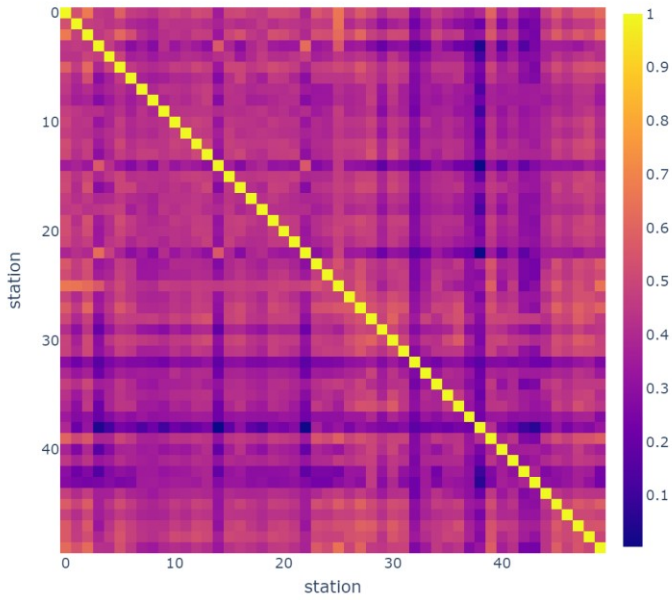


Figure 6.4 Heatmap that visualizes the spatial dependency between stations, in NYCBS system dataset. The low correlation values, which are close to zero, can be attributed to the unstructured nature of the bike stations network.

Chapter 7

A Spatial-Temporal Graph Convolutional Recurrent Network for Traffic Flow Estimation

7.1 Graph Neural Networks

Graph Convolutional Networks (GCNs) are a type of neural network that operate on graph-structured data. The model equations for GCNs typically involve three main components: message passing, aggregation and update operations.

Message Passing: In message passing, information is propagated through the graph by passing messages from neighboring nodes to update the node representations. The message passing equation can be defined as follows:

$$h_i^l = \sigma(\sum_{j \in N_i} \frac{1}{c_{ij}} W^l h_j^{l-1}) \quad (18)$$

where:

- h_i^l represents the hidden representation of node i at layer l.
- σ denotes the activation function.
- N_i represents the set of neighboring nodes of node i.
- c_{ij} represents the normalization constant for the edge connecting nodes i and j.
- W^l represents the learnable weight matrix for layer l.

Aggregation: After the message passing step, an aggregation operation is performed to combine the updated node representations into a single representation for each node. The aggregation equation can be defined as follows:

$$a_i^l = \sigma(\sum_{j \in N_i} h_j^l) \quad (19)$$

where a_i^l represents the aggregated representation of node i at layer l.

These equations can be applied iteratively over multiple layers to capture increasingly complex patterns and dependencies in the graph data. The final node representations can then be used for various downstream tasks such as node classification, link prediction, or graph-level predictions.

Update: All pooled messages are passed through an update function, usually a learned neural network.

7.2 LSTM Model

Long Short-Term Memory (LSTM) [125] is a type of recurrent neural network (RNN) architecture that is designed to overcome the vanishing gradient problem and capture long-term dependencies in sequential data. It achieves this by using a memory cell and three main gates: the input gate, forget gate, and output gate. The equations for the LSTM model are as follows:

Input Gate i_t:

$$i_t = \sigma(W_i\,[h_{t-1}, x_t] + b_i) \tag{20}$$

Forget Gate f_t:

$$f_t = \sigma(W_f\,[h_{t-1}, x_t] + b_f) \tag{21}$$

Output Gate o_t:

$$o_t = \sigma(W_o\,[h_{t-1}, x_t] + b_o) \tag{22}$$

Candidate Cell State $\tilde{c}_t$:

$$\tilde{c}_t = \tanh(W_c\,[h_{t-1}, x_t] + b_c) \tag{23}$$

Cell State c_t:

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{24}$$

Hidden State h_t:

$$h_t = o_t * \tanh(c_t) \tag{25}$$

In the above equations:

- i_t represents the input gate activation at time step t.
- f_t represents the forget gate activation at time step t.
- o_t represents the output gate activation at time step t.
- $\tilde{c}_t$ represents the candidate cell state at time step t.
- c_t represents the cell state at time step t.
- h_t represents the hidden state at time step t.
- x_t represents the input at time step t.
- W_i, W_f, W_o, W_c are weight matrices, and b_i, b_f, b_o, b_c are bias vectors.

7.3 ST-GCRN Model

In our model, we use previous timesteps as input features in order to predict the next one, two or three timesteps as the output. The spatio-temporal aspect is resulting from the historical values of each feature for a specific station as well as the feature values of the stations which are connected to that specific station. The stations are connected either physically (Hangzhou metro) either dynamically according to the stations usage (NYCBS).

In this research, a graph G = (V,E) is used in order to describe the topological structure of the stations network. As described in Table 7.1, Hangzhou metro stations network has 80 nodes and 6320 edges and NYCBS network has 50 nodes 2450 edges respectively.

	Hangzhou metro	Bike NYCBS
Number of nodes	80	50
Number of edges	6320	2450

Table 7.1 The topological structure of the stations network in Hangzhou metro and NYCBS.

In this work, a neural network architecture is implemented which can process timeseries data over a graph. We first apply a Graph Convolution Layer to the inputs and then we pass the results through a LSTM layer and a Dense layer, as shown in Figure 7.1.

In our proposed framework, historical values of features on a number of previous timesteps $(t-n,\dots,t-1,t)$ are used as inputs, in order to predict the transportation flow on a number of the next timesteps $(t+1,\dots,t+n)$. Each node in the graph starts with an initial state and that state is updated through GCN layer, by receiving "messages" from the other nodes that is connected to. All the attribute vectors of any node in the graph are transformed by the application of an aggregation function. In our framework, the aggregation function is the mean value. Then, temporal features feed LSTM layer. Finally, the predicted values are produced from the dense layer.

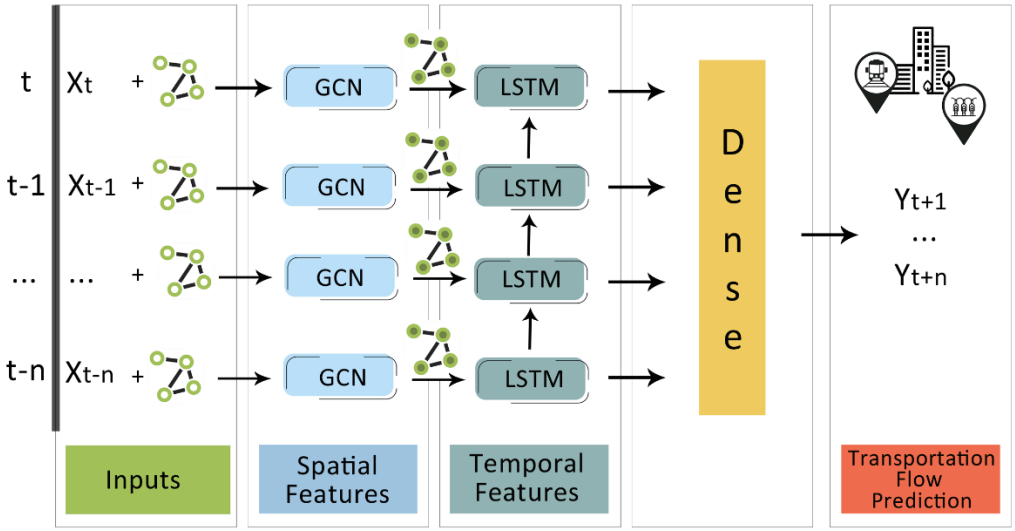


Figure 7.1 ST-GCRN Model architecture. A Graph Convolution Layer is applied to the inputs, followed by passing the results through an LSTM layer and a Dense layer.

7.4 Experimental setup

7.4.1 Data Preparation

Hangzhou Metro System: The dataset was processed so as to have the incoming and outcoming passenger number per time and station. The time window was set to 15 minutes. As there were differing scales in values, the input data were normalized, based on mean value and standard deviation.

NY City Bike System (NYCBS): The initial data were transformed in order to have the number of bikes per time and station, with 60 minutes interval. By using the start and end station of each route, we created the spatial connections between stations. We were not based on their physical connection but on the connection based on the usage of bikes and the trips between stations, which is more realistic and practically useful. The data were normalized, based on mean value and standard deviation.

Both datasets were split into two parts to create a training set and a testing set. The model was trained using 80% of the rows, while the remaining 20% was assigned to the test set.

7.4.2 Baselines

In this subsection, the performance of our proposed model is evaluated. After extensive experiments on Machine and Deep Learning methods so as to manage to include pertinent models in our work and achieve their optimal performance, we finally chose four methods as a benchmark to our proposed model. The experiments were carried out on Hangzhou metro and NY City Bike System. The chosen models are the following:

- **MO-RF** [168]: Multi-Output Random Forest Regression (MO-RF) is a multi-output regression method which utilizes MultiOutputRegressor. This approach involves fitting a single regressor for each target variable. Since each target variable is represented by its own regressor, it is possible to obtain insights about the target by examining its corresponding regressor. However, MultiOutputRegressor cannot benefit from correlations between targets since it fits only one regressor per target [169].
- **LSTM** [125]: Long-short term Memory neural network is widely used in time-series forecasting tasks due to its strong capacity to discover and utilize the information concealed in time-series sequences.
- **GRU** [170]: Gated Recurrent Unit (GRU) network generally performs similarly to LSTM on many tasks, but in some cases, GRU seems to outperform LSTM, as it is faster to train, has a simpler structure and fewer parameters than LSTM and performs better on large datasets or sequences.
- **Transformers** [171]: Transformers is a technique that stands out for its use of self-attention and differential weighting of the importance of each component of the input data. Transformers process the entire input all at once.

In addition to the aforementioned models, we incorporated several existing advanced methods proposed in other literature to evaluate the predictive capabilities of our proposed model. The inclusion of these alternative methods in our evaluation enables us to conduct a comprehensive assessment of the performance and effectiveness of our approach.

The models used as benchmarks for passenger flow estimation in Hangzhou metro, are the following:

- **STHGCN/DSTHGCN** [163]: A dynamic spatio-temporal hypergraph neural network.
- **SARGCN** [164]: A Split-Attention Relational Graph Convolutional Network.
- **PB-GRU** [172]: A deep learning model composed of Parallel multi-graph convolution and stacked Bidirectional unidirectional Gated Recurrent Unit (PB-GRU).
- **AMGC-AT** [165]: A multi-view convolution module with a spatial–temporal self-attention module and a gated convolution network.
- **STDGRL** [173]: A Spatio-Temporal dynamic Graph Relational Learning model.

The models used as benchmarks for bike demand estimation in NY City Bike System, are the following:

- **ST-GNN** [160]: A Spatiotemporal Graph Neural Network.
- **AST-GCN** [161]: An Attention-based ST-GCN.
- **GCN-Multigraph** [174]: A Multi-graph Convolutional Neural Network model.
- **MVGCN** [175]: A Multi-View Graph Convolutional Network.

7.4.3 Model Hyperparameters

Our model utilizes three previous timestamps of historical data and predicts the transportation flow at the next one timestamp.

The experiments are carried out on a system with an Intel Core i7 CPU @ 3.4GHz 3.40GHz processor and 8 GB RAM. The model is developed based on Python 3.9.7, Tensorflow and Keras 2.8.0.

