SAPIENZA
UNIVERSITÀ DI ROMA

Trip Phase Recognition and Transport Mode Classification Through Mobile Sensing Technologies

Faculty of Civil and Industrial Engineering

Department of Civil, Constructional, and Environmental Engineering

Thesis submitted for the degree of Doctor of Philosophy (PhD)

Supervisor: Professor Guido Gentile

PhD Student: Seyed Hassan Hosseini

Cycle XXXVII, A.Y. 2024-2025

January 09, 2025

Abstract

As urban populations grow and sustainability challenges increase, the ability to automatically detect transport modes and recognize trip phases using mobile sensor data has become critical for urban planning, public transport management, and environmental monitoring. This thesis presents a comprehensive approach to transport mode detection and trip phase recognition, utilizing GPS data from mobile devices. The core objective is to develop robust machine learning and deep learning models capable of classifying transportation modes such as walking, cycling, driving, and transit use (bus, metro, etc.) with high accuracy while simultaneously identifying trip phases, including access, egress, and waiting times.

By leveraging large-scale datasets such as the GeoLife and Sussex-Huawei Locomotion datasets, the study applies advanced data preprocessing techniques and designs multiple classification algorithms, including Random Forest and Convolutional Neural Networks (CNNs), to effectively distinguish between different transportation modes. A key contribution of this research is developing a novel framework for trip phase recognition, which segments journeys into distinct stages, enabling the automated calculation of critical transit metrics such as waiting time at public transport stops, access and egress times, and distance to and from transit stations.

The results of this study have wide-ranging implications, including optimizing public transportation systems, improving commuter experience, reducing carbon emissions by encouraging sustainable transportation choices, and providing policymakers and urban planners with actionable insights based on real-world mobility patterns. Furthermore, the thesis presents new key performance indicators (KPIs) to evaluate the accessibility level of public transit stations. This work advances transportation research and lays the groundwork for future developments in smart city initiatives and digital mobility services.

Keywords

Transport mode detection, GPS data, machine learning, deep learning, trip phase recognition

Acknowledgments

First and foremost, I would like to extend my deepest gratitude to my PhD advisor, Prof. Guido Gentile, for allowing me to be part of his esteemed team. His insightful guidance, constant encouragement, and invaluable support have been instrumental in shaping both this work and my academic growth.

I am eternally grateful to my mother, whose unwavering love and strength have been my foundation. To my father, who sadly is no longer with us, I owe so much of who I am; his memory has been a source of inspiration and resilience throughout this journey. To my brothers and sister, I thank you for always believing in me, even during self-doubt, and for your boundless encouragement that propelled me forward. You have taught me resilience, perseverance, and the importance of family.

My heartfelt thanks also go to my lifelong friends, whose unwavering belief in my abilities has inspired me and given me strength.

With immense appreciation

SeyedHassan Hosseini

Contents

List of Tables

List of Figures

List of acronyms

Acronym	Meaning
TMD	Transport Mode Detection
TPR	Trip Phase Recognition
ML	Machine learning
DL	Deep learning
KNN	k-nearest neighbours
TMD	Transport Mode Detection
CNN	Convolutional Neural Network
DNN	Deep Neural Network
LSTM	Long Short Term Memory
RNN	Recurrent Neural Network
ReLU	Rectified Linear Unit
GPS	Global Positioning System
API	Application Programming Interface
GTFS	General Transit Feed Specification
RF	Random Forest

List of Publications

1. Hosseini, Seyed Hassan, and Guido Gentile. "Smartphone-Based Recognition of Access Trip Phase to Public Transport Stops Via Machine Learning Models." Transport and Telecommunication Journal 23.4 (2022): 273-283.
2. Hosseini, Seyed Hassan, et al. "Inferring Station Numbers in Metro Trips Using Mobile Magnetometer Sensor via an Unsupervised K-means Clustering Algorithm." 2023 8th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS). IEEE, 2023.
3. S.Hosseini, S. Pourkhosroa, L. M. B. Miristice, F. Viti, G. Gentile. ,Automated Passengers Trip Phase Recognition and Public Transit Accessibility Level Analysis via Machine Learning Models Using GPS Data, The 103rd Transportation Research Board (TRB) Annual Meeting, January 7 – 11, 2024.
4. Hosseini, Seyed Hassan, et al. "GPS-Based Trip Phase and Waiting Time Detection to and from Public Transport Stops Via Machine Learning Models." *Transportation Research Procedia* 78 (2024): 530-537..
5. S.Hosseini, G. Gentile, L. M. B. Miristice, F. Viti , " Deep Learning-Based Approach to Recognize Passengers Transport Mode and Trip Phases ",24 EEEIC International Conference on Environment and Electrical Engineering & 8 I&CPS Industrial and Commercial Power Systems Europe, June 17-20, 2024. (Presented)
6. S.Hosseini, S. Pourkhosroa, L. M. B. Miristice, F. Viti, G. Gentile., " Deep Neural Networks for Identifying Modes of Transport using GPS Data ", Conference in Emerging Technologies in Transportation Systems (TRC-30), September 2- 4, 2024, Greece. (Presented)

Chapter 1

Introduction

1. Introduction

This chapter aims to introduce the task of automatically identifying modes of transportation from GPS data and recognizing passengers' trip phases. This involves leveraging GPS trajectories to classify the type of transportation mode used, such as walking, cycling, driving, or public transit. It starts with a quick summary of the problem of transport mode detection, looking at its application in different studies and highlighting the functions of sensors and cell phones in particular. The chapter then covers the main functions and real-world uses of transport mode detection. It delves into the goals and motivations behind the research, defining the problem in detail and developing the main research questions. It sets the stage for our research contributions by addressing open research gaps and highlighting recent developments and persistent problems in the field. The chapter concludes by outlining the thesis's general structure.

1.1 Background

Understanding the patterns and motivations behind human mobility is essential for environmental science, public health, and urban planning. The growing prevalence of smartphones has enabled real-time monitoring of urban trips, potentially facilitating faster responses in these fields. The primary objective of this thesis is to explore how smartphones can be used for near-real-time recognition of transportation modes. Moreover, mobile devices now serve as a primary means of tracking individuals' movements, thanks to the inclusion of motion sensors. These devices are especially well-suited for monitoring movement as users typically carry them and are equipped with various inertial sensors such as accelerometers, magnetometers, gyroscopes, GPS, and other sensors. Collecting comprehensive movement data allows researchers and developers to acquire valuable insights for tracking, assessing, and categorizing passengers' trajectories in different categories. Applications employing motion data have shown substantial influence in diverse domains, including transportation [1], [2], and sport [3].

Smartphone operating systems have advanced to offer robust APIs that enable the detection of various activities. These APIs [4], [5] can differentiate between different human activities, such as standing, walking, running, cycling, or being in a vehicle. This capability is achieved by seamlessly integrating sensor data with advanced algorithms that analyze movement patterns in real-time. As a result, Smartphones have become essential instruments in areas such as health and fitness monitoring, navigation, and even safety applications, demonstrating their ability to improve daily life through intelligent technology. Nevertheless, Some public APIs, such as Google's, offer basic transportation mode classifications, but with limited functionality. These APIs can determine whether a user is in a vehicle but do not provide distinctions between specific transport modes such as car, bus, or metro. Instead, general classifications such as walking, driving, or bicycling are offered, without the capability to identify public transport modes in detail. These APIs often struggle with consistently identifying specific modes of transport, especially in complex urban environments, limiting their effectiveness in certain use cases.

1.2 Use case of transport mode detection

The applications of transport mode detection are diverse and tailored to address specific challenges. This section focuses on the use cases of mode detection and the importance of accurately identifying transport modes. The significance of transport mode detection in enhancing transportation systems and services is thoroughly explored by highlighting various use cases.

1.2.1 Travel surveys

Travel surveys are now indispensable for collecting vital information for transportation planning and decision-making. These surveys can provide current data on individuals' and households' characteristics, economic status, and travel patterns. Furthermore, they enhance our comprehension of how trips are connected to everyday tasks' choice, location, and timing. This enables us to improve our travel forecasting techniques and strengthen our capacity to anticipate shifts in daily travel patterns. In this case, transportation-based surveys are an essential tool for various applications in transportation studies, such as trip generation, trip distribution, modal split/direct demand, discrete choice modeling, and route assignment. Most travel survey methods collect data through two distinct approaches: (1) personal and household surveys and (2) detailed single-trip data[6]. Surveys commonly collect data on the origin, destination, purpose, start and end times of trips, and the transport mode used. This information is essential for analyzing travel behavior, identifying mobility patterns, and understanding trip duration and purpose.

One issue with travel surveys is that respondents often find them particularly burdensome due to the frequent daily follow-ups required, which can result in a low response rate. Moreover, more sophisticated techniques have become reliant solely on GPS devices. Several studies have utilized users' GPS data to generate maps of their trajectories. Subsequently, users get a chance to visually authenticate and rectify any detected inaccuracies on the maps [7]. Working with GPS data presents challenges. These challenges include technical issues related to the reliability of GPS signals, battery consumption, and the complexity of data processing. Discussions also encompass social concerns such as privacy and participant compliance[8]. Nevertheless, the recording of GPS data is often plagued by problematic and unreliable information, primarily stemming from errors related to manual data entry. Specifically, errors arise in the identification of trip origins and destinations, resulting in inaccuracies [9]. Furthermore, mistakes in self-reported demographic data and the inherent variability of human behavior can impair the prediction effectiveness of the models[10]. There is a need to automatically extract trip details information from raw GPS data without human intervention, as this approach minimizes the errors associated with human reporting.

Furthermore, tracking data offers several advantages over traditional surveys that rely on interviews or the manual reporting of trip diaries. Users' effort is significantly reduced, making it possible to collect data across several days. Additionally, it is easy to scale up to many users when data collection is done using smartphone applications. Moreover, GPS tracking enables the collection of comprehensive and precise data, such as visited locations, precise departure time, and travel distance. Consequently, tracking data can serve as a complement to traditional surveys, which are not well-suited for collecting this type of information. Moreover, privacy concerns may arise, particularly when there is a significant delay between data collection and the respondents' completion of the surveys [7]. Inferring travel information, such as modes of transportation, using GPS tracks can effectively address the aforementioned issues.

1.2.2 Transportation studies

Cities conduct transportation studies to comprehend the movement patterns of individuals within their boundaries. These studies empower municipalities and urban managers to make well-informed decisions to enhance immediate traffic conditions and facilitate long-term urban development. Conventional transportation studies depend on extensive surveys and these surveys provide solely fixed information from a particular timeframe, which might not fully encompass the entirety of people's mobility patterns. Transportation studies that incorporate transport mode detection via

smartphones offer the advantages of being continuous, scalable, and providing more detailed insights into travel nuances that traditional methods might miss. Travelers normally have two main alternatives to travel from one place to another: motorized (cars, buses, or trains) and non-motorized (bicycles or walking) modes of transportation. In this regard, the efficient planning and construction of public transportation networks depends on high-quality data. In many transportation studies, travel surveys serve as valuable tools and are frequently utilized in academic research [11], [12]. Additionally, manual trip diaries and questionnaires are two labor-intensive, time-consuming, and prone to error traditional means of gathering travel data. These drawbacks emphasize the need for increasingly sophisticated and automated methods to precisely record and examine travel patterns.

To understand travelers' preferences, monitoring passengers' behavior is crucial for developing a service that is more attuned to the needs of its users. Mobile phones are a ubiquitous device that most passengers regularly carry during their daily commutes. Their usage has now reached full penetration in most countries, making smartphones an invaluable resource for urban planners to understand and analyze passenger behavior and travel patterns more precisely. Most of the mobile phones are equipped with a variety of sensors. GPS and inertial sensors are common, and collecting high-resolution, continuous data on individuals' movements is now possible. This wealth of data provides an unprecedented opportunity to infer detailed travel behavior, including detecting transport modes, with greater accuracy and efficiency.

1.2.3 Carbon monitoring

The use of private modes of transport is a significant contributor to individual carbon emissions. Understanding how individuals and communities utilize various transportation modes can aid in reducing these emissions. Some smartphone applications track different types of transportation and convert them into corresponding emission values. By observing the variations in pollution levels among other modes of transportation, individuals can make informed decisions on their preferred mode of transportation[13]. Furthermore, gamification[14] can also be a way to motivate users, allowing them to compare their emissions with those of their community.

1.2.4 Smartphones and sensors

The widespread use of smartphones, which are equipped with a range of sensors, has revolutionized the ability to detect different modes of transportation accurately. These sensors enable smartphones to gather data on movement patterns, speed, direction, and environmental conditions, and accurately identifying various transportation modes. Key points on how smartphones achieve this include their reliance on essential sensors such as GPS, accelerometers, gyroscopes, magnetometers, barometers, and microphones. These sensors enable accurate tracking and identification of different transportation modes, allowing for calculating corresponding emission values. By monitoring this movement over a period, various transport mode identification algorithms may accurately categorize different forms of transportation. Motion sensors such as accelerometers and gyroscopes can also detect a device's movement. On the other hand, barometers, which are instruments used to measure pressure, can determine the current elevation of a device and any elevation changes. These sensors are currently widely used in cell phones due to their usefulness in numerous applications.

1.3 Research motivation and objectives

The necessity for effective and sustainable transportation systems has increased due to population growth. As urban populations expand, the demand for transportation rises, leading to more congestion and the urgent need for efficient transportation solutions. Additionally, growing awareness of climate change and pollution has heightened the push for sustainable transportation options that reduce greenhouse gas emissions and reliance on fossil fuels. Effective transportation systems also play a crucial role in enhancing the quality of life by reducing commute times, improving accessibility, and ensuring safer travel conditions. Moreover, understanding and optimizing urban mobility is crucial to alleviating traffic congestion, minimizing environmental impact, and enhancing the overall quality of life. In this context, identifying transport modes and trip phases is vital, as it provides comprehensive insights into travel behaviors and patterns. Developing reliable and accurate models for trip phase recognition and transport mode detection using sparse GPS data is the motivation behind this research. The primary objective of the research effort is to use sparse GPS data analysis to better understand human travel behavior. Furthermore, this research culminates in a framework that identifies various passenger trip phases in urban environments, such as access, egress time and distance, and waiting times at public transport stops. Additionally, it introduces new key performance indicators (KPIs) to assess the accessibility levels of public transit stations. The specific aims of the research are as follows:

1. Development of algorithms that are capable of accurately detecting various modes of transport using data obtained from GPS sensors.

2. Automating trip phase recognition techniques to identify different trip stages (e.g., access, egress, waiting time) within journeys using sparse GPS data.

3. Presenting new public transport accessibility and performance KPI metrics via attracted trips by public transit stops.

4. Promote sustainable mobility solutions by providing detailed insights into travel behavior, this research aims to support the development of smarter, more sustainable urban mobility solutions. The findings can inform transportation planning, policy-making, and the creation of innovative mobility services tailored to the needs of modern urban environments.

By achieving these objectives, the research aims to contribute significantly to the field of transport mode detection and trip phase recognition to provide valuable tools for understanding human travel behavior from minimal and unobtrusive data sources.

1.4 Problem definition and research questions

Transport Mode Detection (TMD) involves classifying the mode of transit used by passengers based on sensor data collected from mobile devices, such as GPS. The core challenge lies in accurately processing this data to identify various modes of transport, including walking, cycling, and driving, across different environments. A crucial extension of this problem is trip phase detection, which focuses on identifying distinct phases within a journey, such as access, egress, and waiting times at public transport stops. Detecting these trip phases is essential for providing a detailed understanding of passenger behavior and travel patterns. Both transport mode detection and trip phase recognition are critical in advancing the automation of urban mobility analysis, supporting sustainable transportation planning, and enhancing public transit systems. This technique can be implemented in

two ways: as a client-server configuration, where the device sends raw data to a server for transport mode detection, or as a direct implementation on the embedded device itself.

To fully understand and achieve this objective, dissecting the primary issue into multiple smaller components is necessary. The research questions can be summarized as follows:

1. How to preprocess GPS signals and apply ML and deep neural networks?

2. What are the essential features of an optimal approach for identifying the mode of transportation at different stages of a trip (such as stationary, walking, cycling, bus, automobile, train, or subway)?

3. Can raw GPS data be effectively utilized to accurately determine trip phases, such as access, egress, and passenger waiting times, at public transit stops?

4. What model architecture is best suited for a Convolutional Neural Network (CNN) in transport mode detection?

5. Is it possible to evaluate the accessibility level of each public transport stop using data collected from passengers who used that specific stop?

Our attempt to answer these questions led to the following contributions:

1. A robust classification algorithm is proposed that leverages ML and DL techniques to categorize trip segments into various modes of transport.

2. A deep learning-based framework is presented for detecting passengers' trip phases in urban environments.

3. New indicators are introduced to evaluate the performance of public transit stations from the user's perspective.

Finally, the evaluation scenario is precisely defined by employing three distinct sets that are carefully separated to prevent any contamination of the test set.

1.5 General thesis research structure

The structure of this thesis is as follows: Section 2 presents an in-depth review of the existing works relevant to the topic. Section 3 offers a comprehensive explanation of the methodology, including datasets, ML and DL architectures, and the trip phase detection framework. Additionally, it includes a concise analysis of the extracted features. The results are discussed in Section 4. Finally, Section 5 presents the conclusions, summarizing the main findings and their implications for practice. Limitations of the research and potential directions for future work are also discussed.

Chapter 2

Literature Review

2. Literature Review

This chapter provides a comprehensive review of the current state-of-the-art techniques in transport mode detection, as well as methods for detecting access, egress, and waiting times from passenger trajectories. Initially, it outlines the various data sources employed in transport mode detection and examines their respective results. Subsequently, it delves into data preprocessing methodologies, differentiating between approaches for models that utilize feature-extracted data and those that rely on raw signal data. Finally, it provides an in-depth analysis of the advanced ML and DL models that are currently employed in the field of transport mode detection and trip phase recognition.

2.1 Sensors data and datasets

Recent studies in transport mode detection have shown promising results by leveraging a variety of diverse data sources. Data collected from various sensors in mobile phones offer unique advantages and challenges. This section presents an overview of the various sensors available for detecting different transportation modes, along with the corresponding outcomes achieved.

2.1.1 Global positioning system

Global Positioning System (GPS) is a satellite-based navigation system that allows for accurate determination of position and velocity. It operates through a network of satellites that transmit signals to GPS receivers, which then calculate their precise location on Earth. GPS is widely used in various applications, including navigation, mapping, and tracking. Recent studies highlight its application in monitoring physical activity, providing detailed insights into human movement, speed, and location accuracy under different conditions[15]. The combined GPS data points can be utilized to calculate the velocities, and accelerations, enabling the deduction of the individual's mode of transportation. Using GPS in smartphones has revolutionized data collection and positional tracking, providing significant accuracy and versatility. GPS data is frequently used for various applications, including crowd-sourced data collection and monitoring of personal activities through mobile apps[16]. Additionally, utilizing GPS data from individuals raises significant privacy issues, as tracking transportation modes requires continuous monitoring of a person's location. There are several concerns regarding the use of GPS data, including specific trade-offs related to accuracy and battery consumption [17].

2.1.2 Accelerometer and magnetometer

The utilization of accelerometers in mobile phones has significantly advanced over the past decade, contributing to a wide array of applications and research in various fields. Accelerometers, which measure the acceleration of the device in 3-dimensional space, have become integral to the functionality of modern smartphones[18]. Transport mode recognition with an accelerometer sensor involves using the accelerometer to determine distinct acceleration patterns in different directions and then matching these patterns to certain forms of transportation. The accelerometer sensor also quantifies the gravitational acceleration. The presence of the gravitational component on different axes depends on the orientation of the device. The readings are adjusted to eliminate the influence of

gravity, resulting in an orientation-independent representation of the device's acceleration. This sensor is widely employed for user interface enhancements, such as automatic screen rotation and motion-based gaming, as well as for health and fitness tracking through activity recognition and step counting [19]. Recent studies have explored their potential in more complex health monitoring systems, detecting falls in the elderly, and monitoring physical rehabilitation[20]. Furthermore, accelerometers are pivotal in enhancing the accuracy of navigation systems when GPS signals are weak or unavailable[21]. The integration of accelerometers with machine learning algorithms has opened new avenues for context-aware applications and human-computer interaction [22]. Overall, the deployment of accelerometers in mobile phones represents a versatile and evolving technology, fostering innovative solutions across diverse domains. The accelerometer can be sampled at different frequencies, which affects both battery usage and sensor accuracy. One key advantage of the accelerometer is its lower privacy concerns compared to GPS, as tracking a phone's acceleration reveals less personal information than pinpointing its exact location.

Another sensor commonly found in most mobile phones is the magnetometer. This sensor detects magnetic field strength along three axes, measured in μT [23], [24] .It measures the Earth's magnetic field by utilizing the Hall effect. However, the sensor is subject to noise from its surrounding environment. The noise can originate from the magnetic fields generated by the smartphone's internal components. Noise from magnetic fields can also come from nearby devices in the surrounding environment[24]. Although this sensor cannot be used alone for transport mode detection, its combination with multiple sensors enhances the accuracy of the results.

Several studies have utilized a combination of sensors to predict modes of transport. In[25], an innovative input set for improving transport mode detection using Long Short-Term Memory (LSTM) neural networks was applied. The study enhances detection accuracy by integrating data from accelerometers, gyroscopes, and magnetometers, which capture diverse aspects of movement and orientation. This multi-sensor approach significantly improves the LSTM model's ability to accurately distinguish between modes of transport such as walking, running, biking, and driving. Furthermore, in another study [26] , features derived from an accelerometer, gyroscope, and magnetometer were used as inputs for a RF classifier, achieving an initial classification accuracy of 82.3 %. However, after applying a custom algorithm for post-processing the data, the overall accuracy improved to 95 %. The primary issue with studies that utilize multiple sensors to predict modes of transport is the lack of consideration for battery consumption, along with the potential for the mobile phone to become cumbersome when recording data from several sensors simultaneously.

2.1.3 GeoLife dataset

GeoLife dataset[27], collected from 2007 to 2012, comprises GPS data from 180 individuals living in five different Chinese cities, with the aim of analysing their commuting behaviours. Only 10 percent of the entire dataset was labeled. In this thesis, the focus is solely on the labeled data. Each trajectory is associated with one or more modes of transport, with each mode transition marked by a timestamp, allowing for specific labels to be assigned to each point in the trajectory. The dataset includes the following transport modes (classes): walk, bike, car, taxi, bus, train, subway, boat, airplane, and motorcycle. Following the recommendations of the GeoLife user guide [27], the classes for boat, airplane, and motorcycle are excluded, while the classes for taxi and car, as well as train and subway,

are merged. It is important to note that the data points are not sampled at a uniform rate; some trajectories have a sampling rate of 1 or 2 Hz, while others average as low as 0.02 Hz.

2.1.4 Sussex-Huawei Dataset

The University of Sussex-Huawei Locomotion (SHL) dataset [28] is an extensive annotated dataset capturing various modes of transportation. Collected over seven months with one user in 2017, it involves three participants who saved data for three days engaged in eight different transportation modes in real-life settings in the United Kingdom. The dataset includes multi-modal data from a body-worn camera and four smartphones, carried at typical body locations, 750 hours of labeled locomotion data: Car (88 hours), Bus (107 hours), Train (115 hours), Subway (89 hours), Walk (127 hours), Run (21 hours), Bike (79 hours), and Still (127 hours).

There are different types of saved sensors in dataset including accelerometer (x, y, z in m/s^2), gyroscope (x, y, z in rad/s), magnetometer (x, y, z in μT) where μT stands for microtesla , which is a unit of magnetic field strength, orientation (quaternions as x, y, z vector), gravity (x, y, z in m/s^2), linear acceleration (x, y, z in m/s^2), ambient pressure (hPa), ambient light (lx), battery level (0-100%) and temperature (°C), satellite reception (ID, SNR, azimuth, elevation), WiFi reception (SSID, RSSI, frequency, capabilities), mobile phone cell reception (network type, CID, LAC, MCC, MNS, signal strength), location (latitude, longitude, altitude, accuracy).

2.2 Data Preprocessing

Data preprocessing for transport mode detection using GPS data involves several essential steps to clean, transform, and prepare the data for ML and DL. Initially, data collection is carried out, capturing GPS data (latitude, longitude, altitude, speed, direction, timestamp) in predefined frequencies. The data cleaning process involves handling missing values through removal or imputation, applying filters on data to reduce noises, and removing outliers in both datasets. In this section, all the steps including data cleaning, filtering, and windowing into different segments are covered.

2.2.1 Cleaning and filtering

A prevalent preprocessing method involves utilizing a digital filter to minimize noise and fluctuations in the data. This technique is crucial for enhancing data quality by removing unwanted random variations and jitter, which can obscure the underlying signal. By applying a digital filter, the data becomes smoother and more consistent, making it easier to analyze and interpret.

2.2.1.1 GPS filtering

GPS data is always subject to some errors, which are discussed in detail in [29]. These include satellite clock inaccuracies, leading to timing errors and atmospheric conditions such as ionospheric and tropospheric delays that affect signal speed and propagation. Multipath effects, where GPS signals reflect off surfaces such as buildings and mountains, can introduce additional errors. Orbital errors due to imprecise satellite positions are another significant factor. Additionally, receiver noise

and signal blockage from physical obstructions such as buildings and trees can degrade GPS accuracy. It is essential to filter out noise and achieve smoother GPS data for improved accuracy and reliability. In this study [30], a modified Kalman filter was applied to remove outliers from the calculated speed and acceleration values. In another study, Gauss kernel smoothing [31] technique was applied to remove speed and acceleration outliers from the collected data. Before training any ML and DL model for transport mode detection, it is crucial to filter GPS data to remove noise and correct inaccuracies caused by sensor errors. Proper filtering ensures that the data used for training is clean and reliable, leading to more accurate predictions and improved model performance.

2.2.2 Segmentation

The process of windowing the data involves dividing the sensor signal into smaller segments that can be individually classified. Windowing as shown in **Figure 2.1** is an essential part of the preprocessing step, allowing for easier comparison of segments when each contains the same number of data points. It simplifies the process of identifying patterns and extracting features. Some models require a fixed input size, meaning the input data must always have a consistent number of data points. Windows are defined by a specific number of samples, providing discrete segments of data for analysis. These windows can also overlap, which allows for the capture of transitional or evolving patterns between segments, enhancing the model's ability to detect subtle changes in the dataset. This overlap improves the continuity of the data and ensures that key features are not missed between separate windows, contributing to more accurate and robust results in ML and DL applications.

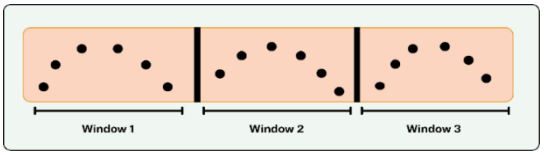

Figure 2.1: Windowing time series data

The sizes of windows vary due to the different models that use these windows. In this paper[32], various window sizes, ranging from 5 seconds to 240 seconds, were tested with a RF classifier, and it was found that 120-second windows were the most effective. Another study [33] reported a 60-second window with a 40% overlap achieved the best results. Therefore, window sizes and overlaps must be set and optimized individually for specific models and the extracted features.

There are also some studies [34], [35] tested a smaller window size (10 seconds) and used it as the main window size of data. In contrast, other researchers [36], applied a larger window size of 30 seconds with 50% overlap. The overlap is more necessary for larger windows, as multiple transport modes can be present in such large windows.

2.2.3 Feature extraction and selection

Feature selection and extraction are fundamental steps in ML models and significantly influence model performance, computational efficiency, and interpretability. In ML, feature selection involves identifying and retaining only the most relevant features from the dataset, thereby reducing dimensionality. This reduction mitigates the curse of dimensionality, decreases overfitting risk, enhances generalization, and accelerates model training.

2.2.3.1 Machine learning feature extraction

In transport mode detection studies, GPS data provides valuable spatial and temporal information, including metrics such as speed, acceleration, jerk, and heading. By utilizing extracted features from GPS data, ML algorithms can accurately classify transport modes, enhancing applications in urban planning, personalized transportation services, and mobility behavior research.

Currently, two predominant methodologies exist in the state-of-the-art for transport mode detection. The first methodology involves the extraction of time and frequency domain features from mobile sensor signals, which are subsequently utilized as inputs for ML models. The second methodology leverages deep learning layers to extract features by identifying spatial relationships within the signal data, which are then employed for classification purposes.

Moreover, there are statistical, frequency domain-based, time-domain-based, peak-based, and segment-based features, as shown in **Table 2.1.** The table provides the feature type and feature names, and the studies that employ these particular features. The statistical features refer to the computed measures derived from the values of the data points. The time-domain features extract characteristics from the entire signal. The frequency-domain features extract information from the frequency domain by the utilization of Fast Fourier Transform (FFT). The following features will be extracted from each segment of the trajectories after the data has been divided into windows.

Feature type	Feature	Reference
Statistical	Mean, Standard Deviation, Minimum Value,Maximum Value	[25], [33], [37], [38]
	Variance, Median, Range	[33], [37]
	Interquartile Range, Kurtosis, Skewness, RMS	[37]
	Minimum Reduction, Maximum Reduction,Maximum Increase, Minimum Increase,Covariance, Harmonic Mean, Quadratic Mean, Arithmetic Mean of Instant Exchange,Quadratic Mean of Instant Exchange	[33]
	Integral, Double Integral, Auto-Correlation	[37]
	Mean-Crossing Rate	[37]

	Zero-Crossing Rate	[37]
Frequency	FFT DC 1,2,3,4,5,6 Hz , Spectral Entropy,Spectrum Peak Position, Wavelet Entropy,Wavelet Magnitude	[37]
	Spectral Energy	[37]
	Spectral Centroid, Spectral Spread, Spectral Flatness, Spectral Roll-Off, Spectral Crest, Spectral Kurtosis	[37]
Peak	Volume, Intensity, Length, Kurtosis, Skewness	[37], [39]
Segment	Variance of Peak Features, Peak Frequency,Stationary Duration, Stationary Frequency	[37], [39]

Table 2.1: Features extracted with motion sensor data from various studies

2.2.3.2 Deep learning feature input

In DL approaches, especially one-dimensional convolutional neural networks, feature extraction is intrinsic to the network architecture. Convolutional neural networks (CNNs) automatically extract hierarchical features through convolutional and pooling layers, where early layers capture low-level features and deeper layers capture high-level features. Feature extraction in 1-D Convolutional Neural Networks (CNNs) for time series data is a critical process that leverages the network's ability to learn hierarchical features directly from raw data, without the need for manual feature engineering. This method has gained significant attention in recent years due to its effectiveness and adaptability in various time series analysis tasks.

In the context of time series data, 1-D CNNs apply convolutional operations along the temporal dimension. The process begins with a convolutional layer that consists of a set of filters (or kernels) that slide over the input time series data. Each filter extracts local patterns by performing element-wise multiplications and summations across the input segments it covers.

A significant advantage of using 1-D CNNs for time series data is their ability to automatically and adaptively learn features that are most relevant to the task at hand, such as classification, regression, or anomaly detection[40].

2.3 Mode of transport classification techniques

Mode of transport classification techniques encompasses a variety of methodologies aimed at identifying the mode of transit being used, such as walking, cycling, driving, or using public transit. These techniques leverage data from various sources, particularly sensors, to determine the mode accurately. Various ML models are employed for mode of transport classification. Traditional models such as Decision Trees, Support Vector Machines (SVM), and k-Nearest Neighbors (k-NN) [41] [42] have been widely used due to their simplicity and interpretability. These models are trained on

labeled datasets where the true mode of transport is known, enabling them to learn the patterns associated with each mode. In recent years, DL techniques, particularly Convolutional Neural Networks (CNNs) [36], [43] and Recurrent Neural Networks (RNNs), have shown superior performance in transport mode classification tasks. CNNs are adept at extracting spatial features from sensor data, making them suitable for processing sequences of accelerometer and GPS readings. RNNs, and especially Long Short-Term Memory (LSTM) [25], [44], [45] networks, are effective in capturing temporal dependencies in sequential data, which is crucial for understanding transitions between different modes of transport.

2.3.1 Machine learning-based models

ML models are categorized into two main types: supervised learning and unsupervised learning [46]. Supervised learning entails the process of extracting a function from training data that has been labeled. The training data contains a collection of observations known as instances, indicated as (x,y). An instance is a single observation of the training data that is utilized to train a model. Unsupervised learning, on the other hand, is closely associated with pattern recognition. In unsupervised scenarios, the dataset lacks a known outcome. The primary objective of unsupervised learning is often to determine which objects should be grouped. This technique aims to identify similarities among groups or discover intrinsic clusters within the data. According to the literature, among most of the ML models used for transport mode detection, the RF algorithm showed the best performance. The logic behind this model will be explored in more detail.

2.3.1.1 Random Forest

Random Forest is an ensemble learning method primarily used for classification and regression tasks. It builds multiple decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees. The approach was first introduced by [47]. There are specific steps to build a random forest-based model. RF operates by first using bootstrap sampling to create multiple subsets of the original training data, where each subset is formed by randomly selecting samples with replacement, allowing some samples to appear multiple times while others can be excluded. For each bootstrap sample, a decision tree is then constructed, introducing additional randomness by selecting a random subset of features at each node and determining the best split from this subset. This method reduces the correlation between trees and enhances the model's robustness. After all trees are built, RF makes predictions by aggregating the outputs of the individual trees, using majority voting for classification tasks and averaging the predictions for regression tasks.

Table 2.2 compares two RF models using different sensor data and their overall accuracy. One model, using accelerometer (ACC), gyroscope (GYR), and sound data, achieved an overall accuracy of 93%, while another model, using accelerometer (ACC), gyroscope (GYR), and magnetometer (MAG) data, achieved a higher accuracy of 95%. This highlights the impact of different sensor combinations on the model's performance in recognizing or classifying data. Various studies have employed different ML models, including k-nearest neighbors (KNN) and decision trees, for transport mode detection.

However, RF consistently achieved the best results, outperforming other models in terms of accuracy and robustness. According to the literature, among various ML models, the RF algorithm

consistently delivers the best results for classifying different modes of transport. Its robust performance is attributed to its ensemble approach, which reduces overfitting and enhances accuracy, particularly in transport mode detection tasks that involve complex and noisy datasets. In this thesis, RF was chosen as the primary ML model for training and testing due to its superior performance in prior research.

The model's results were then compared against those generated by deep learning models to assess the relative strengths and weaknesses of traditional ML versus DL approaches in transport mode classification. This comparison provides a comprehensive evaluation of both methods in terms of accuracy, computational efficiency, and suitability for real-world applications.

Model type	Sensor data	Overall accuracy	Reference
RF	ACC, GYR, SOUND	93 %	[38]
RF	ACC,GYR, MAG	95 %	[33]

Table 2.2: State-of-the-art models for classification of transport mode with RF

2.3.2 Deep learning models

DL models have emerged as powerful tools for transport mode detection, utilizing the rich data provided by GPS to predict transit modes with greater accuracy. Unlike traditional ML methods, DL-based approaches automatically learn hierarchical features from raw data, capturing the inherent complexities and patterns within the data. Many studies have focused on models that rely on manually extracted features from GPS data, these models must be revised. Hand-crafted feature-based models require extensive domain expertise, involve time-consuming feature engineering, and often need help to generalize across different datasets or environments. Furthermore, manually extracted features may not capture the full complexity of the data, limiting the models' accuracy and scalability. By contrast, DL models can overcome these challenges by learning from the data, offering improved performance and adaptability.

