FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Transportation Mode Detection from crowdsourced GPS Trajectory Datasets

Akilu Rilwan Muhammad

MAPi: Doctoral Programme in Computer Science
of the Universities of Minho, Aveiro and Porto

Supervisor: Ana Aguiar

Co-Supervisor: João Mendes-Moreira

January 20, 2026

Transportation Mode Detection from crowdsourced GPS Trajectory Datasets

Akilu Rilwan Muhammad

MAPi: Doctoral Programme in Computer Science
of the Universities of Minho, Aveiro and Porto

January 20, 2026

Dedicated to my parents, Alhaji Rilwan Muhammad and Zulai Rilwan,
whose unwavering support and boundless love have illuminated my path.
To my dear wife, Maryam Umar Saleh, your enduring patience and
unwavering encouragement have been my steady companions.
And finally to my beloved children, Abdurrahman, Abubakar, Zulaiha,
and Umar, your distant yet abiding presence has fuelled my determination.

Resumo

O rápido ritmo de urbanização e o contínuo crescimento da população global estão a exercer uma pressão significativa sobre os sistemas de transporte, exigindo o desenvolvimento de soluções inteligentes e rentáveis para enfrentar os desafios emergentes. Uma das vias promissoras para melhorar a eficiência do transporte reside na utilização eficaz das vastas quantidades de dados espaço-temporais gerados de forma colaborativa por dispositivos móveis, como smartphones, transportados pelos indivíduos. Além disso, os avanços nas ferramentas de mineração de dados e nas técnicas de aprendizagem automática têm-se revelado fundamentais para analisar esses dados, permitindo a extração de informações valiosas sobre os padrões de mobilidade.

Os dados de trajetórias GPS constituem um recurso crucial para investigar os padrões de mobilidade humana e avaliar a sua influência nos sistemas de transporte. No entanto, um desafio significativo reside na determinação precisa do modo de transporte para cada trajeto, uma vez que as trajetórias GPS não contêm informações explícitas sobre o modo de viagem. Assim, esta tese tem como objetivo abordar esta limitação através de uma abordagem de aprendizagem supervisionada, englobando a análise de características e técnicas-chave, e propondo, em última instância, abordagens inovadoras para identificar modos de transporte a partir de conjuntos de dados de trajetórias GPS. A realização bem-sucedida deste objetivo exige uma compreensão aprofundada e um pré-processamento minucioso dos dados brutos de trajetórias.

Para este fim, esta tese investiga a eficácia de várias técnicas de engenharia de características para extrair informações relevantes de trajetórias GPS. Além disso, propõe um novo método de seleção de características de subespaço de classe que identifica as características mais discriminativas para cada modo de transporte, aumentando assim a precisão e robustez da detecção de modo. Além disso, é proposto um framework de classificação hierárquica, aproveitando os erros de classificação de um classificador plano padrão para melhorar o desempenho geral da classificação. A tese também analisa o impacto do desequilíbrio de classes no desempenho dos classificadores de detecção de modo de transporte (TMD) e explora estratégias para mitigar seus efeitos.

Finalmente, o desempenho dos métodos e modelos propostos é avaliado em conjuntos de dados de trajetória GPS do mundo real. Os resultados experimentais demonstram a superioridade do framework proposto para a previsão correta do modo de transporte.

Abstract

The rapid pace of urbanisation and the continued growth of the global population are exerting significant pressure on transportation systems, demanding the development of intelligent and cost-effective solutions to address emerging challenges. One promising avenue for enhancing transportation efficiency lies in the effective utilisation of the vast quantities of crowdsourced spatiotemporal data generated by mobile devices, such as smartphones, carried by individuals. Furthermore, advancements in data mining tools and machine learning techniques are proving instrumental in analysing this data, enabling the extraction of valuable insights into mobility patterns.

GPS trajectory data constitutes a crucial resource for investigating human mobility patterns and evaluating their influence on transportation systems. A significant challenge, however, lies in accurately determining the mode of transport for each journey, as GPS trajectories lack explicit travel mode information. Accordingly, this thesis aims to address this limitation through a supervised learning approach, encompassing the analysis of key features and techniques, and ultimately proposing novel approaches for identifying transportation modes from GPS trajectory datasets. The successful realisation of this objective necessitates a comprehensive understanding and meticulous pre-processing of the raw trajectory data.