The selection range of hyper parameter values for both datasets is determined through extensive experiments in order to produce the optimal results for the evaluation metrics

Hangzhou Metro System: To optimize all metrics and achieve a balanced result, we selected a value of 100 for the epochs and a batch size value of 64. As depicted in Figure 7.2(a), this decision was based on the lowest MAPE value achieved, considering that additional epochs above 100 did not yield any significant improvement in the other metrics (MAE, MSE). Similarly, as depicted in

Figure 7.3 (a), the learning rate value that was selected was 10^{-4} so as to ensure the minimization of all metrics. Root Mean Squared Propagation [176] was used as optimizer and LSTM Units were set to 256.

NY City Bike System (NYCBS): To optimize all metrics and achieve a balanced result, we selected a value of 200 for the epochs and a batch size value of 64. As depicted in Figure 7.2 (b), this decision was based on the lowest RMSE and MAE although MAPE value had a small increase. Similarly, as depicted in

Figure 7.3 (b), the learning rate value that was selected was 10^{-4}, the value at which all metrics reached their minimum. According to [130], Adam method is "computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/parameters". LSTM Units were set to 128.

The hyper-parameters of the model are summarized in Table 7.2 for both datasets.

Parameters	Hangzhou metro	Bike NYCBS
Batch size	**64**	**64**
Epochs	**100**	**200**
Optimizer	**RMSProp**	**Adam**
Learning rate	**10-4**	**10-4**
LSTM Units	**256**	**128**

Table 7.2 The hyper-parameters of the model that were finally selected after extensive experiments, for both datasets.

7.5 Experimental Results

7.5.1 Evaluation metrics

To evaluate the performance of our method in comparison with the baseline models, three evaluation metrics were used, Average Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). In the context of Machine and Deep learning:

MAE refers to the magnitude of difference between the prediction of an observation and the true value of that observation.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|f(i) - h(i)| \quad (26)$$

RMSE shows how far estimations fall from measured true values using Euclidean distance.

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{N}}(\mathrm{f(i)} - \mathrm{h(i)})^2} \quad (27)$$

MAPE refers to the average absolute percent error for each time period minus actual values divided by actual values.

$$\mathrm{MAPE} = \frac{1}{\mathrm{N}}\sum_{1}^{\mathrm{n}}\left|\frac{\mathrm{f(i)-h(i)}}{\mathrm{h(i)}}\right| \quad (28)$$

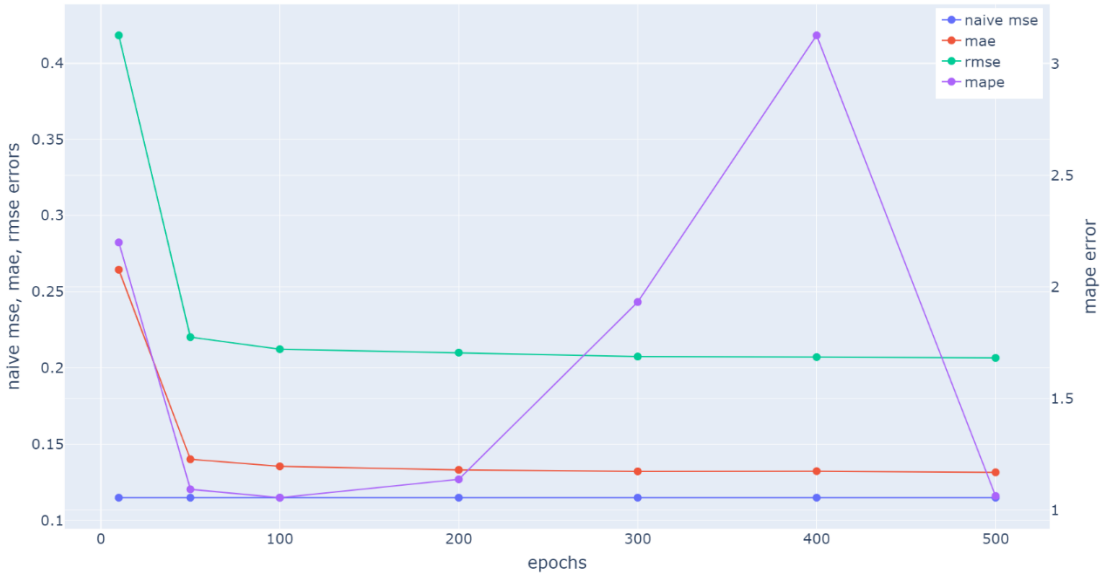
here $f(i)$ is the predicted value, and $h(i)$ is the actual value at the time i. Moreover, N is the total number of the features in dataset.

7.5.2 Hyperparameter tuning

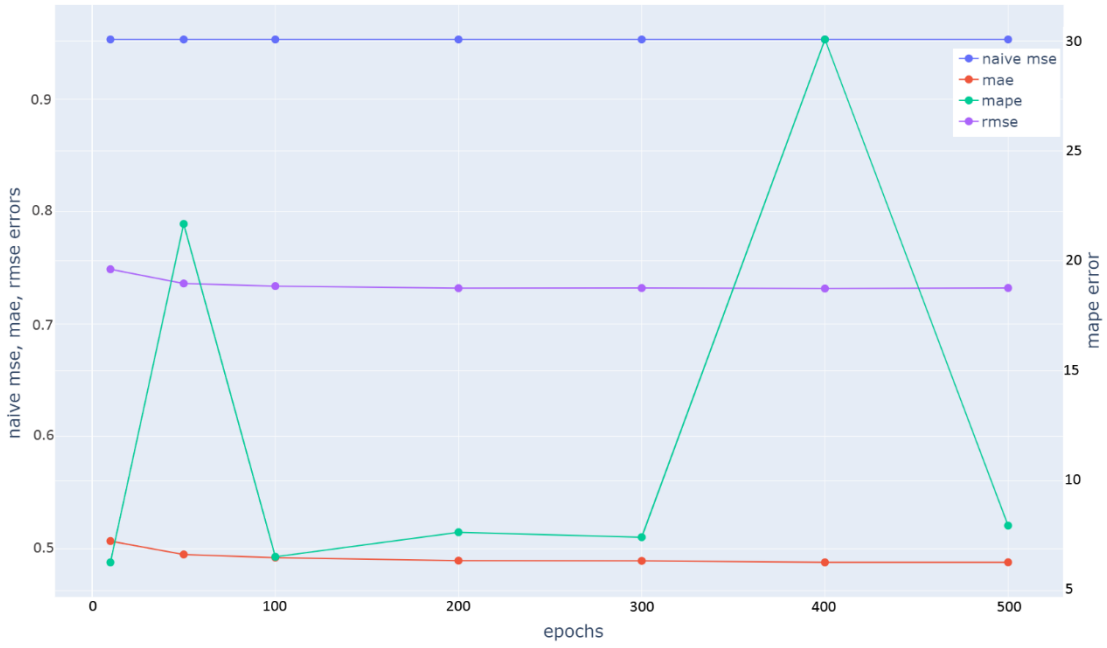
In Figure 7.2 and

Figure 7.3 we can observe how two basic parameters (number of epochs and learning rate) of the model for Hangzhou Metro System and NYCBS dataset, influence the metrics' values (MAE, RMSE, MAPE).

We can observe that MAPE fluctuates a lot as the number of epochs increases, while MAE and RMSE do not seem to be affected. The same is the case with the learning rate, where, as learning rate increases, the MAPE has a smaller variation compared to the epochs increase, but still shows a sharp increase and then a decrease when the rate reaches a high value, in Hangzhou Metro System. The loss increases as the learning rate is increased because the loss "bounces around" and even diverges from the minima due to the parameter adjustments. A large learning rate typically speeds up the learning process but results in an inefficient final set of weights. A model may learn a more ideal set of weights with a smaller learning rate, but training may take much longer. Therefore, as expected in our case, a learning rate that is too large, leads to an unstable MAPE value as the average loss increases [178].

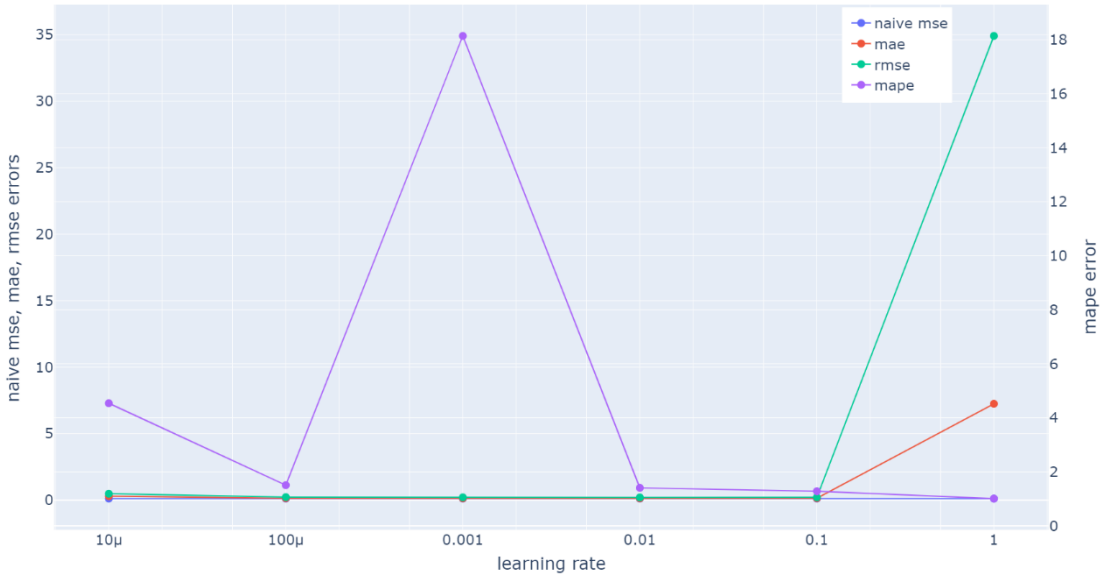


(a)

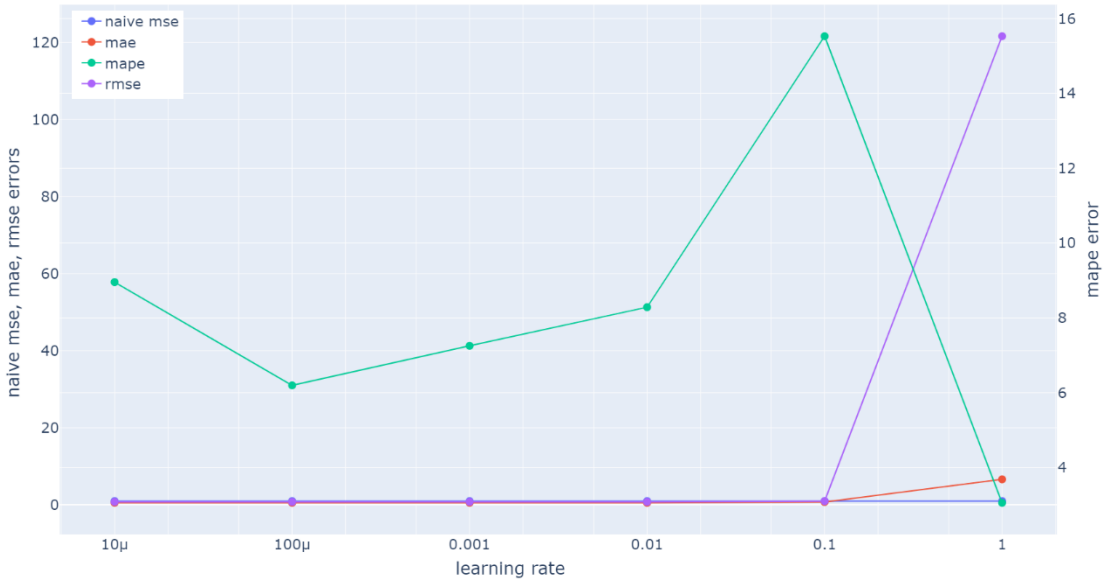


(b)

Figure 7.2 Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) in accordance with epoch values used in ST-GCRN for (a) Hangzhou Metro System dataset; (b) NYCBS dataset. As the number of epochs increases, we can observe significant fluctuations in the MAPE, while the MAE and RMSE do not appear to be affected.