2.3.2.1 CNN and RNN

This study [48] addressed the limitations of traditional ML models, which rely heavily on manually crafted features and are susceptible to human error and bias. The CNN model was designed to automatically extract high-level features from raw GPS data, capturing essential motion characteristics such as speed, acceleration, jerk, and bearing rate. Several preprocessing steps have been implemented to improve the quality of the GPS logs before feeding them into the CNN. Various CNN configurations were tested to determine the optimal model. This process involved experimenting with different architectural designs, including layer types, depths, and parameter settings, to find the best configuration for the task at hand. The goal was to identify the model that offered the highest accuracy or performance based on the given data and requirements. Ultimately achieving a high accuracy of 84.8% with the best CNN configurations. The model successfully classified five different transportation modes including walking, bike, bus, car, and train.

In another study [49], a CNN-based method has been introduced for identifying transportation modes using smartphone sensors. The preprocessing steps involve collecting raw sensor data from accelerometers, gyroscopes, magnetometers, and barometric pressure sensors, which are then smoothed with a low-pass filter to eliminate noise. The data is segmented into fixed-length windows and normalized, converting it into a format suitable for CNN processing. The CNN architecture includes convolutional layers for feature extraction, pooling layers for dimensionality reduction, and fully connected layers for classification. The study investigated several transportation modes, including walking, running, cycling, driving, and bus travel, and achieved an overall accuracy of 98.6% in distinguishing between these modes. This high accuracy underscores the model's effectiveness, particularly in differentiating similar modes such as bus and car travel, which pose significant challenges due to their similar acceleration and braking patterns. The authors highlight the potential applications of this approach in urban planning, traffic management, and personalized services, demonstrating that CNN can learn and extract more expressive features than traditional ML methods. The study shows the robustness of using CNNs for transportation mode detection on widely available smartphone platforms.

As stated in the research by [34], a system to detect transportation modes using only accelerometer data from smartphones has been developed. The preprocessing steps involved collecting raw accelerometer data, removing gravity, smoothing the data to eliminate noise, and then converting the data into a one-dimensional magnitude format to avoid issues related to phone orientation. The processed data was divided into small windows, which were then fed into a CNN designed specifically for this task.

The CNN architecture used in this study included multiple convolutional layers, max pooling layers, and fully connected layers. The model was trained to classify the data into seven transportation modes: stationary, walking, bicycling, driving, taking a bus, taking the subway, and taking a train. The authors implemented and tested different CNN architectures to find the optimal configuration for transportation mode detection and achieved a high accuracy rate of 94.48% in identifying the different modes of transportation. This accuracy was significantly higher than those achieved by traditional ML models such as Support Vector Machines (SVM) and Adaptive Boosting, which were used as benchmarks in the study. The CNN's superior performance is attributed to its ability to automatically extract and learn discriminative features from the raw sensor data, as opposed to relying on manually crafted features.

The study also demonstrates the effectiveness of using CNNs for transportation mode detection, highlighting their robustness and efficiency in real-time applications. The proposed system is energy-efficient, as it relies solely on the accelerometer sensor, making it suitable for widespread use on various smartphone platforms. This research has potential applications in urban transportation planning, personalized travel recommendations, and targeted advertising.

A deep learning-based algorithm that integrates long-short-term-memory (LSTM) and convolutional neural network (CNN) layers was presented in [43] to classify eight different modes of transportation using data from smartphone sensors. The study leverages the Sussex-Huawei Locomotion-Transportation (SHL) dataset, which includes sensor data from accelerometers, gyroscopes, linear accelerations, magnetometers, gravity sensors, pressure sensors, and orientation

information. The preprocessing steps involved data integrity checks to ensure uniform labels, addressing class imbalances by oversampling and under-sampling, applying a low-pass filter, and standard scaling. The model architecture includes seven streams of input data, each corresponding to a different sensor, followed by augmentation, LSTM, convolutional, and max pooling layers, which are then merged and processed through additional convolutional and fully connected layers for classification. The algorithm was trained using high-performance computing resources, with the final model achieving an F1 score of 98.96 % on the private test set. The modes of transportation classified include still, walk, run, bike, car, bus, train, and subway. Despite the high accuracy, the model exhibited challenges in distinguishing between train and subway modes, attributed to their similar sensor characteristics.

In this research [50], a comprehensive study on detecting transportation modes using DL techniques tries to classify all motorized and non-motorized transportation modes such as still, walk, run, climbing upstairs, climbing downstairs, bicycle, motorbike, car, metro, train, high-speed rail (HSR), tram, and metrobus. The preprocessing steps involved merging sensor data from accelerometer, gyroscope, and magnetometer sensors, calculating the signal vector magnitude, and applying a moving average low-pass filter to reduce noise. The data was normalized and split into training, validation, and test sets in a journey-independent manner to ensure robustness.

The authors proposed using a Long Short-Term Memory (LSTM) network combined with their healing algorithm for mode detection. The performance of raw sensor data, knowledge-based features, and features obtained via auto-encoder as inputs to the LSTM network was compared. The knowledge-based features, consisting of 27 statistical time-domain features, outperformed the other input types, demonstrating the effectiveness of domain-specific feature engineering over raw data and automatically learned features. The sensors used for prediction included accelerometers, gyroscopes, and magnetometers, all of which provided data on three axes (x, y, and z). The authors examined various hyper-parameters of the LSTM network, such as learning rate, unit size, dropout rate, batch size, and the number of frames, to optimize the performance. The best results were achieved with a learning rate of 0.0005, unit size of 600, dropout rate of 50%, and batch size of 128. The study concluded that the proposed approach offers a robust, orientation-independent, and generic solution for activity recognition and transport mode detection, achieving an impressive accuracy of 95.5%.

In [51], to detect different modes such as bus, train, car, walking, and stationary, a novel approach combines feature cloning and feature fusion techniques within a Convolutional Neural Network (CNN) framework to enhance the performance of transportation mode detection. The sensor data from accelerometers, gyroscopes, and sound sensors were selected and pre-processed by normalizing the data, reshaping it for the CNN input, and splitting it into training and testing sets. The proposed CNN model was trained using the Adam optimizer for 200 epochs with a learning rate of 0.001. The preprocessing steps involved shuffling the dataset, normalizing feature values, reshaping data into suitable dimensions for the CNN, and splitting the dataset into training (85%) and testing (15%) subsets. Feature augmentation was achieved using a vanilla neural network (VNN) to generate enhanced features, which were then fused with original features to form a more comprehensive dataset. This feature fusion technique helped overcome noise and improved the classifier's performance. The CNN model's architecture includes multiple layers of convolutions and pooling

operations, utilizing the Rectified Linear Unit (ReLU) as the activation function and max-pooling for down sampling. The final layer uses a softmax activation function for multi-class classification. The transportation modes detected in the study include stationary, walking, car, bus, and train.

The sensors used for prediction are accelerometers, gyroscopes, and sound sensors. The CNN model's performance was evaluated on three distinct datasets, with the fused and cloned datasets showing superior results compared to conventional ML methods. The proposed feature fusion and cloning-based FC-TMD method achieved high accuracy, with the best results reaching 99.89 % accuracy. This demonstrates a significant improvement over traditional methods and showcases the effectiveness of the proposed DL approach in transportation mode detection.

Findings from [52], discusses the T2Trans model, which is designed for transportation mode detection using sensor data from smartphones. The preprocessing step involves transforming all datasets into a unified matrix and applying a fixed-length sliding window to split the raw data into sequences, which are then fed into the model. Dirty data removal and normalization are conducted to clean and standardize the data. The model uses data from accelerometers, gyroscopes, magnetometers, and barometric pressure sensors.

These sensors are fused to capture the three elements of the x, y, and z axes due to the unknown orientation of the smartphone. The model employs a time convolution network (TCN) to extract temporal features, followed by multiple fully connected layers to produce transportation mode predictions using the Softmax function. The modes of transportation reported in the study include stationary, walking, running, cycling, car, bus, train, and subway. The results show that T2Trans achieved significant improvements over baseline algorithms, with an accuracy of 86.42% on the SHL dataset and 89.13% on the HTC dataset. The performance metrics included accuracy, precision, recall, and F1-score, with T2Trans outperforming other models such as CNN, LSTM, and various ML algorithms.

In this thesis, the primary data source utilized for training the DL model was GPS data. In our prior studies published and included in the appendix of this thesis, various collections of sensor data, including linear accelerometers and magnetometers, were employed to detect transit modes.

2.4 Access and egress phase

There are several methods to calculate walking time to public transportation stops and waiting time inside bus and metro stations. The commonly used techniques include self-reported walking by users about their actual walking distance or time from home or work office to public transport stations [53], [54]. In [53] walking distances were computed using a detailed methodology involving origin-destination survey data and geographic information systems (GIS). The analysis utilizes the 2003 montreal origin-destination (OD) survey, which records detailed trip data for residents in the montreal region. Only trips that involved walking to transit stops were considered, excluding those using other modes or special services, and focusing on home-based trips to ensure consistency. Each trip's origin and destination were geocoded for accuracy. The closest transit stop serving the first route used in each trip was identified using 2003 stop location data, with walking distances measured via the street network. Similarly, walking distances from the destination to the nearest stop on the last route were measured. The model controlled for various factors, such as competing routes, neighborhood characteristics, trip details, and individual characteristics, including variables such as the number of

intersections, distance to downtown, population density, and personal attributes such as household size and vehicle ownership.

Linear regression was employed to examine the relationship between walking distances and various influencing factors, while multi-level regression was utilized to account for nesting within routes and stops, resulting in more accurate estimations. A generalized model, which did not incorporate individual characteristics, was subsequently developed to create variable service areas for each transit stop. This model utilized aggregated data to estimate walking distances. Buffers were generated around each transit stop based on network distances, excluding freeways, and represented different percentiles (mean, 75th, and 85th) to illustrate the areas from which most riders access transit services. The overlapping service areas were then analyzed to identify gaps (areas lacking transit service) and redundancies (areas served by multiple stops/routes), thereby optimizing transit coverage and highlighting inefficiencies.

In another research [54] data was primarily obtained from a questionnaire survey conducted among metro passengers at six stations along Nanjing Metro Line 2, including Xuezelu, Maqun, Xiamafang, Xinjiekou, Jiqingmendajie, and Youfangqiao. These stations were selected based on principles ensuring coverage of major central city stations, adequate distance between sites, and inclusion of various station types (regular, terminal, and transit). The survey, conducted during peak hours in September 2015, distributed 1200 questionnaires and collected 1100 valid responses (91.67% response rate), with 611 respondents (60%) reporting walking to the metro stations. The dataset included demographic information such as gender, age, education level, occupation, and monthly income, as well as travel behavior details such as travel frequency and purpose.

The survey data were complemented by map materials and information from the Nanjing Urban Planning Bureau, and processed using ArcGIS 9.2 to establish a comprehensive dataset. Statistical methods applied included a negative exponential function for walking distance attenuation, network analysis for visualizing pedestrian catchment areas, analysis of covariance (ANOVA) to examine the relationship between walking distance and demographic characteristics, and K-means cluster analysis to categorize passengers.

The study found that demographic characteristics significantly impact walking distances to metro stations, with notable spatial decay effects and varying pedestrian catchment areas based on station type and urban context. Middle-class passengers, generally more educated, tended to walk longer distances. These findings offer valuable insights for optimizing urban and transport planning, emphasizing the need to prioritize low-income communities and ensure public transport connectivity within 750m of metro stations.

In [55], the dataset and methodology used for computing walking distances are crucial elements of the research. The study involved direct observation and tracking of 139 pedestrians from Glen Park and Rockridge BART stations in the San Francisco Bay Area. These participants were followed to their final destinations on weekdays, providing real-world data on pedestrian behavior. The observations included various details such as the starting and ending points of the trips, the routes chosen, and the activities performed during walking.

To compute the walking distances, the researchers used a methodical approach: Direct Observation ensured accurate tracking of the routes taken and the actual distances covered by the participants. Route Mapping involved mapping the routes taken by pedestrians using detailed observations, allowing the researchers to visualize and measure the exact paths chosen by the pedestrians, as opposed to merely calculating straight-line distances. Distance Measurement Tools included the use of geographic information system (GIS) tools and mapping software to provide precise calculations of the walking distances based on the mapped routes.

The combination of direct observation and GIS technology allowed the study to accurately capture the nuances of pedestrian behavior, including the factors influencing route choice and the various activities performed while walking. This meticulous approach provided valuable insights into pedestrian travel patterns and highlighted the importance of considering actual walking paths rather than theoretical shortest distances in urban planning and transit-oriented development.

This research [56] utilized data from two types of surveys: a commuter survey targeting employees in various businesses to collect data on their commuting patterns and a public transport stop survey conducted with public transport users at stops to gather information on walking distances and durations. Self-reported data on walking distances and durations were collected, with a focus on self-reported durations due to their higher reliability.

Data cleaning involved visual inspection of maps to identify reporting errors, setting cut-off boundaries of 3 km for distance and 30 minutes for duration, and excluding outliers and zero values to ensure accuracy. The computation of walking distances included both directly using self-reported distances and converting self-reported durations into distances using an average walking speed of 80 meters per minute, based on norms from previous studies. Statistical analysis was performed using SPSS and STATA programs to calculate means, medians, percentiles, and standard deviations for both self-reported and calculated walking distances, ensuring robust analysis and comparison across different city sizes and types of public transport stops.

The results showed that walking trips to local public transport stops averaged between 328 to 520 meters, while trips to railway stations averaged between 528 to 688 meters, with longer trips observed in both the smallest city (Hamar) and the largest city (Oslo). The study highlighted that shorter walking distances increase the likelihood of public transport use, but higher frequencies and direct connections are more critical for attracting users, especially in smaller cities. This comprehensive approach provided valuable insights for urban planners aiming to enhance public transport services and promote sustainable mobility.

The primary issues with the aforementioned studies include time consumption, reliance on human intervention, and susceptibility to errors, which represent significant drawbacks for many of these approaches. One of the main contributions of this study is the development of a framework designed to automatically extract behavioral data, as well as calculate walking distances and times from passenger information.

This research [57] used GPS data loggers to collect recorded 77,209 trips across various modes of transportation, including auto, bus, bicycle, and walk. The walking or bicycling distance to transit was calculated using GPS trajectories projected onto a road network in the Geographic Information System (GIS) environment. This approach allows for precise measurement of the actual distance traveled to access transit stops, as opposed to self-reported distances or algorithm-estimated distances based on street network configurations. This methodology provides a detailed and accurate representation of the distances and times people travel to access public transport, using advanced GPS tracking and GIS analysis techniques.

Moreover, 41 related papers was investigated in [58] as a review to compute the distances people walk to access public transport. The major data sources were national travel surveys, data from mobility surveys that offer detailed origin and destination data to help construct potential walking routes, custom surveys conducted at transport stations where participants indicate their origin and/or destination, sometimes including map tracing, and objective measurement tools such as GPS trackers combined with accelerometers to track travel behavior accurately.

The methods used to measure walking distance and time to public transport stations include self-reported data where participants record their trips in travel diaries, often suffering from underreporting and rounding errors. Moreover, Following participants where researchers follow participants from the transport station to their destination, providing a realistic representation of walking behavior; and objective tracking using GPS and accelerometers to measure walking distances and times accurately, which is precise but primarily used in studies focused on general physical activity rather than specifically on walking to public transport.

The document identifies several problems and limitations with the methods used for computing walking distance and time to public transport stations: Self-reported data often suffer from underreporting as participants frequently forget to include short walking trips, leading to inaccuracies, and travel times are typically rounded to the nearest 5 minutes, significantly affecting the precision of short walking stages; reliance on participants' memory can introduce recall bias, especially for frequent or routine trips.

The shortest path calculation method assumes travelers always take the shortest path, which is not always true, and it fails to account for personal route preferences or avoidance of certain areas due to safety or attractiveness concerns, often underestimating the actual walked distance due to detours or intermediate stops. Map tracing requires participants to accurately trace their routes on a map, which can be challenging, leading to potential user errors and inaccuracies, with some studies reporting difficulties experienced by participants in this task.

Following participants can result in the observer effect, where participants alter their behavior if they are aware of being followed, leading to non-representative data, and it poses practical and ethical issues, such as privacy concerns, making it difficult to implement on a large scale; this method is typically only feasible for egress trips and not for access trips.

Objective tracking using GPS and accelerometers, while precise, might influence participants' natural walking behavior due to the devices, and most studies using this method focus on overall physical activity rather than specifically on walking to public transport; this method is also resource-intensive, requiring specialized equipment and analysis, making it more expensive and complex to deploy widely.

General issues across these methods include the context specificity of walking distances and times, which are influenced by the local built environment, culture, and public transport system

characteristics, and the wide range of reported distances and times makes it difficult to generalize findings across different studies and locations.

Accelerometers and GPS [59] have been applied for seven days in October 2012 to track their physical activity and school trips. The methods for computing walking distance and time to public transport stations are detailed as follows: Data Collection recording at 1-second intervals to track movement and location. For School-Trip Identification, trips were identified as GPS tracks on weekdays that terminated at school before the end of the school day or originated from school. Trips were manually coded by a researcher with local knowledge, assessing second-by-second GPS points using the tracking analyst tool (ArcGIS). Each trip was manually verified to ensure accurate mode classification.

In this investigation [60] transit users were asked to provide information on trip purposes, starting and ending locations, access and egress modes, transit routes, and demographic characteristics. Respondents could complete the survey either through paper-based questionnaires or on a laptop, which helped avoid manual data entry errors.

Additionally, respondents had the option to choose their starting and ending locations through an interactive map, enhancing location accuracy. Walking distance between home and the transit stop, the dependent variable in this study, was measured as the shortest path in the street network. This was done using the Network Analysis function in ArcGIS. To ensure data accuracy, trips with a walking distance longer than one mile (1609.3 meters) were removed from the analysis. After data pre-processing, a total of 7887 trips were included in the study. The mean walking distance was about 317 meters.

In another study [61] a survey was carried out over a 4-day period in late August 2009. The data includes socioeconomic and demographic information such as income levels, age, gender, and occupation of the respondents; trip-specific information detailing the purpose of the trip, availability of alternative modes of transportation, and frequency of BRT use; and physical characteristics of the stations and corridors measured using GIS (Geographic Information System), which included the station's side street network, path distances, straight-line distances, and the total length of side streets within a 600 m radius around the station.

Additionally, user perceptions were gathered as respondents rated their walking access experiences in terms of protection, comfort, enjoyment, and directness. The distance from home to public transport stations was computed by asking respondents to point out on a map their approximate origins or destinations and their walking routes to or from the BRT stations. These reported walk paths were geo-coded into a GIS along with each BRT station's side street network to calculate relevant distances.

This research [62] facilitated the segmentation of trips into different phases by observing changes in GPS speed and accelerometer activity levels. For example, segments with high GPS speed and low accelerometer activity were identified as motorized travel, whereas segments with moderate speed and higher accelerometer activity indicated active transportation, such as cycling or walking. Self-reported travel modes were utilized to further verify and refine these trip segmentations by comparing them with the GPS-derived speeds and routes. Finally, the integrated data were mapped using GIS software (ArcView 9.2), providing visual representations of commute routes and the physical activity intensity during different trip phases.

2.5 Waiting time detection

Detecting and reducing the waiting time of passengers plays a crucial role in improving the efficiency and overall experience of the public transportation system. This research [63] explores how different factors influence transit users' perceptions of waiting time. The authors conducted a passenger survey and video-recorded waiting passengers at various types of transit stops and stations. The aim was to compare the differences between survey-reported waiting times and actual waiting times recorded on video. The study used regression analysis to explain variations in reported waiting times based on objectively observed waiting times and several other factors, including station and stop amenities, weather, time of day, personal demographics, and trip characteristics. The results indicated that waiting at stops without amenities was perceived to be at least 1.3 times longer than the actual waiting time.

The availability of basic amenities, such as benches and shelters, significantly decreases perceived waiting times for passengers. Additionally, the study revealed that women waiting in environments perceived as insecure reported feeling their waits were considerably longer than the actual duration, in contrast to the waits reported by men under similar conditions. However, the introduction of amenities notably alleviated this perception gap. The authors advocate for prioritizing the provision of basic amenities at transit stations and stops across the entire transit system, particularly at low-frequency route stops and in areas perceived as less safe. This approach aims to enhance security measures and reduce perceived waiting times for all passengers.

2.6 Public transit accessibility evaluation from attracted passengers data

Public transportation (PT) typically does not provide direct service to a passenger's exact origin or final destination, often requiring individuals to walk a certain distance either before boarding or after disembarking at a transit stop. Consequently, the walkability and accessibility of the area surrounding the station significantly influence the appeal of public transit. The ease with which passengers can navigate from the station to their destination directly impacts their overall satisfaction and willingness to use PT services.

Evaluating bus and metro stations requires a multifaceted approach that leverages various geospatial data sources to map transit stops, routes, and their corresponding service schedules. General Transit Feed Specification (GTFS) data plays a crucial role in this evaluation, as it allows for a thorough analysis of transit frequency and reliability, which are essential for understanding service effectiveness.

In addition to schedule data, real-time GPS tracking systems provide valuable insights into actual arrival and departure times, enabling transit agencies to monitor service performance dynamically and identify delays or inconsistencies in the schedule. Furthermore, passenger flow data obtained from smart card systems reveals usage patterns, helping to understand peak travel times, station popularity, and areas requiring additional service or resources.

Qualitative insights can be gathered from passenger satisfaction surveys, which capture user experiences, perceptions of safety, comfort, and overall service quality. This feedback is critical for identifying areas for improvement and enhancing the overall passenger experience.

Walking distance to and from public transit stops and waiting time at stations are among the main indicators to evaluate public transit stops. Walking distance affects how easily passengers can reach transit stops from their origins or final destinations; shorter, well-connected walking paths encourage

greater transit use and ensure inclusivity, particularly for those with mobility challenges. Accurate measurement of waiting time at stations provides insights into service reliability and frequency, directly influencing passenger satisfaction and transit efficiency.

Long wait times can deter users and reduce the attractiveness of public transit. Therefore, minimizing walking distances and waiting times can significantly improve the usability, convenience, and appeal of transit systems, leading to higher ridership and more sustainable urban mobility.

In this paper [58] the average waiting time at the station was determined by measuring the duration from the moment a passenger arrived at the station until they boarded a vehicle. It was assumed that the average waiting time is half of the average interval between services for all lines passing through the station within an hour. Additionally, the walking time to the nearest station was calculated by conducting a network analysis from specific points of interest (POIs) to the closest public transport stop for all modes. The distance from each POI to the station was converted into walking time, based on an assumed average walking speed of 80 meters per minute. No investigations have evaluated the accessibility performance of public transport stops by analyzing GPS-based trips.

The final contribution of this study is the introduction of novel key performance indicators (KPIs) that provide real-time insights into the average waiting time experienced by passengers at specific bus and metro stations, as well as the average access distance and time for each transit stop, calculated based on the trips attracted to that stop. This innovative methodology leverages raw data from GPS trajectories collected directly from passengers, enabling a detailed and accurate analysis of transit system performance. By examining these GPS trajectories, the study can capture actual waiting times and access distances, offering a more precise evaluation of public transit accessibility and efficiency compared to traditional methods. This approach enhances the understanding of passenger behavior and provides a robust framework for improving transit operations and planning.

These real-time KPIs provide valuable insights for transit agencies, enabling them to monitor service efficiency and passenger experiences at various stations. Moreover, they facilitate data-driven decision-making aimed at improving service delivery, optimizing schedules, and enhancing overall user satisfaction. This contribution not only fills a gap in existing literature but also offers practical tools for transit operators to better understand and respond to passenger needs.

Chapter 3

Methodology

The methodology section is structured to present the proposed framework, data handling, analytical processes, and ML and DL explanation. First, the proposed framework is introduced, detailing both the ML and DL components used for transport mode detection and trip phase recognition. This framework is central to the study, integrating various models and techniques to predict and classify travel behaviors from GPS data. Following the introduction of the framework, the two main datasets utilized for predictive modeling are explained.

Next, the preprocessing and filtering steps applied to the raw data are described. These steps are crucial to ensure data quality and accuracy before analysis. The process includes cleaning and filtering out noise, handling missing values, and removing irrelevant data points. The segmentation and windowing of the data are then presented. These techniques break down continuous data streams into manageable segments, enabling more effective feature extraction and pattern recognition. The rationale behind the chosen windowing and segmentation methods is also elaborated.

The logic behind ML and DL models employed in the study are introduced. This includes a discussion of the algorithms, architectures (e.g., CNNs), and training processes used for the predictive tasks. The models are fine-tuned to accurately detect transport modes and recognize trip phases from the processed data. The trip phase recognition step is then explained, emphasizing the underlying logic for calculating access and egress times, as well as the distances. Finally, the methodology section covers the method used to evaluate public transit accessibility, which involves analyzing the data to assess how accessible transit systems are based on passenger movement and service patterns. This evaluation is essential for understanding service efficiency and identifying potential improvements in transit accessibility.

Our proposed approach closely follows the general workflow diagram for transportation mode detection and trip phase recognition tasks as shown in **Figure 3.1**. The methodology employed in this thesis integrates advanced ML and DL techniques to analyze GPS data, leveraging the Geolife and Sussex-Huawei Locomotion dataset. Several steps were implemented to develop an accurate model for predicting transit modes. The main steps are data analysis, data preprocessing, model training, model evaluation, and post-processing.

Data preprocessing involved several critical steps: cleaning and filtering the raw data using techniques such as filtering to eliminate noise, and windowing the continuous data streams into fixed-length segments to facilitate temporal analysis. Feature extraction focused on deriving kinematic attributes from GPS data, including speed, acceleration, jerk, and bearing rate, using Vincenty formula. ML models, particularly RF, were trained on these features to classify transport modes.

Moreover, DL models, especially Convolutional Neural Networks (CNNs) were utilized to capture hierarchical and temporal features for trip phase recognition. Finally, the trip phase detection algorithm uses the model prediction results to identify different trip phases. Moreover, public transport accessibility and performance evaluation algorithms present new KPIs for each single public transit station.

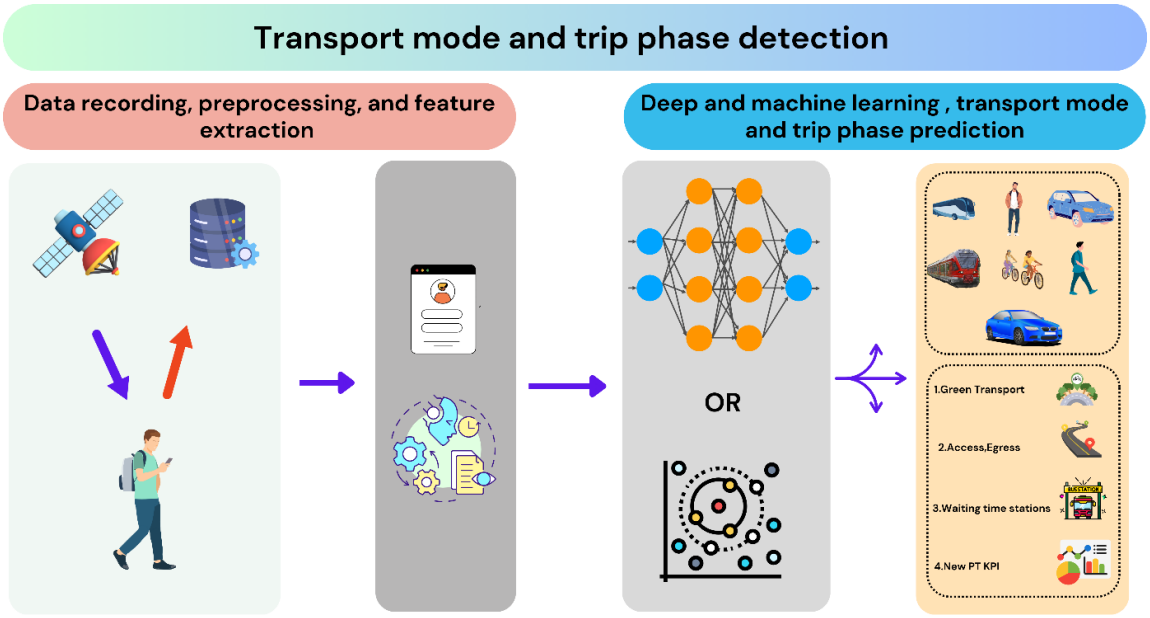

Figure 3.1: Detailed workflow diagram of the proposed approach

3.1 Data Sources

Two primary public data sources are used for training our M and DL models. The first source is the Geolife dataset, which provides extensive labeled GPS data collected from various modes of transportation, including walking, biking, car, and public transit. This dataset offers a rich set of GPS data for understanding and predicting travel modes.

The second source is the Sussex dataset. It includes GPS data and integrates other mobile sensor data, such as accelerometer, gyroscope, and magnetometer readings. Together, these datasets form a robust foundation for developing sophisticated algorithms that can effectively recognize and predict various trip phases, ultimately improving the accuracy and reliability of our travel behaviour analysis.

3.1.1 GPS data (GeoLife)

Geolife dataset, which was prepared by Microsoft Research Asia over a period spanning from April 2007 to August 2012, involving contributions from 182 individual users. The dataset is comprehensively annotated with various transportation modes: airplane, bicycle, boat, bus, car, running, subway, taxi, train, and walking. Nevertheless, this particular research is restricted to examining solely ground-based transportation modes. This dataset covers a diverse range of users' outdoor activities, including daily routines such as commuting to and from home and work and leisure and sports activities. These activities include dining at restaurants, shopping, cycling, sightseeing, and hiking.

This available recorded trajectory dataset can be used in many research areas, such as mining patterns of mobility of single users, user activity detection, location-based social networks, location privacy, and recommending appropriate locations. The whole dataset consists of 17,621 trips with an accumulated distance of 1,292,951 kilometers and a total time duration of 50,176 hours. Trips are recorded by different GPS phones and GPS loggers in different frequencies, and almost 92 percent of the trajectories were saved in the frequency of 1-5 seconds [64].

In the Geolife dataset only 69 users have labeled their trajectory. Therefore, GPS data from these 69 users were selected, whereas the remaining 113 users were excluded. The complete Geolife dataset for these 69 users comprises a total of 12,517,364 rows of GPS coordinates. During the second phase of Geolife dataset analysis, GPS points without labels were removed, resulting in 5,352,505 rows used for the training process.

Geolife has certain instances where the timestamps of two consecutive GPS points for the same user are identical. This may not significantly impact distance calculations, but it poses substantial issues for velocity determination. A zero time interval between consecutive GPS entries results in an undefined or infinite velocity, introducing significant noise that can adversely affect the algorithm's performance. To mitigate this issue, entries with identical consecutive timestamps were filtered out by removing one of the duplicates. This preprocessing step is crucial to maintain the integrity of the data and enhance the reliability of the subsequent ML and DL analyses. **Figure 3.2** and **3.3** illustrate the dataset before and after the deletion of identical rows where the timestamps are equal.

	time	lat	lon	alt	label	user
2008-10-02 16:29:30	37.237749	96.279998	10899.0	metro	10
2008-10-02 16:29:31	37.237474	96.280578	10896.0	metro	10
2008-10-02 16:29:31	37.237615	96.280288	10899.0	metro	10
2008-10-02 16:29:33	37.237334	96.280861	10892.0	metro	10

Figure 3.2: One sample of identical two consecutive timestamps before filtering

	time	lat	lon	alt	label	user
2008-10-02 16:29:30	37.237749	96.279998	10899.0	metro	10
2008-10-02 16:29:31	37.237474	96.280578	10896.0	metro	10
2008-10-02 16:29:33	37.237334	96.280861	10892.0	metro	10
2008-10-02 16:29:34	37.237193	96.281156	10892.0	metro	10

Figure 3.3: One sample of identical two consecutive timestamp after filtering

3.1.2 Sussex dataset

The Sussex-Huawei Locomotion (SHL) dataset [28] is an extensive, well-annotated resource designed for multimodal analytics of locomotion and transportation using mobile devices. Collected data with one and three users over seven months and three days in 2017. Data were gathered using all sensors from four smartphones carried at typical body locations, capturing various modes of transportation such as walking, cycling, and driving across different environments, primarily around London and the southeast of the United Kingdom. The dataset contains 28 context labels, detailing transportation modes, user postures, indoor/outdoor locations, road and traffic conditions, social interactions, and meal times.

The sensors used in smartphones include accelerometers, gyroscopes, magnetometers, barometers, ambient light sensors, and GPS. The data collection frequency for these sensors varied, with the GPS data being recorded at a frequency of 1 Hz (once per second). This rich dataset supports various research applications, including transportation analytics, activity recognition, radio signal

propagation, and mobility modeling, providing a valuable resource for developing machine-learning systems to recognize transportation modes and other contextual information.

In order to accurately detect passengers' waiting time at stations, labeled standing data is required. However, the Geolife dataset lacks labeled standing data, which poses a challenge for this detection task. To address this gap, the Sussex-Huawei Locomotion Dataset, was utilized specifically extracting all GPS points labeled as Stand: Outside, Stand: Inside, Still Sit: Outside, and Still Sit: Inside. This extraction enriches the main training dataset with detailed and diverse standing and still positional data, enhancing its comprehensiveness for further analysis and applications.

Over 190 trips labeled as still were extracted from the dataset. After applying a filter to exclude trips containing less than 200 GPS points (200 seconds), the number of labeled trips was reduced to 132, resulting in a total of 22.5 hours of data. **Figure 3.4** shows some labeled trips that were saved in the UK.

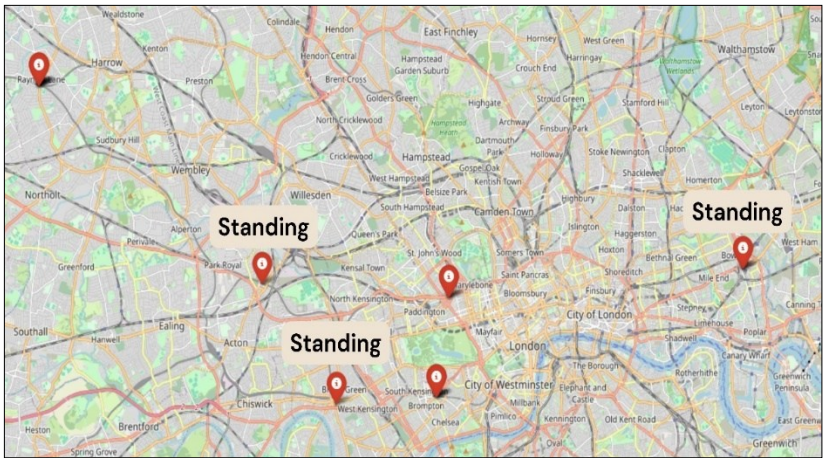

Figure 3.4: Recorded labeled standing data (UK)

Figure 3.5 shows integrating transportation data from two sources: GeoLife (China), which includes multiple transportation modes such as biking, walking, car, bus, and train, and Sussex (UK), focusing on 132 trips totaling 22 hours with standing label. The primary reason for incorporating Sussex standing data was to support one of the key objectives of this study: detecting passengers' waiting times at public transit stations. This data plays a critical role in identifying and accurately measuring the time passengers spend standing or waiting at bus and metro stations, which is essential for evaluating transit service efficiency and enhancing passenger experience. By integrating this data, the framework aims to provide real-time insights into waiting patterns, contributing to more effective public transit accessibility analysis.