To this end, this thesis investigates the effectiveness of various feature engineering techniques for extracting relevant information from GPS trajectories. Furthermore, it proposes a novel class-subspace feature selection method that identifies the most discriminative features for each transportation mode, thereby enhancing the accuracy and robustness of mode detection. Additionally, a hierarchical classification framework is proposed, leveraging the misclassification errors of a standard flat classifier to improve overall classification performance. The thesis also analyses the impact of class imbalance on the performance of transportation mode detection (TMD) classifiers and explores strategies to mitigate its effects.

Finally, the performance of the proposed methods and models are evaluated on real-world GPS trajectory datasets. The experimental results demonstrate the superiority of the proposed framework towards correct prediction of transportation mode.

Acknowledgments

All praise and thanks are due to Almighty God, whose guidance and blessings have made the completion of this thesis possible. Without His infinite mercy, none of this would have been achieved.

I would like to express my deepest gratitude to my parents. Their unwavering love, support, and sacrifices have been instrumental in my success. Their continuous encouragement and unwavering belief in me have been an invaluable source of strength.

To my family, thank you for your endless support, understanding, and patience. Although geographically distant during my studies, you remained my greatest source of strength. Your love, prayers, and unwavering support from afar have been a constant source of motivation.

I am deeply indebted to my supervisors, Professor Ana Aguiar and Professor João Mendes-Moreira, for their invaluable guidance, patience, and expertise. Their insightful feedback, constructive criticism, and unwavering support have been instrumental in shaping this research. I am truly grateful for their dedication and the opportunities they have provided me.

I would also like to express my sincere thanks to all members of the IT-Lab at FEUP. Their collaborative spirit, helpful discussions, and friendly camaraderie fostered a stimulating and supportive research environment.

Finally, I express my sincere appreciation to *Instituto de Telecomunicações (IT)* for providing me with excellent research facilities, a supportive environment, and the resources necessary for the successful completion of my work.

This research would not have been feasible without the crucial support provided by the following projects: S2MovingCity (CMUP-ERI/TIC/0010/2014), Safe Cities (POCI-01-0247-FEDER-041435), and Route 25 (C645463824-00000063).

Akilu Rilwan Muhammad

"It is not befitting for anyone with knowledge
to give up learning."

- Malik ibn Anas
Jãmi' Bayãn al-'Ilm, 423

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
AUC	Area Under the Curve
BN	Bayesian Network
BSC	Base Station Controller
BST	Base Transceiver Station
CDR	Call Detail Records
CNN	Convolutional Neural Networks
DNN	Deep Neural Networks
DT	Decision Trees
EAE	Ensemble of Autoencoders
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GIS	Geographic Information System
GPS	Global Positioning System
GPU	Graphical Processing Unit
HAR	Human Activity Recognition
HMM	Hidden Markov Model
IMU	Inertial Measurement Unit
k-NN	k-Nearest Neighbours
LA	Location Area
LR	Logistic Regression
LSTM	Long Short-Term Memory
ML	Machine Learning
MNOs	Mobile Network Operators
PNT	Positioning, Navigation, and Timing
RF	Random Forest
ROC	Receiver Operating Characteristics
SFFS	Sequential Forward Floating Selection
SHAP	SHapley Additive exPlanations
SMF2016	SenseMyFEUP 2016 Dataset
SVC	Support Vector Classifier
SVM	Support Vector Machines
TMD	Transportation Mode Detection
TN	True Negative
TP	True Positive
TPR	True Positive Rate
XGBoost	Extreme Gradient Boosting

Chapter 1

Introduction

This chapter establishes the context and motivation for the research presented in this thesis. We outline the key scientific contributions of this work and provide an overview of the document's structure.

1.1 Research Context and Motivation

Rapid urbanisation and population growth have led to increasing global challenges in urban transportation systems. Population growth remains a primary global concern. By 2050, the world population is estimated to reach around 9.7 billion people, with nearly 70% projected to live in urban areas [132]. Given this alarming figure, understanding urban-scale people's transportation behaviour is crucial for urban planning. This understanding is essential for constructively interpreting citizens' travel behaviour and assessing transportation-related policies and actions. Knowledge of people's travel behaviour, including mode choice and travel patterns, is essential for informed decision-making in urban planning, transportation policy, and infrastructure investment. By analysing travel behaviour, policymakers can identify bottlenecks, optimise transportation networks, and promote sustainable mobility options. Moreover, understanding travel behaviour can help to predict future transportation demands, which can inform long-term planning and investment decisions. Additionally, analysing travel behaviour patterns can help to identify emerging trends and challenges, such as the increasing use of ride-sharing services and the growing popularity of electric vehicles.