(a)



(b)

Figure 7.3 Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) in accordance with learning rate values used in ST-GCRN for (a) Hangzhou Metro System dataset; (b) NYCBS dataset. With an increase in the learning rate, the MAPE demonstrates relatively less variation compared to an increase in epochs. However, it still exhibits a sharp increase followed by a decrease when the rate reaches a high value in Hangzhou Metro System (a).

7.5.3 Flow estimation results

In Figure 7.4 and Figure 7.5, the blue line represents the average values of the actual transportation flow for one station over time while the red line represents the predicted values. In the Hangzhou metro dataset it is clear that the model was able to capture the overall flow trend more accurately than in the NYCBS dataset and this

is reasonable and expected since the latter is a more unstructured network with more unpredictable traffic flow.

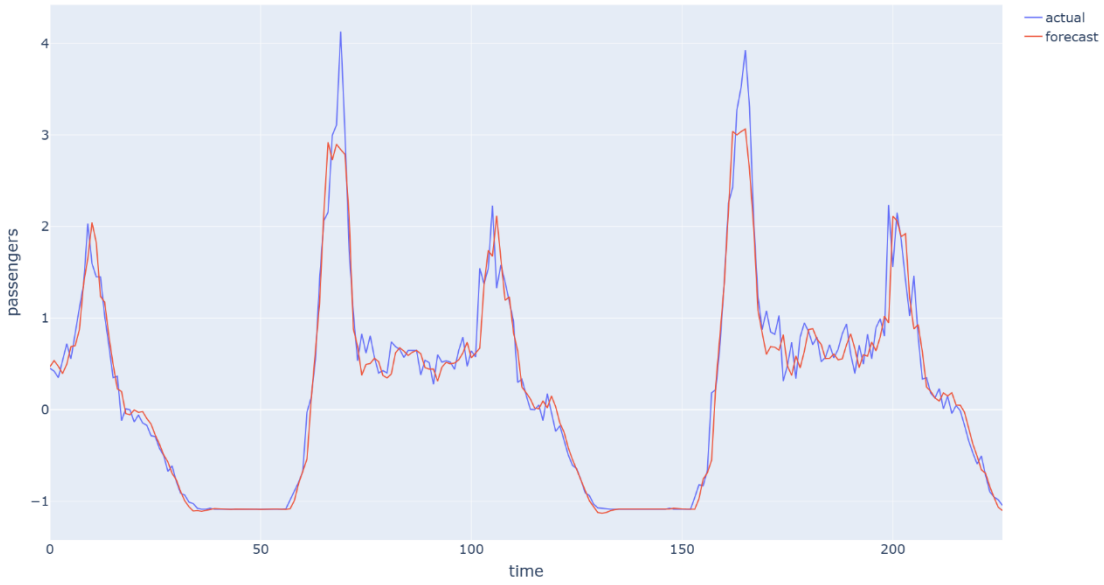


Figure 7.4 The average values of the actual transportation flow for one station over time in accordance with the predicted values for Hangzhou Metro System dataset with a sample rate of 15 minutes. The model was able to capture the overall flow trend.

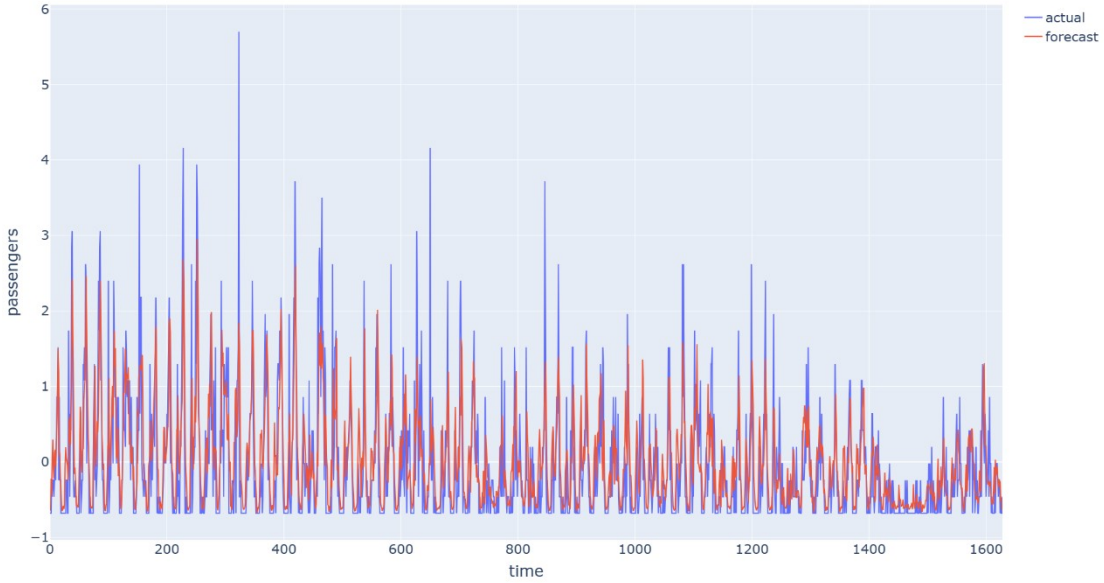


Figure 7.5 The average values of the actual transportation flow for one station over time in accordance with the predicted values for NYCBS dataset with a sample rate of 60 minutes. Due to the unstructured nature of the dataset, the model was able to capture the overall flow trend, but it struggled to accurately predict certain peak values.

In Figure 7.6 and Figure 7.7, MAE, RMSE and MAPE values are depicted for 15 minutes, 30 minutes and 60 minutes time interval, for both datasets. In Hangzhou

metro dataset, each error value increases as the sampling rate increases whereas in NYCBS dataset, MAE and RMSE decrease. Therefore, in a dynamic system like the bike system where the relationships between stations are not predetermined and change dynamically, the sampling rate must be higher than in a more static system like the subway so as to make a better estimation in transportation flow.

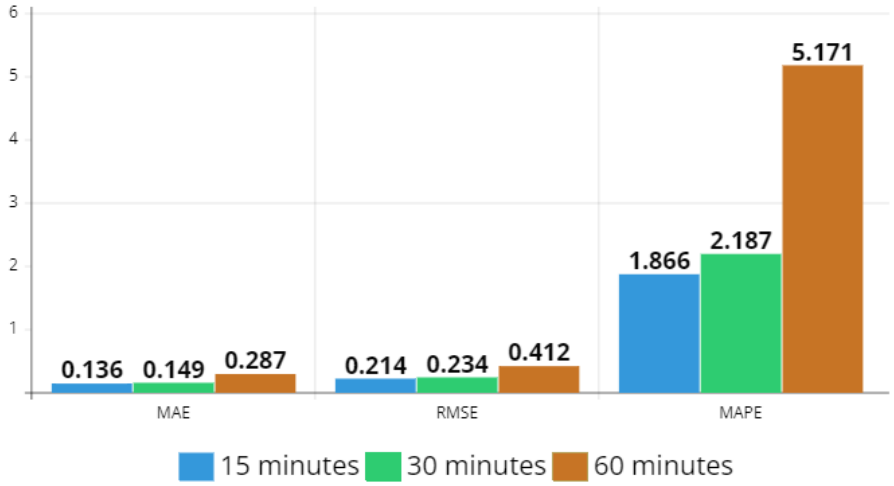


Figure 7.6 MAE, RMSE and MAPE values for 15 minutes, 30 minutes and 60 minutes time interval for Hangzhou Metro System dataset. Each error value increases as the sampling rate increases.

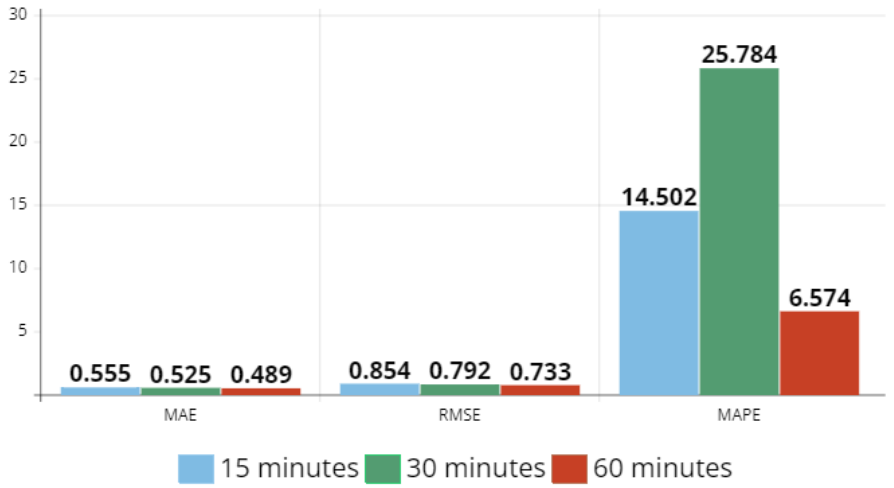


Figure 7.7 MAE, RMSE and MAPE values for 15 minutes, 30 minutes and 60 minutes time interval for NYCBS dataset. Each error value of MAE and RMSE decreases as the sampling rate increases.

7.5.4 Comparison with Alternative Models

We compared ST-GCRN to all baselines with 15 minutes, 30 minutes and 1 hour prediction time intervals, for short-term, middle-term and long-term estimation respectively. In these experiments, our model utilizes three previous timestamps of historical data and predicts the transportation flow at the next one timestamp.

Table 7.3 and Table 7.4 show the comparison of ST-GCRN with existing models from previous works on Hangzhou metro and NYCBS datasets respectively. The bold data indicate the best results. As we mentioned in subsection 5.1, MAPE is prone to variations so we preferred to focus on MAE and RMSE error values minimization.

The main inferences from the estimation findings are distilled as follows:

1. ST-GCRN can achieve the lowest MAE and RMSE, among all the existing proposed frameworks, as presented in Table 7.3 and Table 7.4 with an error decrease of 98%, in both datasets.

2. As MAPE is concerned, our framework outperforms the other frameworks except for STDGRL in Hangzhou metro dataset with a small difference though.

Models	15 minutes			30 minutes			60 minutes		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
STHGCN/ DSTHGCN [163]	11.400/ 11.080	26.770/ 26.990	-	12.900/ 13.010	28.220/ 27.840	-	-	-	-
SARGCN [164]	22.480	36.220	13.940	23.460	37.830	14.990	25.290	41.590	17.600
PB-GRU [172]	22.130	36.550	13.300	22.900	38.330	13.750	23.910	40.020	14.870
AMGC-AT [165]	19.670	31.240	-	30.680	50.790	-	-	-	-
STDGRL [173]	23.720	46.860	**0.210**	24.370	49.290	**0.210**	26.580	57.390	**0.230**
ST-GCRN (proposed)	**0.136**	**0.214**	1.866	**0.149**	**0.234**	2.187	**0.287**	**0.412**	5.171

Table 7.3 Comparison of ST-GCRN with existing models from previous works proposed in other literature, on Hangzhou metro dataset.