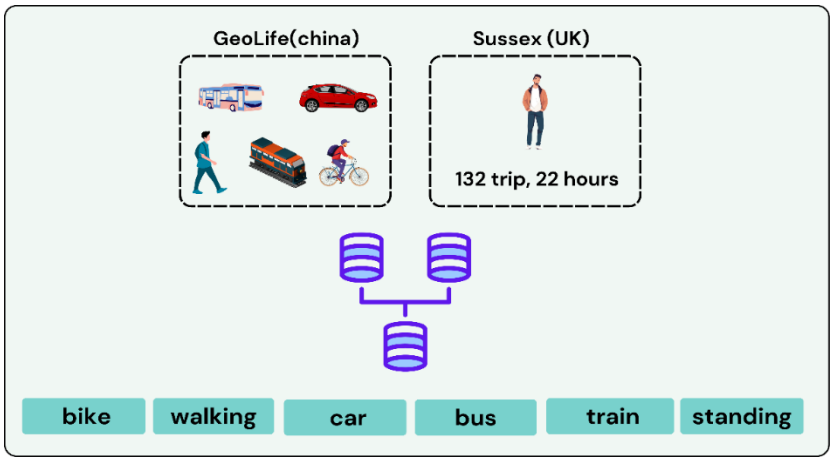

Figure 3.5: Merging data of two different datasets

3.1.3 Trip segmentation

In the next stage of analyzing the Geolife dataset, the focus was on extracting individual trips for each user. To accomplish this, a new trip was defined when the time difference between the timestamps of two consecutive GPS data points exceeded a specified threshold of 1200 seconds following the roles in this research [48]. This threshold was chosen to distinguish between different trips effectively. Finally, a total of 6,399 individual trips were identified recorded with all 64 users. The distribution of these trips, categorized by modes of transportation, is illustrated in the graphs below, which provide a visual representation of the various transportation modes and the frequency of journeys.

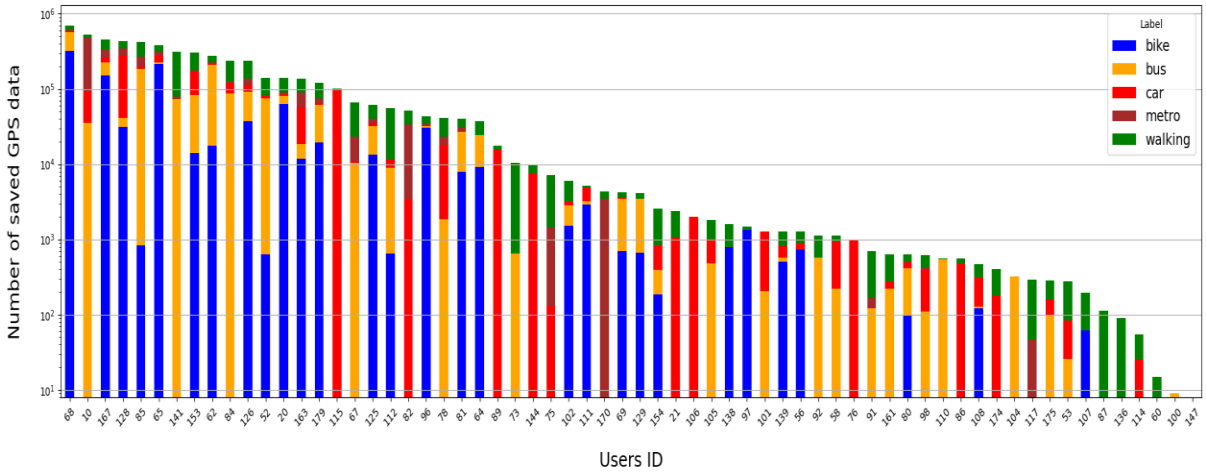

Figure 3.6 : Number of saved GPS points collected by users, categorized by mode of transport

Figure 3.6 shows the number of GPS points saved by each user, categorized by mode of transport. It is evident that user 68 recorded the highest number of GPS points, whereas user 147 recorded the lowest number. For the next stage, five specific modes of transportation were selected as the main sources of labeled data: bike, bus, car, train, and walking. Therefore, other modes such as airplane, boat, running, and motorcycle, were deleted from the dataset. Additionally, taxis were grouped under the car category, and subways were combined with trains under the train category.

Figure 3.7 displays the breakdown of transportation modes by percentage. Each segment of the chart represents a different mode of transport and its proportion of the total saved GPS data. This

visualization indicates that walking has the highest proportion of saved GPS data, followed by bus, bike, metro, and car.

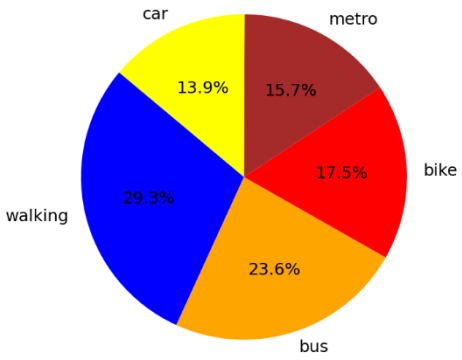

Figure 3.7: The proportion of saved GPS data for each mode of transport in Geolife

Figure 3.8 illustrates the distribution of saved trips among users, showing a pronounced skewness with a few users recording a significantly higher number of trips. User 128 stands out with nearly 800 saved trips, followed by users such as 68, 153, and 85, each with over 400 trips. Understanding this distribution can help in targeting interventions to boost engagement and in efficiently allocating resources to improve system performance.

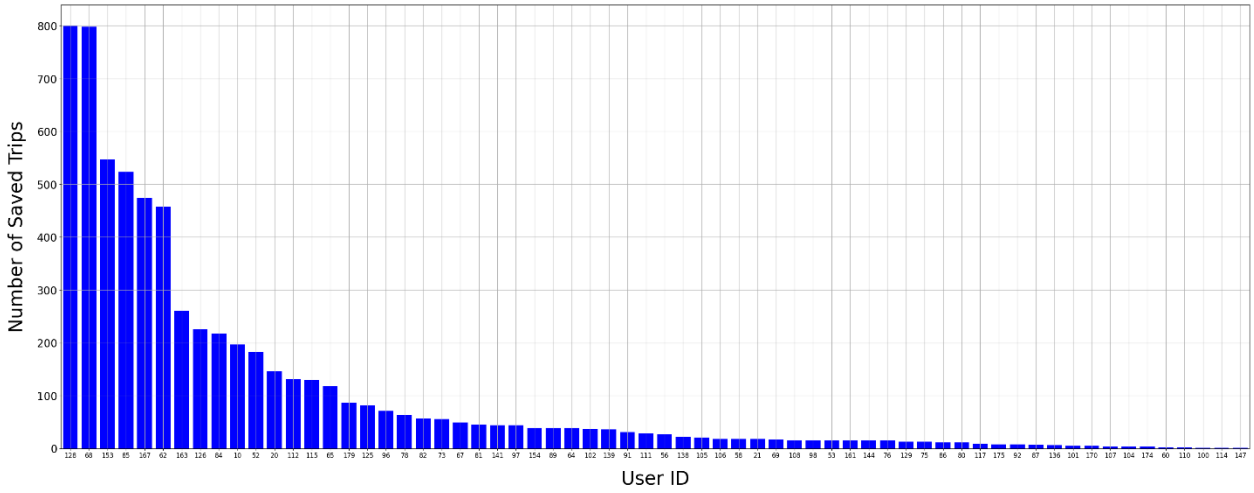

Figure 3.8: Number of trips saved by each user

3.2 Motion attributes from raw GPS data

Plenty of motion characteristics can be extracted from GPS point-based datasets via their geographical coordinates and timestamps. The most significant feature between two consecutive GPS points in a trajectory is the relative distance between these two points (e.g. RD = Relative Distance), which can be computed from many different approaches. The main two widely used formulas are Haversine and Vincenty formula. Vincenty formula, developed by thaddeus vincenty in 1975 [65],

provide accurate methods for calculating distances between two points on the earth's surface using an ellipsoidal model. These formulae are especially useful for geodesic calculations, where the earth's shape is approximated as an oblate spheroid rather than a perfect sphere. **Equation 3.1** and **3.2** shows vincenty formula and are known for their high precision and are often used in geodetic applications.

$$\begin{gathered}
\sin\ \alpha = \sqrt{(\cos\ U_2 \sin\ \lambda)^2 + (\cos\ U_1 \sin\ U_2 - \sin\ U_1 \cos\ U_2 \cos\ \lambda)^2} \\
\cos\ \alpha = \sin\ U_1 \sin\ U_2 + \cos\ U_1 \cos\ U_2 \cos\ \lambda \\
\alpha = \arctan\ \frac{\sin\ \alpha}{\cos\ \alpha} \\
\sin\ \alpha = \frac{\cos\ U_1 \cos\ U_2 \sin\ \lambda}{\sin\ \alpha} \\
\cos^2\ \alpha = 1 - \sin^2\ \mathrm{a} \\
\cos\ (2\alpha) = \cos\ \alpha - \frac{2\sin\ U_1 \sin\ U_2}{\cos^2\ \alpha} \\
\mathrm{C} = \frac{\mathrm{f}}{16}\cos^2\ \mathrm{a}[4 + \mathrm{f}(4 - 3\cos^2\ \mathrm{a})] \\
\lambda + (1 - \mathrm{C})\mathrm{f}\sin\ \alpha\{\mathrm{a} + \mathrm{C}\sin\alpha[\cos\ (2\alpha) + \mathrm{C}\cos\ \alpha(-1 + 2\cos^2\ (2\alpha))]\}
\end{gathered} \tag{3.1}$$

When λ has converged to the desired degree of accuracy (10^{-12} corresponds to approximately 0.06 mm), the second phase of evaluation should apply:

$$\begin{gathered}
\mathrm{u}^2 = \cos^2\alpha\left(\frac{a^2 - b^2}{b^2}\right) \\
A = 1 + \frac{u^2}{16384}(4096 + u^2[-768 + u^2(320 - 175u^2)]) \\
\mathrm{B} = \frac{\mathrm{u}^2}{1024}(256 + \mathrm{u}^2[-128 + \mathrm{u}^2(74 - 47\mathrm{u}^2)]) \\
\Delta\alpha = B\sin\ \alpha\{\cos(2\alpha)\ \ + \frac{1}{4}\ \mathrm{B}(\cos\ \alpha[-1 + 2\cos^2\ (2\alpha)\}] \\
\text{Distance } = \mathrm{bA}(\alpha - \Delta\alpha) \\
\alpha = \text{ angular separation between points} \\
\mathrm{f} = \text{ flattening of the ellipsoid}(\frac{1}{298.257223563}\ in\ \text{WGS - 84 }) \\
\mathrm{U_i} = \arctan\ (1 - \mathrm{f})\tan\ \varphi_\mathrm{i} \\
\varphi = \text{ Latitude of the points} \\
\mathrm{L} = \text{Longitude of the points} \\
\lambda = \text{Difference in longitude of the points on the auxiliary sphere} \\
\mathrm{a} = \text{Length of semi - major axis of the ellipsoid (radius in equator)} \\
b = (1 - \mathrm{f})\mathrm{a} = \text{Length of semi - minor axis of the ellipsoid (radius at the poles)}
\end{gathered} \tag{3.2}$$

Moreover, **Equation 3.3** shows the formula for haversine :

$$\begin{gathered} a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2) \\ c = 2 \cdot \operatorname{atan}2(\sqrt{a}, \sqrt{(1-a)}) \\ \text{Distance} = R \cdot c \\ \varphi = \text{Latitude in Radians},\ \lambda = \text{Longitude in Radians}, \\ R = \text{Earth radius (6,371 km)} \end{gathered} \tag{3.3}$$

The time interval between two GPS points Δt is another primary quantity that can be extracted from the dataset by finding the difference between two consecutive timestamps. Most of the other relevant kinematic features are extracted from these two main quantities, such as speed, acceleration or deceleration, and Jerk. Speed or velocity is the distance variation in the given time interval, which represents how fast a user is moving or traveling. Acceleration is the rate of change of velocity of an object with respect to time. It is a vector quantity and is measured in meters per second squared m/s^2. Jerk is the rate of change of acceleration with respect to time. It represents how quickly the acceleration of an object is changing and is measured in meters per second cubed m/s^3. The aforementioned kinematic attributes for GPS data are computed based on the following formulas:

$$\Delta d = \text{vicinity}\,(p_1[\text{latitude}, \text{longitude}], p_2[\text{latitude}, \text{longitude}]\ ) \tag{3.4}$$

$$\Delta t = t_2 - t_1 \tag{3.5}$$

$$v = \frac{\Delta d}{\Delta t} \tag{3.6}$$

$$a = \frac{\Delta v}{\Delta t} \tag{3.7}$$

$$j = \frac{\Delta a}{\Delta t} \tag{3.8}$$

In Equations 3.4, 3.5, 3.6, 3.7, 3.8 p_1 and p_2 are shown as the latitude and longitude of two consecutive GPS point. The distance (Δd) between two GPS points, combined with the time difference (Δt) between those points allow the calculation of key motion parameters. The mean speed v representing the commuter's average speed between the two points. The mean acceleration or deceleration a is the change in speed over the time interval. Moreover, the j as a jerk is the change in acceleration between the two consecutive GPS points. These calculations provide insights into the commuter's motion dynamics. Using the equations mentioned above, the kinematic features of all GPS points of the whole dataset are calculated.

There is another feature that is the rate of variation in the heading direction and is different in various modes of transport. For instance, cars and public transport have to obey the rules and drive alongside the designed roads, while walking or cycling can have higher bearing rate variation concerning the car or public transport facilities. The bearing rate measures the angle between the line connecting two consecutive points and a reference axis, a south-north axis (the Magnetic or True North axis). The bearing rate is used as an extra motion feature in the absolute difference between the bearing of two GPS points. The range of the bearing rate is a degree from till.

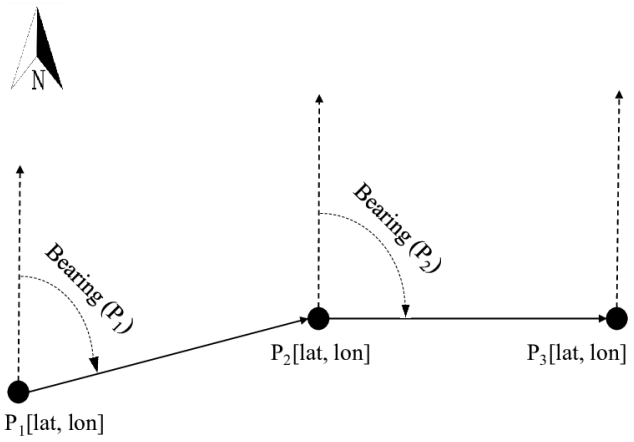

Figure 3.9: Bearing feature of GPS data points

Figure 3.9 illustrates the bearings and positions of three GPS points P_1, P_2 P_3 .It shows the direction (bearing) from P_1 to P_2 and from P_2 to P_3, along with the straight-line distances between these consecutive points. This setup is used in navigation and geospatial analysis to determine the distance and direction between points, aiding in route planning, movement tracking, and geographical mapping.

To calculate the bearing between two GPS points:

$$\theta = \text{atan } 2(\sin(\Delta\lambda) \cdot \cos(\phi_2), \cos(\phi_1) \cdot \sin(\phi_2) - \sin(\phi_1) \cdot \cos(\phi_2) \cdot \cos(\Delta\lambda)) \quad (3.9)$$

In **equation 3.9**, θ is the angle in radians and ϕ_1 and ϕ_2 are the latitudes of the first and second points in radians and $\Delta\lambda$ is the difference in the longitudes of the two points in radians. Steps to Calculate Bearing are as follow:

In **equation 3.10,** latitudes and longitudes from degrees to radians are converted:

$$\begin{aligned} \phi_1 &= \text{lat } 1 \times \frac{\pi}{180} \\ \phi_2 &= \text{lat } 2 \times \frac{\pi}{180} \\ \Delta\lambda &= (\text{lon } 2 - \text{ lon } 1) \times \frac{\pi}{180} \end{aligned} \quad (3.10)$$

Apply the bearing formula ($\mathbf{3.11}$):

$$\theta = \text{atan } 2(sin(\Delta\lambda) \cdot cos(\phi_2), cos(\phi_1) \cdot sin(\phi_2) - sin(\phi_1) \cdot cos(\phi_2) \cdot cos(\Delta\lambda)) \quad (3.11)$$

Finally, convert the result from radians to degrees in **equation 3.12**:

$$\text{Normalize the result to a range of } 0° \text{ to } 360° : \text{bearing } = (\theta \text{ degrees } + 360) \bmod 360 \quad (3.12)$$

Figures 3.10 and **3.11** show a detailed view of a commuter's journey, highlighting variations in speed, acceleration, jerk, and bearing across walking and bus travel modes. **Figure 3.10** shows the

route, starting with a red segment for bus and transitioning to a green segment for walking travel, moving eastward and then north. The speed plot in **Figure 3.11** indicates higher, more consistent speeds during the bus segment, while the walking segment shows lower, variable speeds. The acceleration plot reflects these patterns, with peaks and troughs during bus travel corresponding to stops and starts, and smaller, irregular changes during walking. The jerk plot displays pronounced spikes in the bus segment, indicating abrupt changes in acceleration, and smaller spikes during walking, denoting smoother changes. The bearing plot reveals steadier values with occasional sharp changes during bus travel, likely due to turns, and more frequent, irregular changes during walking. Together, these visualizations provide comprehensive insights into the commuter's motion dynamics throughout the trip.

Figure 3.10: A random bus-walking trip extracted from GeoLife dataset

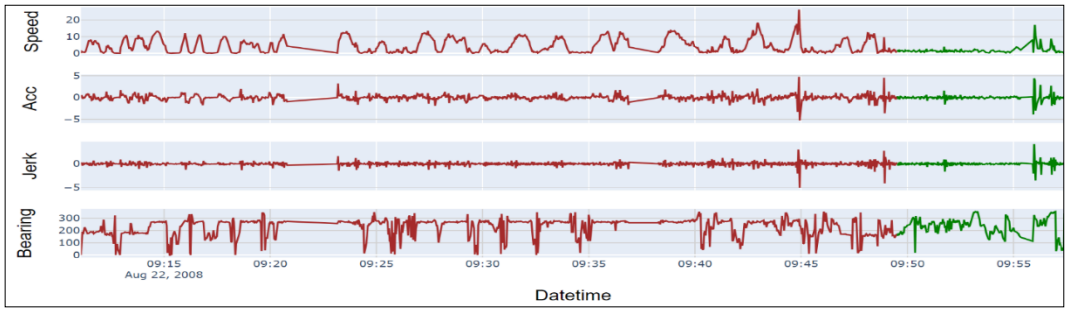

Figure 3.11: Corresponded speed, acceleration, jerk, and bearing for the selected trip (full trip data)

There are several examples of trips utilizing different modes of transport, extracted from the GeoLife dataset. For each trip, raw GPS data was plotted on a map, visually representing the trajectory of the traveler. In addition to the spatial path, the corresponding speed, acceleration, jerk, and bearing were also plotted, providing a comprehensive view of the trip dynamics. These plots highlight variations in motion, allowing for detailed analysis of how speed, directional changes, and acceleration patterns evolve. This multifaceted approach aids in understanding the complexities of each trip and contributes to more accurate transport mode detection and trip phase recognition.

Figure 3.12: A random car trip extracted from GeoLife dataset

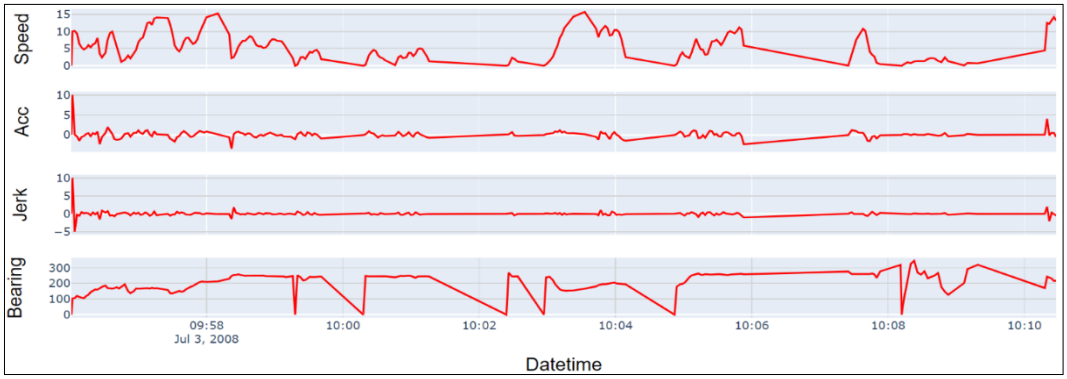

Figure 3.13: Corresponded speed, acceleration, jerk, and bearing for the selected trip (full trip data)

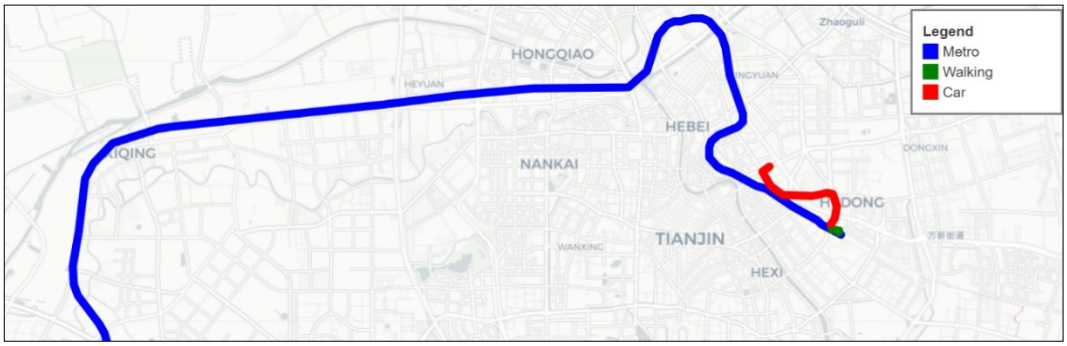

Figure 3.14: A random metro-walking-car trip extracted from GeoLife dataset

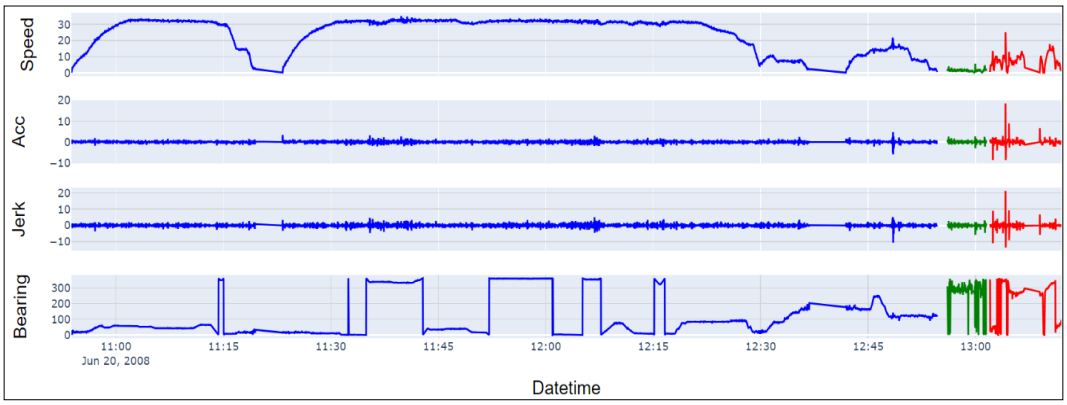

Figure 3.15: Corresponded speed, acceleration, jerk, and bearing for the selected trip (full trip data)

Figure 3.16 : A random bike trip extracted from GeoLife dataset

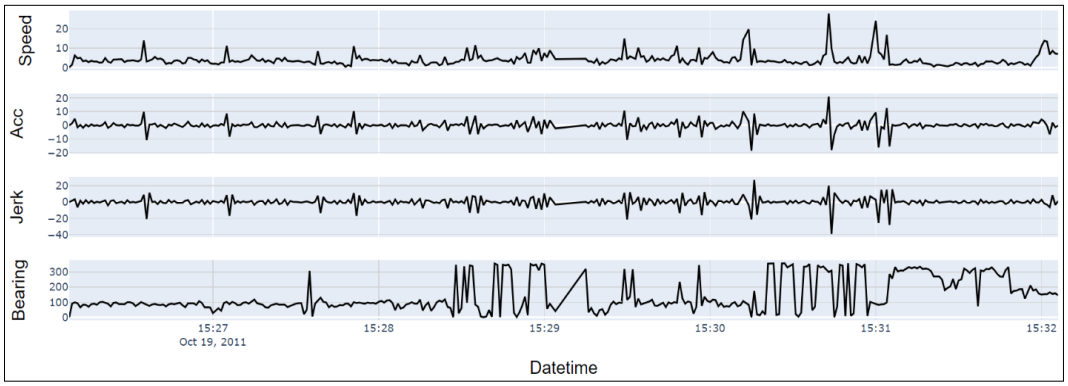

Figure 3.17 : Corresponded speed, acceleration, jerk, and bearing for the selected trip (full trip data)

Figures 3.12, 3.13, 3.14, 3.15, 3.16, and **3.17** show the behaviour of speed, acceleration, jerk, and bearing for each single trip.

In the GeoLife dataset, out of 6,399 labeled trips, only 5,575 trips contained more than 60 GPS points. Trajectories with fewer data points lack valuable information and can adversely affect the performance of the algorithm. To maintain data quality and optimize the performance of the model, all trajectories with fewer than 60 GPS points were removed. This filtering step ensures that only trips with sufficient data density are used, thereby improving the reliability of the results and enhancing the accuracy of the transport mode detection algorithm.

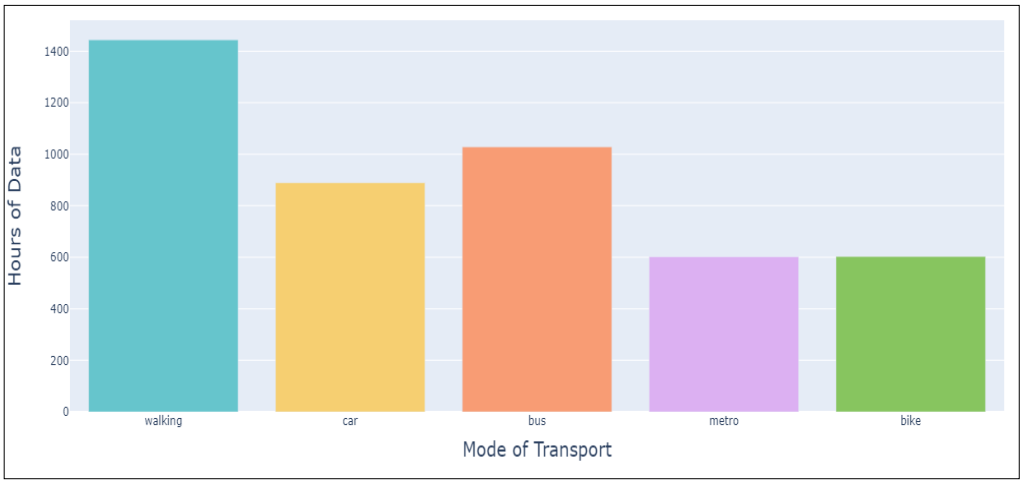

Figure 3.18 : Hours of labeled data for each mode of transport

Figure 3.18 displays the total hours of data recorded for different modes of transport: walking, car, bus, metro, and bike. Walking has the highest recorded data, amounting to over 1,400 hours, indicating it is the most frequently used mode of transport among the recorded trips. Car and bus follow, each with over 900 and 1,000 hours respectively, showing significant usage. Metro and bike have the least amount of data, each with slightly over 600 hours. This plot highlights the varying reliance on different transport modes within the dataset.

3.3 Preprocessing

GPS preprocessing involves several essential steps to enhance data quality and ensure accurate analysis. The process begins with detecting and removing outliers, which are often the result of GPS errors, signal reflections, or multipath effects that can significantly distort trajectory analysis. After eliminating these outliers, the Savitzky-Golay filter is applied to smooth the remaining data, reducing noise while preserving important features of the trajectory. This preprocessing is crucial for improving the reliability of the data and ensuring that subsequent analyses, such as transport mode detection, yield more accurate results..

3.3.1 Outliers

The next phase of data preprocessing involves detecting and managing outliers in the dataset. Using the maximum threshold speed and acceleration for each transport mode, following this research paper [48] as specified in **Table 3.1**, any GPS points with unrealistically high speeds and accelerations are discarded.

Mode	maximum speed (m/s)	maximum acceleration (m/s^2)	number of outliers GPS
Walk	7	3	29932
Bike	12	3	6616
Bus	34	2	1030
Car	50	10	616
Train	34	3	99846

Table 3.1: Thresholds for speed and acceleration categorized with different modes

Table 3.1 shows a detailed analysis of transportation modes based on their maximum speed, and maximum acceleration. Walking, characterized by a maximum speed of 7 m/s and a maximum acceleration of 3 m/s², exhibits 29,932 GPS outliers. This high outlier count suggests significant variability in pedestrian movement, likely due to frequent stops, changes in direction, and variable speed. Bike, with a higher maximum speed of 12 m/s and the same acceleration threshold of 3 m/s², shows fewer outliers (6,616), indicating more predictable and consistent travel patterns compared to walking. Public transit modes such as buses and trains display unique operational profiles; buses, with a maximum speed of 34 m/s and a lower acceleration limit of 2 m/s², record 1,030 outliers, reflecting relatively stable routes with occasional deviations. Despite having the same speed limit as buses but a slightly higher maximum acceleration of 3 m/s², trains exhibit an extraordinarily high number of outliers at 99,846, potentially due to frequent speed adjustments and high sensitivity to track conditions and signal variations. Car, capable of reaching up to 50 m/s with a maximum acceleration of 10 m/s², presents the lowest outlier count at 616, underscoring a high degree of travel consistency and reliability. This comprehensive analysis underscores the influence of operational parameters on the variability and predictability of different transportation modes.

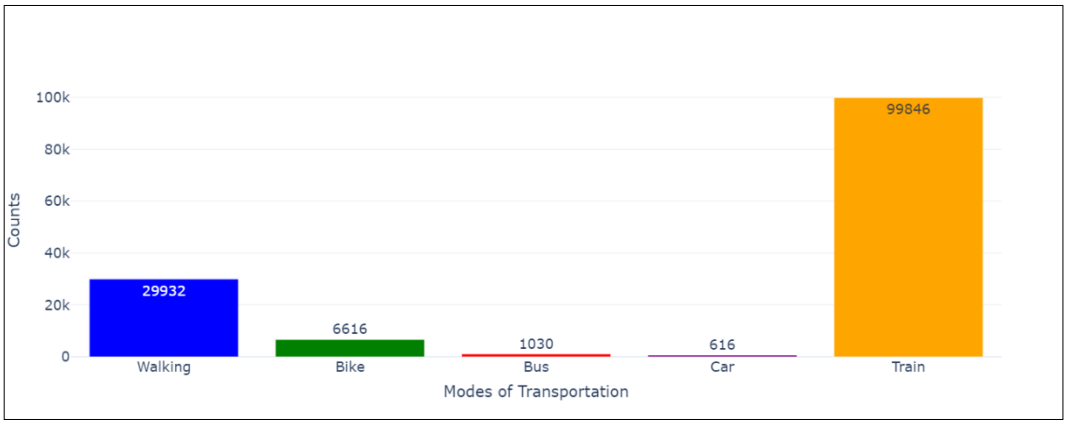

Figure 3.19: Number of observed outliers in each mode of transit

There are different numbers of outliers in each mode of transport. **Figure 3.19** presents a comparative analysis of the number of GPS-detected outliers across various modes of transportation: walking, bike, bus, car, and train. For clarity, each mode is represented by a distinct colour. The y-axis denotes the count of outliers, while the x-axis lists the transportation modes. Walking shows a significant number of outliers at 29,932, depicted in blue, indicating high variability in pedestrian movement, likely due to frequent changes in speed and direction. Biking, represented in green, has 6,616 outliers, reflecting more consistent travel patterns than walking. The Bus mode, marked in red, registers 1,030 outliers, suggesting relatively stable operations with occasional deviations. Cars, shown in purple, exhibit the fewest outliers at 616, indicating high predictability and consistent travel behavior. The most striking observation is the train mode, represented in orange, which has an exceptionally high number of outliers at 99,846. This vast number could be attributed to the frequent speed adjustments and potential data collection issues specific to train travel, such as signal variability and track conditions. Overall, the **Figure 3.19** underscores the varying degrees of travel consistency across different transportation modes, with trains and walking showing the highest levels of variability.

Figures 3.20, 3.21 and **3.22** show the time series data for speed, and acceleration, across a trip, with outliers highlighted in red. In the speed subplot, sharp spikes indicate sudden changes in velocity, marking these points as outliers. The acceleration subplot shows similar abrupt spikes corresponding to rapid starts or stops, with outliers reflecting significant deviations from typical behavior.