A land transportation network encompasses a variety of infrastructure and modes of transport that facilitate the movement of people from one location to another. Transportation mode refers to the specific means of transport individuals use, such as cars, buses, trains, bicycles, or walking. Understanding the distribution of transportation modes is crucial for effective urban planning, traffic management, and sustainable transportation policies. By analysing these modal distributions, researchers and policymakers can gain valuable insights into travel behaviour patterns, identify trends, and inform evidence-based decision-making. For instance, understanding the factors influencing the choice of transportation mode can help to promote sustainable transportation options,

such as public transit and cycling, and reduce reliance on private vehicles. Additionally, analysing transportation modes' distribution can help identify areas with high levels of traffic congestion and inform strategies to alleviate congestion, such as investing in public transportation infrastructure or implementing traffic management measures.

Despite the advantages of mobility and accessibility offered by transportation networks, urban areas continue to face significant challenges in urban planning and transportation management. Traditional methods for collecting transportation data, such as household surveys and interviews, are often time-consuming, costly, and prone to low response rates and incomplete information [31]. Moreover, these methods usually rely on self-reported data, which can be subject to recall bias. To address these limitations and gain a more accurate understanding of urban mobility patterns, there is a pressing need for efficient and reliable methods for collecting and analysing large-scale transportation data [17].

Fortunately, the proliferation of smartphones and mobile devices equipped with various sensors has opened up new opportunities for collecting large-scale, fine-grained mobility data through crowdsensing (or participatory sensing) [106, 107, 114]. By leveraging the collective sensing capabilities of mobile devices, researchers and policymakers can gather real-time, location-based information on transportation, traffic congestion, and other relevant factors. This data can include the Global Positioning System (GPS) trajectories, accelerometer data, Wi-Fi logs, and Bluetooth signals, providing a rich source of information for analysing urban mobility patterns. However, while crowdsensing offers valuable data, it does not directly provide information about the mode of transportation used by individuals.

Advanced data mining and machine learning techniques are required to extract transportation mode information from GPS trajectories. However, accurate and reliable transportation mode detection remains challenging due to noisy and incomplete data, diverse urban environments, and privacy concerns. Extracting meaningful features from GPS trajectories and developing robust machine-learning models to classify transportation modes are crucial steps toward understanding urban mobility patterns and informing transportation policies.

This thesis addresses these challenges by developing robust machine learning models to accurately classify transportation modes from real-world GPS trajectory data. This research will employ supervised machine learning techniques to classify transportation modes directly from GPS trajectory data without relying on external data sources.

1.2 Thesis Statement

Given the challenges of accurate transportation mode detection (TMD) using real-world mobile crowdsourced GPS trajectory data, what are the most effective approaches for improving the accuracy and robustness of TMD models, considering factors such as class imbalance, feature engineering, and classification methodologies?

1.3 Research Objectives

This thesis aims to investigate the effectiveness of machine learning techniques in accurately classifying transportation modes (walking, cycling, car, bus, train) from GPS trajectory data, thereby contributing to a better understanding of urban mobility patterns. Accordingly, this study aims to achieve the following objectives:

- Develop and evaluate feature selection and classification techniques to improve the accuracy of transportation mode detection, particularly for challenging classes with overlapping characteristics. This objective is detailed in Chapter 5.
- To investigate the effectiveness of hierarchical classification approaches in developing TMD models, analysing the impact of different hierarchical structures, and exploring methods for optimising hierarchical classifiers for TMD. This objective is addressed in detail in Chapter 6.
- To comprehensively analyse the impact of class imbalance on TMD performance, investigating the interplay of class imbalance with other factors. This is described in detail in Chapter 7.

1.4 Research Contributions

This research makes significant contributions to the field of Transportation Mode Detection (TMD), specifically addressing the challenges of accurately identifying modes of transportation from noisy and incomplete GPS data. The key contributions of this thesis are:

- We proposed a novel class-subspace feature selection method for TMD. This method uniquely captures the most relevant features for individual transportation modes by identifying discriminative subspaces within the feature space. By focusing on these informative subspaces, the proposed method improves the accuracy of mode detection, especially in challenging scenarios with overlapping class distributions (Chapter 5, publication [89]).
- We proposed *HiClass4MD* hierarchical classifier for TMD. *HiClass4MD* leverages the misclassification error of the standard flat classifier to learn the class between different transportation modes. The detailed algorithm is presented in Chapter 6 (publication [92]).
- Investigated the impact of class imbalance and overlap on TMD classifiers. This work is discussed in Chapter 7. Part of this work is published in [93].
- A comprehensive evaluation of the proposed methods on two real-world GPS trajectory datasets and comparison with existing state-of-the-art techniques.