Models	15 minutes			30 minutes			60 minutes		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
ST-GNN [160]	-	-	-	-	-	-	2.580	3.860	7.900
AST-GCN [161]	-	-	-	1.880	-	-	-	-	-
GCN-Multigraph [174]	-	-	-	-	-	-	-	4.003	-
MVGCN [175]	-	-	-	-	-	-	2.600	4.150	-
ST-GCRN (proposed)	**0.555**	**0.854**	**14.502**	**0.525**	**0.792**	**25.784**	**0.489**	**0.733**	**6.574**

Table 7.4 Comparison of ST-GCRN with existing models from previous works proposed in other literature, on NYCBS dataset.

Table 7.5 and Table 7.6 present a comparison between ST-GCRN and various machine learning and deep learning models created specifically for this study, using the Hangzhou metro and NYCBS datasets. In Table 7.5, the symbol "*" indicates that

the prices are excessively high and go beyond the compared prices of the other models.

1. It is observed that RF performed marginally better than our proposed framework only in Hangzhou Metro dataset due to its relative small amount of data as well as the structured spatial layout of the station network. In all other case it is clear that ST-GCRN outperforms the other models in terms of MAE and RMSE values.
2. LSTM and GRU, in both datasets, have relative performance as MAE and RMSE indicate and only in MAPE they have a substantial difference.
3. Transformers have not performed well compared with the other models. Although the research community has tried to develop Transformer variants for longer sequences and these methods are quite efficient, their preconceived notions about the structure of the attention matrix, acquired through training, may not always be suitable for tasks beyond natural language processing (NLP). A very high MAPE value is observed that seems to be due to zero values indicating no connections between stations as well as due to big variance in data values as depicted in Figure 6.1 and Figure 6.2.

Models	15 minutes			30 minutes			60 minutes		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
MO-RF [168]	**0.117**	**0.176**	**1.451**	**0.111**	**0.179**	**1.200**	**0.172**	**0.286**	**0.929**
LSTM [125]	0.174	0.272	1.362	0.197	0.295	2.783	0.490	0.700	4.821
GRU [170]	0.172	0.266	1.895	0.189	0.293	6.218	0.397	0.579	2.709
Transformers [171]	0.766	1.028	*	0.760	1.022	*	0.767	1.019	*
ST-GCRN (proposed)	**0.136**	**0.214**	1.866	**0.149**	**0.234**	2.187	**0.287**	**0.412**	5.171

* The prices are excessively high and go beyond the compared prices of the other models.

Table 7.5 Performance comparison between ST-GCRN and various Machine learning and Deep learning models on Hangzhou Metro System dataset.

Models	15 minutes			30 minutes			60 minutes		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
MO-RF [168]	0.574	0.866	43.726	0.542	0.797	10.623	0.508	0.738	6.655
LSTM [125]	0.632	0.880	**3.876**	0.575	0.824	12.783	0.566	0.798	6.185
GRU [170]	0.663	0.890	5.127	0.596	0.834	**6.693**	0.560	0.795	**6.122**
Transformers [171]	0.700	0.923	79.907	0.692	0.920	76.385	0.693	0.920	103.553
ST-GCRN (proposed)	**0.555**	**0.854**	14.502	**0.525**	**0.792**	25.784	**0.489**	**0.733**	6.574

Table 7.6 Performance comparison between ST-GCRN and various Machine learning and Deep learning models on NYCBS dataset.

7.5.5 Modeling results with different estimation horizons

In this subsection, we describe the results that arise from various experiments on different estimation horizons. Table 7.7 shows the change of MAE, RMSE and MAPE values at different estimation horizons i.e. 1, 2 or 3 timesteps ahead from 3 previous samples.

1. The best performance was accomplished by predicting 1 next timestep for both datasets. It is noticeable that the structured form of the metro network does not allow a longer term forecasting as it might be expected.
2. However, with little variation in the error value we were able to predict also the second and the third next timesteps from the previous three samples.

Different estimation horizons	15 minutes			60 minutes		
	Hangzhou Metro			**Bike NYCBS**		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
1 next from 3 previous	0.136	0.214	1.866	0.489	0.733	6.574
2 next from 3 previous	0.215	0.356	2.484	0.543	0.809	7.686
3 next from 3 previous	0.335	0.553	2.962	0.606	0.893	9.039

Table 7.7 MAE, RMSE and MAPE values on different estimation horizons.

7.5.6 Comparison between the two datasets

The performance of the model is assessed using two datasets with distinct characteristics, as described in subsection 6.2.1. Mean Absolute Scaled Error (MASE) is a metric used in time series estimation to evaluate the accuracy of forecasts produced by an algorithm. Its value is given by the ratio of MAE for algorithm and MAE of naïve forecast and it provides an insight into how well a forecasting algorithm is performing in comparison to this naive forecast. If the value of MASE is greater than one (1), it suggests that the algorithm is performing poorly. By applying ST-GCRN model to the datasets we use for the model's evaluation, as presented in Figure 7.8 and Figure 7.9, the following points are observed:

1. The decrease in the sampling rate of the Hangzhou metro system leads to a reduction in the MASE. On the other hand, it has been noticed that in the NYCB system, where the connections between stations are dynamically established, an increase in the sampling rate results in a decrease in the MASE value.
2. As far as long-term horizon transportation flow estimation is concerned, NYCB dataset performs better in all three cases of 1, 2 and 3 timesteps ahead from 3 previous samples.

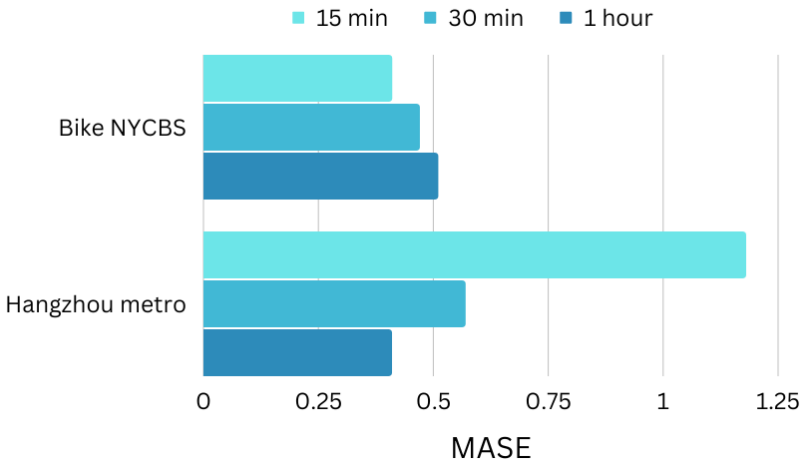


Figure 7.8 Performance of the two datasets compared with each other with 15 minutes, 30 minutes and 1 hour prediction time intervals based on Mean Absolute Scaled Error (MASE). The decrease in the sampling rate of the Hangzhou metro system leads to a reduction in the MASE whereas in the NYCB system results in a decrease in the MASE value.

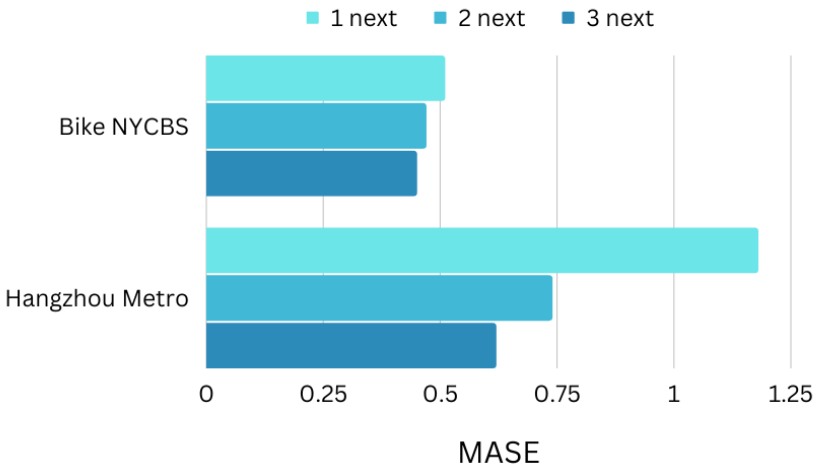


Figure 7.9 Performance of the two datasets compared with each other with estimation horizons 1, 2 and 3 timesteps ahead based on Mean Absolute Scaled Error (MASE). NYCBS dataset performs better in all three cases of 1, 2 and 3 timesteps ahead from 3 previous samples.

7.5.7 Computation efficiency

The experiments are carried out on a system with an Intel Core i7 CPU @ 3.4GHz 3.40GHz processor and 8 GB RAM. The training time needs about 20.59 seconds/epoch for Hangzhou Metro and 242.45 for NYCBS, while the inference time just takes a few seconds as depicted in Table 7.8. Given that the training process is conducted offline, the current running efficiency is considered sufficient for real-life metro and bike flow prediction systems.

	Hangzhou Metro	Bike NYCBS
Number of parameters	283915	76427
Inference time (seconds)	1.14	9.97
Training time (seconds/epoch)	20.59	242.45

Table 7.8 Number of parameters, Inference and Training time for Hangzhou metro and NYCBS.

Chapter 8

Conclusion

8.1 Discussion

This thesis aims to address issues related to smart cities by focusing on the identification of citizens' transportation modes and on predicting transportation flow in diverse systems. To overcome these challenges, the research emphasizes in:

(a) Establishing a robust and efficient Transportation Mode Detection (TMD) system that utilizes sequential data from various smartphone sensors. This system is designed to enhance detection accuracy compared to current methods.
(b) Creating a Spatial-Temporal Graph Convolutional Recurrent Network for Transportation Flow Estimation (TFE). This innovative method can identify dynamic spatial correlations between stations and is capable of making long-term forecasts.

We have successfully managed the preliminary sensor data (in TMD) and smart card swiping data (in TFE) through efficient cleaning and preprocessing techniques, which involve imputation, data normalization, and feature extraction. This approach managed to rectify errors, address missing data, and mitigate high-dimensionality issues.

More analytically:

(a) This study has investigated the efficacy of various Machine and Deep Learning approaches in Transportation Mode Detection (TMD), utilizing multimodal smartphone sensor data. The suggested Bayesian-optimized LSTM model exhibits remarkable performance, accurately identifying eight transportation modes with 99.7% accuracy and a 99.8% F1-Score. Notably, it surpasses other cutting-edge methods. However, despite the positive outcomes, certain limitations were encountered. The restricted availability of annotated data, particularly for specific classes, posed challenges for the model in distinguishing between certain modes, such as bus, train, and subway.

To tackle the challenges derived from the proposed LSTM model approach for TMD, a Transformers model was introduced, denoted as TMD-BERT. The objective was to predict the transportation mode used, taking into account the information from one minute prior to forecast the mode for the next minute. The TMD-BERT model leverages the advantages of the bidirectional encoder representations from transformers approach in addressing the TMD problem. By treating the entire sensor data sequence as a unified entity, the model facilitates improved modeling of long-range dependencies. Through a comprehensive experimental evaluation, the

model demonstrates exceptional performance, achieving a 99.8% F1-Score in comparison to the current state of the art.

Although the two models ended up with similar overall performance, the TMD-BERT model made a better prediction than LSTM model, in all means of transport. Moreover, the limitations of LSTM-model were successfully treated by TMD-BERT model as follows:

The limited number of annotated data—especially for specific classes: The presence of varying proportions among class categories in the dataset is not a concern in this study. Firstly, BERT appears to effectively manage imbalanced classification, eliminating the necessity for employing standard data augmentation methods to address this imbalance issue [28]. Secondly, the results indicate that all classes are predicted irrespective of the number of samples for each class.