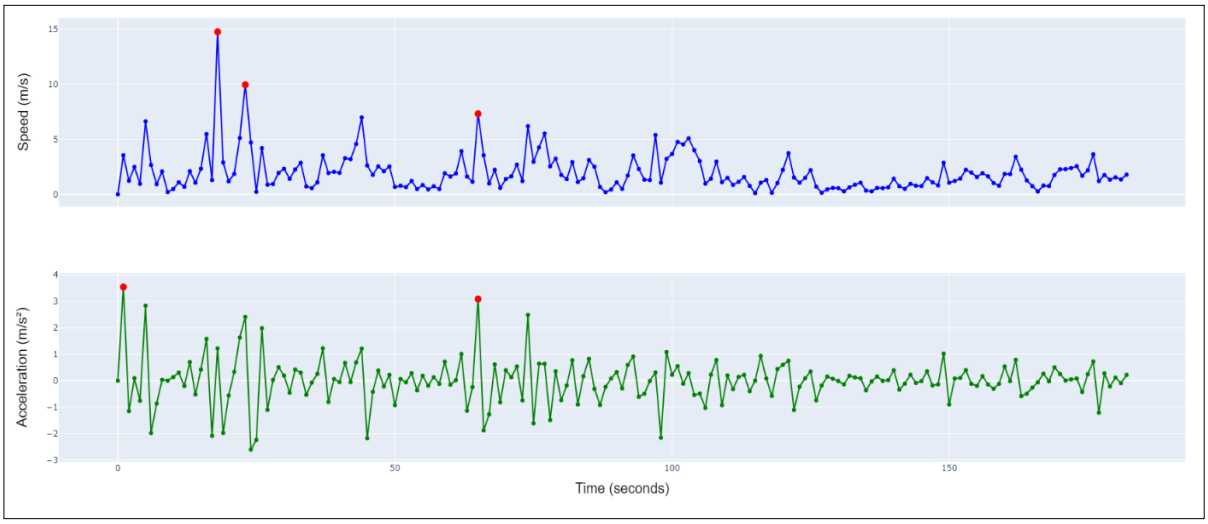

Figure 3.20: Outliers in a trip (labeled as walking)

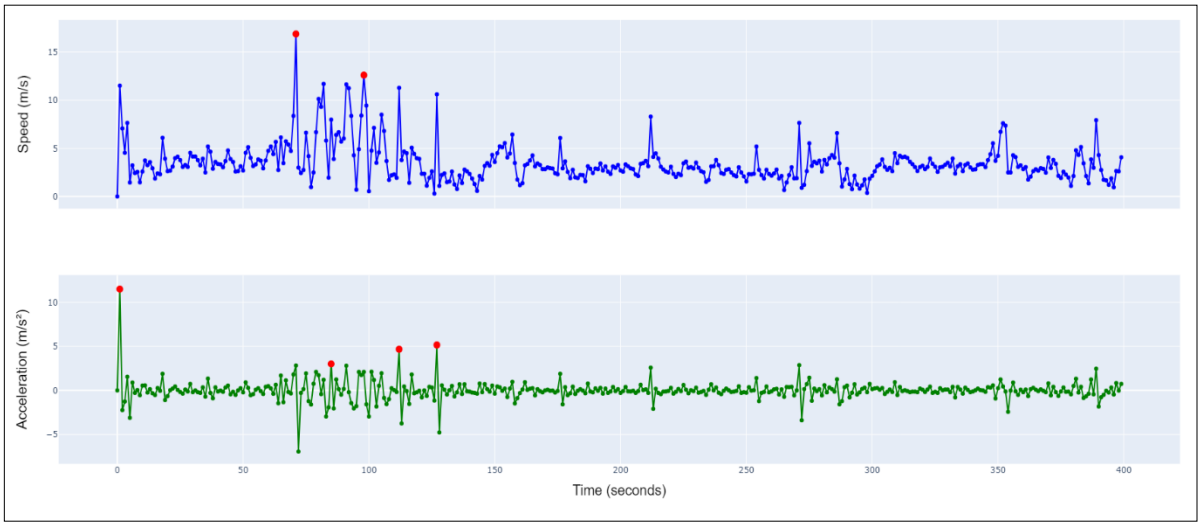

Figure 3.21: Outliers in a trip (labeled as bike)

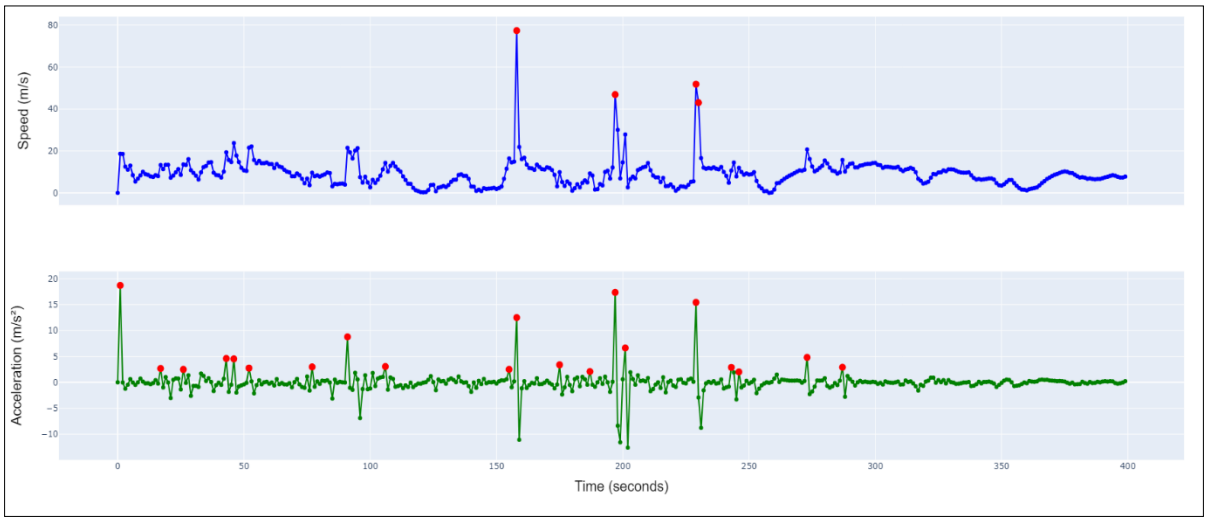

Figure 3.22: Outliers in a trip (labeled as bus)

3.3.2 Smoothing trips

Smoothing speed, acceleration and jerk data is essential for ensuring precise and reliable data for training classification models. Various sophisticated techniques are employed for this purpose. One widely used method is Savitzky-Golay filter, which applies a low-degree polynomial fit to successive subsets of the data through the method of least squares. This technique is particularly advantageous for preserving critical features of the data, such as peak values and overall structural integrity, while effectively smoothing out noise. Savitzky-Golay filter excels at preserving essential features of the data, such as peaks and overall shape, while reducing noise. It offers flexibility by allowing adjustment of the polynomial order and window size, and provides a higher degree of smoothness, making it suitable for applications requiring a very clean signal. Savitzky-Golay filter is more computationally intensive, which can be a drawback for real-time processing or large datasets, and can produce less reliable results at the edges of the data range. Additionally, choosing the appropriate polynomial order and window size requires consideration, as improper settings can lead to overfitting or insufficient smoothing.

3.3.3.1 Applying Savitzky-Golay filter

The Savitzky-Golay [66] filter is a digital filter that can smooth a set of data points. It is particularly effective for smoothing noisy data while preserving the original shape and features of the signal, which is often lost with other smoothing techniques such as moving averages. The Savitzky-Golay filter works by fitting successive subsets of adjacent data points with a low-degree polynomial by the method of linear least squares. The idea is to perform a least-squares fit to a polynomial of a certain degree within a moving window of data points.

Window and Polynomial Degree: window Size (N) : The number of data points used in each local polynomial fit. The window size must be an odd number ($N = 2m + 1$), where m is the half-window size.

Polynomial Degree (k) : the degree of the polynomial used to fit the data in each window.

Goal: the goal is to find the coefficients $a_0, a_1, \dots, a_k$ of the polynomial that best fits the data points within each window (**Equation 3.14**):

$$y(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_k x^k \tag{3.14}$$

Least-Squares Polynomial Fit:
For a given window of N points (x_i, y_i), where i $\in$ $[-\mathrm{m}, +\mathrm{m}]$, the goal is to minimize the sum of the squared differences between the observed data y_i and the values predicted by the polynomial of degree k :

$$S = \sum_{i=-m}^{m} \left[y(x_i) - \left(a_0 + a_1 x_i + a_2 x_i^2 + \cdots + a_k x_i^k \right) \right]^2 \tag{3.15}$$

To minimize S, the partial derivative of S with respect to each coefficient $a_0, a_1, a_2, \ldots, a_k$ are taken and set these derivatives to zero. This will give us a system of $k+1$ equations to solve for the coefficients $a_0, a_1, a_2, \ldots, a_k$.

For each $j \in \{0,1, \ldots, k\}$, the derivative of S with respect to a_j is:

$$\frac{\partial S}{\partial a_j} = -2 \sum_{i=-m}^{m} \left[y_i - \left(a_0 + a_1 x_i + \cdots + a_k x_i^k \right) \right] x_i^j = 0 \tag{3.16}$$

Matrix Formulation:
Define the matrix A and vector $\mathbf{b}$ as follows:

$$A = \begin{bmatrix} 1 & x_{-m} & x_{-m} & \ldots & x_{-m}^k \\ 1 & x_{-m+1} & x_{-m+1} & \ldots & x_{-m+1}^k \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_m & x_m & \ldots & x_m^k \end{bmatrix} \tag{3.15}$$

$$\mathbf{b} = \begin{bmatrix} y_{-m} \\ y_{-m+1} \\ \vdots \\ y_m \end{bmatrix} \tag{3.16}$$

The polynomial coefficients $\mathbf{a} = [a_0, a_1, \ldots, a_k]^T$ can be found by solving the normal equations in **Equation 3.17**:

$$\mathbf{a} = (A^T A)^{-1} A^T \mathbf{b} \tag{3.17}$$

Convolution Coefficients:

Instead of solving the polynomial fit for each window, the filter computes a set of convolution coefficients that are applied to the data. The idea is to precompute these coefficients based on the chosen window size and polynomial degree.

$$\mathbf{c} = (A^T A)^{-1} A^T \tag{3.18}$$

In **Equation 3.18,** $\mathbf{c}$ is the vector of polynomial coefficients. The smoothed value at the center of the window is obtained by convolving these coefficients with the data points in the window.

Applying the Filter:

The convolution operation is performed on the data with the precomputed coefficients:

$$\hat{y}_i = \sum_{j=-m}^{m} c_j y_{i+j} \tag{3.19}$$

c_j is the convolution coefficients, and $\hat{y}_i$ is the smoothed value at the i-th point.

Figure 3.23 illustrates the application of the Savitzky-Golay filter to smooth GPS data for a bus trip, featuring a total of 200 data points. In this example, different colors represent various data series: the original speed is depicted in darker blue, while the Savitzky-Golay smoothed speed is shown in a light blue. The second panel presents acceleration data, with original values in green and the smoothed results in purple. The third panel showcases jerk values, where the original jerk is represented in purple and the smoothed jerk is indicated in yellow. The Savitzky-Golay filter significantly reduces fluctuations in the speed data, providing a clearer trend while preserving key variations. Similarly, the acceleration and jerk panels demonstrate how the filter smooths sharp peaks and troughs, yielding cleaner signals. Overall, this filtering process enhances the readability of the bus trip data and improves its suitability for subsequent analyses, such as transport mode detection or trip segmentation.

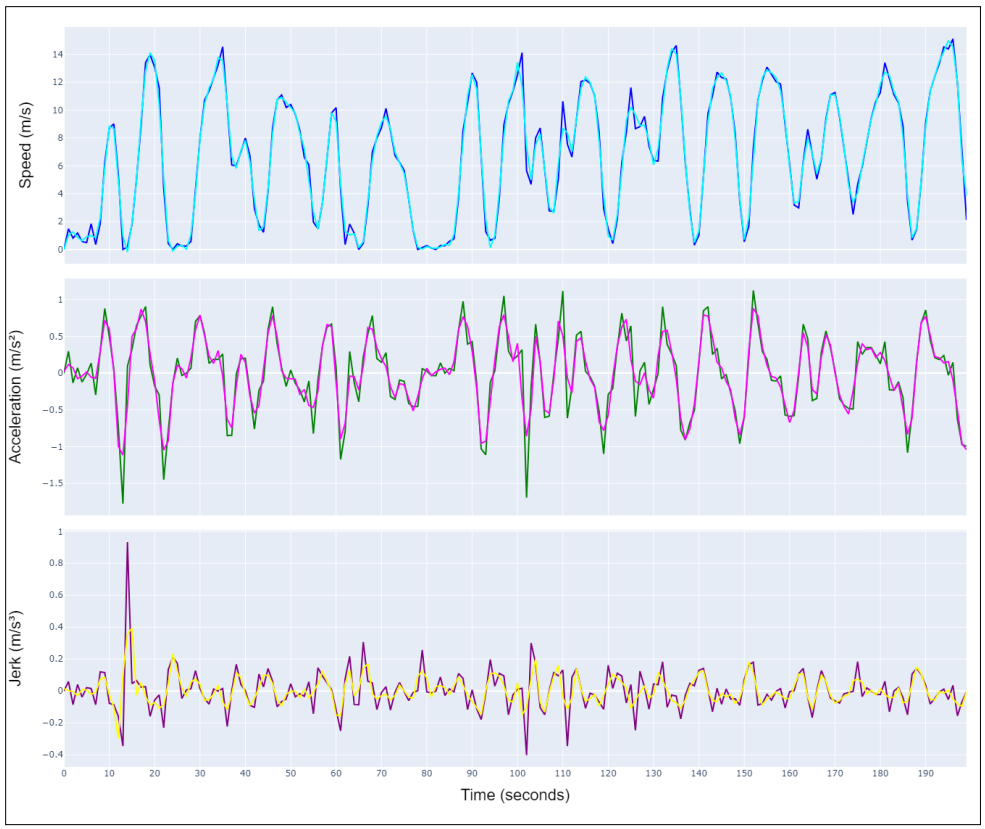

Figure 3.23 : Applying Savitzky-Golay filter on 200 GPS data of a bus trip

Figure 3.24 demonstrates the application of the Savitzky-Golay filter on 200 GPS data points collected during a car trip. In the speed graph, pronounced spikes are visible, indicating rapid variations in speed, which may arise from GPS inaccuracies or sudden accelerations and decelerations during the car trip. The Savitzky-Golay filter effectively smooths these spikes, leading to a more stable representation of speed trends and providing a clearer understanding of the overall speed patterns. The acceleration panel shows moderate fluctuations, with both the original and smoothed data remaining around zero, reflecting typical driving behavior. The jerk panel, capturing changes in acceleration, also benefits from the smoothing process, revealing a more consistent signal that highlights the dynamics of the car trip while filtering out noise.

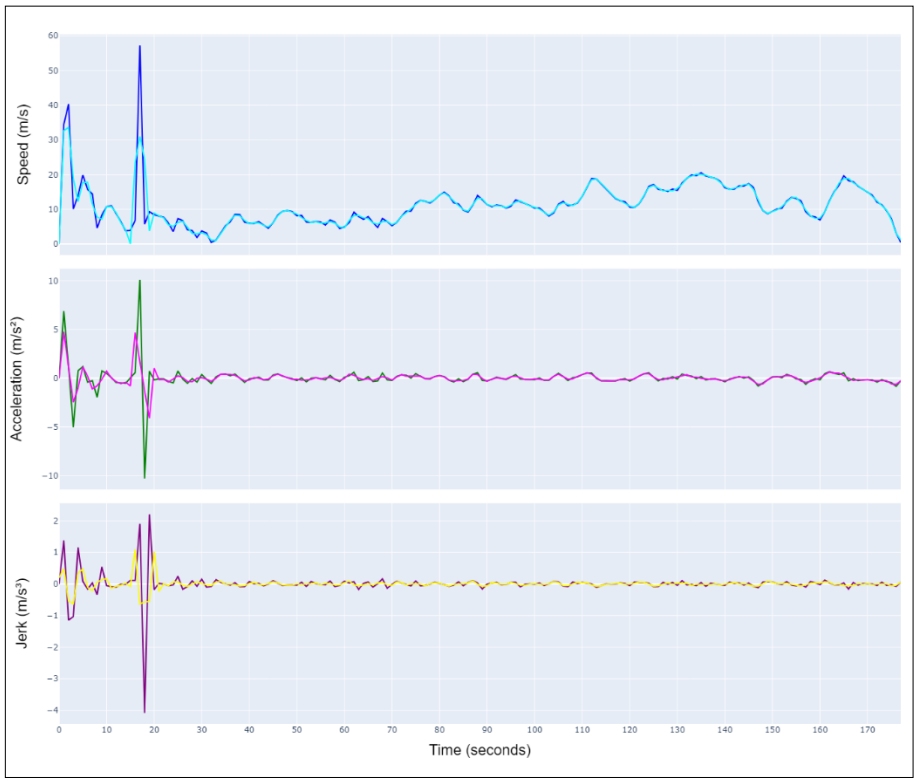

Figure 3.24 : Applying Savitzky-Golay filter on 200 GPS data of a car trip

3.4 Segmentation Problem:

The dataset consists of 6,399 individual trips recorded by 67 users, with each trip labeled according to one or more modes of transport. These modes can include combinations such as walking-bus-walking or single modes such as trains or car. The length of each trajectory varies significantly, adding complexity to the analysis. The primary objective is to segment the trajectories that involve multiple labeled modes into distinct subsections, each corresponding to a unique mode of transport. This segmentation will facilitate a more detailed analysis of user behavior and transportation patterns, allowing for a clearer understanding of how users interact with different modes of transit throughout their journeys.

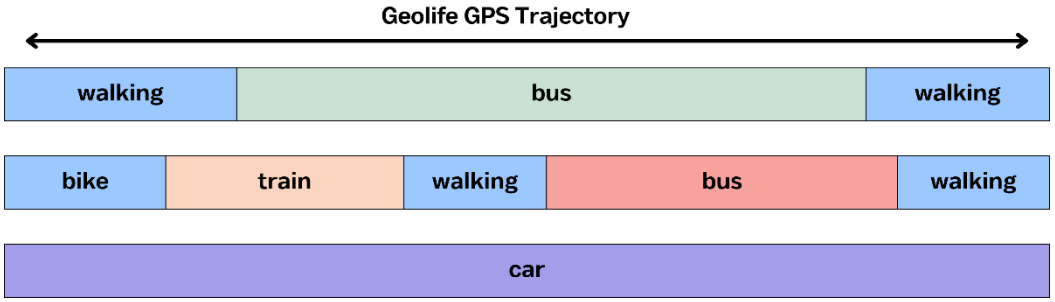

Figure 3.25: Three different labeled trajectories, with single-mode, and multiple modes

Figure 3.25 highlights that for each trip, the data will be segmented into sections, each containing only a single mode of transport. This approach ensures that each segment represents a uniform mode of travel, providing a detailed and accurate analysis of the transportation patterns within each trip.

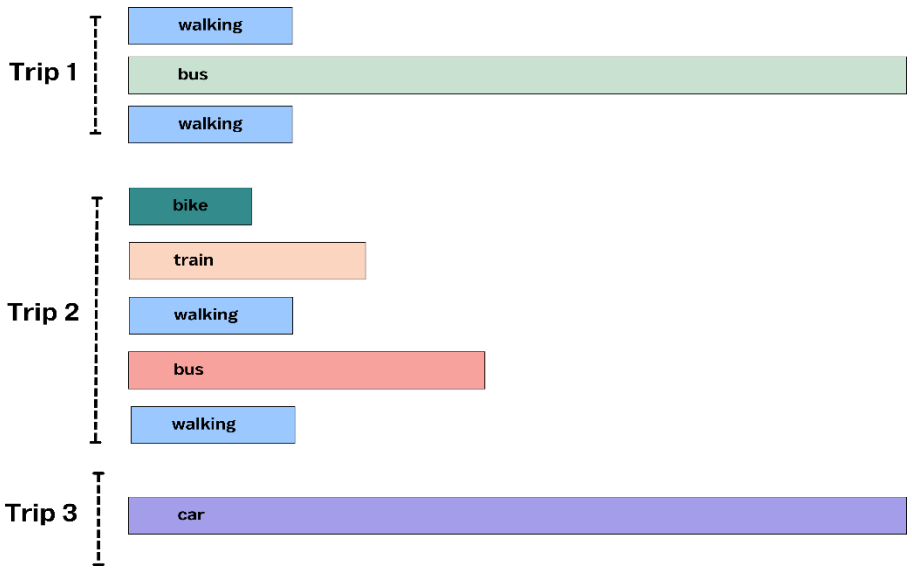

Figure 3.26**:** Trip segments with uniform mode for each segment with a length of more than 60 GPS points

Segments, representing a single mode of transport, can vary in length as shown in **Figure 3.26**. To ensure the reliability and quality of the data, there is a threshold of 60 GPS points per segment. Segments with fewer than 60 GPS points will be excluded from the analysis. Since trajectories with less than 60 rows of GPS dataset, which is equivalent to less than one minute, do not provide significant useful information. After extracting segments containing a minimum of 60 GPS points, the next step involves constructing data windows of uniform size (**Figure 3.27**).

To accomplish this, a function iterates over each segment, where segments represent trips identified by a unique mode of transportation. However, the length of each segment varies. For example, a bus segment may contain 680 GPS points, whereas a walking segment may contain 830 GPS points.

In this study, the impact of window length on the final accuracy of the model was evaluated by testing three different window sizes: 200 seconds, 60 seconds, and 30 seconds. Each window size represents a distinct approach to segmenting the trajectory data, allowing for an analysis of how varying the temporal resolution affects the model's performance.

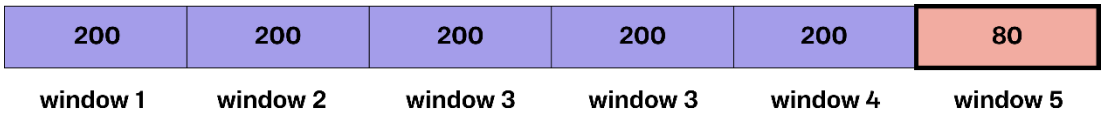

Figure 3.27: Fixed size window size

Various methods can be employed to address the issue of varying lengths in the final segment of each trip and to achieve fixed-sized windows for analysis, several suggestions can be applied:

1. Trimming the final section that does not match the dimensions of the preceding sections. In this case, data will be lost.

2. Padding involves inserting zero values for speed, acceleration, jerk, and bearing until the desired size is reached. This method was selected in our approach.

3. Duplicating the values from the previous section until reaching the predetermined size. Fake data will be added.

The new filtered dataset originally contained 6399 trajectories, each possibly involving multiple modes of transport. After dividing these trajectories into segments, each corresponding to a single mode of transport, the total number of segments increased to 8878. **Figure 3.28** shows the total number of segments with unique labeled modes, whereas **Figure 3.29** illustrates the total number of segments, each uniquely labeled, with zero padding applied. Walking exhibits the highest number of fixed-size segments, with 3,413 segments each containing 200 GPS points, while car, metro, and bike have significantly fewer segments. These are the fixed-window sizes segments of data that will be used to train ML and DL models.

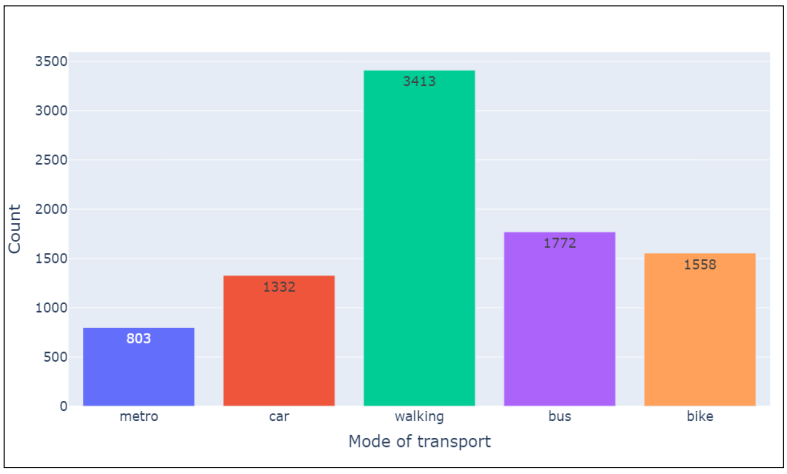

Figure 3.28: Number of trip segments with a unique mode (200 GPS point)

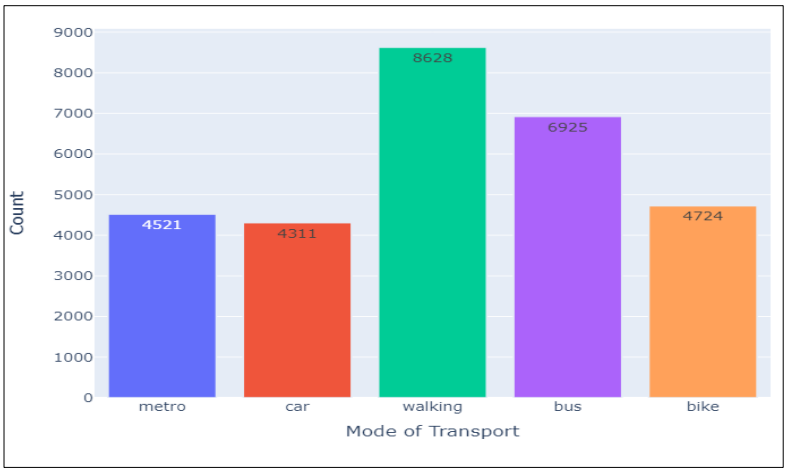

Figure 3.29**:** Total number of windows with 200 length (zero padding)

3.5 Machine learning

Applying ML algorithms to classify GPS data for predicting modes of transport, particularly using the RF algorithm, offers significant advantages. RF, an ensemble learning method, combines multiple decision trees to enhance prediction accuracy and robustness. The RF algorithm excels in its ability to manage large datasets and its resistance to overfitting, making it ideal for handling the complex and noisy nature of GPS data. By leveraging feature importance, it can identify key patterns and variables most indicative of specific transport modes. This leads to highly accurate and reliable predictions, even in diverse and challenging environments. Moreover, the interpretability of RF models allows for better understanding and fine-tuning of the classification process, enhancing the overall model performance. Numerous studies [48], [67], [33] [68] have evaluated various ML models for transport mode detection, and RF consistently demonstrated superior performance.

3.5.1 Decision tree and random forest

Random forests [69] are an ensemble learning method for classification, regression, and other tasks, that operate by constructing a multitude of decision trees during training time. RF operate by constructing multiple decision trees using different subsets of the training data and features. The process begins with bootstrap sampling, where several subsets of the data are created through random sampling with replacement. Each subset trains a distinct decision tree, where at each node, a random subset of features is considered to determine the optimal split, enhancing model diversity. After training, predictions from all trees are aggregated for classification tasks, the majority vote is taken, while for regression, the mean prediction is used. This ensemble approach boosts accuracy and mitigates overfitting by leveraging the collective decision-making of the trees, ensuring robustness and reliability in predictions. The RF algorithm was chosen for transportation mode detection in this study due to its superior performance and robustness compared to other ML classification algorithms. This method excels in handling large datasets with multiple features, reducing overfitting, and delivering high accuracy. It achieves these results by constructing multiple decision trees and combining their outputs to make final predictions, enhancing model reliability. A detailed explanation of the underlying logic behind the RF algorithm, including how it aggregates predictions from individual trees, will be provided later in this section.

3.5.1.1 Decision tree

In a RF, a decision tree is one of many individual trees that work together to make a collective prediction. Each decision tree in the forest is built using a random subset of the data and a random subset of features. The final output of the RF is based on the majority vote (for classification) or the average prediction (for regression) of all the individual trees.

Entropy (H) : A measure of impurity or randomness in the dataset determines the best feature to split the data. Lower entropy indicates higher purity of the subset. For a given set of training examples S, entropy is defined as:

$$H(S) = -\sum_{i=1}^{c} p_i \log_2 p_i \tag{3.20}$$

where p_i is the proportion of samples in S that belong to class i, and c is the number of classes.

Gini Index (G) : is a metric used in decision trees to measure the impurity of a dataset:

$$G(S) = 1 - \sum_{i=1}^{c} p_i^2 \tag{3.21}$$

Information Gain (IG) : It is a measure of the effectiveness of a feature in reducing uncertainty. Used to select the feature that provides the highest information gain for splitting the data. The reduction in entropy from a split:

$$IG(S, A) = H(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} H(S_v) \tag{3.22}$$

where S_v is the subset of S for which feature A has value v.

3.5.1.2 Bagging (Bootstrap Aggregating)

Random Forests operate by constructing multiple decision trees using different subsets of the training data and features. From a dataset D of size N, create B bootstrap samples D_i by sampling with replacement, each of size N. Sample trains a distinct decision tree, where at each node, a random subset of features is considered to determine the best split. For classification, aggregate predictions using majority voting:

$$\hat{y} = \text{mode}\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_B\} \tag{3.23}$$

3.5.1.3 Random feature selection

To build each decision tree in a RF model, rather than considering all M features to split each node, a random subset of m features ($m < M$) is selected. The optimal split is then determined from these m features. This method ensures tree diversity, reduces correlations between trees, and improves the model's robustness and accuracy by preventing any single feature from dominating the splits across all trees, thus enhancing generalization and reducing the risk of overfitting.

3.5.1.4 Feature importance

The importance of a feature can be estimated by looking at how much the impurity (e.g., Gini index, entropy) decreases when a feature is used in a split:

$$I(A) = \sum_{t \in \text{ nodes}} \frac{N_t}{N} \Delta i(t) \tag{3.24}$$

where N_t is the number of samples reaching the node t, N is the total number of samples, and $\Delta i(t)$ is the decrease in impurity from splitting on feature A.

3.6 Feature selection and extraction

Feature selection and extraction are pivotal in transport mode detection studies utilizing GPS data, significantly improving model performance and interpretability. Feature selection involves identifying critical attributes such as speed, acceleration, jerk, stop frequency, and average speed that are instrumental in distinguishing between modes such as walking, cycling, driving, and public transit. Advanced techniques including RF importance measures and Principal Component Analysis (PCA) help rank and reduce features, ensuring only the most informative attributes are included. Feature extraction transforms raw GPS data into meaningful features that capture the nuances of travel behavior. This process includes calculating metrics such as distance traveled, time intervals between GPS points, directional changes, and movement patterns indicative of specific transport modes. For instance, frequent stops and low average speed suggest walking, while high speed and infrequent stops indicate driving. Integrating spatial and temporal features allows for a comprehensive analysis and robust classification. Effective feature selection and extraction enhance accuracy, reduce computational costs, and improve generalizability across diverse datasets and contexts.

3.6.1 Feature extraction

Numerous studies have explored feature extraction to enhance transport mode detection using GPS data, focusing on identifying and transforming relevant attributes to improve model performance. Different statistical features will be extracted from various time windows to capture the temporal dynamics of the data.

1. Average speed : $$\frac{\sum_{i=1}^{\mathrm{N}} v_i}{t_{\mathrm{N}} - t_1} \tag{3.25}$$

where v_i is the speed between consecutive GPS points and t_i and t_N are the starting and ending times respectively.

2.Average accerelarion : $$\frac{\sum_{i=1}^{\mathrm{N}} a_i}{t_{\mathrm{N}} - t_1} \tag{3.26}$$

where a_i is the acceleration between consecutive GPS points and t_i and t_N are the starting and ending times respectively.

3.Average jerk: $$\frac{\sum_{i=1}^{\mathrm{N}} j_i}{t_{\mathrm{N}} - t_1} \tag{3.27}$$

where j_i is the jerk between two consecutive GPS points and t_i and t_N are the starting and ending times respectively.

4. Length (total window distance) : $\sum_{i=1}^{N-1} d_i$ (3.28)

where d_i is the distance between consecutive GPS points.

5. Top three velocities: $\max \text{ three } (v_1, v_2, \dots, v_{n-1})$ (3.29)

6. Top two velocities: $\max \text{ two } (v_1, v_2, \dots, v_{n-1})$ (3.29)

7. Top one velocities: $\max \text{ one } (v_1, v_2, \dots, v_{n-1})$ (3.29)

8. Top three accelerations: $\max \text{ three } (a_1, a_2, \dots, a_{n-1})$ (3.30)

9. Top two accelerations: $\max \text{ two } (a_1, a_2, \dots, a_{n-1})$ (3.30)

10. Top one accelerations: $\max \text{ one } (a_1, a_2, \dots, a_{n-1})$ (3.30)

11. Stop duration: $\sum_{i=1}^{n} (\text{speed}_i == 0)$ (3.31)

Stop duration counts the number of points where speed equals zero (representing stops).

12. Movements ratio: $\frac{\text{stop duration}}{n}$ (3.32)

In this equation , n is the number of GPS points in each window.

13. Speed variance: $\frac{1}{n}\sum_{i=1}^{n} \left(\text{speed}_i - \text{avg\_speed} \right)^2$ (3.33)

The variance of speed values, which measures how much speed varies from the average speed.

14. Accleration variance: $\frac{1}{n}\sum_{i=1}^{n} \left(\text{speed}_i - \text{avg\_speed} \right)^2$ (3.34)

The variance of acceleration values, which measures how much acceleration varies from the average speed.

15. Bearing change rate: $\frac{\sum_{i=1}^{n} \text{bearing}_{i+1} - \text{bearing}_i}{duration}$ (3.34)

From each data segment, the aforementioned features will be calculated to capture the statistical features within specific segments. There are numerous additional features that can be extracted from GPS data, such as altitude changes, heading direction, and stop durations. However, in this study, we focused on utilizing the most relevant and essential features for our analysis, ensuring that the model remains both efficient and effective in achieving accurate results.

3.6.2 Feature importance

Feature importance in transport mode detection involves identifying the most related attributes or characteristics derived from GPS data for distinguishing between different modes of transport. This process is critical for enhancing the accuracy and efficiency of ML models. By evaluating the contributions of various features such as speed, acceleration, jerk, bearing, and their statistical derivatives, models can more effectively differentiate between walking, cycling, driving, and other transport modes.

The math behind feature importance in RF involves measuring the contribution of each feature to the prediction accuracy. Each tree in the forest splits nodes based on features that reduce impurity (e.g., Gini impurity or entropy). The importance of a feature is quantified by the total decrease in impurity that the feature achieves across all the nodes and trees where it is used for splitting.

Specifically, for a given feature, its importance is calculated by summing up the impurity reductions Δ_i for all nodes it splits averaged over all trees in the forest. This score is then normalized. The sum of importance for all features is equal to one. This method provides a straightforward way to rank features by their predictive power, guiding feature selection and model refinement.

Various studies adopt different time window lengths to extract statistical features for ML and DL models, which can notably influence model performance, particularly in time-series analysis. The selection of appropriate time windows is crucial, as it affects the ability of the models to capture relevant temporal patterns and trends, whether dealing with GPS trajectories or sensor data. The choice of time windows allows the models to strike a balance between capturing fine-grained, short-term fluctuations and identifying broader, long-term patterns.

Different time windows were selected to determine the optimal time window for feature extraction that maximizes the performance of ML and DL models. By experimenting with three different time windows 200, 60, and 30 seconds, we aim to evaluate how these varying time intervals impact the model's ability to capture both short-term fluctuations and long-term trends within the data.

The 200-second window captures long-term, sustained trends, which are useful for identifying gradual changes in behaviour. The 60-second window balances short-term events and long-term trends, offering a mid-range perspective. The 30-second window focuses on short-term, high-frequency events, such as sudden changes in speed or direction.

Our goal is to systematically analyze the performance of deep and ML models using features extracted from each time window and identify the window that yields the most accurate and robust predictions. This multi-scale approach enables us to explore the trade-offs between granularity and temporal scope, ultimately identifying the best time window for optimizing model performance.

3.7 Deep learning methods

Applying DL methods for transport mode detection using GPS data has shown significant promise in enhancing the accuracy and reliability of identifying transportation modes. By leveraging the rich spatiotemporal information embedded in GPS trajectories, DL models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), can automatically learn complex patterns and dependencies within the data. These models can effectively capture the differences between various transport modes, such as walking, cycling, driving, and public transit, by analyzing features

such as speed, acceleration, jerk, and bearing. The application of DL in this domain typically involves preprocessing GPS data to extract relevant features, training the models on labeled datasets, and fine-tuning them to improve classification performance. Studies have demonstrated that DL approaches outperform traditional ML methods in terms of accuracy and robustness[48], making them a valuable tool for intelligent transportation systems, urban planning, and personalized mobility services.

3.7.1 1D CNNs for transport mode detection

Convolutional Neural Networks (CNNs) come in different types based on the structure of the input data, including 1D and 2D CNNs. 1D CNNs are used for sequential data such as time series or sensor data, making them suitable for analyzing transport mode detection features such as speed and acceleration over time. They perform convolution operations along one dimension, capturing temporal patterns efficiently. 2D CNNs, on the other hand, are commonly used for image data such as satellite imagery or urban traffic analysis. Both types of CNNs offer powerful feature extraction capabilities, depending on the nature of the data**.**

The main reasons to use 1D CNN in transit mode detection studies are as follows:

1. **Nature of GPS Data (sequential data:):** GPS data is inherently sequential, representing the movement of an entity over time. This time-series nature is well-suited to 1D CNNs, which excel at extracting patterns from sequences by applying filters along the temporal axis.

2. Comparison with 2D CNNs (data structure): 2D CNNs are designed for grid-like data such as images, where spatial relationships are key. Applying 2D CNNs to sequential data such as GPS trajectories would require transforming the data into a 2D format, which can be artificial and might lose temporal dependencies.

3. Complexity and overhead: Converting GPS sequences into a 2D matrix adds unnecessary complexity and computational overhead. This transformation does not inherently benefit the task of detecting transport modes, which relies on temporal patterns rather than spatial ones.

4. **Efficiency in feature extraction**: 1D CNNs are highly effective at extracting local temporal features from sequential data. By applying convolutional filters along the time axis, patterns in speed, direction, and acceleration, which are critical for distinguishing different transport modes can be captured.

5. Simplicity and speed: Compared to RNNs, such as LSTMs and GRUs, 1D CNNs are generally simpler and faster to train. RNNs involve complex gate mechanisms and require sequential processing, which can be computationally intensive and slower due to the need for maintaining dependencies across time steps. 2D CNNs and RNN/LSTM models each possess unique strengths and applications; however, 1D CNNs offer an optimal balance of simplicity, efficiency, and effectiveness for transport mode detection using GPS data. Their proficiency in capturing temporal patterns, combined with scalability and faster training times, makes 1D CNNs an excellent choice for this task. Utilizing 1D CNNs enables reliable and efficient transport mode detection, significantly enhancing the capabilities of intelligent transportation systems and urban mobility solutions.

In this study, due to the advantages of 1D CNN, particularly its ability to effectively capture temporal patterns in sequential data, the DL model was designed and trained using a 1D CNN architecture.