1.5 List of Publications

The research presented in this thesis has led to the following publications.

Conference proceedings

1. Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. Transportation mode detection from GPS data: A data science benchmark study. In 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pages 3726–3731. IEEE, 2021.

2. Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. Inferring transportation mode using pooled features from time and frequency domains. In 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), pages 3985–3990. IEEE, 2023.

3. Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. *HiClass4MD*: A hierarchical classifier for transportation mode detection. In 2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC), pages –. IEEE, 2024.

4. Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. "Characterising class imbalance in transportation mode detection: An experimental study," in Intelligent Data Engineering and Automated Learning – IDEAL 2024, V. Julian, et al., Eds., vol. 15347, Lecture Notes in Computer Science. Cham, Switzerland: Springer, 2025, pp. 123-130. doi: 10.1007/978-3-031-77738-7_6.

Journal (under review)

1. A. R. Muhammad, A. Aguiar, and J. Mendes-Moreira, "From attribution to selection: harnessing SHAP values for class-subspace feature selection in transportation mode detection" [submitted to the *International Journal of Data Science and Analytics*].

1.6 Thesis Outline

The rest of this thesis is structured as follows.

Chapter 2

This chapter comprehensively reviews state-of-the-art techniques in TMD. We discuss various data types used in TMD studies, their advantages and limitations, and the methodologies employed to identify modes of transportation from GPS trajectory data. Additionally, we critically analyse existing approaches, highlighting their strengths, weaknesses, and limitations.

Chapter 3

This chapter characterises the datasets used throughout this thesis and details the preprocessing techniques employed. We also present a benchmark study to evaluate the performance of different

TMD algorithms on the dataset.

Chapter 5

This chapter presents a novel class-subspace feature selection method to enhance the accuracy of TMD classifiers. The method uses game-theoretic principles to identify discriminative subspaces within the feature space, pinpointing the most relevant features for each transportation mode. This targeted approach addresses challenges such as overlapping class distributions and limited discriminative features in minority classes. Focusing on these informative subspaces improves the classification performance, particularly in complex scenarios where traditional methods struggle.

Chapter 6

This chapter introduces the *HiClass4MD*, a hierarchical classification framework designed to improve the predictive models of TMD systems. This method builds upon the misclassification errors of standard flat classifiers to construct a hierarchical structure that effectively discriminates between transportation modes. By strategically learning and refining the class hierarchy, *HiClass4MD* enhances classification performance, particularly in scenarios with overlapping class distribution and imbalance. The chapter explains the algorithm, its design rationale, and its application to real-world GPS trajectory datasets. Comprehensive evaluations demonstrate the framework's ability to improve performance overall.

Chapter 7

Chapter 7 investigates the impact of class imbalance and overlapping feature distributions on the performance of TMD classifiers. This chapter delves into how these challenges affect classification accuracy and explores strategies to mitigate their effects. By analysing real-world GPS trajectory datasets, the chapter provides insights into the behaviour of classifiers under these conditions and evaluates techniques to improve their robustness.

Chapter 8

Chapter 8 concludes the thesis by summarizing the key findings and contributions of the research. It also discusses potential future research directions based on this thesis's work.

Chapter 2

State of the Art

This chapter presents a broad overview of transportation mode detection (TMD).

2.1 Data Sources

2.1.1 Mobile Phone Network Data

The base transceiver stations (BTSs), called cell towers, constitute a fundamental component of cellular networks. Equipped with multiple antennas, these stations facilitate wireless communication between the network and mobile phones. A single BTS typically covers a limited geographical area, known as a cell. A group of BTSs within a specific location area (LA) is managed by a base station controller (BSC). The BSC oversees various network operations, including call setup and handover procedures, ensuring seamless communication as mobile devices move between cells [64].

Mobile Switching Centres (MSCs) are crucial in managing and coordinating multiple BSCs. MSCs establish and terminate end-to-end connections between mobile devices and facilitate communication with external networks. MSCs handle the complexities of mobility management and handover procedures, ensuring uninterrupted service during device usage. An integral component of the MSC is the visitor location register (VLR), a database that maintains real-time records of the locations of all mobile subscribers within its coverage area. This is illustrated in Figure. 2.1).