Difficulty in detecting certain classes, such as bus, train and subway even in the 1-before case: The accuracy per class before and after applying PCA algorithm to our model is presented in Figure 5.12. All the classes showed exceptional performance in all cases.

The proposed methods have been validated on real-world data and compared with a wide range of Machine and Deep Learning techniques.

(b) Moreover, this thesis introduced ST-GCRN, a Spatial-Temporal Graph Convolutional Recurrent Network, as a solution to accurately estimating transportation flow. The technique effectively detects dynamic spatial correlations between stations and performs long-term estimation. The model captures both spatial and temporal dependencies in transportation flow data.

The efficacy of the proposed framework is assessed using real-world datasets from Hangzhou metro and New York City's Citi Bike system. The experiments reveal that ST-GCRN outperforms current state-of-the-art baselines, achieving a 98% reduction in estimation errors for the metro system and a 63% reduction for the bike-sharing system, compared to the current state of the art. The model not only predicts future transportation flow in both short and long terms but also across various time horizons, enhancing accuracy and practicality compared to existing baselines. The evaluation results demonstrate the superior predictive performance of ST-GCRN across all estimation horizons, surpassing other models based on various evaluation metrics. This underscores the effectiveness of ST-GCRN in spatio-temporal traffic forecasting.

8.2 Limitations and Future research

While this study has yielded commendable results, it is essential to acknowledge the existence of certain limitations. The constrained availability of annotated data, particularly for specific modes of transportation, coupled with the considerable computational complexity of applying the proposed methods. Furthermore, the

reliance on smartphone data for transportation mode detection raises concerns, as it excludes individuals, such as residents of developing countries or the elderly, who may not own smartphones. Therefore, the investigation of alternative methods that will also take into account the important factor of security and protection of personal data becomes imperative to effectively deal with these limitations.

It is noteworthy that the current work lacks the incorporation of external contextual factors, such as climate conditions, economic variables, and social events. Recognizing the impact of these factors on transportation patterns is crucial for enhancing the model's accuracy and generalizability. Additionally, the study does not delve into the identification of patterns in transportation zones or high-traffic areas. This absence hinders a comprehensive understanding of the dynamics of transportation flow.

To tackle with these limitations, in future research, our plan is to investigate the integration of external contextual factors, such as climate conditions, economic factors, and social events. This inclusion of additional variables is anticipated to improve the model's ability to make accurate transportation mode identification and estimations of transportation flow by considering the impact of various external factors on the system. As a result, we expect to enhance the overall generalizability of the proposed model.

Moreover, our aim is to integrate the prediction of congestion patterns in transportation zones or high-traffic areas. This entails forecasting or estimating the levels and patterns of traffic congestion in specific stations or transportation modes that frequently encounter significant traffic volume or congestion. The goal of this prediction is to foresee when and how severe congestion might occur in these areas, offering valuable insights into the dynamics of transportation flow. By doing so, we can facilitate proactive measures to handle and alleviate congestion. To achieve this objective, we will scrutinize historical transportation data, encompassing information on transportation volume, speed, density, and other pertinent factors such as road infrastructure, events, and weather conditions. Harnessing this data, we will attempt to construct predictive models capable of anticipating congestion patterns. These models will empower transportation planners, traffic management authorities, and commuters to make well-informed decisions regarding route planning, transportation management strategies, and travel time estimation.

In conclusion, our forthcoming endeavors include the integration of Explainable Machine Learning techniques to augment the interpretability of the developed models and offer meaningful explanations for their predictions. This may entail the inclusion of methods like feature importance analysis, rule-based models, surrogate models, or attention mechanisms to enhance the interpretability of the models. Through the incorporation of Explainable Machine Learning in our upcoming research, our aim is to enhance transparency, trust, and accountability in our models. We anticipate that this approach will not only improve the understanding of

our predictions but also provide valuable insights for stakeholders and users, enabling them to make well-informed decisions based on the model's output.

References

[1] Vuchic, V. (2017). Transportation for livable cities. Routledge.

[2] Levine, J., Grengs, J., & Merlin, L. A. (2019). From mobility to accessibility: Transforming urban transportation and land-use planning. Cornell University Press.

[3] Dillahunt, T. R., & Veinot, T. C. (2018). Getting there: Barriers and facilitators to transportation access in underserved communities. ACM Transactions on Computer-Human Interaction (TOCHI), 25(5), 1-39.

[4] Canning, D., & Fay, M. (1993). The effects of transportation networks on economic growth.

[5] Deng, T. (2013). Impacts of transport infrastructure on productivity and economic growth: Recent advances and research challenges. Transport Reviews, 33(6), 686-699.

[6] Othman, A. G., & Ali, K. H. (2020). Transportation and quality of life. Planning Malaysia, 18.

[7] Sallis, J. F., Frank, L. D., Saelens, B. E., & Kraft, M. K. (2004). Active transportation and physical activity: opportunities for collaboration on transportation and public health research. Transportation research part A: policy and Practice, 38(4), 249-268.

[8] Verma, A., Rahul, T. M., & Dixit, M. (2015). Sustainability impact assessment of transportation policies–A case study for Bangalore city. Case Studies on Transport Policy, 3(3), 321-330.

[9] Titos, G., Lyamani, H., Drinovec, L., Olmo, F. J., Močnik, G., & Alados-Arboledas, L. (2015). Evaluation of the impact of transportation changes on air quality. Atmospheric Environment, 114, 19-31.

[10] Karekla, X., Gkiotsalitis, K., & Tyler, N. (2020). The impact of a passenger-safety-driven acceleration limit on the operation of a bus service. Accident analysis & prevention, 148, 105790.

[11] Guo, Y., Chen, Z., Stuart, A., Li, X., & Zhang, Y. (2020). A systematic overview of transportation equity in terms of accessibility, traffic emissions, and safety outcomes: From conventional to emerging technologies. Transportation research interdisciplinary perspectives, 4, 100091.

[12] Meixell, M. J., & Norbis, M. (2008). A review of the transportation mode choice and carrier selection literature. The International Journal of Logistics Management, 19(2), 183-211.

[13] Gentile, G., & Nökel, K. (2016). Modelling public transport passenger flows in the era of intelligent transport systems. Springer Tracts on Transportation and Traffic, 10, 641.

[14] Zhu, L., Yu, F. R., Wang, Y., Ning, B., & Tang, T. (2018). Big data analytics in intelligent transportation systems: A survey. IEEE Transactions on Intelligent Transportation Systems, 20(1), 383-398.

[15] Andersen, A., Karlsen, R., & Yu, W. (2018). Green transportation choices with IoT and smart nudging. Handbook of Smart Cities: Software Services and Cyber Infrastructure, 331-354.

[16] Tewolde, G. S. (2012, May). Sensor and network technology for intelligent transportation systems. In 2012 IEEE International Conference on Electro/Information Technology (pp. 1-7). IEEE.

[17] Drosouli, I., Voulodimos, A., Mastorocostas, P., Miaoulis, G., & Ghazanfarpour, D. (2023). A Spatial-Temporal Graph Convolutional Recurrent Network for Transportation Flow Estimation. Sensors, 23(17), 7534..

[18] Zhu, L., Yu, F. R., Wang, Y., Ning, B., & Tang, T. (2018). Big data analytics in intelligent transportation systems: A survey. IEEE Transactions on Intelligent Transportation Systems, 20(1), 383-398.

[19] Gargoum, S., & El-Basyouny, K. (2017, August). Automated extraction of road features using LiDAR data: A review of LiDAR applications in transportation. In 2017 4th International Conference on Transportation Information and Safety (ICTIS) (pp. 563-574). IEEE.

[20] Xiong, Z., Sheng, H., Rong, W., & Cooper, D. E. (2012). Intelligent transportation systems for smart cities: a progress review. Science China Information Sciences, 55, 2908-2914.

[21] Ali, S. (2014). Sensors and mobile phones: evolution and state-of-the-art. Pakistan journal of science, 66(4).

[22] Drosouli, Ifigenia, Athanasios Voulodimos, Georgios Miaoulis, Paris Mastorocostas, and Djamchid Ghazanfarpour. "Transportation mode detection using an optimized long short-term memory model on multimodal sensor data." Entropy 23, no. 11 (2021): 1457

[23] Siuhi, S., & Mwakalonge, J. (2016). Opportunities and challenges of smart mobile applications in transportation. Journal of traffic and transportation engineering (english edition), 3(6), 582-592.

[24] S. Richoz et al., "Human and Machine Recognition of Transportation Modes from Body-Worn Camera Images," 2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Spokane, WA, USA, 2019, pp. 67-72, doi: 10.1109/ICIEV.2019.8858537.

[25] S. Richoz, L. Wang, P. Birch and D. Roggen, "Transportation Mode Recognition Fusing Wearable Motion, Sound, and Vision Sensors," in IEEE Sensors Journal, vol. 20, no. 16, pp. 9314-9328, 15 Aug.15, 2020, doi: 10.1109/JSEN.2020.2987306.

[26] Devillaine, F., Munizaga, M., & Trépanier, M. (2012). Detection of Activities of Public Transport Users by Analyzing Smart Card Data. Transportation Research Record, 2276(1), 48-55. https://doi.org/10.3141/2276-06.

[27] Ni, Qian, and Yida Guo. "Passenger Flow Control in Subway Station with Card-Swiping Data." In International Conference on Big Data Engineering and Technology, pp. 3-9. Cham: Springer International Publishing, 2022.

[28] Jeon, Soobin, Eil Kwon, and Inbum Jung. 2014. "Traffic Measurement on Multiple Drive Lanes with Wireless Ultrasonic Sensors" Sensors 14, no. 12: 22891-22906. https://doi.org/10.3390/s141222891.

[29] Prasetyo, Moch Agung, Roswan Latuconsina, and Tito Waluyo Purboyo. "A proposed design of traffic congestion prediction using ultrasonic sensors." Int J Appl Eng Res 13, no. 1 (2018): 434-441.

[30] Fadeev, A., and S. Alhusseini. "Determining the public transport demand by validation data of the electronic tickets." In IOP Conference Series: Materials Science and Engineering, vol. 734, no. 1, p. 012148. IOP Publishing, 2020.

[31] Senneth, Leon &Wolfson, Our& Yu, Philip & Xu, Bo. “Transportation Mode Detection using Mobile Phones and GIS Information”, GIS: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems. 54-63, 2011.

[32] Xiao, Yu & Low, David &Bandara, Thusitha& Pathak, Parth&Beng Lim, Hock &Goyal, Devendra & Oliveira Santos, Jorge &Cottrill, Caitlin & Pereira, Francisco &Zegras, Chris & Ben-Akiva, Moshe, “Transportation activity analysis using smartphones”, 2012.

[33] Widhalm, P., P. Nitsche, and N. Brandie, “Transport mode detection with realistic Smartphone sensor data”, 21st International Conference on Pattern Recognition (ICPR 2012), 11-15 Nov. 2012, Piscataway, NJ, USA: IEEE.

[34] Drosouli, Ifigenia, Athanasios Voulodimos, and Georgios Miaoulis. "Transportation mode detection using machine learning techniques on mobile phone sensor data." In Proceedings of the 13th ACM International Conference on PErvasive Technologies Related to Assistive Environments, pp. 1-8. 2020.

[35] Prabha, R., and Mohan G. Kabadi. "Overview of data collection methods for intelligent transportation systems." The International Journal Of Engineering And Science (IJES) 5, no. 3 (2016): 16-20.