3.7.2 Convolutional neural network

Convolutional neural networks were first used by [70] to forecast digital handwritten pictures and have proved their success in several computer vision tasks. In this study [71] more than 1000 photos were classified with an acceptable performance. CNN has also been used to map sequential data, including natural language processing and speech recognition[72], [73]. A CNN architecture typically consists of a series of layers in which the input data is transformed by each layer to the output layer of a collection of neurons. In CNN, the convolutional layers (convolutional kernels) convolve several local filters with raw input data and generate translation invariant local features and the subsequent pooling layers extract features with a fixed length over sliding windows of the raw input data.

3.7.3 Input Layer

The predefined GPS data after preprocessing and windowing has been segmented in such a way that each segment contains 200, 60, and 30 GPS points. The decision to use 200 GPS points per segment in this study was made to enable a direct comparison with previous research, which employed similar segmentation techniques. This allows for a consistent and meaningful evaluation of our results against existing findings in the field. Other window lengths were also selected to compare their effectiveness in improving model performance.

There are four main features in each segment of data including speed, acceleration, jerk, and bearing. In the CNN, the input shape for all instances must be the same length. To address this challenge, all segments are restricted to a fixed-size matrix with $200 * 4$ where 200 is the height and 4 is the width of input data. For the remainder of the explanation, the window length of 200 GPS points will be used to explain the logic.

3.7.4 Convolutional layer

The first layer of a neural network after the input layer defines filters, also known as feature detectors, with a height specified by the kernel size. Utilizing a single filter in the first layer allows the neural network to learn only one feature, which is typically insufficient for complex data. Therefore, multiple filters are defined to enable the network to learn various distinct features from the input data in the first layer. The weights for each filter are stored in the columns of the resulting output matrix. Given the defined kernel size and the input matrix length, each filter will have its own set of weights that contribute to feature detection. Each kernel learns specific features in a signal (e.g., a rapid acceleration), while another might detect gradual trends (e.g., a slow increase in speed).

The result from the first CNN layer will be fed into the second CNN layer. Different filters have been applied to be trained on this level. Each neuron in the convolutional layer's output is connected to the small region (receptive field) of the previous layer, where the size of the receptive field is equivalent to the filter. The output value of the neuron is computed by operating dot product between the parameters of the filter and the entries of its receptive field. Convolving the same filter across the whole surface of the input volume creates a 2-dimensional map in the output volume, also called a feature map or activation map.

Performing similar operations for all layer's filters creates several activation maps. Stacking the feature maps of all filters along the depth dimension creates the 3-d output volume of the layer. Since the size of the input volume is small in our application, we use a small ($1\times3\times C$) convolution filters throughout for all convolutional layers. For each layer, C indicates the number of channels in the layer's input volume and in our case, it is one, due to using a 1D convolutional network. Using

smaller receptive fields leads to reducing the number of parameters and mitigating the overfitting problem [74].

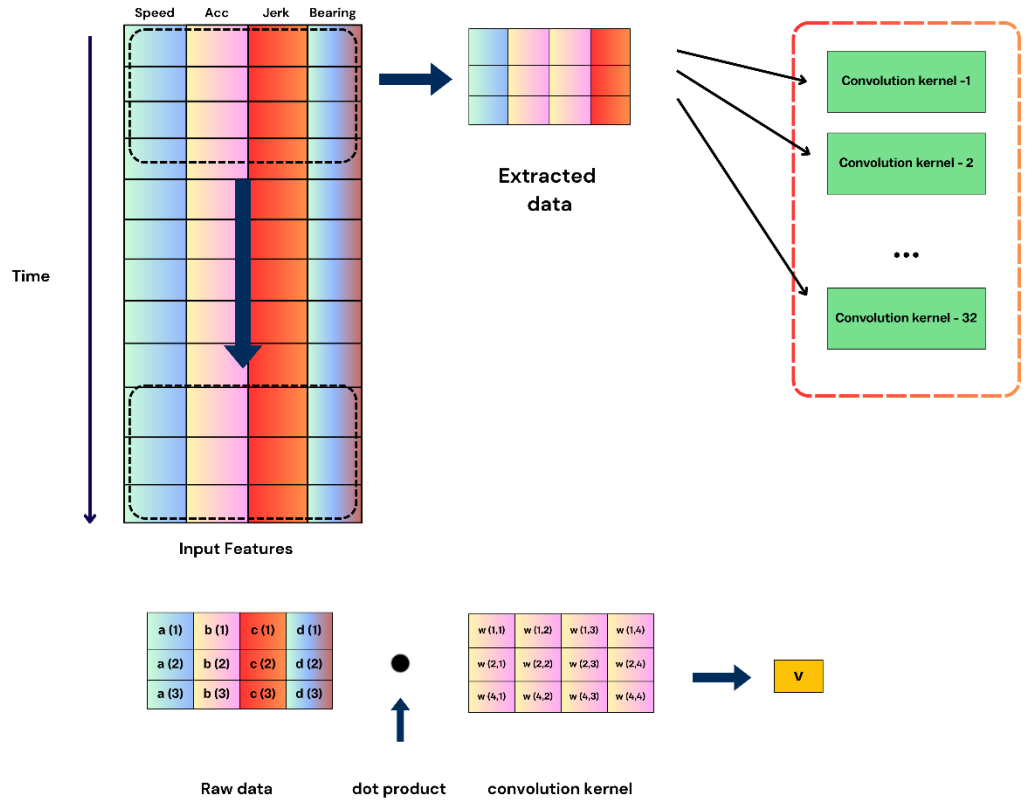

Figure 3.31: 1D Convolutional Neural Network Processing of Sequential Input Features

Figure 31 shows the application of a 1D Convolutional Neural Network (1D CNN) to time-series data with multiple features, such as Speed, Acceleration, Jerk, and Bearing. In this process, the data is segmented into small time windows, and each segment is analyzed by convolutional kernels that slide along the time axis. These 1D convolutions focus on capturing temporal patterns and dependencies within the data while simultaneously considering multiple features. The kernels perform dot product operations between the input data and the kernel weights, producing feature maps that highlight specific patterns detected in the time-series data. This approach enables the 1D CNN to effectively analyze and learn from temporal data, making it well-suited for tasks such as time-series classification.

More filter details

Filter Shape: each filter has weights $\mathbf{W} \in \mathbb{R}^{3\times4}$. For 32 filters, and $\mathbf{W}_k$ where $k \in \{1,2,\ldots,32\}$.

Bias: Each filter has a bias term b_k.

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} & w_{1,4} \\ w_{2,1} & w_{2,2} & w_{2,3} & w_{2,4} \\ w_{3,1} & w_{3,2} & w_{3,3} & w_{3,4} \end{bmatrix}$$

The weight matrix **W** represent the kernel applied in the convolution.Suppose the kernel size $\mathbf{W} = \mathbf{3}$ and the number of features is 4.

Output calculation

For each filter k and each valid timestep t ($t \in \{1,2,\ldots,198\}$):

$$z_k[t] = \sum_{i=0}^{2} \sum_{j=1}^{4} w_{i,j}^{k} \cdot x_{t+i,j} + b_k \tag{3.31}$$

$$y_k[t] = \mathrm{ReLU}(z_k[t]) = \max(0, z_k[t]) \tag{3.32}$$

where:

$w_{i,j}^{k}$ are the weights of the k-th filter.

$x_{t+i,j}$ are the input values at the current position.

b_k is the bias for the k-th filter

$y_k[t]$ is the activated output for the $k-th$ filter at the $t-th$ time step

Output shape

The convolution reduces the number of time steps from 200 to:

$$\text{Output Length } = L - K + 1 = 200 - 3 + 1 = 198 \tag{3.32}$$

Number of filters (channels) remains 32. Therefore, the output shape after the first convolutional layer is $(198,32)$.

3.7.5 Activation layer

Following the acquisition of the convolved output volume, the neurons are frequently followed by an activation procedure to inject nonlinearity into the model. Among the several forms of activation functions, we use non-saturating nonlinearity. The function, which is referred to the Rectified Linear Units (ReLU), substitutes all negative values in the feature map with a value of zero. Compared to other functions such as tanh(x) and the sigmoid function, the CNN with ReLU has a substantially quicker learning rate[75].

$$y_k[t] = \mathrm{ReLU}(z_k[t]) = \max(0, z_k[t]) \tag{3.33}$$

where $y_k[t]$ is the activated output for the $k-th$ filter at the $t-th$ time step.

3.7.6 Batch normalization

Batch normalization is a method for automatically standardizing the inputs to a deep-learning neural network layer. Batch normalization greatly accelerates the training process of a neural network and, in certain situations, enhances model performance via a minor regularization impact. The layer normalizes the inputs, resulting in a mean of zero and a standard deviation of one. The layer keeps track of statistics for each input variable during training and utilizes them to normalize the data.

Batch normalization layer can be used to normalize inputs before or after the prior layer's activation function[76].

Compute the mean and variance:

For a given mini-batch $B = \{x_1, x_2, \dots, x_m\}$, where x_i are the activations of the layer, compute the mean μ_B and variance σ_B^2 :

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$$
$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{3.34}$$

Normalize the batch:
Normalize the activations using the computed mean and variance:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{3.35}$$

x_i represents the individual input value (feature) from the batch that needs to be normalized and ϵ is a small constant added for numerical stability to avoid division by zero. μ_B is the mean of the batch. It calculates the average of all input values in the batch and σ_B^2 is the variance of the batch, calculated from all input values in the batch.

Scale and shift:
Apply learnable parameters γ (scale) and β (shift) to the normalized activations:

$$y_i = \gamma \hat{x}_i + \beta \tag{3.36}$$

where γ and β are parameters learned during the training process. The final output of the batch normalization layer is then passed to the next layer in the neural network.

3.7.7 Pooling layer

The pooling layer's goal is to provide spatial and scale invariance, reduced computation, and control overfitting by reducing the dimensionality of each feature map by computing the average of all elements within each feature map, resulting in a single value and spatially down-sampling the convolution layer [77]. Global Average Pooling is a function that computes the average output of each feature map in the preceding layer. It also does not have any trainable parameters. In the classification layer, global average pooling across feature maps can be used, which is easier to comprehend and less prone to overfitting than standard fully linked layers [72]. Given an input tensor of shape (batch_size, steps (200), features (4)), the GlobalAveragePooling1D layer computes the average of each feature map over all steps. The resulting output tensor has shape (batch_size, features).

For an input tensor x with dimensions (steps, features):

$$\text{GAP}\ (x) = \left[\frac{1}{n}\sum_{i=1}^{n} x_{i,j}\right]\ \forall j \in [1, \text{features}\ ] \tag{3.37}$$

where n is the number of steps, and $x_{i,j}$ is the value of the j-th feature at step i.

3.7.8 Dropout

Deep neural networks consist of multiple non-linear hidden layers, enabling them to model highly complex relationships between inputs and outputs. However, when trained on minimal data, many of these intricate patterns may simply reflect sampling noise, appearing in the training set but not in the actual test data, even when both come from the same distribution. This leads to overfitting, for which several techniques have been developed to mitigate. Overfitting is a key issue in CNNs architecture, due to the enormous number of weights and intricate interactions between inputs and outputs. Dropout is the most feasible and extensively utilized method for dealing with the overfitting problem in CNNs. The methodology removes the input units, together with all of their incoming and outgoing connections, from the network with a probability of P at each update throughout the training phase. At each level, the parameters associated with the reduced network are learned[78].

Dropout randomly "dropping out" or deactivating a subset of neurons during training. Specifically, at each training iteration, a certain percentage of neurons (defined by the dropout rate) are randomly selected and temporarily removed from the network, meaning their contributions are ignored for that iteration. This forces the network to learn redundant representations, as it cannot rely on any single neuron, thereby promoting a more robust and generalized model.

3.7.9 Fully connected layer with SoftMax activation

Several Fully-Connected (FC) layers can be used to complete the CNN design. Every neuron in the FC layer, such as the typical multilayer perceptron, is linked to all neurons in the preceding layer and calculated via element-wise multiplication. In our model, 100 and 50 neurons has been used to link the two final layers to the prior layer. Except for the last FC layer, the rest are involved in feature extraction. Following that, the recovered high-level features from the previous layers are input into the final FC layer for classification, where the SoftMax activation function is used to build a probability distribution across the transportation labels.

As a very fundamental and crucial network layer in DNNs, the SoftMax layer is commonly utilized in multiclassification tasks. The last layer will decrease the vector of height 50 to a vector of five and six since the goal is to predict five and six modes of transport (bike, bus, car, walking, train, still). Another matrix multiplication is used to accomplish this reduction. SoftMax is employed as the activation function. It pushes the neural network's outputs (number of selected modes) to add up to one. As a result, the output number will indicate the likelihood of each of the six classes. The output layer uses the logistic regression algorithm SoftMax to build a probability distribution across the categorization labels.

$$\text{Softmax}\,(x_i) = \frac{\exp\,(x_i)}{\sum_j^X \exp\,(x_j)} \tag{3.38}$$

where x is a set including all the received stimulation x_i of the output layer.

3.7.10 CNN configurations

The number of layers, the sequence of layers, and the number of filters in each convolutional layer can create a wide range of CNN configurations. To determine the optimal network for our purpose, various architectures has been generated (**Table 3.2**). Instead of performing an exhaustive and computationally expensive search over all CNN hyperparameters, we adopt an efficient manual search approach. We start with shallow networks with a low number of filters and gradually increase

both the number of layers and the filters in each layer. This allows us to evaluate whether the model achieves higher accuracy. For simplicity, we present only the most important configurations from the many networks tested. This highlights the impact of four key hyperparameters on prediction quality: the layer pattern, the presence or absence of certain layers, the network depth, and the number of filters in each layer. The configuration of Column A produced the optimal results for training the deep learning model.

	A	B	C	D	E
Input Layer	Shape of the input layer is (1×200×4 200 GPS points and 4 features)				
Convolutional	32	64	128	256	256
Batch Normalization	yes	yes	yes	yes	yes
Convolutional	64	128	256	512	512
Batch Normalization	yes	yes	yes	yes	yes
Convolutional	64	64	128	256	256
FC	100	200	200	100	200
Dropout	No	0.2	0.2	0.2	No
FC	50	100	100	50	100
Dropout	No	0.2	0.2	0.2	No
FC	5,6	5,6	5,6	5,6	5,6

Table 3.2 : Multiple CNN configurations per column.

3.7.11 Training process

After completing all the preprocessing and segmentation of GPS data, the next crucial step is to feed this data into a DL model for training. The objective of the training process is to optimize the parameters of the layers' filters to minimize the loss function. Categorical cross-entropy is used as the loss function to measure the error in the output layer. This error, representing the discrepancy between the actual and desired outputs, is defined by the loss function. During each iteration of the back-propagation process, the Adam optimizer is employed to minimize this loss, thereby enhancing the model's accuracy.

The Adam optimizer is used to update the model parameters during the backpropagation process, with five iterations to train the model. The key strategy is to divide the learning rate by 10 over each iteration, progressively reducing it. This allows the model to fine-tune the weights more precisely, leading to more stable convergence and reducing the risk of overshooting the minimum of the loss function.

Adam is well-suited for problems involving large datasets and numerous parameters, and it has recently seen broader adoption in DL applications. The batch size is set to 256, with an initial learning rate of 0.01. The parameters in the convolutional and fully connected layers are initialized according to the proposed scheme [79] .

The early stopping method is employed to determine the optimal number of epochs for training the most effective CNN model, thus mitigating the risk of overfitting. This technique involves computing training and validation metrics (such as accuracy) after each epoch. The epoch number

that yields the highest validation score is selected as the optimal number of epochs. Although the validation set is not utilized by the training algorithm, its true outputs are known and used to evaluate the model's performance after each epoch. Training is halted as soon as the validation performance plateaus or begins to degrade. This process, known as early stopping, enhances the model's generalization capability.

3.8 Transport mode detection framework

After training the model and optimizing its configuration, the model can be utilized to make predictions on new, unseen GPS segment data. The trained model will predict the mode of transport for each segment, offering valuable insights into travel patterns. To apply the pre-trained model to new segments, it is essential to ensure that all preprocessing steps are consistently applied to the incoming data. This also includes segmenting the GPS data into fixed-size windows. These preprocessed and segmented data windows are then fed into the model for prediction.

3.8.1 Segments predictions

The goal is to predict the corresponding mode of transport for each individual segment, thereby ensuring accurate and efficient mode classification. **Figure 32** shows the comprehensive process of transport mode detection using GPS data and a deep or ML model. Initially, GPS data is recorded from devices such as smartphones or GPS trackers, capturing coordinates, timestamps, and additional details such as altitude and speed. This raw data undergoes preprocessing, which involves cleaning to remove noise and outliers, filtering to smooth the data, segmenting based on time, distance, or movement changes, and extracting features such as speed, acceleration, heading, stop duration, and distance. The preprocessed data is then normalized for consistency. Next, this data is fed into a deep or machine-learning model. The model undergoes training with labeled data to learn associations between data patterns and transport modes, involving steps such as model selection, training, hyperparameter tuning, and validation. Once trained and validated, the model is deployed to predict transport modes for new, unseen GPS data, analyzing each data segment to determine whether it corresponds to walking, driving, cycling, or other modes. The final output consists of these predictions, represented different transport modes.

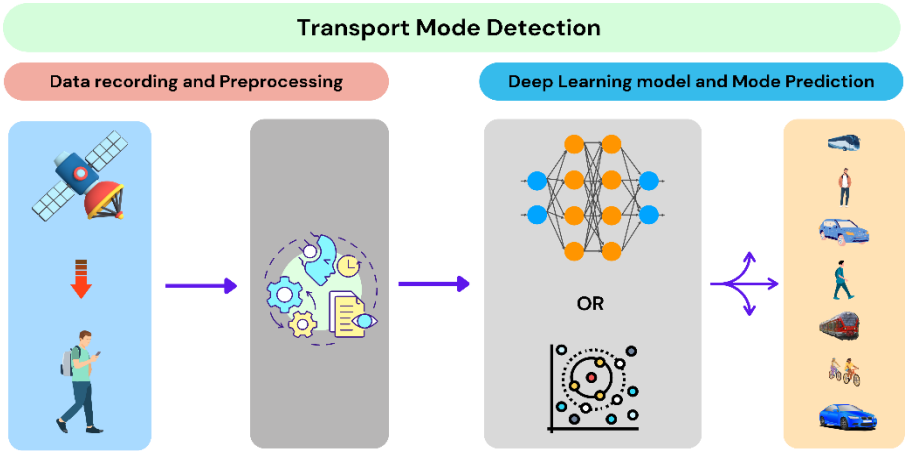

Figure 3.32: Transport mode detection baseline

3.8.2 Post Processing (motorized – non-motorized)

The motorized–non-motorized algorithm categorizes transport modes into motorized and nonmotorized segments. It starts by defining a function categorize_mode(mode), which takes a transport mode as input and returns "nonmotorized" for modes such as walking, cycling, and standing, and "motorized" for modes such as car, bus, and metro.

The algorithm initializes counters for motorized and nonmotorized segments. It then processes each segment in the input data, using the categorize_mode function to determine the category of each transport mode. Based on the category, it increments the corresponding counter. After processing all segments, the algorithm outputs is the total number of segments in each category.

This process provides a clear distinction between motorized and nonmotorized segments, offering valuable insights for applications such as environmental impact assessments, urban planning, and personal fitness tracking.

ALGORITHM 1 CONVERT RESULTS TO MOTORIZED AND NONMOTORIZED

```
1:  Input: A sequence of predicted mode of transport segments
2:  Output: Categorized segments into motorized and non-motorized
3:  function categorize_mode(mode)
        if mode in ["walking", "cycling", "standing"] then
            return "nonmotorized"
        else if mode in ["car", "bus", "metro"] then
            return "motorized"
4:  begin
5:      forEach item in predictedModes :
6:          Modes[i] = categorize_mode( Modes[i])
7:          return  Modes
    end
```

3.8.3 Smoothing (motorized – non-motorized)

The Smooth Motorized and Nonmotorized algorithm is designed to refine the classification of transport modes by ensuring consistency in the categorization of sequential segments. The input is a sequence of segment data values representing the user's aggregated transport modes. The output is the number of segments for each mode of transport, smoothed for accuracy. The core function compares two consecutive segments: if both segments are classified as "nonmotorized," the function calculates the mode with the highest probability between them and returns this mode.

This smoothing process ensures that minor inaccuracies or inconsistencies in the classification of nonmotorized segments are corrected, leading to a more coherent and reliable representation of transport modes. By iterating through the entire sequence of segments, the algorithm applies this check to each pair of consecutive segments, thus refining the overall categorization. This approach helps in achieving a more accurate distinction between motorized and nonmotorized modes, which is

essential for applications that rely on precise transport mode detection, such as urban planning, transportation management, and personal fitness tracking.

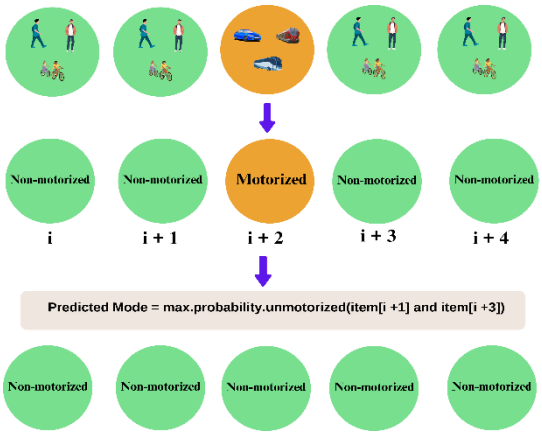

Figure 3.33: Smoothing algorithm

Figure 3.33 presents a sliding window approach to detect transportation modes based on a sequence of time steps (i, i+1, i+2, etc.). At each time step, a mode of transportation, such as walking, cycling, or driving, is assigned and categorized into either motorized or non-motorized modes. In the example shown, a transition to a motorized mode occurs at time step i+2, while the surrounding time steps are non-motorized. To improve the accuracy of mode detection, the algorithm considers a window of multiple time steps (from i-1 to i+4), rather than relying solely on the prediction for each individual time step. The final mode prediction is determined by calculating the maximum probability of non-motorized modes at adjacent steps, particularly at i+1 and i+3. This method allows the algorithm to account for the context provided by surrounding data points, smoothing out abrupt changes in predicted modes and ensuring more accurate transportation mode classification. As a result, the algorithm can effectively handle error predictions of modes of transportation by leveraging the sliding window technique to reduce noise and improve the robustness of its predictions. The reverse scenario can also occur, where there are predominantly motorized segment predictions, but a single non-motorized segment appears. In this case, the sliding window approach will help smooth the prediction by favoring the motorized mode, overriding the lone non-motorized segment.

3.9 Trip phase recognition and public transit stop evaluation

The purpose of this framework is to extract valuable insights from individual trips and to introduce new Key Performance Indicators (KPIs) for assessing the performance of each transit station. The input data for this framework includes the trip data of each user and the outputs are the following insights:

1. Segments of the trip classified as either green (eco-friendly) transport or motorized transport.
2. The access phase to public transport, including the time and distance covered.
3. The egress phase of each trip, detailing the portion of the journey from the end of the trip to the final destination.
4. The waiting time of users, indicating how long users waited to board public transport such as a metro or bus.

From an infrastructure point of view, the framework provides several key performance indicators (KPIs) for each public transport station, whether metro or bus:

1. The real-time and offline average waiting time, including hourly averages, reflecting the waiting time passengers experienced at specific public transport stops.
2. Access and egress time and distance, indicating the time and distance people covered when traveling to specific public transport stops. These data points help to better understand the performance and accessibility level of these stops.

The Trip Phase and Mode Detection Framework as shown in **Figure 3.34** is a detailed system for analyzing travel behavior using GPS data, encompassing data recording and preprocessing, deep learning-based mode prediction, and trip phase recognition. Initially, GPS data is collected from devices such as smartphones or GPS trackers, which provide detailed information including coordinates, timestamps, and potentially additional parameters such as speed and altitude. This raw data is then subjected to rigorous preprocessing steps to enhance its quality and usability. Preprocessing includes data cleaning to eliminate noise and outliers, data filtering to smooth abrupt changes, segmentation to divide continuous data into meaningful intervals based on time, distance, or changes in travel patterns, and feature extraction to derive critical variables such as speed, acceleration, heading, stop duration, and distance traveled.

Next, the preprocessed data is fed into a prediction model, to recognize patterns in the data that correspond to different modes of transport. The training process involves using labeled data to help the model learn the relationship between input patterns and transport modes, adjusting its internal parameters through multiple iterations to minimize prediction errors.

Once trained, the model is capable of predicting the mode of transport for each segment of the GPS data, identifying whether the user is walking, cycling, driving, or using public transport. Following mode prediction, the framework focuses on trip phase recognition, which categorizes each segment of the trip into specific phases. These phases include Access/Egress (initial and final segments of the trip where the user accesses or exits the main transport mode), Waiting Time (periods where the user is stationary, awaiting transport).

For each phase , the framework present both the distance traveled and the time spent, providing a comprehensive and detailed analysis of the user's travel behavior. This categorization allows for the assessment of various trip aspects, such as the proportion of time spent waiting versus moving, the distance covered using green transport modes, and the extent of reliance on motorized transport. By integrating data preprocessing, prediction phase, and detailed phase recognition, this framework offers a robust tool for understanding and optimizing travel patterns, making it invaluable for urban planners, transportation managers, and individuals seeking to analyze their personal travel habits.

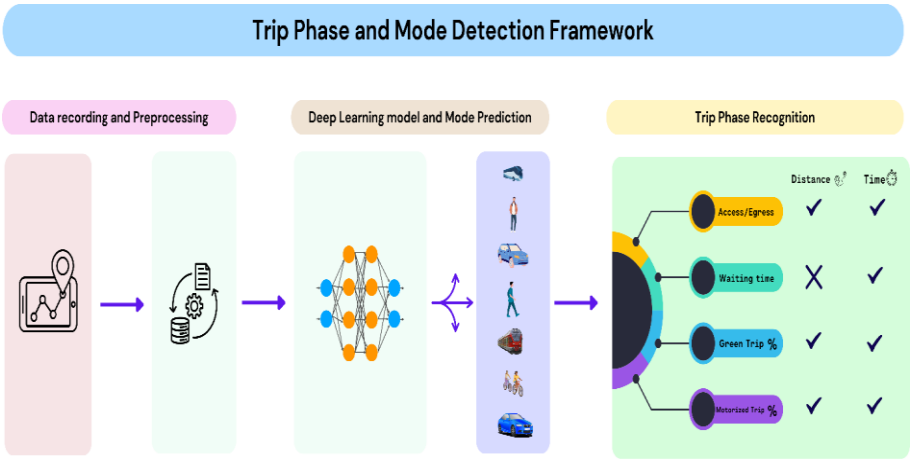

Figure 3.34: Trip Phase Recognition and Public Transit Stop Evaluation (baseline)

3.9.1 Access, Egress and Waiting Time

One of the primary objectives of this thesis is to develop an automated framework for accurately determining passengers' access, egress to and from, and waiting times at public transit stations.

Figure 3.35 **:** Actual access, egress, and waiting at stations

Figure 3.35 show a transportation scenario focused on the actual access, egress, and waiting at stations for two passengers navigating through a city. The map highlights metro stations in orange and bus stations in purple. Access refers to how passengers come to the transit system, such as walking or cycling, while egress involves their exit from the system at various points up to their destination. The dotted lines indicate the walking paths to stations. The visualization focuses on these crucial aspects of the passengers' experience (walking to stops), and the time spent waiting at stations, which are vital components in optimizing travel efficiency and improving user experience in urban public transport systems.

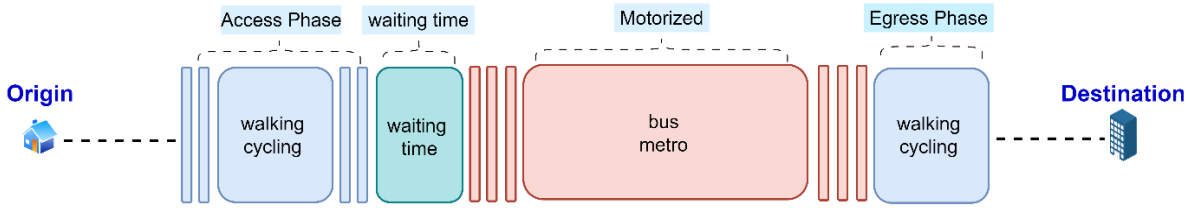

Figure 3.36: Different phases of a trip

Figure 3.36 provides a clear depiction of a typical multimodal journey from origin to destination, breaking down the travel process into distinct phases: the access phase, waiting time, the motorized phase, and the egress phase. Starting from the origin, such as a home, the access phase consists of modes such as walking or cycling that passengers use to reach a transit station, such as a bus stop or metro station. This phase is essential in determining how easily passengers can access the public transportation network. After reaching the station, passengers experience a waiting time, which is the period spend before boarding their mode of motorized transport, such as a bus or metro. The motorized phase represents the main portion of the journey where passengers travel via public transport, covering the bulk of the distance between the origin and destination. Once passengers reach their final transit stop, the egress phase begins. Similar to the access phase, this involves walking or cycling from the station to the final destination, such as an office building or other location. Each phase in this process, including access, waiting, motorized transport, and egress, plays a critical role in the overall travel experience and efficiency. The main goal is detecting and recognizing each trip phase via trip phase detection algorithm. By understanding each phase, transit authorities can better address issues such as waiting times, accessibility, and the connectivity of transit options, all of which directly influence passenger satisfaction and the efficiency of the transit system.

3.9.2 Access, egress, and waiting time detection logic

The access phase (**Figure 3.37**) will be detected from a sequence of predicted transport modes. It is defined by including all walking, bike, and standing activities from the beginning of the trip up to the index where the bus or metro is first observed. After identifying the first instance bus or metro, aggregate all segments predicted by the model as walking, bike, and standing into the access phase. Then, compute the total distance and duration of this phase for each single trip.

The waiting time is detected by checking for standing activities occurring between the last walking or bike activity and the first bus or metro activity, setting a flag when walking or bus is detected, and appending subsequent standing activities to the waiting time list until a bus or metro is encountered. This approach ensures accurate segmentation of different trip phases, crucial for understanding passenger behavior and improving transportation planning and management.

$$\text{Access Time} = \sum_{i=1}^{n} t_i \tag{3.39}$$

$$\text{Access Distance} = \sum_{i=1}^{n} d_i \tag{3.40}$$

$$\text{Egress Time} = \sum_{i=1}^{n} t_i \tag{3.41}$$

$$\text{Egress Distance} = \sum_{i=1}^{n} d_i \tag{3.42}$$

In equations **3.39** and **3.40,** i is the index of the first 200 GPS points in a trip and n is the index of segments before starting motorized. Additionally, in equations **3.41** and **3.42,** i is the index of the last 200 GPS points in a trip, and n is the index of the last labeled motorized segments.

In equation **3.43**, i shows the index of ending walking segments and n demonstrates the index where the motorized part starts. For each equation, the sum of all segments are access, egress, and waiting time and distance.

$$\text{Waiting Time} = \sum_{i=1}^{n} w_i \tag{3.43}$$

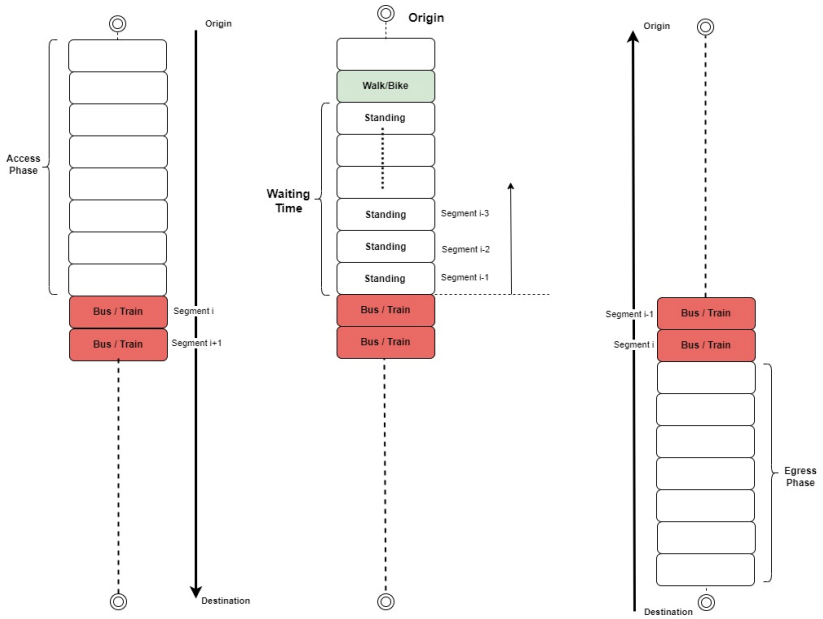

Figure 3.37: Finding access, egress, and waiting phases in a single trip

3.9.3 Public transit accessibility evaluation from attracted trips data

Walking distance to and from public transit stops, as well as waiting time at stations, are critical metrics in evaluating the effectiveness, accessibility, and overall attractiveness of public transit systems. The walking distance directly influences the ease with which passengers can reach transit stops from their points of origin or their final destinations. Shorter and more accessible walking distances make public transit a more viable and appealing option, particularly for those who may face mobility challenges, such as the elderly, individuals with disabilities, or those traveling with young children. Accurately detecting walking distances and utilizing this data to enhance the accessibility of public transit stations is crucial for improving the overall efficiency and user experience of public transportation systems

Waiting time at transit stations also plays a significant role in determining passenger satisfaction and service quality. Accurate, real-time measurement of waiting times provides valuable information on the reliability and frequency of transit services. Extended waiting periods can be highly frustrating for passengers, leading to reduced trust in the system and, ultimately, a decline in ridership. When passengers face long or unpredictable waiting times, they may perceive the system as unreliable, which can discourage them from using public transportation in favor of more predictable alternatives, such as driving.

Figure 3.38 : Attracted trips by PT stops

Figure 3.38 shows how different attracted trips by public transit stops can produce new KPIs to evaluate the accessibility level of each public transit stop. These indicators can help urban planners measure how accessible a transit stop is for users, ensuring the efficiency of the public transport network. The figure emphasizes the role of each station in attracting trips and the potential to use this data to enhance transit planning and station placement decisions for better urban mobility.

To evaluate the accessibility of a bus or metro station, the following key performance indicators (KPIs) and their corresponding formulas can be applied:

1. Average Access Time (AAT):

$$\text{AAT} = \frac{\sum_{i=1}^{n} T_i}{n} \tag{3.44}$$

Where T_i is the access time for user i, and n is the total number of attracted trips by public transit stations.

2. Average Access Distance (AAD):

$$\text{AAD} = \frac{\sum_{i=1}^{n} D_i}{n} \tag{3.45}$$

Where D_i is the access distance for user i, and n is the total number of attracted trips by public transit stations.

3.Average Waiting Time (AWT): $$\text{AWT} = \frac{\sum_{i=1}^{n} W_i}{n} \tag{3.46}$$

Where W_i is the waiting time for user i and n is the total number of trips.

In the methodology section, an advanced framework for detecting transportation modes and recognizing trip phases using GPS data is proposed. The methodology integrates both ML and DL techniques, which are fine-tuned to predict travel behaviors effectively. Two primary datasets, the Geolife and Sussex-Huawei Locomotion datasets, were utilized to train the models. These datasets

provide extensive, labeled data on various modes of transport, enabling the extraction of crucial features such as speed, acceleration, jerk, and bearing rates.

Preprocessing was essential for ensuring data quality, involving steps such as noise filtering, handling missing values, and segmenting continuous data streams into manageable windows for temporal analysis. The RF algorithm was employed for transport mode classification due to its robustness, while Convolutional Neural Networks (CNNs) were utilized for capturing hierarchical temporal features to recognize different trip phases. The study also explored techniques for evaluating public transit accessibility, contributing to the improvement of transit systems based on the passenger's movements.