Mobile network operators track device locations for various operational purposes, estimating device positions based on the nearest BTS connection [13]. While location precision varies with tower density, this approach enables large-scale, passive data collection without user intervention. Mobile phone network data can be broadly categorized into two types: event-driven and network-driven [64]. *Event-driven data* is generated by user activity, such as phone calls and SMS messages, e.g., call detail records (CDRs) [36]. *Network-driven data* is collected regardless of user activity and is triggered by events like location updates or handovers [26, 99]. Network operators utilize network-driven data to ensure optimal service quality for their subscribers.

Passively collected mobile phone network data offers a distinct advantage. Users do not require active participation, potentially enabling the study of even hard-to-reach populations. However,

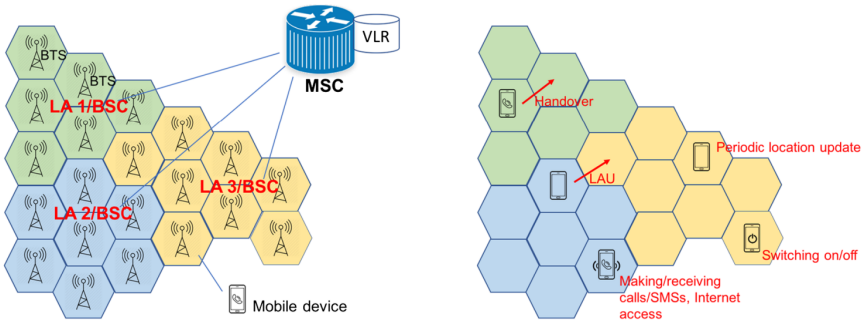


Figure 2.1: An illustration of the mobile network infrastructure and network event handover [64].

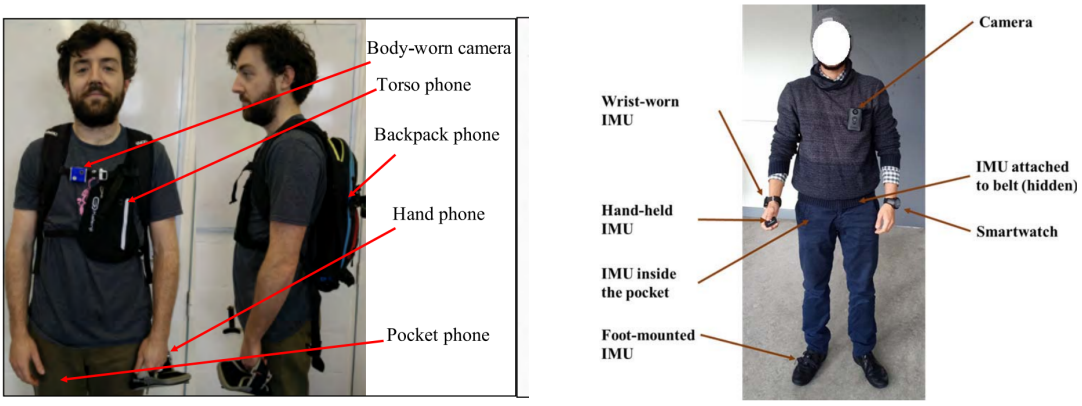
mobile phone network data is not easily accessible. Due to privacy concerns and potential business competition, mobile network operators (MNOs) often hesitate to share data with researchers. Furthermore, mobile phone network data is characterised by poor spatio-temporal granularity. The temporal granularity of mobile data depends on the type used. Network-driven data is often denser; it is collected irrespective of phone usage but not recorded at regular intervals. Regarding spatial accuracy, mobile data is often less precise, as the location data is the location of the BTS, not the exact phone location.

Several studies have explored using cellular data for TMD with varying degrees of success. For instance, Bachir et al. [13] proposed an unsupervised method to classify trips into road and rail categories. However, its inability to handle multi-modal and uncertain journeys limits this method's performance. Qu et al. [100] proposed a transportation mode split model to estimate the shares of three transportation modes within each census tract. Nevertheless, the model's predictive accuracy can vary across different regions. Li et al. [76] proposed a novel public TMD system that accurately detects subway, train, and bus trips. However, the system's accuracy is contingent on the quality and coverage of cellular data, potentially limiting its performance in remote areas, during rapid user movement, or in crowded environments.