[36] Carpineti, C.; Lomonaco, V.; Bedogni, L.; Di Felice, M.; Bononi, L., Custom Dual Transportation Mode Detection by Smartphone Devices Exploiting Sensor Diversity. In Proceedings of the 14th Workshop on Context and Activity Modeling and Recognition (IEEE COMOREA 2018), Athens, Greece, 19–23 March 2018.

[37] Wang, L.; Gjoreski, H.; Ciliberto, M.; Lago, P.; Murao, K.; Okita, T.; Roggen, D. Summary of the sussex-huawei locomotion-transportation recognition challenge 2020. In Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers (UbiComp-ISWC '20); Association for Computing Machinery: New York, NY, USA, 2020; pp. 351–358. https://doi.org/10.1145/3410530.3414341.

[38] Jahangiri, A.; Rakha, H.A. Applying Machine Learning Techniques to Transportation Mode Recognition Using Mobile Phone Sensor Data. IEEE Trans. Intell. Transp. Syst. 2015, 16, 2406–2417. https://doi.org/10.1109/TITS.2015.2405759.

[39] Xiao, Z.; Wang, Y.; Fu, K.; Wu, F. Identifying Different Transportation Modes from Trajectory Data Using Tree-Based Ensemble Classifiers. ISPRS Int. J. Geo-Inf. 2017, 6, 57. https://doi.org/10.3390/ijgi6020057.

[40] Gjoreski, H.; Ciliberto, M.; Wang, L.; Morales, F.J.O.; Mekki, S.; Valentin, S.; Roggen, D. The University of Sussex-Huawei Locomotion and Transportation Dataset for Multimodal Analytics with Mobile Devices. IEEE Access 2018, 6, 42592–42604 https://doi.org/10.1109/ACCESS.2018.2858933.

[41] Lu, D.-N.; Nguyen, D.-N.; Nguyen, T.-H.; Nguyen, H.-N. Vehicle Mode and Driving Activity Detection Based on Analyzing Sensor Data of Smartphones. Sensors 2018, 18, 1036. https://doi.org/10.3390/s18041036.

[42] Qin, Y.; Luo, H.; Zhao, F.; Wang, C.; Wang, J.; Zhang, Y. Toward Transportation Mode Recognition Using Deep Convolutional and Long Short-Term Memory Recurrent Neural Networks. IEEE Access 2019, 7, 142353–142367. https://doi.org/10.1109/ACCESS.2019.2944686.

[43] Sharma, A.; Singh, S.K.; Udmale, S.S.; Singh, A.K.; Singh, R. Early transportation mode detection using smartphone sensing data. IEEE Sens. J. 2021, 21, 15651–15659. https://doi.org/10.1109/JSEN.2020.3009312.

[44] Vakili; Meysam; Ghamsari; Mohammad; Rezaei; Masoumeh. Performance Analysis and Comparison of Machine and Deep Learning Algorithms for IoT Data Classification. arXiv 2020, arXiv:2001.09636. Available online: https://arxiv.org/abs/2001.09636 (accessed on 12/11/2023).

[45] Delli Priscoli, F.; Giuseppi, A.; Lisi, F. Automatic Transportation Mode Recognition on Smartphone Data Based on Deep Neural Networks. Sensors 2020, 20, 7228. https://doi.org/10.3390/s20247228.

[46] Fang, S.; Fei, Y.; Xu, Z.; Tsao, Y. Learning Transportation Modes from Smartphone Sensors Based on Deep Neural Network. IEEE Sens. J. 2017, 17, 6111–6118. https://doi.org/10.1109/JSEN.2017.2737825.

[47] Asci, G.; Guvensan, M.A. A Novel Input Set for LSTM-Based Transport Mode Detection. In Proceedings of the 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 11-15 March, Kyoto, Japan, 2019; pp. 107–112. https://doi.org/10.1109/PERCOMW.2019.8730799.

[48] Liang, X.; Zhang, Y.; Wang, G.; Xu, S. A Deep Learning Model for Transportation Mode Detection Based on Smartphone Sensing Data. IEEE Trans. Intell. Transp. Syst. 2020, 21, 5223–5235. https://doi.org/10.1109/TITS.2019.2951165.

[49] Vaswani, Ashish; Shazeer, Noam; Parmar, Niki; Uszkoreit, Jakob; Jones, Llion; Gomez, Aidan N.; Kaiser, Lukasz; Polosukhin, Illia (2017-06-12). "Attention Is All You Need". arXiv:1706.03762

[50] Feng, Zhangyin, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, et al. 2020. ‘‘CodeBERT: A Pre-Trained Model for Programming and Natural Languages,” February. https://arxiv.org/abs/2002. 08155v4.

[51] ESzu-Yin Lin, Yun-Ching Kung, Fang-Yie Leu, Predictive intelligence in harmful news identification by BERT-based ensemble learning model with text sentiment analysis, Information Processing & Management, Volume 59, Issue 2, 2022, 102872, ISSN 0306-4573, https://doi.org/10.1016/j.ipm.2022.102872.

[52] Annamoradnejad, Issa, and Gohar Zoghi. "Colbert: Using bert sentence embedding for humor detection." arXiv preprint arXiv:2004.12765 1, no. 3 (2020).

[53] Swathi Pola, Manna Sheela Rani Chetty,Behavioral therapy using conversational chatbot for depression treatment using advanced RNN and pretrained word embeddings,Materials Today: Proceedings, 2021, ISSN 2214-7853,https://doi.org/10.1016/j.matpr.2021.02.521.

[54] Sherstinsky, A. (2020). Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. Physica D: Nonlinear Phenomena, 404, Article

[55] J. Saxe and K. Berlin. (2017) “eXpose: a character-level convolutional neural network with embeddings for detecting malicious URLs, file paths, and registry keys.” Cornell University, New York, USA, https://arxiv.org/abs/1702.08568v1.

[56] Drosouli, Ifigenia, Athanasios Voulodimos, Paris Mastorocostas, Georgios Miaoulis, and Djamchid Ghazanfarpour. "TMD-BERT: A Transformer-Based Model for Transportation Mode Detection." Electronics 12, no. 3 (2023): 581.

[57] Le, Xuan Hien & Ho, Hung & Lee, Giha & Jung, Sungho. (2019). Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting. Water. 11. 1387. 10.3390/w11071387.

[58] X. Xiao, D. Zhang, G. Hu, Y. Jiang, and S. Xia. (2020) “CNN-MHSA: a convolutional neural network and multi-head self-attention combined approach for detecting phishing websites.” Neural Networks, 125, 303-312, http://doi.org/10.1016/j.neunet.2020.02.013.

[59] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pretraining of deep bidirectional transformers for language understanding,” in NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational

Linguistics: Human Language Technologies - Proceedings of the Conference, 2019. ISBN 9781950737130.

[60] Chen, M., Radford, A., Wu, J., Jun, H., Dhariwal, P., Luan, D., & Sutskever, I. (2020). Generative Pretraining From Pixels. ICML.

[61] H. Chen et al., "Pre-Trained Image Processing Transformer," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 12294-12305, doi: 10.1109/CVPR46437.2021.01212.

[62] Carion, Nicolas & Massa, Francisco & Synnaeve, Gabriel & Usunier, Nicolas & Kirillov, Alexander & Zagoruyko, Sergey. (2020). End-to-End Object Detection with Transformers.

[63] Jumper, J., Evans, R., Pritzel, A. et al. Highly accurate protein structure prediction with AlphaFold. Nature 596, 583–589 (2021). https://doi.org/10.1038/s41586-021-03819-2.

[64] Fuchs, F.B., Worrall, D.E., Fischer, V., & Welling, M. (2020). SE(3)-Transformers: 3D Roto-Translation Equivariant Attention Networks. ArXiv, abs/2006.10503.

[65] Xu, M., Dai, W., Liu, C., Gao, X., Lin, W., Qi, G., & Xiong, H. (2020). Spatial-Temporal Transformer Networks for Traffic Flow Forecasting. ArXiv, abs/2001.02908.

[66] KyoHoon Jin, JeongA Wi, EunJu Lee, ShinJin Kang, SooKyun Kim, YoungBin Kim, TrafficBERT: Pre-trained model with large-scale data for long-range traffic flow forecasting, Expert Systems with Applications,Volume 186,2021,115738,ISSN 0957-4174,https://doi.org/10.1016/j.eswa.2021.115738.

[67] Yeong J, Velasco-Hernandez G, Barry J, Walsh J. Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review. Sensors (Basel). 2021 Mar 18;21(6):2140. doi: 10.3390/s21062140. PMID: 33803889; PMCID: PMC8003231.

[68] O. Banos, M. Damas, H. Pomares, I. Rojas, On the use of sensor fusion to reduce the impact of rotational and additive noise in human activity recognition, Sensors 12 (2012) 8039–8054.

[69] H. Durrant-Whyte, Sensor models and multisensor data fusion, Journal of Robotics Research 6 (1988) 97–113.

[70] E. Waltz, J. Llinas, Multisensor Data Fusion, Artech House, Boston, 1990.

[71] A. Essien, I. Petrounias, P. Sampaio, S. Sampaio, A deep-learning model for urban traffic flow prediction with traffic events mined from twitter, World Wide Web (2020) http://dx.doi.org/10.1007/s11280-020-00800-3, URL http: //link.springer.com/10.1007/s11280-020-00800-3.

[72] Ma, Yanli, Xuefeng Guan, Jun Cao, and Huayi Wu. "A multi-stage fusion network for transportation mode identification with varied scale representation of GPS trajectories." Transportation Research Part C: Emerging Technologies 150 (2023): 104088.

[73] Chen, Runze, Tao Ning, Yida Zhu, Song Guo, Haiyong Luo, and Fang Zhao. "Enhancing transportation mode detection using multi-scale sensor fusion and spatial-topological attention." In Adjunct Proceedings of the 2023 ACM International Joint Conference on Pervasive and Ubiquitous Computing & the 2023 ACM International Symposium on Wearable Computing, pp. 534-539. 2023.

[74] Kong, Fanyu, Yufeng Zhou, and Gang Chen. "Multimedia data fusion method based on wireless sensor network in intelligent transportation system." Multimedia Tools and Applications 79 (2020): 35195-35207.