Chapter 4

Results and Discussion

This chapter starts by defining the selected metrics for evaluating the final results, followed by a detailed presentation of the deep and ML outcomes obtained through feature extraction techniques. The chapter also presents the training process of various ML and DL, exploring how they were fine-tuned and optimized to maximize their performance. Additionally, the chapter examines the specific challenges and considerations encountered when working with datasets, offering insights into the implications of the chosen methodologies for broader applications in the field.

The results of our evaluation demonstrate the robustness and reliability of our transport mode detection model and trip phase recognition framework. Utilizing the confusion matrix, accuracy, precision, recall, and F1 score metrics, the model exhibited high accuracy and balanced performance across various transport modes. The precision and recall scores indicate a strong ability to correctly identify each transport mode, with minimal false positives and false negatives. The F1 score further underscores the model's effectiveness in handling imbalanced classes, ensuring both precision and recall are adequately represented. For the trip phase recognition algorithm, the mean absolute percentage error (MAPE) revealed a low average percentage error, indicating that the algorithm predictions closely match the actual trip phases.

Overall, these results validate the algorithm's potential for practical application in real-world transport mode and trip phase detection scenarios. Finally, the results of the public transit stop evaluation will be presented, following the identification and calculation of new Key Performance Indicators (KPIs) for each public transit station. This evaluation aims to assess the performance and efficiency of transit stops by analyzing these newly defined KPIs, providing insights into the operational strengths and areas for improvement within the public transit system. The chapter will conclude with a discussion of how these findings can inform future transit planning and optimization efforts.

4.1 Performance Metrics

Performance evaluation is a fundamental aspect of ML and DL processes, as it quantifies the effectiveness of the designed algorithm. Several methodologies are available to measure the performance of a machine-learning model and validate its applicability in real-world scenarios. In this study, the confusion matrix, accuracy, precision, recall, and F1 score were utilized to evaluate the performance of our transport mode detection model. Additionally, for the trip phase recognition model, MAPE was employed to assess model performance.

4.1.1 Confusion matrix for multi-class classification

A confusion matrix for multi-class classification is a square matrix of size $N \times N$, where N is the number of classes. The rows represent the actual classes, while the columns represent the predicted classes. Each cell (i, j) in the matrix indicates the number of instances where the actual class is i and the predicted class is j.

4.1.1.1 True Positives (TP)

Diagonal elements (i, i) and represent the number of times that class i was correctly predicted.

4.1.1.2 False Positives (FP)

Non-diagonal elements in each column (j, k) where $j \neq k$. These represent the number of times instances of other classes were incorrectly predicted as class j.

4.1.1.3 False Negatives (FN)

Non-diagonal elements in each row (i, j) where $i \neq j$. These represent the number of times instances of class i were incorrectly predicted as other classes.

4.1.1.4 True Negatives (TN)

Elements that are not in the row or column of a specific class but are correctly classified as not being that class.

4.1.1.5 True Negative Ratio (TNR)

The True Negative Ratio (TNR), also known as Specificity, measures the proportion of actual negatives that are correctly identified as negatives. It is calculated using the formula:

$$TNR = TN / (TN + FP)$$

Where:

TN is the number of True Negatives and FP is the number of False Positives.

Interpretation: A high TNR indicates the model's ability to correctly exclude a specific class when it is not present.

4.1.1.6 False Negative Ratio (FNR)

The False Negative Ratio (FNR), also known as Miss Rate, measures the proportion of actual positives that are incorrectly classified as negatives.

It is calculated as:

$$FNR = FN / (FN + TP)$$

Where:

FN is the number of False Negatives and TP is the number of True Positives.

Interpretation: A low FNR indicates the model's effectiveness in minimizing misclassification of positive instances.

4.2 Accuracy formula using confusion matrix

Accuracy is a commonly used metric for evaluating classification models, including RF and 1D CNN models. It measures the proportion of correctly predicted instances out of the total instances. In the context of predicting different modes of transport (such as walking, car, metro), accuracy can provide a straightforward measure of how well the model is performing and is calculated as the ratio of the number of correct predictions to the total number of predictions.

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} \text{TP}_i}{\sum_{i=1}^{N} (\text{TP}_i + \text{FP}_i + \text{FN}_i + \text{TN}_i)} \quad (4.1)$$

The ratio of correctly predicted instances to the total instances.

4.2.1 Precision

Precision is a metric used to evaluate the accuracy of a classification model, particularly in the context of binary and multi-class classification problems. It measures the proportion of true positive predictions among all instances that were predicted to belong to a particular class.

Precision for Class i :

$$\text{Precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i} \tag{4.2}$$

The ratio of correctly predicted instances of class i to the total instances predicted as class i.

4.2.2 Recall (Sensitivity, True Positive Rate)

Recall, also known as Sensitivity or True Positive Rate (TPR), is a metric used to evaluate the performance of a classification model. It measures the ability of the model to correctly identify all relevant instances of a particular class. In other words, recall focuses on the proportion of actual positives that are correctly classified by the model.

Recall (Sensitivity) for Class i :

$$\text{Recall}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i} \tag{4.3}$$

The ratio of correctly predicted instances of class i to the total instances of actual class i.

4.2.3 F1 Score

The F1 Score is a metric used to evaluate the performance of a classification model, particularly in scenarios where the data is imbalanced. It is the harmonic mean of precision and recall, providing a single metric that balances both the precision and recall of the model.

$$\text{F1} - \text{Score}_i = 2 \times \frac{Precision_i \times Recall_i}{Precision_i + Recall_i} \tag{4.4}$$

It's the harmonic mean of precision and recall for class i.

4.2.4 Mean Absolute Percentage Error

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n} \left|\frac{A_i - F_i}{A_i}\right| \times 100 \tag{4.5}$$

Where:

n is the number of observations

A_i is the actual value

F_i is the forecasted or predicted value

MAPE expresses the error as a percentage, which makes it useful for understanding the prediction error in relative terms.

4.3 Machine Learning Results

The RF model, utilized as the primary ML model for transport mode detection, leverages a range of features derived from motion data. After applying the Savitzky-Golay filter and other preprocessing steps to each trip's data, trips with fewer than 60 GPS points are removed. The remaining trips are then segmented into varying sizes for prediction, including segments of 30, 60, and 200 GPS points, to assess model performance across different granularity levels.

From each data segment, the primary features, as explained in the previous section, are extracted to serve as input for the ML models. The dataset is split into training and test sets, with 80 percent allocated for training and 20 percent for testing. Three different window lengths are trained and tested, ensuring no overlap between the windows. Once the model is trained using 26,904 labeled windows for five transit modes, with a window size of 200, predictions are made on the test set. Evaluation metrics such as accuracy, along with classification reports detailing precision, recall, and F1-score, are generated to assess the model's performance. The accuracy metric gives a simple score on the correctness of the predictions, while the classification report provides deeper insights into how well the model performs across different transport modes.

	Precision	Recall	F1-score
bike	0.87	0.82	0.84
bus	0.84	0.78	0.81
car	0.80	0.74	0.77
metro	0.89	0.80	0.85
walking	0.82	0.97	0.89

Table 4.1: Transport mode classification performance, 200 seconds time window

Table 4.1 shows the results of transport mode detection. Each time window consists of 200 GPS points, enabling the capture of various dynamic behaviors across different modes such as biking, walking, car, and public transit. The evaluation metrics show a balanced performance across all modes. Biking achieves a precision of 0.87 and a recall of 0.82, while walking stands out with the highest recall of 0.97 and an F1-score of 0.89, indicating high sensitivity. The metro achieves a precision of 0.89, showcasing the model's ability to correctly identify trips with fewer false positives. The car mode shows slightly lower recall (0.74), reflecting a more challenging classification scenario. These metrics highlight the robustness of the model, leveraging diverse features to ensure reliable predictions across different transport modes. The final average accuracy arrived at 84 percent**.**

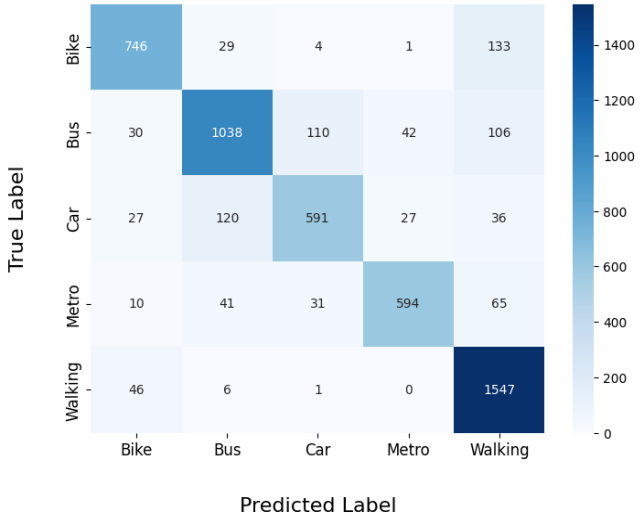

Figure 4.1: Confusion matrix, 200 seconds time window

Figure 4.1 illustrates the confusion matrix provides a detailed breakdown of the model's performance in classifying transport modes across biking, bus, car, metro, and walking. Each row represents the true label, while each column indicates the predicted label. The model correctly classified 746 biking instances but misclassified 133 as walking and 29 as bus. For buses, 1038 instances were accurately identified, though 110 were misclassified as cars. Walking achieved the highest correct predictions (1547), with very few misclassifications. This matrix highlights the model's strong accuracy in classifying walking and moderate confusion between bus, car, and metro modes. The errors show where the model struggles, especially between car and bus classifications.

	Average F-score	Average precision	Average recall
Best random forest	83.2	84.4	82.2
[48]	62.4	63.2	63.5

Table 4.2**:** Comparison between previous research

Table 4.2 shows the performance of the best RF model, with an average precision of 84.4 %, recall of 82.2 %, and F1-score of 83.2, indicating its high capability in correctly classifying transportation modes. In contrast, the results from the study [47] with the same preprocessing and windowing procedure (200 GPS points) show significantly lower performance, with an average precision of 63.2%, recall of 63.5%, and F1-score of 62.4. This substantial difference suggests that the RF model used in this study is more effective, due to the better feature engineering, and model tuning, which improved prediction accuracy across various transport modes.

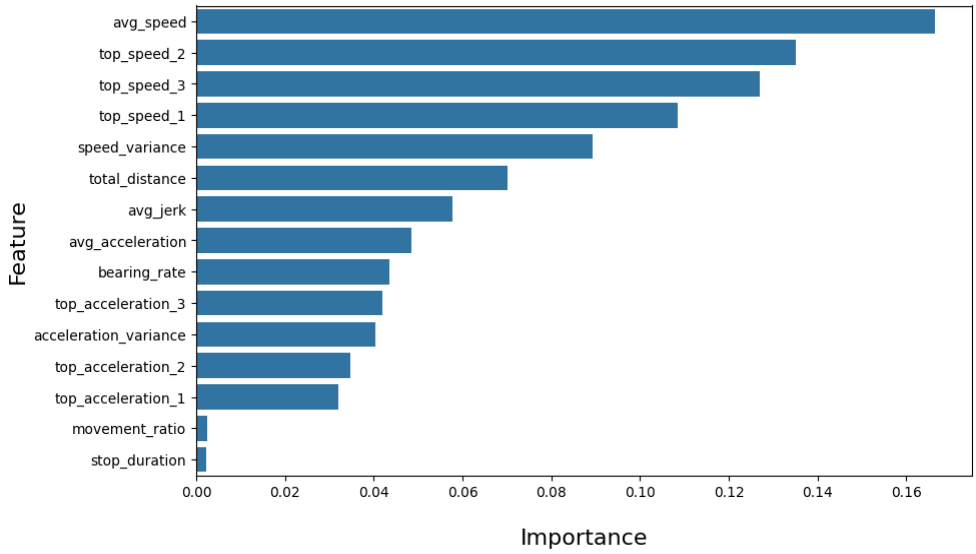

Figure 4.2: Feature Importance, 200 seconds time window

Figure 4.2 presents the feature importance of 200 window size, ranking the features that contribute most to the RF model's predictions in transport mode classification. Feature importance values are computed based on how much a feature reduces the uncertainty (or impurity) at each split in the decision trees that make up the RF. The importance score of each feature is normalized so that the sum of all feature importances equals 1, making it a relative measure rather than an absolute one. The most influential feature is average speed, followed closely by top_speed_2 and top_speed_3, highlighting the critical role of high-speed recordings in distinguishing between different transport modes. Other significant features include top_speed_1, speed_variance, and total distance, which add valuable context about speed fluctuations over time. Additionally, total distance and average jerk contribute to understanding movement dynamics. While features such as bearing rate, stop duration, and movement ratio are less influential, they still play an important role in capturing behaviors such as direction changes and stationary periods, which are essential for distinguishing walking or biking. **Figure 4.2** offers insights into how the model leverages various aspects of movement data to predict transport modes effectively.

To compare the effects of different time windows when working with GPS data, the model is trained, tested, and evaluated using three different window sizes: 200, 60, and 30 GPS points per window. The preprocessing steps remained consistent across all different time windows. For each segment, the primary features, as described in the previous section, were extracted and used as input for the ML models. By varying the window size, the goal is to assess how the granularity of the time window impacts the classification accuracy and feature importance. Larger windows, such as 200 points, typically capture broader movement patterns, while smaller windows, such as 30 points, offer finer, more immediate details about the movement. The results for each window size help us understand the trade-offs between detailed data and general patterns.

Table 4.3 shows the transport mode classification performance using a 60-second time window with 80736 separated segments that reveals balanced results across several transport modes. Bike achieves a precision of 0.83, recall of 0.76, and an F1-score of 0.80, indicating a good balance

between correct and missed classifications. Bus shows similar performance with an F1-score of 0.76, while car has a slightly lower F1-score of 0.70 due to a recall of 0.63. Metro performs well with an F1-score of 0.85. Walking has the highest recall at 0.96, reflecting strong sensitivity, and an F1-score of 0.86. The total average accuracy arrives to 80 percent.

	Precision	Recall	F1-score
bike	0.83	0.76	0.80
bus	0.80	0.72	0.76
car	0.78	0.63	0.70
metro	0.88	0.82	0.85
walking	0.75	0.96	0.84

Table 4.3 : Transport mode classification performance, 60 seconds time window

Figure 4.3 shows confusion matrix for the transport mode classification using a 60-second time window illustrating the model's accuracy and misclassifications across different modes. Walking is the most accurately predicted class with 4406 correct predictions, showing minimal confusion with other modes. Bus also performs well with 2899 accurate predictions, but there is some confusion with car (314 instances) and walking (496 instances).

Biking is correctly identified 2029 times but is sometimes confused with walking (500 instances). Car and metro have moderate accuracy, with 1572 and 1963 correct predictions, respectively, and some misclassifications with other transport modes. This matrix highlights areas where the model excels and where more confusion arises, particularly between modes such as bus and car, and bike and walking.

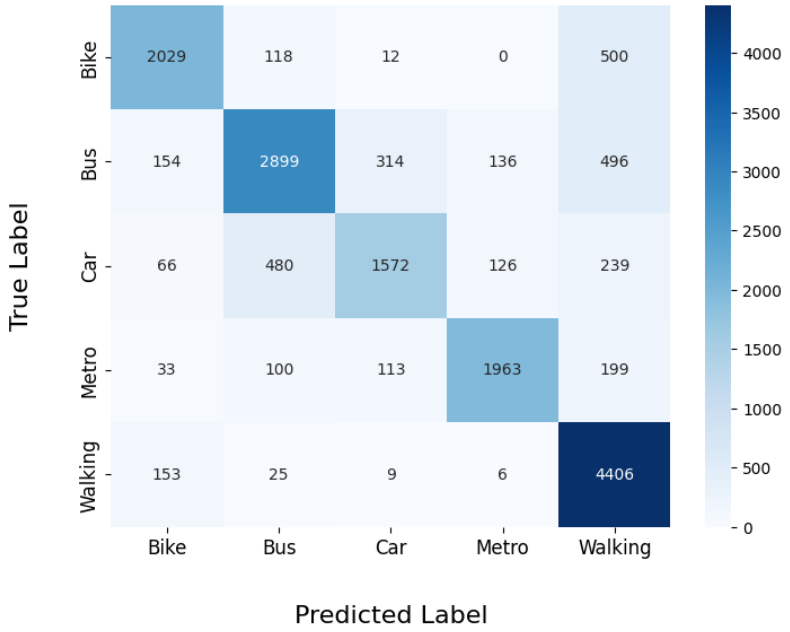

Figure 4.3 : Confusion matrix, 60 seconds time window

Figure 4.4 shows the feature importance plot for the transport mode classification model, using a 60-second time window, highlights average speed as the most influential feature, indicating that the overall speed of movement significantly impacts the model's predictions. Following closely are the top two speed values (top_speed_1, top_speed_2, top_speed_3), which provide insight into the highest speeds during the time window. Speed variance and total distance also play critical roles, showing the importance of both speed variability and distance covered. Other features, such as top acceleration and bearing rate, contribute to refining the classification by capturing changes in movement dynamics and direction. Less influential, but still important, features such as stop duration and movement ratio help distinguish between more stationary or smooth-moving modes such as walking and biking. This ranking shows how the model relies on speed and movement characteristics to classify different transport modes effectively.

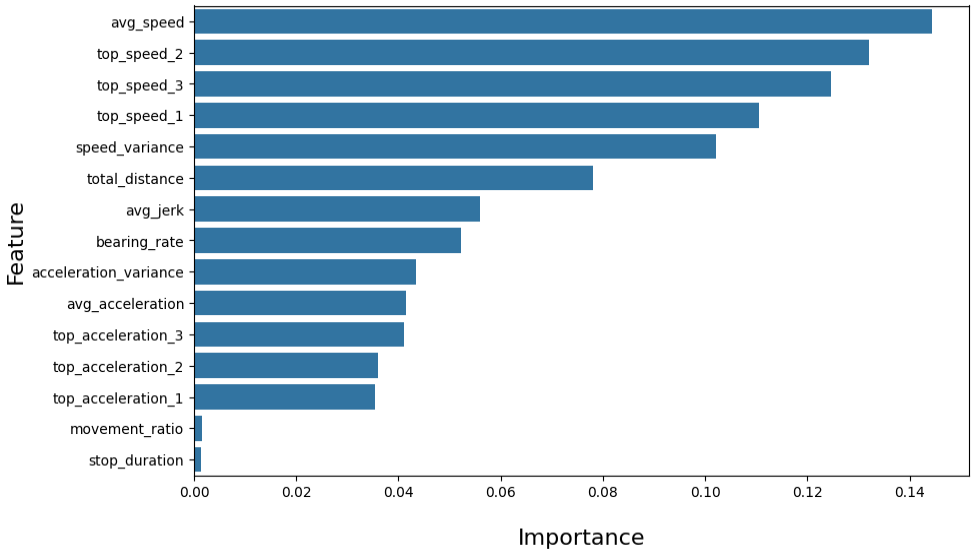

Figure 4.4 : Feature importance, 60 seconds time window

Table 4.4 and **Figures 4.5** and **4.6** illustrate the model's performance using 30-second time windows, including the confusion matrix and feature importance. The results highlight a reduction in accuracy compared to larger time windows, reflecting how shorter windows may capture less comprehensive patterns in movement data, leading to more misclassifications. Final accuracy arrives to 77 percent.

This comparison underscores the importance of time window size in determining classification accuracy, with larger windows generally offering better performance due to their ability to encapsulate broader movement behavior and patterns.

	Precision	Recall	F1-score
bike	0.82	0.73	0.77
bus	0.76	0.70	0.73
car	0.76	0.59	0.66
metro	0.88	0.78	0.83
walking	0.72	0.95	0.82

Table 4.4 : Transport Mode Classification Performance, 30 seconds time window

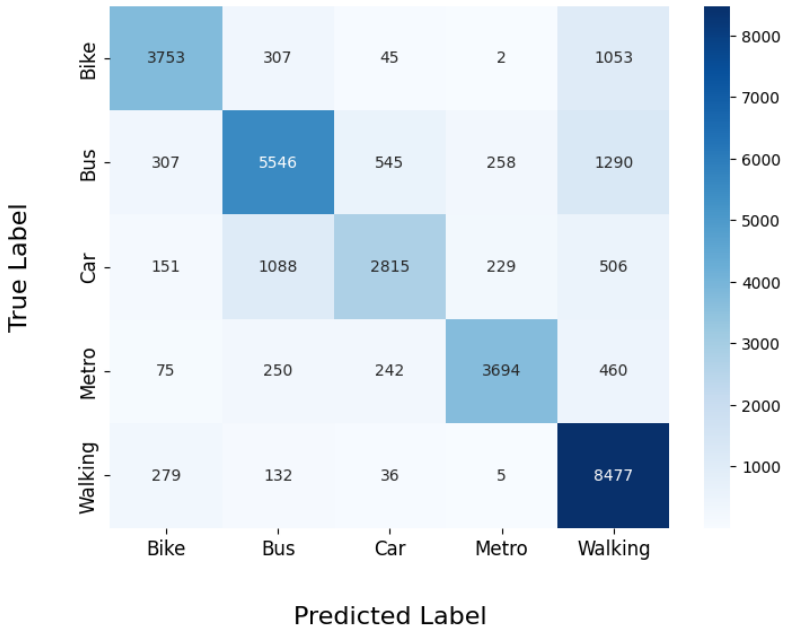

Figure 4.5: Confusion matrix, 30 seconds time window

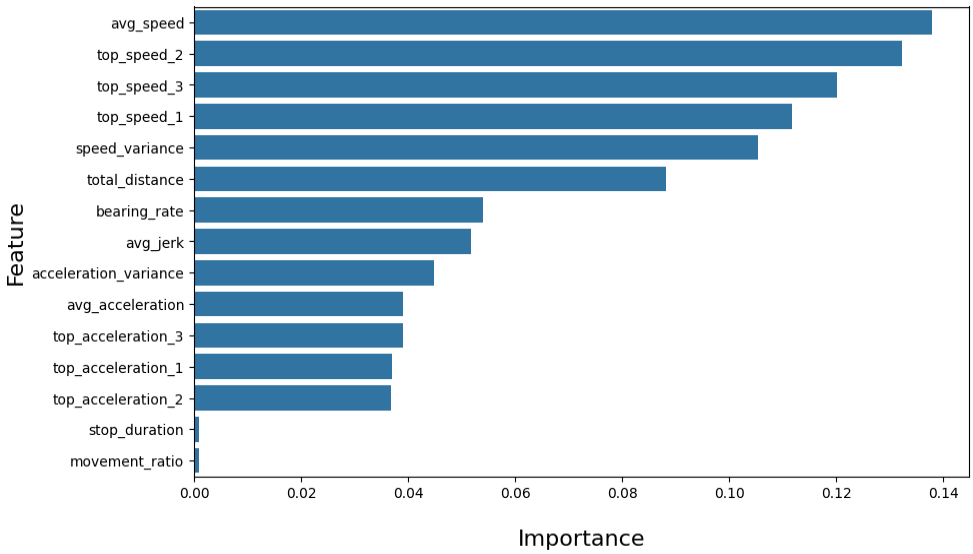

Figure 4.6: Feature importance, 30 seconds time window

The comparison of the results across different time windows (200, 60, and 30 seconds) reveals some significant insights into the trade-offs between granularity and overall classification performance. The 200-second window generally produced the best results, offering an average accuracy of 84%. The precision, recall, and F1-scores for this larger window were robust across all transport modes, with particularly strong performance in distinguishing between biking, walking, and metro modes. The longer window allowed the model to capture more sustained patterns, particularly useful in distinguishing between modes with distinct travel behaviors such as walking and metro.

The 60-second window balanced short-term and long-term trends, resulting in slightly lower performance compared to the 200-second window, with an average accuracy of 80%. While walking still achieved a high recall of 0.96, the F1-scores for bus and car modes dropped, reflecting increased misclassifications. This suggests that while this window size is more responsive to variations, it doesn't always capture the broader behavioral trends as effectively.

Finally, the 30-second window captured very short-term behaviors but struggled with overall classification accuracy, achieving an average of 77%. The precision and recall scores for biking and car modes were particularly affected. This reflects the model's difficulty in classifying modes accurately when relying on smaller data windows, likely due to insufficient data to capture broader movement trends. Thus, while smaller windows can offer finer details, they tend to underperform in capturing long-term travel behaviors compared to larger windows.

4.4 Hyperparameter Tuning

Hyperparameter tuning offers a significant advantage in improving the performance and generalization ability of ML models. Unlike model parameters, which are learned during training, hyperparameters must be set before the learning process begins, and their impact on model performance can be substantial. Tuning techniques such as GridSearchCV or RandomizedSearchCV systematically explore different configurations of hyperparameters, ensuring that the model does not underperform due to poorly chosen settings. For instance, in RF, hyperparameters such as the number of trees (n-estimators), the maximum depth of each tree (max-depth), and the number of features to consider at each split (max-features) can all significantly affect the balance between model accuracy and overfitting. Effective hyperparameter tuning also increases model robustness and ensures better

generalization to unseen data, leading to more reliable and consistent predictions across different datasets. Additionally, automated tuning processes such as GridSearchCV leverage cross-validation to prevent overfitting, ensuring that the selected hyperparameters provide strong performance across multiple folds of the data rather than just on a specific train-test split. Hyperparameter tuning was tested across different time window lengths (30, 60, and 200 seconds), and the variations in performance were minimal.

The first parameter, n-estimators, specifies the number of decision trees to be constructed in the forest, with values of [100, 200] indicating the model will try 100, and 300 trees, respectively. The best value after hyperparameter tuning is 200. A higher number of trees generally leads to better performance by reducing the variance of the predictions, as the ensemble effect becomes stronger. However, this comes at the cost of increased computational time. The second parameter, max-depth, controls the maximum depth of each tree, with values of [10, 20]. The optimized value is 20. Limiting the depth helps to prevent overfitting, as overly deep trees may become too specific to the training data, capturing noise and leading to poor generalization on unseen data.

The min-samples-split parameter, with values [2, 5], determines the minimum number of samples required to split an internal node. Lower values such as 2 allow more frequent splitting, leading to more complex trees, while higher values such as 10 create simpler, more generalized trees. The best value is 2. This parameter helps control how the model handles small and potentially noisy splits in the data. Similarly, min-samples-leaf controls the minimum number of samples required to be at a leaf node, with values [1, 2] and value 1 is the best option. Setting a higher min-samples-leaf prevents the model from creating overly specific leaf nodes that could lead to overfitting.

In conjunction with this parameter grid, GridSearchCV test each combination of hyperparameters and identify the best configuration. GridSearchCV performs an exhaustive search over the hyperparameter space defined in param-grid, evaluating each combination using cross-validation, which is controlled by the cv = 5 argument. This means the dataset is split into five parts, with the model being trained on four parts and validated on the remaining one, ensuring a robust estimate of performance. Additionally, the use of stratified splits (stratify = y in train-test-split) ensures that the class distribution in the training and test sets mirrors that of the original dataset, particularly important when working with imbalanced datasets to avoid skewed training outcomes.

4.5 Machine learning (six modes)

The previous results were obtained from applying and evaluating a RF model aimed at recognizing modes of transportation, specifically car, bus, train, walking, and cycling. The objective of this evaluation was to benchmark the model's performance against similar studies or papers that also focused on these five modes of transportation. By using the same set of transportation modes for training and testing, the goal was to create a standardized comparison and validate the effectiveness of the trip phase recognition model. **Table 4.5** presents the results of training the ML model using the Geolife and Sussex datasets, which include transportation modes such as car, train, bus, cycling, walking, and standing. In total, 6,443 separate trips were identified. The time window used as the core of the trip phase recognition algorithm was set to 60 seconds. The final model will be trained using 81,789 labeled windows, each with a uniform size of 60 seconds. **Table 4.5** shows the performance of the trained model across various transportation modes in terms of precision, recall, and F1-score.

For biking, the model demonstrates a solid performance with a precision of 0.83, meaning 83 % of predicted biking instances are correct, though the recall of 0.76 suggests it misses some actual biking trips. Bus trips have a slightly lower precision of 0.81 and a recall of 0.72, resulting in an F1-score of 0.76, indicating room for improvement in capturing all bus instances. Car trips show a noticeable gap, with a precision of 0.78 and a recall of 0.65, meaning the model struggles more with identifying car trips, reflected in the lower F1-score of 0.71. In contrast, the model performs very well for metro trips, with a high precision of 0.90 and recall of 0.81, leading to an F1-score of 0.85. Finally, for walking, the model has a relatively lower precision of 0.75 but compensates with a strong recall of 0.96, leading to an F1-score of 0.84, showing that the model effectively captures most walking instances despite some false positives. Overall, the majority of standing segments were classified with high accuracy.

	Precision	Recall	F1-score
bike	0.83	0.76	0.80
bus	0.81	0.72	0.76
car	0.78	0.65	0.71
metro	0.90	0.81	0.85
standing	0.91	0.98	0.94
walking	0.75	0.96	0.84

Table 4.5**:** Transport mode classification performance, 60 seconds time window

The confusion matrix presented in **Figure 4.7** illustrates the performance of a classification model in predicting transportation modes based on true labels (y-axis) and predicted labels (x-axis). The matrix reveals that the model performs well for certain classes such as Walking and Bus with the majority of true Walking instances (4383) correctly classified, and 2938 instances of Bus also accurately predicted. However, the model struggles with Bike and Walking classifications, as there is misclassification, such as 493 Bike instances being predicted as Walking and 492 Car instances predicted as Bus. The diagonal values represent the correct predictions for each transportation mode, while the off-diagonal values indicate misclassifications. This matrix provides insight into the model's strengths and weaknesses, highlighting areas for potential improvement, particularly in distinguishing between similar modes such as Bike and Walking.

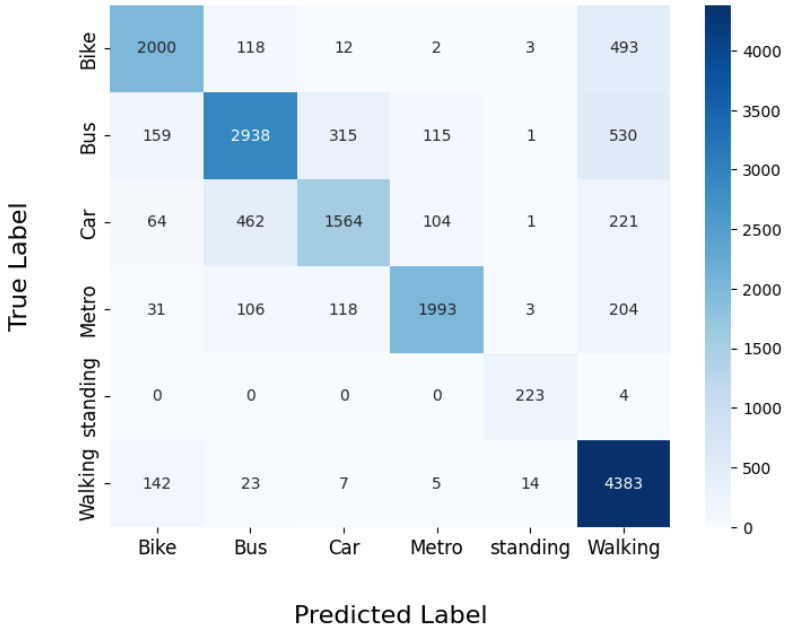

Figure 4.7: Confusion matrix, 60 seconds time window

The feature importance plot in **Figure 4.8** illustrates that average speed is the most crucial predictor for the classification model, followed by the top speed metrics across different segments (top_speed_3, top_speed_2, top_speed_1), highlighting the significant role of speed-related features. Speed variance, total distance, and top acceleration values also contribute meaningfully, indicating that variations in speed and travel distance help differentiate transportation modes. Bearing rate, average jerk, and acceleration variance reflect changes in direction, velocity, and ride stability, further enhancing the model's predictive accuracy. Lower-ranked features, such as movement ratio and stop duration, have less impact, suggesting that while these contribute, speed and acceleration metrics are the most influential in classifying transportation modes.

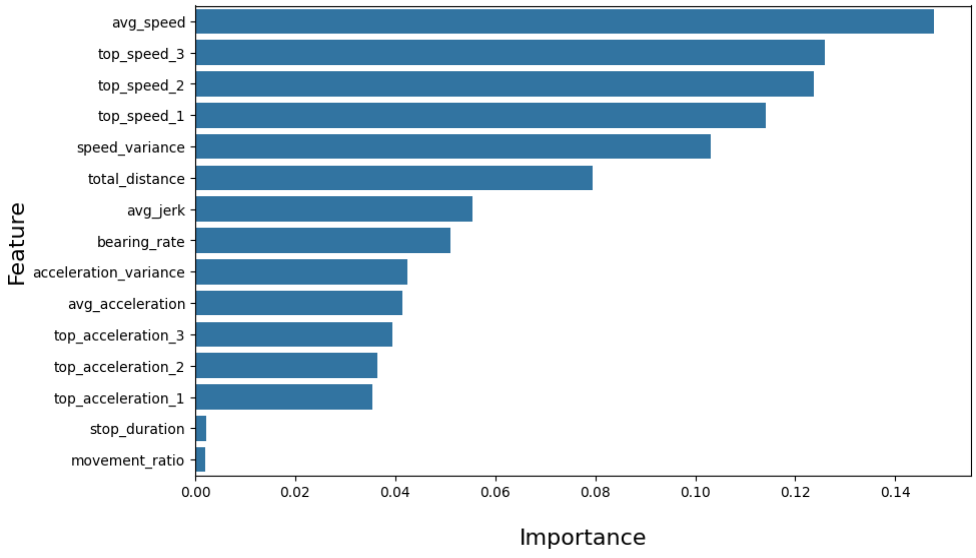

Figure 4.8: Feature Importance, 60 seconds time window

4.6 Deep Learning Results

A Convolutional Neural Network (CNN) was developed using Keras to assess its performance through various metrics. The process began with importing essential libraries for model construction, training, and evaluation. Keras was employed for building the CNN architecture, with tensorFlow as the backend, utilizing the Adam optimizer, which is known for its efficiency in gradient-based optimization, particularly in managing sparse gradients. To evaluate the model's performance, metrics such as precision, recall, F1-score, and accuracy were computed using Scikit-learn, a widely recognized library for statistical and ML applications.

The best model architecture starts with an input layer of shape (200, 4), indicating that the model is designed to process sequences of 200 GPS points, each characterized by four distinct features. This setup is typical in time-series analysis or sensor data processing, where the temporal sequence and multivariate input structure are important for pattern recognition. The reason for training our DL model with 5 modes and a 200 GPS window size is to ensure comparability with previous research. By adopting the same number of modes and a similar GPS window size as used in earlier studies, we aim to benchmark our model's performance and consistently validate our findings. The architecture incorporates three 1D convolutional layers (Conv1D), with increasing filter sizes of 32, 64, and 64, respectively. This progression allows the model to progressively capture more intricate patterns within the data by leveraging the convolutional layers ability to detect local dependencies. Each convolutional layer is followed by Batch Normalization, which normalizes activations, stabilizing the learning process and enabling faster convergence by mitigating internal covariate shifts.

After the convolutional layers, a GlobalAveragePooling1D layer is used to reduce the dimensionality of the feature maps. This layer pools the average value of each feature map across the time steps, converting the 1D sequence into a fixed-length vector representation, which can then be processed by fully connected (Dense) layers. The model includes two dense layers, with 100 and 50 units, respectively, each employing the ReLU (Rectified Linear Unit) activation function to introduce non-linearity and ensure the model can learn complex representations. To mitigate overfitting, dropout layers are included after each dense layer, with a dropout rate of 20 %, which helps prevent the model from becoming overly reliant on any one subset of neurons by randomly dropping 20 % of neurons during each training iteration.

The output layer consists of 5 and 6 modes, corresponding to a multi-class classification task with 5 and 6 possible categories. The softmax activation function is applied in the output layer to ensure that the predicted outputs represent a probability distribution across the classes, making it suitable for multi-class classification problems.