[75] Richoz, Sebastien, Lin Wang, Philip Birch, and Daniel Roggen. "Transportation mode recognition fusing wearable motion, sound, and vision sensors." IEEE Sensors Journal 20, no. 16 (2020): 9314-9328.
[76] M. Cornacchia, K. Ozcan, Y. Zheng, S. Velipasalar, A survey on activity detection and classification using wearable sensors, IEEE Sens. J. 17 (2017) 386–403.
[77] M. Yu, A. Rhuma, S.M. Naqvi, L. Wang, J. Chambers, A posture recognition-based fall detection system for monitoring an elderly person in a smart home environment, IEEE Trans. Inf. Technol. Biomed. 16 (2012) 1274–1286.
[78] I. You, K.-K.R. Choo, C.-L. Ho, A smartphone-based wearable sensors for monitoring real-time physiological data, Comput. Electr. Eng. 65 (2017) 376–392.
[79] A. Bayat, M. Pomplun, D.A. Tran, A study on human activity recognition using accelerometer data from smartphones, in: E.M. Shakshuki (Ed.), Proceedings of the Ninth International Conference on Future Networks and Communications, Elsevier Science BV, Amsterdam, 2014, pp. 450–457.
[80] M. Janidarmian, A.R. Fekr, K. Radecka, Z. Zilic, A Comprehensive analysis on wearable acceleration sensors in human activity recognition, Sensors 17, (2017) 26.
[81] B Vidya, Sasikumar P,Wearable multi-sensor data fusion approach for human activity recognition using machine learning algorithms, Sensors and Actuators A: Physical, Volume 341, 2022, 113557, ISSN 0924-4247, https://doi.org/10.1016/j.sna.2022.113557.
[82] Chen, Jingcheng, Yining Sun, and Shaoming Sun. 2021. "Improving Human Activity Recognition Performance by Data Fusion and Feature Engineering" Sensors 21, no. 3: 692. https://doi.org/10.3390/s21030692.
[83] C. V. San Buenaventura and N. M. C. Tiglao, "Basic Human Activity Recognition based on sensor fusion in smartphones," 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, Portugal, 2017, pp. 1182-1185, doi: 10.23919/INM.2017.7987459.
[84] Xaviar, Sanju, Xin Yang, and Omid Ardakanian. "Robust Multimodal Fusion for Human Activity Recognition." arXiv preprint arXiv:2303.04636 (2023).
[85] Z.B. Xiao, Y. Wang, K. Fu, F. Wu, Identifying Different transportation modes from trajectory data using tree-based ensemble classifiers, ISPRS Int. J. Geo Inf. (2017).
[86] Available online: https://www.microsoft.com/en-us/download/details.aspx?id=52367&from=https%3A%2F%2Fresearch.microsoft.com%2Fen-us%2Fdownloads%2Fb16d359d-d164-469e-9fd4-daa38f2b2e13%2F (accessed on 12 November 2023).
[87] Yu Zheng, Lizhu Zhang, Xing Xie, Wei-Ying Ma, ”Mining interesting locations and travel sequences from GPS trajectories”, In Proceedings of International conference on World Wild Web (WWW 2009), Madrid Spain. ACM Press: 791-800.
[88] Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, Wei-Ying Ma, “Understanding Mobility Based on GPS Data”, In Proceedings of ACM conference on Ubiquitous Computing (UbiComp 2008), Seoul, Korea. ACM Press: 312-321.
[89] Yu Zheng, Xing Xie, Wei-Ying Ma, “GeoLife: A Collaborative Social Networking Service among User, location and trajectory”, Invited paper, in IEEE Data Engineering Bulletin. 33, 2, 2010, pp. 32-40.
[90] Available online: http://cs.unibo.it/projects/us-tm2017/index.html (accessed on 12 November 2023)
[91] Carpineti C., Lomonaco V., Bedogni L., Di Felice M., Bononi L., "Custom Dual Transportation Mode Detection by Smartphone Devices Exploiting Sensor Diversity",

Proceedings of the 14th Workshop on Context and Activity Modeling and Recognition (IEEE COMOREA 2018), Athens, Greece, March 19-23, 2018.
[92] Available online: http://www.shl-dataset.org/dataset/(accessed on 23 June 2023).
[93] HristijanGjoreski, Mathias Cilibert, Lin Wang, Francisco Javier Ordonez Morales, Sami Mekki, Stefan Valentin, Daniel Roggen, "Locomotion and Transportation Dataset for Multimodal Analytics With Mobile Devices", The University of Sussex-Huawei, 2018.
[94] Teh, H.Y., Kempa-Liehr, A.W. & Wang, K.IK. Sensor data quality: a systematic review. J Big Data 7, 11 (2020). https://doi.org/10.1186/s40537-020-0285-1.
[95] Guyon, I., Gunn, S., Nikravesh, M., & Zadeh, L. A. (2006). Feature extraction: Foundations and applications (Studies in fuzziness and soft computing). New York: Springer.
[96] Lakshminarayan, K., Harp, S.A. & Samad, T. Imputation of Missing Data in Industrial Databases. Applied Intelligence 11, 259–275 (1999). https://doi.org/10.1023/A:1008334909089.
[97] Cabello-Solorzano, Kelsy, Isabela Ortigosa de Araujo, Marco Peña, Luís Correia, and Antonio J. Tallón-Ballesteros. "The Impact of Data Normalization on the Accuracy of Machine Learning Algorithms: A Comparative Analysis." In International Conference on Soft Computing Models in Industrial and Environmental Applications, pp. 344-353. Cham: Springer Nature Switzerland, 2023.
[98] Jo, Jun-Mo. "Effectiveness of normalization pre-processing of big data to the machine learning performance." The Journal of the Korea institute of electronic communication sciences 14, no. 3 (2019): 547-552.
[99] Subramanian, J., & Simon, R. (2013). Overfitting in prediction models–is it a problem only in high dimensions?. Contemporary clinical trials, 36(2), 636-641.
[100] McDonald, Gary C. "Ridge regression." Wiley Interdisciplinary Reviews: Computational Statistics 1, no. 1 (2009): 93-100.
[101] He, Jianghua, and Prabhakar Chalise. "Nested and repeated cross validation for classification model with high-dimensional data." Revista Colombiana de Estadística 43, no. 1 (2020): 103-125.
[102] Kazemitabar, Jalil, Arash Amini, Adam Bloniarz, and Ameet S. Talwalkar. "Variable importance using decision trees." Advances in neural information processing systems 30 (2017).
[103] Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324.
[104] André Altmann, Laura Toloşi, Oliver Sander, Thomas Lengauer, Permutation importance: a corrected feature importance measure, Bioinformatics, Volume 26, Issue 10, May 2010, Pages 1340–1347, https://doi.org/10.1093/bioinformatics/btq134.
[105] Ranstam, Jonas, and J. A. Cook. "LASSO regression." Journal of British Surgery 105, no. 10 (2018): 1348-1348.
[106] Upadhyay, Darshana, Jaume Manero, Marzia Zaman, and Srinivas Sampalli. "Gradient boosting feature selection with machine learning classifiers for intrusion detection on power grids." IEEE Transactions on Network and Service Management 18, no. 1 (2020): 1104-1116.
[107] Sedgwick, Philip. "Pearson's correlation coefficient." Bmj 345 (2012).
[108] Sedgwick, Philip. "Spearman's rank correlation coefficient." Bmj 349 (2014).

[109] Al-Jarrah, Omar Y., Paul D. Yoo, Sami Muhaidat, George K. Karagiannidis, and Kamal Taha. "Efficient machine learning for big data: A review." Big Data Research 2, no. 3 (2015): 87-93.
[110] Jiawei Han, Micheline Kamber, Jian Pei, 12 - Outlier Detection, Editor(s): Jiawei Han, Micheline Kamber, Jian Pei, In The Morgan Kaufmann Series in Data Management Systems, Data Mining (Third Edition), Morgan Kaufmann, 2012, Pages 543-584, ISBN 9780123814791, https://doi.org/10.1016/B978-0-12-381479-1.00012-5.
[111] Verleysen, M., François, D. (2005). The Curse of Dimensionality in Data Mining and Time Series Prediction. In: Cabestany, J., Prieto, A., Sandoval, F. (eds) Computational Intelligence and Bioinspired Systems. IWANN 2005. Lecture Notes in Computer Science, vol 3512. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11494669_93.
[112] Mykola Pechenizkiy, Seppo Puuronen, and Alexey Tsymbal. 2006. The impact of sample reduction on PCA-based feature extraction for supervised learning. In Proceedings of the 2006 ACM symposium on Applied computing (SAC '06). Association for Computing Machinery, New York, NY, USA, 553–558. https://doi.org/10.1145/1141277.1141406.
[113] Zebari, Rizgar, Adnan Abdulazeez, Diyar Zeebaree, Dilovan Zebari, and Jwan Saeed. "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction." Journal of Applied Science and Technology Trends 1, no. 2 (2020): 56-70.
[114] Liu, Raymond, and Duncan F. Gillies. "Overfitting in linear feature extraction for classification of high-dimensional image data." Pattern Recognition 53 (2016): 73-86.
[115] Y. Fu, S. Yan and T. S. Huang, "Correlation Metric for Generalized Feature Extraction," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 12, pp. 2229-2235, Dec. 2008, doi: 10.1109/TPAMI.2008.154.
[116] Mutlag, Wamidh K., Shaker K. Ali, Zahoor M. Aydam, and Bahaa H. Taher. "Feature extraction methods: a review." In Journal of Physics: Conference Series, vol. 1591, no. 1, p. 012028. IOP Publishing, 2020.
[117] Pearson, Karl. "LIII. On lines and planes of closest fit to systems of points in space." The London, Edinburgh, and Dublin philosophical magazine and journal of science 2, no. 11 (1901): 559-572.
[118] Jolliffe, I. (2011). Principal Component Analysis. Berlin: Springer.
[119] Anowar, Farzana, Samira Sadaoui, and Bassant Selim. "Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne)." Computer Science Review 40 (2021): 100378.
[120] Jia, W., Sun, M., Lian, J. et al. Feature dimensionality reduction: a review. Complex Intell. Syst. 8, 2663–2693 (2022). https://doi.org/10.1007/s40747-021-00637-x.
[121] Sachin, Deshmukh. "Dimensionality reduction and classification through PCA and LDA." International journal of computer Applications 122, no. 17 (2015).
[122] Choubey, Dilip K., Manish Kumar, Vaibhav Shukla, Sudhakar Tripathi, and Vinay Kumar Dhandhania. "Comparative analysis of classification methods with PCA and LDA for diabetes." Current diabetes reviews 16, no. 8 (2020): 833-850.
[123] Borade, Sushma Niket, and Ramesh P. Adgaonkar. "Comparative analysis of PCA and LDA." In 2011 International Conference on Business, Engineering and Industrial Applications, pp. 203-206. IEEE, 2011.
[124] Hochreiter, S.; Younger, A.S.; Conwell, P.R. Learning to Learn Using Gradient Descent; Springer: Berlin, Germany, 2001; pp. 87–94.

[125] Hochreiter, S.; Schmidhuber, J. Long Short-term Memory. Neural Comput. 1997, 9, 1735–1780.
[126] Rahimpour, H.; Fugate, Q.D.; Kuruganti, T. Non-Intrusive Energy Disaggregation Using Non-Negative Matrix Factorization with Sum-to-k Constraint. IEEE Trans. Power Syst. 2017, 32, 4430–4441.
[127] Kaselimi, M.; Doulamis, N.; Doulamis, A.; Voulodimos, A.; Protopapadakis, E. Bayesian-optimized Bidirectional LSTM Regression Model for Non-intrusive Load Monitoring. In Proceedings of the ICASSP 2019-IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 2747–2751.
[128] Bardenet, R.; Balázs, K. Surrogating the surrogate: Accelerating Gaussian-process-based global optimization with a mixture cross-entropy algorithm. In Proceedings of the 27th International Conference on Machine Learning (ICML 2010) Omnipress, Haifa, Israel, 21–24 June 2010.
[129] Delli Priscoli, F.; Giuseppi, A.; Lisi, F. Automatic Transportation Mode Recognition on Smartphone Data Based on Deep Neural Networks. Sensors 2020, 20, 7228.
[130] Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. International Conference on Learning Representations. arXiv 2014, arXiv:1412.6980.
[131] Vig, J. A Multiscale Visualization of Attention in the Transformer Model. arXiv 2019, arXiv:1906.05714.
[132] Madabushi, H.T.; Kochkina, E.; Castelle, M. Cost-Sensitive BERT for Generalisable Sentence Classification on Imbalanced Data. arXiv 2019, arXiv:2003.11563.
[133] Edwards, A.L. The Correlation Coefficient. In An Introduction to Linear Regression and Correlation; W. H. Freeman: San Francisco, CA, USA, 1976; pp. 33–46.
[134] Chicco, D.; Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genom. 2020, 21, 6.
[135] Cohen, J. A coefficient of agreement for nominal scales. Educ. Psychol. Meas. 1960, 20, 37–46.
[136] Landis, J.R.; Koch, G.G. The measurement of observer agreement for categorical data. Biometrics 1977, 33, 159–174.
[137] Worldwide Investments in Light-Rail & Metro Rail Projects 2019–2025. STATISTA. Available online: https://www.statista.com/statistics/1142868/investment-in-light-rail-projects-worldwide/ (accessed on 12 November 2023).
[138] Hamed, M.M.; Al-Masaeid, H.R.; Said, Z.M.B. Short-term prediction of traffic volume in urban arterials. J. Transp. Eng. 1995, 121,249–254.
[139] Williams, B.M. Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling. Transp. Res. Rec. 2001, 1776,194–200.
[140] Williams, B.M.; Hoel, L.A. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis andempirical results. J. Transp. Eng. 2003, 129, 664–672.
[141] Milenkovi ́c, M.; Švadlenka, L.; Melichar, V.; Bojovi ́c, N.; Avramovi ́c, Z. SARIMA modelling approach for railway passenger flow forecasting. Transport 2018, 33, 1113–1120.
[142] Wang, Y.; Han, B.; Zhang, Q. SARIMA model-based passenger flow prediction of Beijing subway station. Transp. Syst. Eng. Inf.2015, 15, 205–211.

[143] Liu, T.; Wang, J.; Yang, B.; Wang, X. NGDNet: Non uniform Gaussian-label distribution learning for infrared head pose estimation and on-task behavior understanding in the classroom. Neurocomputing 2021, 436, 210–220.
[144] Voulodimos, A.; Kosmopoulos, D.; Veres, G.; Grabner, H.; Van Gool, L.; Varvarigou, T. Online classification of visual tasks for industrial workflow monitoring. Neural Netw. 2011, 24, 852–860.
[145] Liu, H.; Zheng, C.; Li, D.; Shen, X.; Lin, K.; Wang, J.; Zhang, Z.; Zhang, Z.; Xiong, N.N. EDMF: Efficient Deep Matrix Factorization with Review Feature Learning for Industrial Recommender System. IEEE Trans. Ind. Inform. 2022, 18, 4361–4371.
[146] Kosmopoulos, D.I.; Voulodimos, A.S.; Doulamis, A.D. A System for Multicamera Task Recognition and Summarization for Structured Environments. IEEE Trans. Ind. Inform. 2013, 9, 161–171.
[147] Zou, L.; Shu, S.; Lin, X.; Lin, K.; Zhu, J.; Li, L. Passenger flow prediction using smart card data from connected bus system based on interpretable xgboost. Wirel. Commun. Mob. Comput. 2022, 2022, 5872225.
[148] Xu, Z.; Zhu, R.; Yang, Q.; Wang, L.; Wang, R.; Tong, L. Short-term bus passenger flow forecast based on the multi-featuregradient boosting decision tree. In Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery; Springer International Publishing: Cham, Switzerland, 2020; Volume 1, pp. 660–673.
[149] Liu, T.; Wang, J.; Yang, B.; Wang, X. Facial expression recognition method with multi-label distribution learning for non-verbal behavior understanding in the classroom. Infrared Phys. Technol. 2021, 112, 103594.
[150] Liu, H.; Zheng, C.; Li, D.; Zhang, Z.; Lin, K.; Shen, X.; Xiong, N.N.; Wang, J. Multi-perspective social recommendation method with graph representation learning. Neurocomputing 2022, 468, 469–481.
[151] Huang, J.; Shao, F.; Yang, S. Passenger flow prediction based on recurrent neural networks and wavelet transform. J. Phys. Conf.Ser. 2020, 1486, 022021.
[152] Du, B.; Peng, H.; Wang, S.; Alam Bhuiyan, Z.; Wang, L.; Gong, Q.; Liu, L.; Li, J. Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction. IEEE Trans. Intell. Transp. Syst. 2019, 21, 972–985.
[153] Liu, Y.; Liu, Z.; Jia, R. DeepPF: A Deep Learning Based Prediction Architecture for Metro Passenger Flow. Transp. Res. Part CEmerg. Technol. 2019, 101, 18–34.
[154] Xiong, Z.; Zheng, J.; Song, D.; Zhong, S.; Huang, Q. Passenger flow prediction of urban rail transit based on deep learningmethods. Smart Cities 2019, 2, 371–387.
[155] Emmanuel, T.; Maupong, T.; Mpoeleng, D.; Semong, T.; Mphago, B.; Tabona, O. A survey on missing data in machine learning.J. Big Data 2021, 8, 140.
[156] Sarker, I.H. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. SN Comput.Sci. 2021, 2, 420.
[157] Wang, Y. Graph neural network in traffic forecasting: A review. In Proceedings of the 3rd International Conference on Robotics Systems and Automation Engineering (RSAE), New York, NY, USA, 28 May 2021; pp. 34–39.28.
[158] Jiang, W.; Luo, J. Graph neural network for traffic forecasting: A survey. Expert Syst. Appl. 2022, 23, 117921.
[159] Liang, F.; Qian, C.; Yu, W.; Griffith, D.; Golmie, N. Survey of graph neural networks and applications. Wirel. Commun. Mob.Comput. 2022, 28, 9261537.
[160] Guo, R.; Jiang, Z.; Huang, J.; Tao, J.; Wang, C.; Li, J.; Chen, L. BikeNet: Accurate bike demand prediction using graph neural networks for station rebalancing. In Proceedings of the 2019 IEEE Smart World, Ubiquitous Intelligence & Computing,

Advanced& Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Leicester, UK, 19–23 August 2019; IEEE: PiscatawayTownship, NJ, USA, 2019; pp. 686–693.

[161] Chen, Z.; Wu, H.; O'Connor, N.E.; Liu, M. A comparative study of using spatial-temporal graph convolutional networks for predicting availability in bike sharing schemes. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 19–22 September 2021; IEEE: Piscataway Township, NJ, USA, 2021; pp. 1299–1305.

[162] Hamad, S.Y.Y.; Ma, T.; Antoniou, C. Analysis and Prediction of Bike sharing Traffic Flow–Citi Bike, New York. In Proceedings of the 7th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), Heraklion, Greece,16–17 June 2021; pp. 1–8.

[163] Wang, J.; Zhang, Y.; Wei, Y.; Hu, Y.; Piao, X.; Yin, B. Metro passenger flow prediction via dynamic hypergraph convolution networks. IEEE Trans. Intell. Transp. Syst. 2021, 22, 7891–7903.

[164] [141]Zeng, J.; Tang, J. Combining knowledge graph into metro passenger flow prediction: A split-attention relational graph convolutional network. Expert Syst. Appl. 2023, 213, 118790.

[165] Wu, F.; Zheng, C.; Zhang, C.; Ma, J.; Sun, K. Multi-View Multi-Attention Graph Neural Network for Traffic Flow Forecasting.Appl. Sci. 2023, 13, 711.

[166] Zenodo. A Passenger Flow Data Set Collected in the Metro System of Hangzhou, China [Data Set]; Zenodo: Washington, DC, USA, 2019.

[167] CityBike. Available online: https://citibikenyc.com/system-data (accessed on 21 June 2023).

[168] Glocker, B.; Pauly, O.; Konukoglu, E.; Criminisi, A. Joint classification-regression forests for spatially structured multi-object segmentation. In Computer Vision—ECCV 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 870–881.

[169] Linusson, H. Multi-Output Random Forests. 2013. Available online: https://api.semanticscholar.org/CorpusID:122685952(accessed on 21 June 2023).

[170] Cho, K.; Van Merrienboer, B.; Gülçehre, Ç.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1724–1734.

[171] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA,4–9 December 2017.

[172] Gao, F.; Wang, Z.; Liu, Z. Parallel Multi-Graph Convolution Network For Metro Passenger Volume Prediction. In Proceedings of the IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), Porto, Portugal, 6–9 October 2021;pp. 1–10.

[173] Xie, P.; Ma, M.; Li, T.; Ji, S.; Du, S.; Yu, Z.; Zhang, J. Spatio-Temporal Dynamic Graph Relation Learning for Urban Metro Flow Prediction. IEEE Trans. Knowl. Data Eng. 2023, 1–12.

[174] Chai, D.; Wang, L.; Yang, Q. Bike flow prediction with multi-graph convolutional networks. In Proceedings of the 26th ACMSIGSPATIAL International Conference on

Advances in Geographic Information Systems, New York, NY, USA, 6–9 November2018; pp. 397–400.
[175] Sun, J.; Zhang, J.; Li, Q.; Yi, X.; Liang, Y.; Zheng, Y. Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks. IEEE Trans. Knowl. Data Eng. 2020, 34, 2348–2359.
[176] Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to Sequence Learning with Neural Networks. In Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS), Cambridge, MA, USA, 8–13 December 2014; pp. 3104–3112.
[177] Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
[178] Kolassa, S.; Martin, R. Percentage Errors Can Ruin Your Day (and Rolling the Dice Shows How). Foresight Int. J. Appl. Forecast.2011, 23, 21–27.

List of Publications

RELEVANT TO THE THESIS

Journal articles

[J3] Drosouli, Ifigenia, Athanasios Voulodimos, Paris Mastorocostas, Georgios Miaoulis, and Djamchid Ghazanfarpour. "A Spatial-Temporal Graph Convolutional Recurrent Network for Transportation Flow Estimation." Sensors 23, no. 17 (2023): 7534.

[J2] Drosouli, Ifigenia, Athanasios Voulodimos, Paris Mastorocostas, Georgios Miaoulis, and Djamchid Ghazanfarpour. "TMD-BERT: A Transformer-Based Model for Transportation Mode Detection." Electronics 12, no. 3 (2023): 581.

[J1] Drosouli, Ifigenia, Athanasios Voulodimos, Georgios Miaoulis, Paris Mastorocostas, and Djamchid Ghazanfarpour. "Transportation mode detection using an optimized long short-term memory model on multimodal sensor data." Entropy 23, no. 11 (2021): 1457.

Conference papers

[C3] Ifigenia Drosouli, Athanasios Voulodimos, Paris Mastorocostas, Georgios Miaoulis, Djamchid Ghazanfarpour, “A Graph Neural Network based Learning Model for Urban Metro Flow Prediction”, accepted for presentation at the 22nd International Conference on Machine Learning and Applications (ICMLA) 2023, Florida, USA, December 2023.

[C2] Drosouli, Ifigenia, Athanasios Voulodimos, and Georgios Miaoulis. "Transportation mode detection using machine learning techniques on mobile phone sensor data." In Proceedings of the 13th ACM International Conference on PErvasive Technologies Related to Assistive Environments, pp. 1-8. 2020.

[C1] Drosouli, Ifigeneia, Georgia Theodoropoulou, Georgios Miaoulis, and Athanasios Voulodimos. "A process mining approach for resource allocation management in a bike sharing system." In Proceedings of the 24th Pan-Hellenic Conference on Informatics, pp. 327-333. 2020.

NON-RELEVANT TO THE THESIS

Conference paper

[C1] Voulodimos, Athanasios, Paraskevas Karagiannopoulos, Ifigenia Drosouli, and Georgios Miaoulis. "CGVis: A Visualization-Based Learning Platform for Computational Geometry Algorithms." In Addressing Global Challenges and Quality Education: 15th European Conference on Technology Enhanced Learning, EC-TEL 2020, Heidelberg, Germany,

September 14–18, 2020, Proceedings 15, pp. 288-302. Springer International Publishing, 2020.