To optimize the model, the Adam optimizer is used with different learning rates from the list lr_values = [0.001, 0.0001, 0.00001]. The Adam optimizer, which combines the benefits of Adaptive Gradient Algorithm and Root Mean Square Propagation, is particularly effective for models with sparse gradients and large parameter spaces, such as deep neural networks. For each learning rate, the model is compiled with categorical cross-entropy as the loss function, which is appropriate for multi-class classification tasks as it measures the difference between the true labels and the predicted probability distribution across the classes. Accuracy is used as the performance metric to gauge the overall correctness of the model's predictions during training.

During training, the model checkpoint callback is employed to save the model's weights whenever the validation accuracy improves. This ensures that the best-performing model, in terms of validation accuracy, is preserved during the training process. The model is trained for 20 epochs with

a batch size of 64, using the training data (X_train, y_train) and validated on the test set (X_test, y_test). Moreover, the model has 31,472 parameters, out of which 31,152 are trainable and 320 are non-trainable, occupying approximately 122.94 KB of memory.

Once training is complete, predictions are made on the test set. The predicted probability outputs are converted into class labels using the argmax function, which selects the class with the highest predicted probability for each sample. To evaluate the model's performance across each class, precision, recall, F1-score, and accuracy are computed. Overall accuracy is calculated as the proportion of correct predictions overall predictions made by the model.

A confusion matrix is also generated, which provides a detailed view of the model's performance by showing the true positive, false positive, true negative, and false negative predictions for each class. The confusion matrix is visualized using seaborn to provide an intuitive graphical representation of the classification results for each learning rate. This allows for easy identification of specific classes where the model may struggle, such as misclassifications or imbalanced performance across categories.

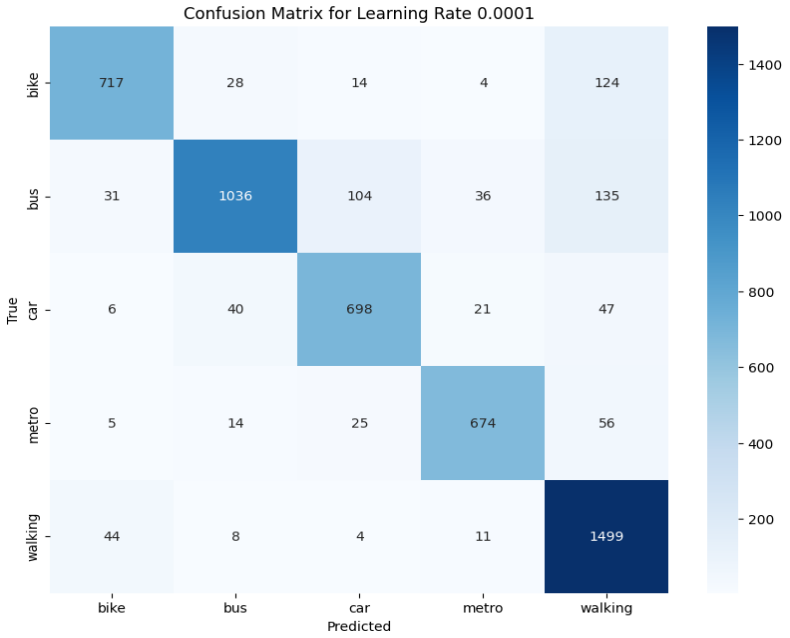

Figure 4.9: Confusion matrix, 200 seconds time window

Figure 4.9 illustrates the performance of a deep learning model used for transport mode classification, visualized through a confusion matrix and supported by precision and recall metrics for five distinct modes: bike, bus, car, train, and walking. The rows in the confusion matrix represent the actual labels, while the columns correspond to the predicted labels, providing a comprehensive overview of how well the model performed across these transport modes. The diagonal values, highlighted in green, show the correctly predicted instances for each mode, while the off-diagonal values indicate misclassifications. For example, the model correctly identified 717 out of 887 bike trips, but it misclassified 28 as bus, 14 as car, and 124 as walking.

The overall performance can also be assessed through the precision and recall scores provided at the bottom. Precision indicates the proportion of correct predictions out of all predicted instances for each mode. For instance, the model achieved a precision of 89.7 % for bike trips, meaning that 89.7 % of all instances predicted as bikes were correct.

On the other hand, recall reflects the model's ability to correctly identify all actual instances of a given mode, such as a recall of 80 % for bike trips, meaning that the model correctly identified 80 % of all actual bike trips. The precision for walking, at 80 %, is the lowest among the five modes, indicating that the model tends to misclassify other modes as walking more frequently than for other modes.

The highest recall score of 95 % is for walking, which suggests the model is better at identifying actual walking trips. These metrics are crucial in evaluating the model's ability to differentiate between modes of transport and highlight areas where misclassifications are more common, providing insight into potential improvements in future model iterations.

Table 4.6 provides an overview of the test accuracy of various studies (200 seconds) used for transport mode classification. The best CNN model achieved the highest test accuracy at 86.55 %, demonstrating its superior performance in accurately classifying transport modes compared to the other studies. The other papers, as indicated by their citations, achieved test accuracies of 84.8% and 74.1%, respectively.

These accuracies reflect the effectiveness of each model in generalizing to unseen data and emphasize the success of the best CNN model in this specific task. This comparison is critical in assessing the effectiveness of different DL approaches and justifies the use of the CNN model as the preferred method for the given task.

Model	Test Accuracy
Best CNN Model	86.55
[48]	84.8
[1]	76.2

Table 4.6**:** Comparison with other studies, 200 seconds time window

The selected trips in **Figure 4.10** were fed into a mode detection prediction model to predict the corresponding modes of transport for each 200 GPS point (GeoLife dataset).

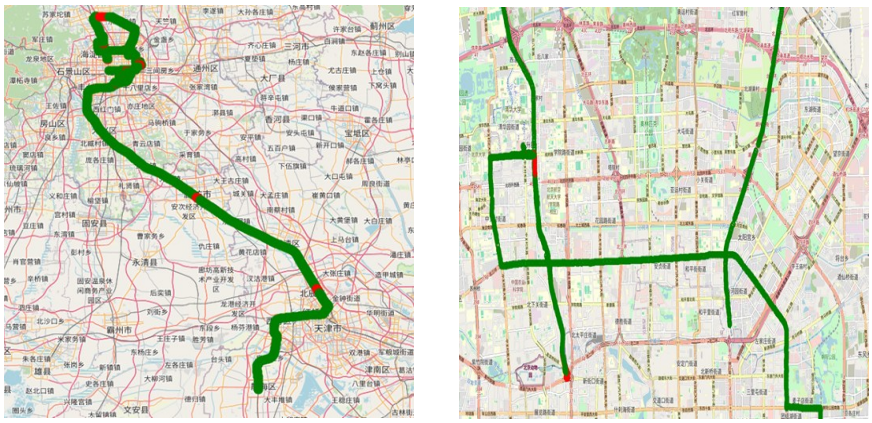

Figure 4.10: Selected trips for prediction, green (true prediction) - red (false prediction)

4.7 Deep learning (six modes)

To incorporate deep learning as a core component of the trip phase recognition framework, the standing labels from the Sussex dataset have been utilized. One key reason for including standing labels in the model was to enable the prediction of passenger waiting times. The confusion matrix with 60 seconds time window illustrates in **Figure 4.11** shows the performance of a DL model designed to classify six modes of transportation: bike, bus, car, metro, standing, and walking. The diagonal elements represent correct predictions, with 2026 trips correctly classified as bike, 3033 as bus, 1265 as car, 2076 as metro, 201 as standing, and 4409 as walking.

The model demonstrates high accuracy in predicting walking, bus, and bike trips. However, there are misclassifications: for instance, 505 bike trips were incorrectly classified as walking, 102 bus trips were misclassified as car, and 30 walking trips were misclassified as bus. These misclassifications, represented by the off-diagonal elements, indicate areas where the model could be improved, particularly in differentiating between similar transport modes such as bus and car. The color gradient in the matrix visually highlights these trends, with darker shades of blue indicating a higher number of predictions. Overall, while the model performs well in several categories, there is room for improvement in distinguishing between some transportation modes.

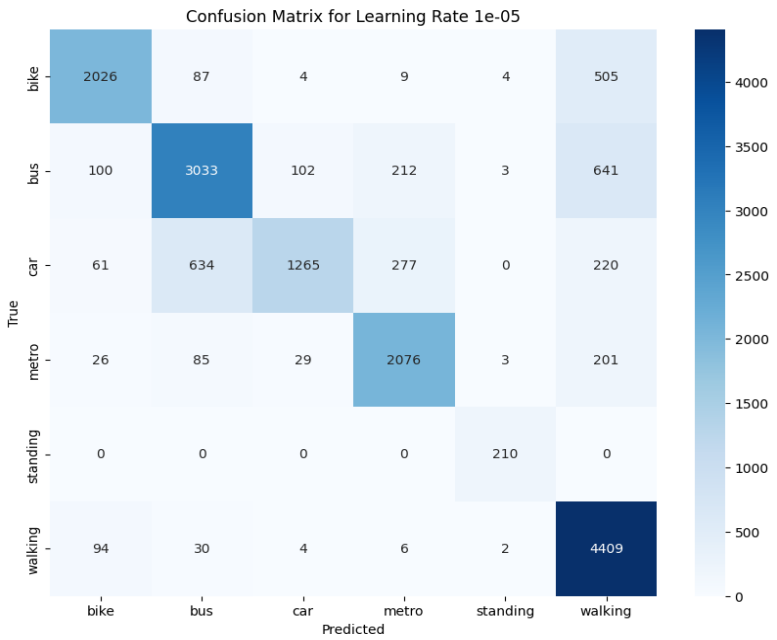

Figure 4.11: Confusion matrix, 200 seconds time window

Table 4.7 shows the performance of the model across different classes can be assessed using precision, recall, and F1-score. For the bike class, the model demonstrates a high precision of 0.87 but a lower recall of 0.76, leading to an F1-score of 0.81, indicating good performance in correctly predicting bikes but missing some true instances. The bus class shows slightly lower performance, with a precision of 0.78, recall of 0.74, and F1-score of 0.76, reflecting similar trends. The car class has a high precision of 0.90 but a lower recall of 0.51, resulting in an F1-score of 0.65, suggesting the model is precise in identifying cars but misses many.

For metro, the model performs well across all metrics, with an F1-score of 0.83 due to balanced precision (0.80) and recall (0.85). The standing class achieves the highest performance, with near-perfect precision (0.92), recall (0.99), and F1-score (0.96), indicating excellent detection. Finally, for walking, the precision is lower at 0.73, but a high recall of 0.97 brings the F1-score to 0.83, indicating the model identifies most walking instances but with some false positives.

	Precision	Recall	F1-score
bike	0.87	0.76	0.81
bus	0.78	0.74	0.76
car	0.90	0.51	0.65
metro	0.80	0.85	0.83
standing	0.94	0.99	0.97
walking	0.73	0.97	0.83

Table 4.7: Transport mode classification performance, 60 seconds time window

4.8 Trip Phase Recognition

The primary objective of the trip phase recognition algorithm is to accurately detect and analyze access, egress, and waiting times for individual trips. Unlike traditional methods, which often rely on Geographic Information Systems (GIS) techniques, user surveys for access and egress times and distances, or tracking users from stations to destinations, this framework leverages trip GPS data as its input. The framework processes this data to output precise access, egress, and waiting times, along with the corresponding distances and durations for each trip phase. This approach provides a more integrated and precise analysis of the temporal and spatial dynamics of trip phases, significantly improving the accuracy and overall results.

The Geolife dataset contains 6,399 user-labeled trips. To test and validate the framework, a subset of trips meeting specific conditions was selected. The primary condition for each trip was that it began with a walking or cycling segment, followed by a bus or metro segment, and concluded with another walking or cycling segment. The goal is to detect walking or cycling as the access phase prior to the start of the motorized portion and to identify the last observed walking or cycling segment after the motorized portion as the egress phase. Additionally, the total distance and time for each phase are computed.

This process ensures that only trips with significant multimodal characteristics are included as shown in **Figure 4.12**, allowing for a more focused analysis of the transitions between different modes of transport. Finally, applying these conditions resulted in the extraction of 566 trips that meet the specified criteria. These trips, characterized by their multimodal nature and sufficient GPS data points, provide a robust dataset for testing and validating the algorithm. This selection ensures that the analysis captures detailed and meaningful insights into the access, egress, and waiting phases of trips, particularly in the context of transitions between different modes of transportation such as walking, biking, buses, and metro services.

Figure 4.12: selected public transport-based trips

Figure 4.13 displays a route map overlaid with two different modes of transport: walking (indicated in green) and bus travel (indicated in red). The green segments represent the walking portions, typically shorter in duration and distance, found at both the start and end of the trip. The red segment in the middle represents the bus travel, which covers a longer distance in a relatively straight path. The goal is to automate the detection of different trip phases, such as walking, cycling, and transit, using GPS data.

Figure 4.13 : One sample of selected public transport-based trips

Each single trip of 566 labeled trips will now be processed through our framework. Initially, trips are divided into segments of 60 GPS points. After completing the necessary preprocessing and cleaning steps, these segments are fed into our DL model, which predicts the corresponding mode of transportation for each segment.

Following the mode prediction, the trip phase recognition algorithm performs postprocessing to identify and extract the access and egress phases of each trip. Additionally, the framework determines the corresponding time and distance for these phases and assesses whether any waiting time is present during the trip. This comprehensive analysis provides a detailed understanding of the trip's phases, enhancing the accuracy of trip phase detection and interpretation. To predict trip phases, all segments labeled as walking or biking by users were extracted before starting bus or metro segments.

4.8.1 Trip Phase Recognition (GeoLife dataset - China)

This section presents the results of applying the trip phase recognition algorithm to a selection of trips from the GeoLife dataset. **Figure 4.14** displays the results of the trip phase recognition framework on a bus-based trip, specifically trip number 366. It shows two distinct phases of the journey: the Access Phase and the Bus Trip phase. The access phase is the portion of the trip where the traveler reaches the bus station on foot, and the bus trip represents the segment after boarding the bus. In the Access Phase, the red path represents the actual route taken by the user, while the green path indicates the model's predicted access route. The two paths closely overlap, although there are some visible discrepancies between the actual and predicted routes. The blue circle highlights the bus station where the access phase ends and the bus trip phase begins. The actual access distance is 428 meters, while the predicted distance is slightly lower at 350 meters. Similarly, the actual access time is 374 seconds, while the model predicts a shorter time of 254 seconds. This analysis provides a clear visual and numerical comparison between the model’s predictions and the actual trip characteristics, demonstrating the algorithm's effectiveness in recognizing different phases of a bus-based trip.

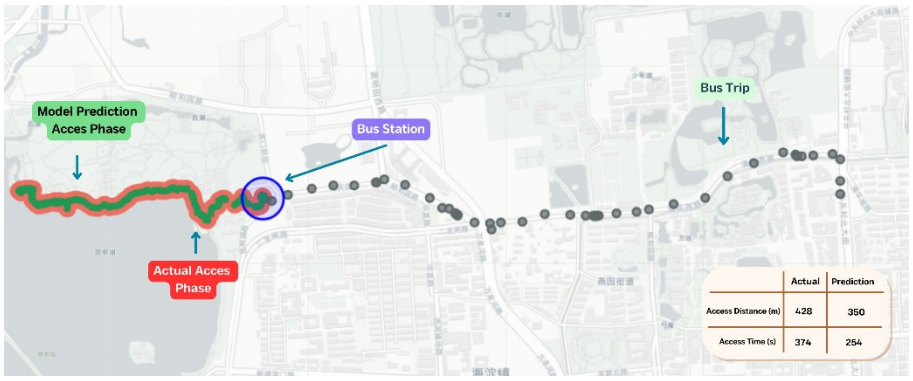

Figure 4.14 : Applying trip phase recognition algorithm on a bus-based trip, access phase

Figure 4.15 demonstrates the results of applying a trip phase recognition algorithm to the "Egress Phase" of a bus-based trip, similar to the previously analyzed access phase. The egress phase refers to the part of the journey where the user departs from the bus and continues on foot. It displays the actual egress path in red and the model's predicted path in green. While the two paths largely coincide, minor differences between the actual and predicted routes are visible, indicating small deviations in the model's accuracy.

The blue circle represents the bus station where the bus trip ends, marking the start of the egress phase. The actual egress distance is 392 meters, whereas the predicted distance is slightly shorter at 312 meters. Similarly, the actual egress time is 308 seconds, while the model predicts 294 seconds. This comparison helps to illustrate the performance of the trip phase recognition algorithm in identifying and predicting the user's behavior during the egress phase, providing insights into potential areas for further refinement in model accuracy.

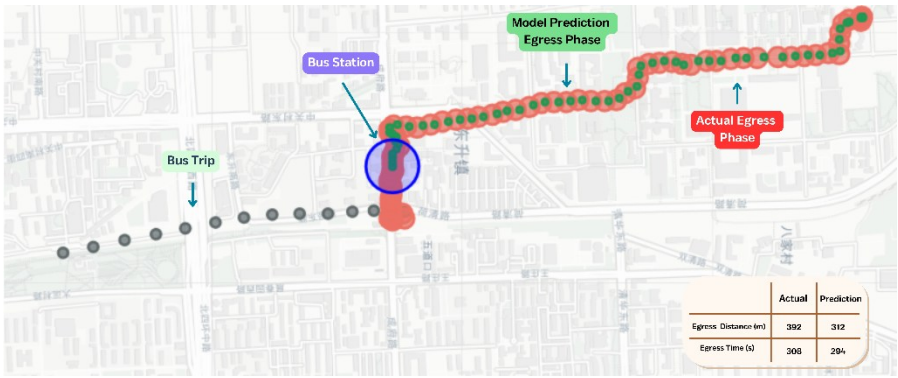

Figure 4.15 : Applying trip phase recognition algorithm on a bus-based trip, egress phase

In the evaluation of the trip phase recognition algorithm, the MAPE was calculated to assess the performance of the predicted distances and time intervals in comparison to the actual values. **Table 4.8** shows the MAPE for different phases of a trip. For the access phase, the MAPE for time is 0.1432, while for distance it is 0.1391, indicating relatively low error rates in predicting access time and distance. In the egress phase, the MAPE for time is slightly higher, with values of 0.1530 and 0.171, reflecting a modest increase in prediction errors compared to the access phase. This suggests that while both phases have manageable error levels, predicting egress times may present a slightly greater challenge.

Phase	MAPE
Access Time	0.1432
Access Distance	0.1391
Egress Time	0.1530
Egress Time	0.171

Table 4.8: MAPE of trip phase recognition framework

4.8.2 Trip Phase Recognition (Roma)

Several trips were recorded in Rome, providing valuable data for the analysis of urban mobility patterns. The raw data for each trip, including latitude, longitude, and timestamps, are processed through the framework. The output consists of the identified trip phases such as walking, cycling, and motorized transport and the waiting times at stations, along with the corresponding distances and durations. This approach allows for a detailed examination of both the spatial and temporal characteristics of urban travel in Rome, offering insights into the city's transportation efficiency and connectivity.

Figure 4.16 illustrates the comparison between actual trip data collection and the results of a DL model applied to analyze that trip. There is a series of GPS data points tracing the route taken during the trip. This raw data captures the movement of a GPS tracker, and provides a general overview of the path traveled, which appears as a sequence of points connected by a line representing the trip's progression across an urban environment.

In the lower section, a trip is broken down into distinct phases. The phases are categorized into Access Phase, Waiting Time, Motorized Phase, and Egress Phase. The Access Phase represents the initial walking segment where the traveler moves towards a mode of public transport. The waiting time indicates a period of stationary behavior, which is typically associated with waiting at a stop or station for the next available transport.

The motorized phase captures the segment where the person is traveling via a motorized vehicle, such as a bus or train, reflecting the core part of the trip. Finally, the egress phase represents the walking phase after leaving the motorized transport, completing the journey by traveling on foot to the final destination. The visualization underscores the application of advanced DL techniques to interpret trip data, transforming raw GPS points into meaningful and actionable insights about the nature of the journey, which could be useful for urban planning, transportation modeling, or optimizing public transport systems by better understanding traveler behavior.

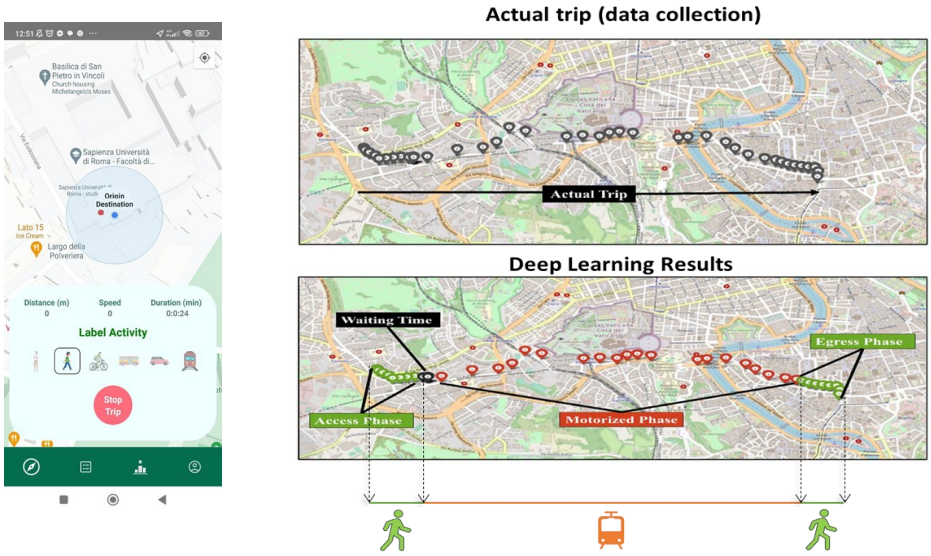

Figure 4.16 : Trip recording application and trip phase detection

4.9 Public Transit Stop Accessibility Evaluation

Evaluating the accessibility of public transit stops is essential for developing efficient and inclusive urban transportation systems. This process involves assessing how easily different population groups can reach transit stops, factoring in walking distance, safety, convenience, and the unique needs of individuals such as the elderly, disabled, and residents of underserved areas. Accessibility evaluation relies on spatial data analysis and an understanding of surrounding infrastructure and land use to pinpoint obstacles that might limit access. By addressing these barriers, cities can improve transit equity, fostering a more inclusive system that supports sustainability, reduces traffic congestion, and ensures access for all.

One of the main contributions of this thesis is the introduction of a novel Key Performance Indicator (KPI) framework to evaluate each public transport station. This evaluation is based on real-time data collected and processed from passengers using the specific station. By leveraging this real-time data, the proposed KPIs provide a more dynamic and accurate assessment of station performance, enabling more informed decision-making for transit authorities

There are 566 labeled public transport-based journeys that were captured from GeoLife dataset, with the condition to start with walking or cycling, continue with a bus or train, and then return to walking and cycling. To identify the locations of public transit stops where trips begin, each trip is plotted on a map, highlighting only the points where passengers have indicated the start of their bus or metro journey.

Figure 4.17 visualizes the entry points for public transport (PT) stations across an urban area, with black dots representing bus station entry points and red dots representing train station entry points. To accurately identify the location of each public transport station, all entry points (i.e., GPS points marking the transition from walking or cycling to bus or metro) for each trip were filtered and categorized. These entry points were grouped into the same category if their distances were less than 20 meters, ensuring that closely situated entry points were clustered together to represent the same station.

This process allows for a more precise mapping of public transport access points across different trips. Moreover, it shows various trips that all originated at the same public transport station. By visualizing the starting points of different trips, the diagram highlights how multiple travelers initiated their journeys from a common station.

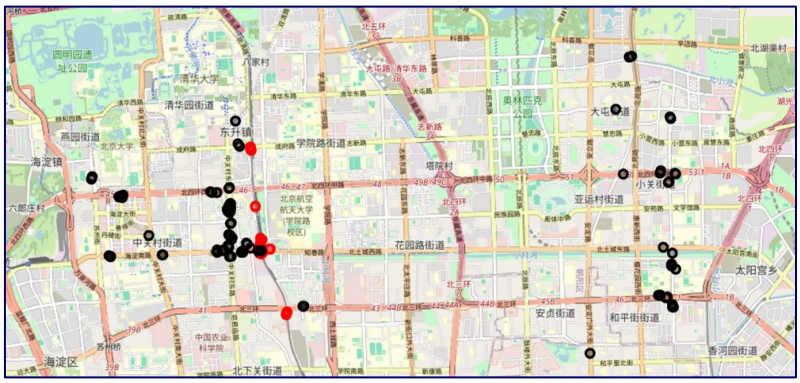

Figure 4.17 : Entry point of PT stations bus (black) and train (red)

Figure 4.18 illustrates the spatial distribution of user-labeled public transport trip starting points around a train station. The red points, each representing a starting location, are densely clustered within approximately 20 meters of the station.

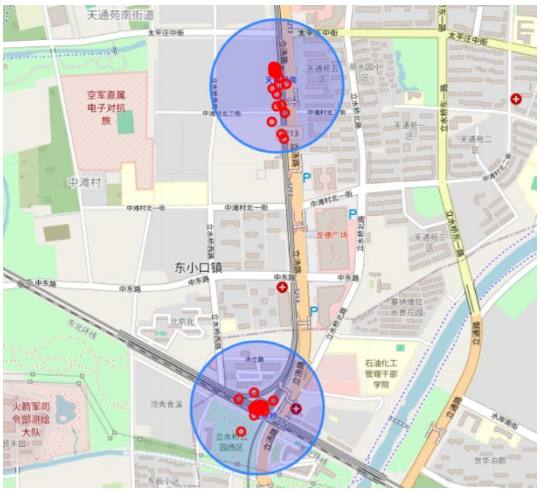

Figure 4.18 : Entry point of PT stations train

Figure 4.19 displays all trips attracted by a single train station. For each of these trips, access distance and time are computed using the trip phase recognition framework.

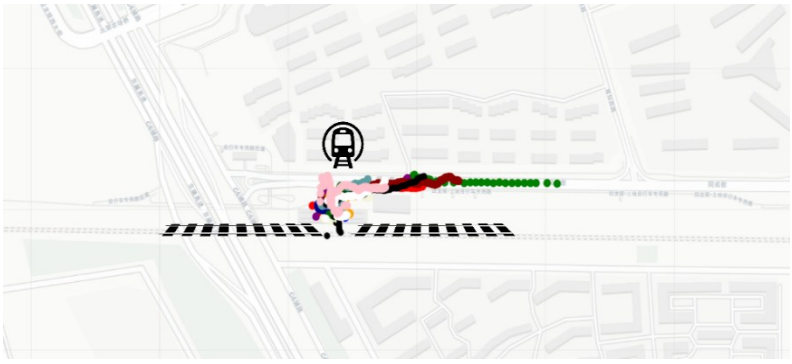

Figure 4.19: Attracted trips by a train station

4.9.1 Kernel Density Estimation

Kernel Density Estimation (KDE) [80] is a non-parametric way to estimate the probability density function (PDF) of a random variable. It is particularly valuable for estimating the underlying distribution of a dataset without assuming a specific parametric form, such as a normal or Poisson distribution. By smoothing the observed data points, KDE generates a continuous estimate of the probability density function. The core concept of KDE is to assign a kernel that is the smooth, bell-shaped function to each data point, and then sum these kernels to create a refined and smooth overall density estimate. Although the Gaussian kernel is commonly used for its desirable properties, alternative kernels such as the Epanechnikov or triangular kernels are also available, each offering different trade-offs between smoothness and computational efficiency, depending on the specific needs of the analysis.

The KDE for a given dataset (actual and predicted values) is defined by the following formula:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right) \tag{4.6}$$

Where $\hat{f}(x)$ is the estimated density at point x and n is the number of data points. Moreover, h is the bandwidth (a smoothing parameter that controls the width of the kernel) and K is the kernel function, which is usually a symmetric, non-negative function that integrates to one. A commonly used kernel is the Gaussian kernel, which is defined as:

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2} \tag{4.7}$$

$$u = \frac{x - x_i}{h} \tag{4.8}$$

In **Equation 4.8** formula, u represent a standardized variable and x is a reference point such as mean and x_i are the data points. The bandwidth h plays a crucial role in KDE. A smaller bandwidth results in a more sensitive estimate that may overfit the data (producing many spikes), while a larger bandwidth creates a smoother estimate but may underfit by over smoothing the data.

Figure 4.20 presents the kernel density estimation (KDE) for all trips attracted by the train station and it illustrates the comparison between actual and predicted access times and distance for public transport trips, with the blue curve representing the density distribution of actual access times and the red curve depicting the predicted access times.

The x-axis represents the access time in seconds, while the y-axis shows the density in terms of occurrences per second. As shown in **Figure 4.20**, the predicted access times (red) and actual access times (blue) follow similar distributions, though there are notable differences, particularly in the early stages where the model tends to overestimate the access time, as evidenced by the higher density of the red curve in the lower access time range.

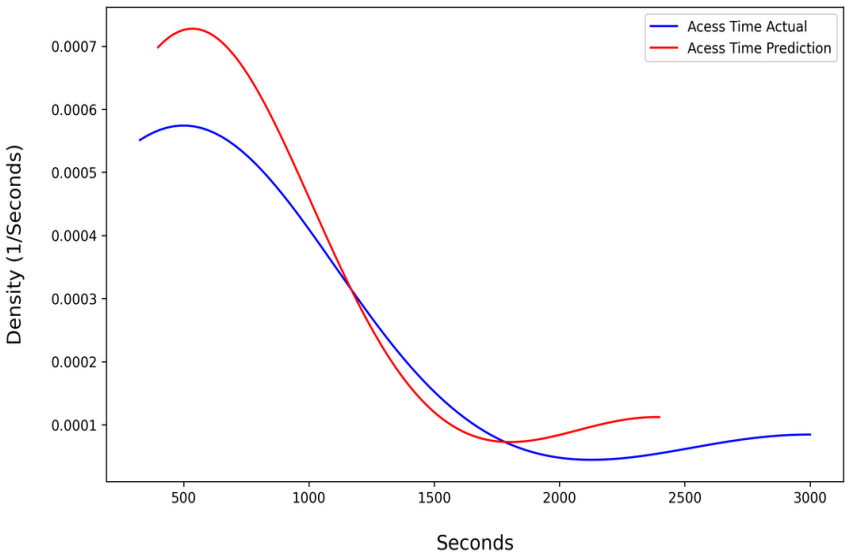

Figure 4.20 : Access Phase time from train station (China)

The KDE plot in **Figure 4.21** compares the actual and predicted access distances for public transport trips, with the blue curve representing the actual access distance distribution and the red curve showing the predicted access distances. In **Figure 4.21** , the x-axis represents the access distance in meters, while the y-axis shows the density in occurrences per meter. The comparison highlights that while the predicted distances (red curve) and the actual distances (blue curve) follow a similar pattern, there is a consistent overestimation by the model in the middle range of distances, as indicated by the higher density of the red curve compared to the blue. This disparity is most prominent between 600 and 1000 meters, where the prediction density peaks higher than the actual access distances.

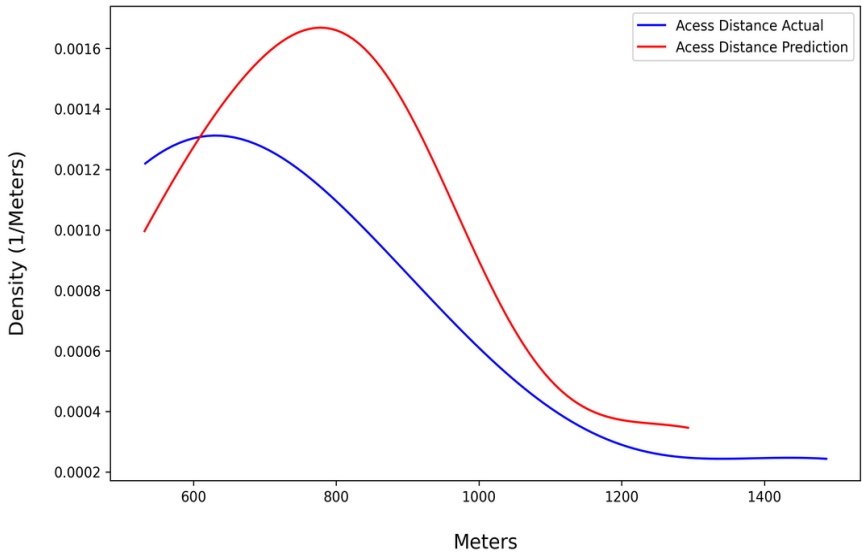

Figure 4.21 : Access Phase distance for train station

Figure 4.22 represents the trips attracted to a specific bus station, with multiple routes converging at a central point marked by the bus icon. The different colored lines correspond to distinct trips, showing the paths taken by passengers to access the bus station from various locations. The map highlights how individuals approach the station from different directions, revealing the spatial distribution of trips and the connectivity of the bus station within the surrounding area. By visualizing these trip paths, the figure provides insights into the accessibility of the bus station, indicating how effectively it serves passengers from nearby regions and the range of areas it attracts. This information can be valuable for analyzing passenger flow, optimizing station accessibility, and improving transit planning efforts to better serve the needs of the community.

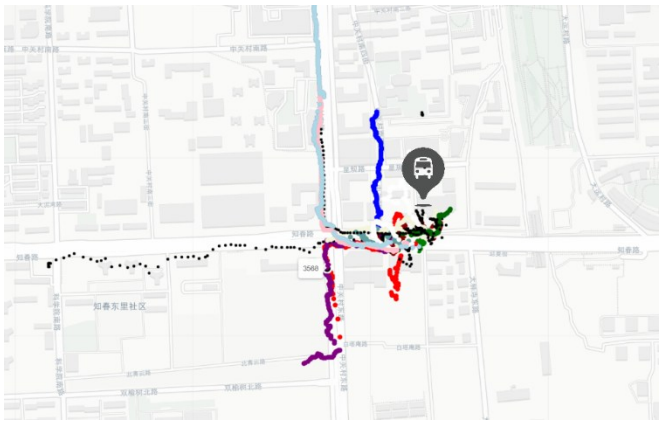

Figure 4.22 : Access Phase distance for a bus station

The two kernel density estimation plots (**Figures 4.23** and **4.24**) provide a comprehensive analysis of both access distance and access time for trips originating from a specific bus station, comparing the actual (blue curves) and predicted (red curves) data. In the **Figure 4.23**, which focuses on access distance, the x-axis represents the distance in meters, while the y-axis shows the density in occurrences per meter.

The predicted access distances (red curve) closely follow the actual distances (blue curve) and tend to overestimate shorter distances between 500 to 2000 meters, indicating that while the prediction model generally aligns well with real-world data, it slightly overpredicts trips with shorter distances.

In **Figure 4.24**, which illustrates access time, the x-axis measures the access time in seconds, and the y-axis represents the density in occurrences per second. Similarly, the predicted access times show a tendency to overestimate shorter trips (between 500 to 2000 seconds), as evidenced by the higher red curve in that range. Despite these overestimations, both figures reflect similar overall patterns, demonstrating that the prediction model is capturing the broad spatial and temporal characteristics of trips attracted to this bus station.

However, these slight overestimations in the shorter ranges suggest opportunities for further refinement of the model to enhance its predictive accuracy. Together, these visualizations provide valuable insights into how well the model predicts access behavior in terms of distance and time, highlighting its strengths in aligning with actual data and areas where it can be improved for more precise predictions.

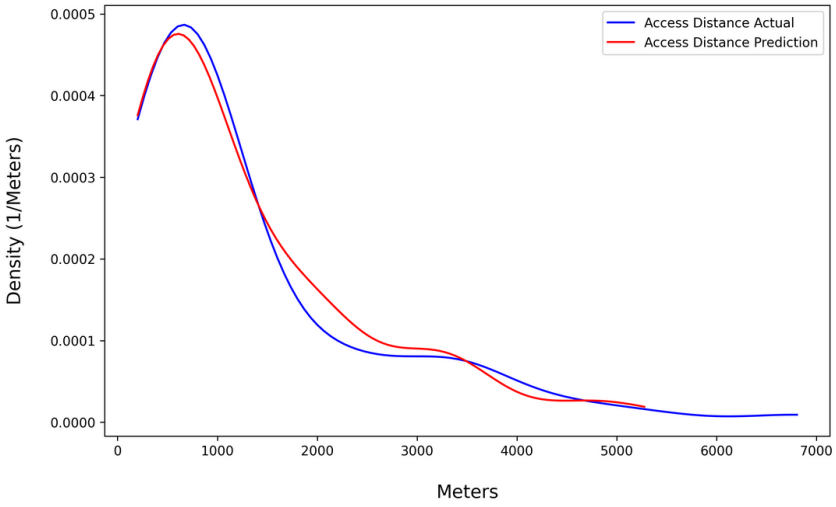

Figure 4.23 : Access Phase distance for a bus station

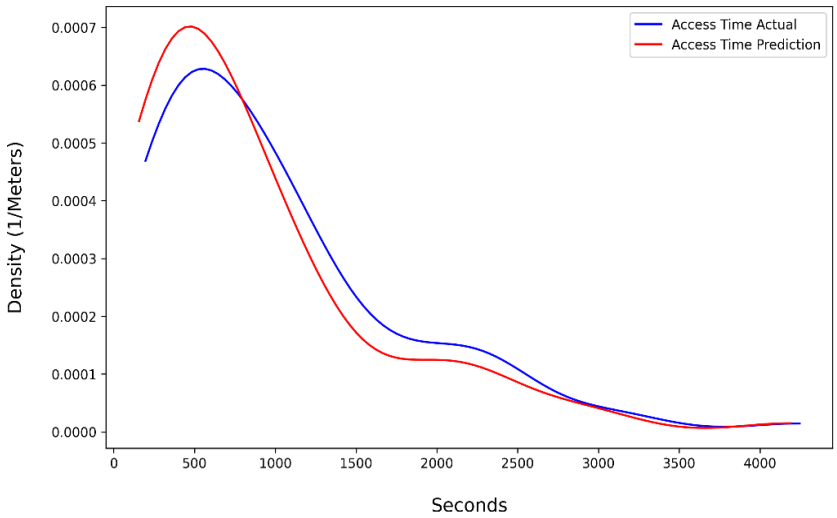

Figure 4.24 : Access Phase distance for a bus station

4.9.2 Public transit accessibility evaluation with bubble map and density color

A bubble map with color gradients representing density and circle radius corresponding to the average access distance for each public transport stop provides a clear visualization of accessibility levels across the network. The size of each circle reflects the average distance passengers travel to reach a specific station, offering a detailed spatial analysis of access patterns. This method helps to identify transit stations with lower accessibility, enabling targeted interventions. Such visualizations serve as powerful tools for urban planners and policymakers, facilitating improvements in public transport systems to make them more user-friendly, equitable, and sustainable.

4.9.2.1 Bus stations

In **Figure 4.25**, each point represents a single bus station, and the corresponding circles around these points indicate the access distance to that station. The access distance is the distance that passengers need to walk or cycle to reach a specific public transport stop. The numbers on the scale (473 to 2381 meters) represent the range of access distances. The color intensity of the circles around each bus station reflects how far passengers are traveling to reach that stop. Darker shades of blue indicate higher access distances (closer to 2381 meters), while lighter shades indicate shorter distances (closer to 473 meters).

This type of visualization is useful for identifying areas where public transport access can be more difficult, as stations with larger and darker circles suggest passengers travel farther. Conversely, stations with smaller, lighter circles may indicate well-positioned stops with shorter access distances. By evaluating these patterns, urban planners can assess the effectiveness of the current transit system and identify potential gaps in service or areas where more stations may be needed to improve access.

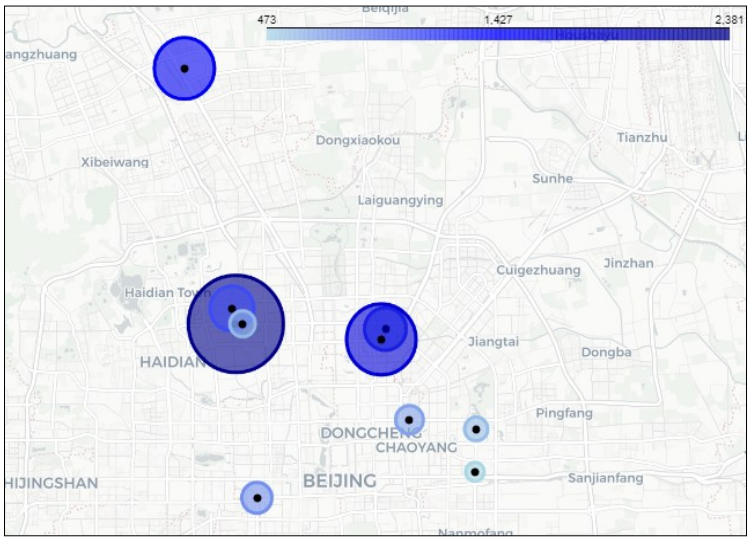

Figure 4.25: Bubble map visualization for bus stations

Figure 4.26 illustrates passengers' trips, with each dot representing a GPS point saved during their journey to a bus station. These GPS points track the actual paths passengers follow to access public transit stops. After calculating the access distance for each trip, the average of all trips attracted to each station is computed. This results in a circle representing the average access distance for

passengers who used that specific bus station. The circles provide a visual summary of the typical distance passengers travel to reach each station, aiding in the analysis of the accessibility of the public transport network.

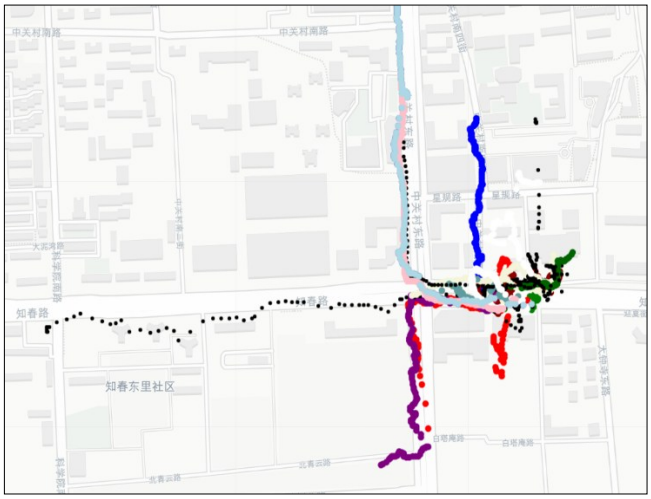

Figure 4.26: Access Phase distance for bus stations

Figure 4.27 illustrates the average access distance to various bus stations, with the size of each circle proportional to the calculated distance. Larger circles signify bus stations where passengers, on average, traveled greater distances to reach the stop, indicating lower accessibility. In certain neighborhoods, there is a significant overlap between the circles, suggesting a disparity in station accessibility. While one station may provide adequate access, another nearby station appears to attract passengers from a much larger area, indicating potential shortcomings in accessibility.

These overlapping zones reveal critical spatial inefficiencies within the public transport network, emphasizing the need for targeted interventions to enhance transit access. Identifying and addressing such disparities can inform more equitable and efficient public transit system designs, promoting better service coverage and improved user experience across the network.

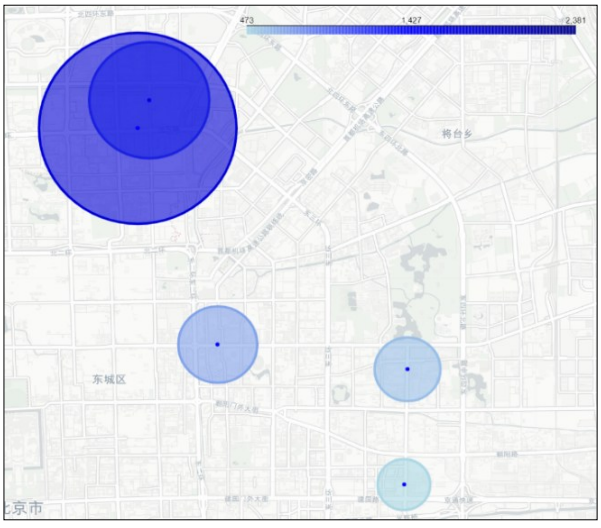

Figure 4.27: More detailed Bubble map visualization for bus stations

4.9.2.2 Train stations

The same methodology was applied to the trips attracted by train stations. In **Figure 4.28**, each point represents a train station, and the surrounding circles depict the access distances to those stations, with the size of each circle proportional to the calculated average distance. Similar to the earlier figures for bus stations, the access distance represents how far passengers must travel to reach the train station, with values ranging from 740 to 2009 meters.

The color gradient of the circles visually conveys these distances, where darker shades of green correspond to higher access distances (closer to 2009 meters), and lighter shades indicate shorter access distances (closer to 740 meters). This visualization provides a comprehensive understanding of accessibility patterns across the train network, highlighting stations with larger catchment areas and potential accessibility challenges.

This visualization helps in identifying areas where train stations are less accessible due to longer access distances, which could suggest the need for improved public transport infrastructure or additional stations to reduce the burden on passengers.

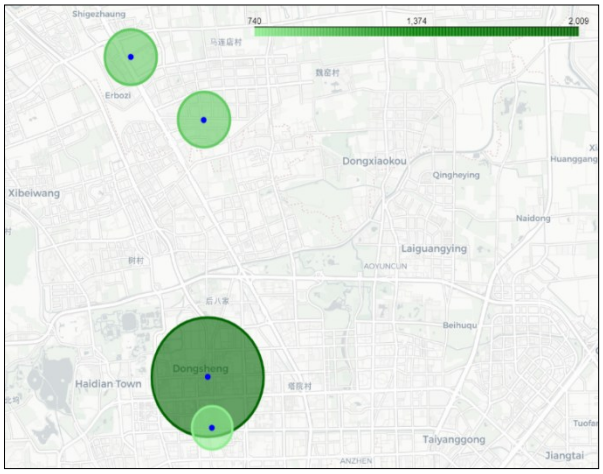

Figure 4.28 : Bubble map visualization for metro stations

Figure 4.28 represents the access distances for train stations. Each blue point marks the location of a train station, while the surrounding circles indicate the average access distance passengers traveled to reach those stations. The size of each circle is proportional to the calculated average distance, with a scale ranging from 740 meters to 2009 meters. The color intensity reflects the access distance: darker shades of green indicate longer distances (closer to 2009 meters), while lighter shades represent shorter distances (closer to 740 meters). This visualization highlights the variability in station accessibility, offering insights into areas where access to the train network may require improvement.

4.9.2.3 Waiting time at public transit stations

One of the capabilities of the framework is detecting passengers' waiting time before boarding a bus or metro. This is achieved by identifying standing segments in the GPS data, where a passenger remains stationary for a certain period. The framework then cross-references these standing segments with subsequent bus or metro trips. If the standing segment directly precedes a transit segment, it is classified as waiting time. This functionality provides valuable insights into passenger behavior and

transit efficiency, enabling a more detailed analysis of the overall journey experience and the potential for optimizing transit schedules and station amenities

Figure 4.29 illustrates the detection of waiting times at public transit stations, with waiting durations ranging from 60 to 284 seconds. Each point represents a transit station, and the color intensity of the surrounding circle indicates the waiting time experienced by passengers. Stations marked with green circles, such as those closer to 60 seconds, reflect shorter waiting times, implying more efficient service at these locations. In contrast, red circles, highlight stations where passengers face significantly longer waiting times, closer to 284 seconds. This visualization offers valuable insights into the operational efficiency of the public transport system, allowing for the identification of stations that may require improvements due to delays. Stations with longer waiting times may suffer from issues such as overcrowding, inefficient scheduling, or a lack of sufficient service frequency, which could be investigated further by urban planners and transport authorities. On the other hand, stations with shorter waiting times provide a benchmark for optimal transit service, offering lessons that can be applied to improve other parts of the network. By addressing these disparities, transit authorities can enhance the overall efficiency, reliability, and user experience of public transport services.

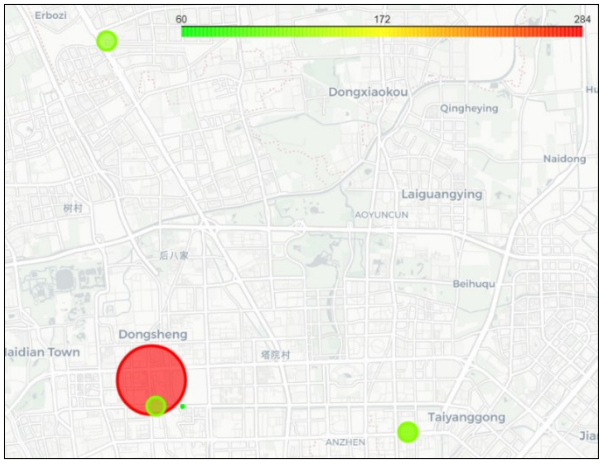

Figure 4.29 : Bubble map visualization for waiting time at public stations

In short, transport mode detection results demonstrate the robustness of the framework in classifying various transport modes, utilizing both RF and DL models. The DL model, which classified six modes (bike, bus, car, metro, standing, and walking), achieved high accuracy across all categories.

The RF model used features such as speed, acceleration, and total distance to classify transport modes. The model achieved strong performance metrics, with an overall accuracy of 83% and balanced precision and recall scores.

Following transport mode detection, trip phase recognition is another critical component of the framework. The trip phase recognition model focuses on identifying the distinct phases of a trip: access, egress, and waiting. This model leverages GPS data to segment trips into these phases, providing a more granular analysis of how passengers interact with public transport systems.

In one case, for a bus trip, the model predicted the access phase with a deviation of only 78 meters from the actual access distance. Similarly, the predicted access time was 120 seconds shorter than the actual time, indicating the model's efficiency in phase recognition.

A key innovation in the trip phase recognition process is the detection of waiting times before boarding buses or metro trains. By identifying standing segments and linking them to subsequent transit trips, the framework calculates waiting times more precisely than traditional methods. This feature provides valuable insights into public transit efficiency, allowing for an evaluation of how long passengers wait at specific transit stops.

These transport mode detection and trip phase recognition capabilities offer comprehensive tools for urban mobility analysis. By identifying the modes of transport used by passengers and accurately segmenting trip phases, the framework provides critical insights into both the behavior of passengers and the performance of public transit systems. This detailed analysis is particularly valuable for urban planners, as it allows them to evaluate transit system accessibility, identify areas for improvement, and make data-driven decisions to enhance public transport efficiency and user experience.

4.10 Future Work

Future research can expand on the present findings by integrating multi-sensor data to enhance classification accuracy, particularly in challenging urban environments where GPS data alone may be insufficient. Combining GPS with data from accelerometers, gyroscopes, and magnetometers can help capture detailed movement dynamics, providing more accurate transport mode differentiation in areas prone to signal interference or blockage, such as indoor spaces and high-density cityscapes. Additionally, the use of barometric pressure data could further improve accuracy by detecting elevation changes, which is especially relevant for distinguishing between different transport modes that involve vertical movement, such as subway or multi-level transit systems.

To increase the model's generalizability, future studies should also consider expanding the dataset to cover a broader range of geographic and demographic contexts. Collecting data from various urban areas with distinct transportation infrastructures, population densities, and behavioral patterns would allow the model to be applied more widely. Using data augmentation techniques to simulate different travel scenarios and conditions can further support this effort, enabling models to perform consistently across regions with varying urban designs and travel behaviors.

Developing lightweight versions of the model for deployment on mobile devices represents another promising avenue for future work. Techniques such as model pruning, quantization, and efficient architecture design could make transport mode detection and trip phase recognition feasible in real-time directly on users' devices. This adaptation would allow for scalable implementation, supporting applications like smart city monitoring and dynamic urban mobility management, where real-time data could improve responsiveness to transit demands and traffic conditions.

Finally, incorporating interactive feedback from users offers a pathway for model enhancement. By allowing users to confirm or modify detected transport modes and trip phases, the model can progressively improve its accuracy based on real-world data. This feedback-driven approach could not only refine model predictions but also foster user engagement, contributing to the development of adaptive systems that reflect changing urban mobility patterns and individual preferences. This

iterative improvement process will be instrumental in achieving a robust, user-centered system for transport mode detection and trip phase recognition across diverse urban settings.

Chapter 5

Conclusion

This research has made several noteworthy contributions to the field of transportation studies, particularly in the domain of transport mode detection and trip phase recognition using mobile sensor technologies such as GPS. The work presented in this thesis focused on addressing the growing challenges of urban mobility by offering automated solutions for detecting transportation modes and recognizing different phases of a trip, such as access, egress, and waiting, which are critical for improving urban transportation systems. The implementation of ML and DL models has demonstrated their effectiveness in delivering accurate results in these contexts.

The first significant contribution of this study lies in its comprehensive approach to transport mode detection, where various transportation modes such as walking, cycling, driving, and public transit were classified using advanced models. By leveraging large-scale datasets, such as the GeoLife and Sussex-Huawei Locomotion datasets, the thesis applied robust data preprocessing techniques, including filtering, segmentation, and feature extraction, to prepare the data for classification. The ML models, including RF and DL models such as Convolutional Neural Networks (CNNs), have proven to be highly effective in classifying transportation modes with high accuracy, even in complex urban environments where mode transitions are frequent.

One of the core achievements of this research is the development of a CNN-based framework capable of automatically learning and identifying transport modes from raw GPS data. Traditional methods often rely on manually engineered features, which are time-consuming and may not capture the full spectrum of data complexity. However, the CNN architecture used in this study bypasses the need for manual feature engineering by learning patterns directly from the data, making it highly adaptable to different urban environments and conditions. The CNN's ability to accurately differentiate between transport modes, such as walking, driving, and various forms of public transit (e.g., bus, metro, and train), highlights the superiority of DL methods in this domain.

Additionally, this research introduced a novel approach to trip phase recognition, which segments journeys into distinct stages such as access (the journey to a transit station), egress (the journey from the transit station to the destination), and waiting times at transit stops. This level of granularity in analyzing trips is crucial for optimizing public transport systems and enhancing the overall commuter experience. The framework developed in this thesis for trip phase recognition provided valuable insights into passenger behavior, identifying inefficiencies such as prolonged waiting times at transit stops and offering actionable data for improving transit services.

Moreover, by applying this trip phase recognition framework, the research introduced new Key Performance Indicators (KPIs) that can be used to assess the performance and accessibility of public transit stations. These KPIs, such as Average Access Time (AAT), Average Access Distance (AAD), and Average Waiting Time (AWT), are essential metrics that can help transit authorities evaluate the effectiveness of their services and identify areas where improvements are needed. The ability to quantify passenger experience using these KPIs provides a valuable tool for policymakers, urban planners, and transportation engineers, enabling them to make data-driven decisions that can improve the efficiency and accessibility of public transportation networks.

A major challenge addressed in this research was dealing with sparse and noisy GPS data, which is a common issue in transportation studies. The preprocessing techniques employed, including outlier removal, noise filtering, and smoothing algorithms, were instrumental in ensuring that the data used for model training was clean and reliable. These methods significantly improved the quality of the GPS data, making it possible for the ML and DL models to perform well even in challenging environments where GPS signals are weak or intermittent. The ability to handle such sparse data

makes the framework developed in this thesis highly robust and suitable for deployment in real-world scenarios where data quality may vary.

The implications of this research extend beyond academic contributions to practical applications in urban planning and transportation management. By providing a framework that can accurately detect transportation modes and recognize trip phases, this research offers urban planners and transportation authorities a powerful tool for understanding mobility patterns and improving public transit services. The findings of this study can help optimize route planning, reduce traffic congestion, and encourage the use of sustainable transportation modes, ultimately contributing to a more efficient and environmentally friendly urban transportation system.

Furthermore, this research emphasizes the role of mobile sensing technologies in modern urban transportation systems. As mobile phones become increasingly equipped with sophisticated sensors, including GPS, accelerometers, gyroscopes, and magnetometers, the potential for real-time transport mode detection and trip phase recognition becomes ever more feasible. This thesis has shown that mobile devices can be leveraged to capture large amounts of mobility data, which, when processed using ML and DL models, can provide highly accurate and actionable insights into urban transportation systems. This represents a significant step forward in the use of mobile technologies for smart city applications.

References

[1] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw gps data for geographic applications on the web," in *Proceedings of the 17th international conference on World Wide Web*, 2008, pp. 247–256.

[2] J. Q. James, "Travel mode identification with GPS trajectories using wavelet transform and deep learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1093–1103, 2020.

[3] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, and P. Havinga, "Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey," in *23th International conference on architecture of computing systems 2010*, VDE, 2010, pp. 1–10.

[4] https://developer.apple.com/library/ios/ documentation/CoreMotion/Reference/CMMotionActivity_class/ index.html#//apple_ref/occ/cl/CMMotionActivity, "CMMotionActivity," [Online; accessed 31-August-2018].

[5] https://developers.google.com/ android/reference/com/google/android/gms/location/ ActivityRecognition, "ActivityRecognition," Google, [Online; accessed 31-August-2018].

[6] J. de Dios OrtÃozar and L. G. Willumsen, *Modelling transport*. John Wiley & Sons, 2011.

[7] P. Stopher and A. Collins, "Conducting a GPS prompted recall survey over the internet," in *Transportation Research Board Annual Meeting, 84th, 2005, Washington, DC, USA*, 2005.

[8] L. Shen and P. R. Stopher, "Review of GPS travel survey and GPS data-processing methods," *Transp Rev*, vol. 34, no. 3, pp. 316–334, 2014.

[9] M. H. Nguyen, J. Armoogum, J.-L. Madre, and C. Garcia, "Reviewing trip purpose imputation in GPS-based travel surveys," *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 7, no. 4, pp. 395–412, 2020.

[10] P. McGowen and M. McNally, "Evaluating the potential to predict activity types from GPS and GIS data," in *Transportation Research Board 86th Annual Meeting*, Citeseer, 2007.

[11] P. R. Stopher and S. P. Greaves, "Household travel surveys: Where are we going?," *Transp Res Part A Policy Pract*, vol. 41, no. 5, pp. 367–381, 2007.

[12] C. D. Cottrill, Francisco Câmara Pereira, Fang Zhao, Inês Ferreira Dias, Hock Beng Lim, Moshe E. Ben-Akiva, and P. Christopher Zegras "Future mobility survey: Experience in developing a smartphone-based travel survey in Singapore," *Transp Res Rec*, vol. 2354, no. 1, pp. 59–67, 2013.

[13] Froehlich, J., Dillahunt, T., Klasnja, P., Mankoff, J., Consolvo, S., Harrison, B., & Landay, J. A, "UbiGreen: investigating a mobile tool for tracking and supporting green transportation habits," in *Proceedings of the sigchi conference on human factors in computing systems*, 2009, pp. 1043–1052.

[14] Wells, S., Kotkanen, H., Schlafli, M., Gabrielli, S., Masthoff, J., Jyllha, A., & Forbes, P, "Towards an applied gamification model for tracking, managing, & encouraging sustainable travel behaviours," *EAI Endorsed Transactions on Ambient Systems*, vol. 1, no. 4, p. e2, 2014.

[15] R. Maddison and C. Ni Mhurchu, “Global positioning system: a new opportunity in physical activity measurement,” *International journal of behavioral nutrition and physical activity*, vol. 6, pp. 1–8, 2009.

[16] K. Merry and P. Bettinger, “Smartphone GPS accuracy study in an urban environment,” *PLoS One*, vol. 14, no. 7, p. e0219890, 2019.

[17] S. Van der Spek, J. Van Schaick, P. De Bois, and R. De Haan, “Sensing human activity: GPS tracking,” *Sensors*, vol. 9, no. 4, pp. 3033–3055, 2009.

[18] Zappi, P., Lombriser, C., Stiefmeier, T., Farella, E., Roggen, D., Benini, L., & Tröster, G, “Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection,” in *Wireless Sensor Networks: 5th European Conference, EWSN 2008, Bologna, Italy, January 30-February 1, 2008. Proceedings*, Springer, 2008, pp. 17–33.

[19] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, “Activity recognition from accelerometer data,” in *Aaai*, Pittsburgh, PA, 2005, pp. 1541–1546.

[20] S. Patel, H. Park, P. Bonato, L. Chan, and M. Rodgers, “A review of wearable sensors and systems with application in rehabilitation,” *J Neuroeng Rehabil*, vol. 9, pp. 1–17, 2012.

[21] O. J. Woodman, “An introduction to inertial navigation,” University of Cambridge, Computer Laboratory, 2007.

[22] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, “Activity recognition using cell phone accelerometers,” *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[23] Y. Cai, Y. Zhao, X. Ding, and J. Fennelly, “Magnetometer basics for mobile phone applications,” *Electron. Prod.(Garden City, New York)*, vol. 54, no. 2, 2012.

[24] Shao, Wenhua, Fang Zhao, Cong Wang, Haiyong Luo, Tunio Muhammad Zahid, Qu Wang, and Dongmeng Li, “Location fingerprint extraction for magnetic field magnitude based indoor positioning,” *J Sens*, vol. 2016, no. 1, p. 1945695, 2016.

[25] G. Asci and M. A. Guvensan, “A novel input set for LSTM-based transport mode detection,” in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, IEEE, 2019, pp. 107–112.

[26] M. A. Guvensan, B. Dusun, B. Can, and H. I. Turkmen, “A novel segment-based approach for improving classification performance of transport mode detection,” *Sensors*, vol. 18, no. 1, p. 87, 2017.

[27] Y. Zheng, “GeoLife User Guide. 2012. url: https://www. microsoft. com/en-us/research/wpcontent/uploads/2016/02,” *User20Guide-1.2. pdf (visited on 01/21/2019)*.

[28] 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*, vol. 6, pp. 42592–42604, 2018.

[29] E. D. Kaplan and C. Hegarty, *Understanding GPS/GNSS: principles and applications*. Artech house, 2017.

[30] J. Jun, R. Guensler, and J. H. Ogle, “Smoothing methods to minimize impact of global positioning system random error on travel distance, speed, and acceleration profile estimates,” *Transp Res Rec*, vol. 1972, no. 1, pp. 141–150, 2006.

[31] N. Schuessler and K. W. Axhausen, “Processing raw data from global positioning systems without additional information,” *Transp Res Rec*, vol. 2105, no. 1, pp. 28–36, 2009.

[32] L. Bedogni, M. Di Felice, and L. Bononi, “By train or by car? Detecting the user’s motion type through smartphone sensors data,” in *2012 IFIP Wireless Days*, IEEE, 2012, pp. 1–6.

[33] M. A. Guvensan, B. Dusun, B. Can, and H. I. Turkmen, “A novel segment-based approach for improving classification performance of transport mode detection,” *Sensors*, vol. 18, no. 1, p. 87, 2017.

[34] X. Liang and G. Wang, “A convolutional neural network for transportation mode detection based on smartphone platform,” in *2017 IEEE 14th international conference on mobile Ad Hoc and sensor systems (MASS)*, IEEE, 2017, pp. 338–342.

[35] X. Han, J. Ye, J. Luo, and H. Zhou, “The effect of axis-wise triaxial acceleration data fusion in cnn-based human activity recognition,” *IEICE Trans Inf Syst*, vol. 103, no. 4, pp. 813–824, 2020.

[36] R. Tambi, P. Li, and J. Yang, “An efficient CNN model for transportation mode sensing,” in *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*, 2018, pp. 315–316.

[37] S. Hemminki, P. Nurmi, and S. Tarkoma, “Accelerometer-based transportation mode detection on smartphones,” in *Proceedings of the 11th ACM conference on embedded networked sensor systems*, 2013, pp. 1–14.

[38] C. Carpineti, V. Lomonaco, L. Bedogni, M. Di Felice, and L. Bononi, “Custom dual transportation mode detection by smartphone devices exploiting sensor diversity,” in *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, IEEE, 2018, pp. 367–372.

[39] Y. Qin, H. Luo, F. Zhao, C. Wang, J. Wang, and Y. Zhang, “Toward transportation mode recognition using deep convolutional and long short-term memory recurrent neural networks,” *IEEE Access*, vol. 7, pp. 142353–142367, 2019.

[40] Z. Wang, W. Yan, and T. Oates, “Time series classification from scratch with deep neural networks: A strong baseline,” in *2017 International joint conference on neural networks (IJCNN)*, IEEE, 2017, pp. 1578–1585.

[41] Fang, Shih-Hau, Hao-Hsiang Liao, Yu-Xiang Fei, Kai-Hsiang Chen, Jen-Wei Huang, Yu-Ding Lu, and Yu Tsao, “Transportation modes classification using sensors on smartphones,” *Sensors*, vol. 16, no. 8, p. 1324, 2016.

[42] S. H. Hosseini and G. Gentile, “Smartphone-based recognition of access trip phase to public transport stops via machine learning models,” *Transport and Telecommunication Journal*, vol. 23, no. 4, pp. 273–283, 2022.

[43] B. Friedrich, C. Lübbe, and A. Hein, “Combining LSTM and CNN for mode of transportation classification from smartphone sensors,” 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, 2020, pp. 305–310.

[44] A. Nawaz, H. Zhiqiu, W. Senzhang, Y. Hussain, I. Khan, and Z. Khan, "Convolutional LSTM based transportation mode learning from raw GPS trajectories," *IET Intelligent Transport Systems*, vol. 14, no. 6, pp. 570–577, 2020.

[45] T.-Y. Ma and S. Faye, "Multistep electric vehicle charging station occupancy prediction using hybrid LSTM neural networks," *Energy*, vol. 244, p. 123217, 2022.

[46] K. Sindhu Meena and S. Suriya, "A survey on supervised and unsupervised learning techniques," in *Proceedings of international conference on artificial intelligence, smart grid and smart city applications: AISGSC 2019*, Springer, 2020, pp. 627–644.

[47] L. Breiman, "Random forests," *Mach Learn*, vol. 45, pp. 5–32, 2001.

[48] S. Dabiri and K. Heaslip, "Inferring transportation modes from GPS trajectories using a convolutional neural network," *Transp Res Part C Emerg Technol*, vol. 86, pp. 360–371, 2018.

[49] G. Yanyun, Z. Fang, C. Shaomeng, and L. Haiyong, "A convolutional neural networks based transportation mode identification algorithm," in *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2017, pp. 1–7.

[50] J. Iskanderov and M. A. Guvensan, "Breaking the limits of transportation mode detection: Applying deep learning approach with knowledge-based features," *IEEE Sens J*, vol. 20, no. 21, pp. 12871–12884, 2020.

[51] Alam, Md Golam Rabiul, Mahmudul Haque, Md Rafiul Hassan, Shamsul Huda, Mohammad Mehedi Hassan, Fred L. Strickland, and Salman A. AlQahtani, "Feature cloning and feature Fusion Based Transportation Mode Detection using convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 4671–4681, 2023.

[52] P. Wang and Y. Jiang, "Transportation mode detection using temporal convolutional networks based on sensors integrated into smartphones," *Sensors*, vol. 22, no. 17, p. 6712, 2022.

[53] A. El-Geneidy, M. Grimsrud, R. Wasfi, P. Tétreault, and J. Surprenant-Legault, "New evidence on walking distances to transit stops: Identifying redundancies and gaps using variable service areas," *Transportation (Amst)*, vol. 41, pp. 193–210, 2014.

[54] J. He, R. Zhang, X. Huang, and G. Xi, "Walking access distance of metro passengers and relationship with demographic characteristics: A case study of Nanjing metro," *Chin Geogr Sci*, vol. 28, pp. 612–623, 2018.

[55] H. Kim, "Walking distance, route choice, and activities while walking: A record of following pedestrians from transit stations in the San Francisco Bay area," *Urban Design International*, vol. 20, pp. 144–157, 2015.

[56] A. Tennøy, M. Knapskog, and F. Wolday, "Walking distances to public transport in smaller and larger Norwegian cities," *Transp Res D Transp Environ*, vol. 103, p. 103169, 2022.

[57] T. Zuo, H. Wei, and A. Rohne, "Determining transit service coverage by non-motorized accessibility to transit: Case study of applying GPS data in Cincinnati metropolitan area," *J Transp Geogr*, vol. 67, pp. 1–11, 2018.

[58] R. Yang, Y. Liu, Y. Liu, H. Liu, and W. Gan, “Comprehensive public transport service accessibility index—a new approach based on degree centrality and gravity model. Sustainability 11: 5634,” 2019.

[59] C. Voss, M. Winters, A. Frazer, and H. McKay, “School-travel by public transit: Rethinking active transportation,” *Prev Med Rep*, vol. 2, pp. 65–70, 2015.

[60] T. Tao, J. Wang, and X. Cao, “Exploring the non-linear associations between spatial attributes and walking distance to transit,” *J Transp Geogr*, vol. 82, p. 102560, 2020.

[61] Y. Jiang, P. C. Zegras, and S. Mehndiratta, “Walk the line: station context, corridor type and bus rapid transit walk access in Jinan, China,” *J Transp Geogr*, vol. 20, no. 1, pp. 1–14, 2012.

[62] M. Oliver, H. Badland, S. Mavoa, M. J. Duncan, and S. Duncan, “Combining GPS, GIS, and accelerometry: methodological issues in the assessment of location and intensity of travel behaviors,” *J Phys Act Health*, vol. 7, no. 1, pp. 102–108, 2010.

[63] Y. Fan, A. Guthrie, and D. Levinson, “Waiting time perceptions at transit stops and stations: Effects of basic amenities, gender, and security,” *Transp Res Part A Policy Pract*, vol. 88, pp. 251–264, 2016.

[64] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, “Understanding mobility based on GPS data,” in *Proceedings of the 10th international conference on Ubiquitous computing*, 2008, pp. 312–321.

[65] T. Vincenty, “Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations,” *Survey review*, vol. 23, no. 176, pp. 88–93, 1975.

[66] R. W. Schafer, “What is a savitzky-golay filter?[lecture notes],” *IEEE Signal Process Mag*, vol. 28, no. 4, pp. 111–117, 2011.

[67] Z. Xiao, Y. Wang, K. Fu, and F. Wu, “Identifying different transportation modes from trajectory data using tree-based ensemble classifiers,” *ISPRS Int J Geoinf*, vol. 6, no. 2, p. 57, 2017.

[68] Z. Lu, Z. Long, J. Xia, and C. An, “A random forest model for travel mode identification based on mobile phone signaling data,” *Sustainability*, vol. 11, no. 21, p. 5950, 2019.

[69] T. K. Ho, “Random decision forests,” in *Proceedings of 3rd international conference on document analysis and recognition*, IEEE, 1995, pp. 278–282.

[70] LeCun, Yann, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel, “Handwritten digit recognition with a back-propagation network,” *Adv Neural Inf Process Syst*, vol. 2, 1989.

[71] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Adv Neural Inf Process Syst*, vol. 25, 2012.

[72] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, “Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition,” in *2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP)*, IEEE, 2012, pp. 4277–4280.

[73] A. Rakhlin, “Convolutional neural networks for sentence classification. GitHub. 2016,” 2021.

[74] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[75] G. Lin and W. Shen, “Research on convolutional neural network based on improved Relu piecewise activation function,” *Procedia Comput Sci*, vol. 131, pp. 977–984, 2018.

[76] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International conference on machine learning*, pmlr, 2015, pp. 448–456.

[77] D. Scherer, A. Müller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” in *International conference on artificial neural networks*, Springer, 2010, pp. 92–101.

[78] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[79] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[80] Y.-C. Chen, “A tutorial on kernel density estimation and recent advances,” *Biostat Epidemiol*, vol. 1, no. 1, pp. 161–187, 2017.