2.1.2 Inertial Measurement Sensor Data

Inertial measurement unit (IMU) sensor data comprises datasets from wearable sensors attached to the human body. Typically, these sensors include accelerometers, gyroscopes, and magnetometers, providing highly detailed and complex data. Such datasets offer significant potential for the development and refinement of machine learning models [40]. This data can be utilised to identify the user's current mode of transportation, such as walking or public transport [140]. However, IMU sensor data is not without limitations. Data acquisition is often costly and time-consuming, limiting participation and potentially compromising the representativeness of the dataset. Additionally, the placement of sensors on specific body locations can influence data quality and may not accurately reflect people's movement patterns, in reality [140].

Recent studies have explored the use of IMU sensor data for TMD. Wang et al. [140] proposed



(a) SHL dataset. (b) CAPTIMOVE dataset.

Figure 2.2: Participants outfitted with several smartphones and cameras for data acquisition in: (a) SHL dataset [58], (b) CAPTIMOVE dataset [5].

a standardised framework for transportation mode recognition using smartphone sensors. However, the study's limitations include a limited number of participants and geographic diversity in data collection. Subsequent research [141] indicated the superiority of deep learning methods over classical machine learning approaches, with the top-performing submission achieving an F1 score of 88.5%. Nevertheless, this study also suffered from limited participant diversity, as data was collected from only two smartphone users. Hasan et al. [60] investigated using smartwatch and smartphone data to predict transportation mode. The study identified notable disparities in physical attributes and activity metrics associated with different modes of transportation. In a related study, Shin et al. [119] presented a method for urban data collection, sensing, and classifying diverse transportation activities using smartphone sensors. This approach enables the classification of transportation modes directly on smartphones.

2.1.3 Global Positioning System Data

The Global Positioning System (GPS) is a satellite-based radio navigation system owned by the United States Space Force [133]. It is a network of 24 or more satellites orbiting the Earth and providing continuous global positioning, navigation, and timing (PNT) services [61]. Each satellite emits radio signals containing information about its precise location, status, and time. GPS receivers on Earth detect these signals and calculate the distance between the receiver and each satellite using the known speed of light and the measured signal travel time. The receiver can determine its exact geographic coordinates by triangulating the distances from multiple satellites, including latitude, longitude, and altitude.

The increasing prevalence of GPS-enabled devices, such as smartphones, has led to the development of studies that focus on analysing mobility patterns and behaviour based on mobility data [118], including the identification of modes of transport. When satellites are visible, GPS data

can be obtained with precise spatial and temporal accuracy. The widespread availability of smartphones with sensing capabilities has enabled a cost-effective approach to obtaining GPS trajectory data for TMD studies. In addition to facilitating a cost-effective data collection, mobile crowdsensing enables this data to be collected at a high granularity, such as every 1 second in [107], and at sampling rates of around 3-5 seconds in the GeoLife project [163]. The GeoLife project pioneered the TMD studies from GPS trajectory data in 2008 [161]. Since then, GPS data have been used to estimate mode of transport use by leveraging on the accuracy of GPS technology, as evidenced by the increasing number of publications [31, 90, 91, 160, 159] on the topic in recent years.

Recent advancements in data mining and machine learning have intensified research efforts to utilise the vast quantities of data collected by GPS-enabled devices for travel mode inference. Numerous researchers have employed GPS-enabled devices to gather data for mode detection studies. For instance, Bolbol et al. [20] proposed an SVM-based method for mode detection, emphasising the importance of speed and acceleration as key features. They evaluated the discriminative power of various features for six different modes of transportation. Stenneth et al. [126] developed a TMD framework that integrates GPS data with transportation network knowledge. Their framework achieves state-of-the-art accuracy on a relatively small dataset of GPS trajectories using various machine-learning classification algorithms. However, the framework's performance is limited by the size and diversity of the dataset.

In recent years, the breakthrough performance of deep neural networks in various research domains, such as computer vision and natural language processing, has prompted its utilization in the context of TMD [69]. Endo et al. [44] introduce a novel approach that models raw GPS trajectory data into 2D image structures. This transformation automatically extracts high-level features using a fully connected deep neural network. The core idea involves representing trajectory data as images, where pixel values represent the user's duration at specific locations. These image-based features are integrated with manual features and fed into a classical classifier for transportation mode prediction. However, a limitation of this approach is that it solely relies on duration time and lacks consideration of spatio-temporal information. Dabiri and Heaslip [31] propose using convolutional neural networks (CNNs) for mode detection from GPS trajectories. The study autonomously extracts high-level features from raw GPS data by leveraging CNN architectures and identifying five travel modes. This approach stands out for its ability to operate solely on GPS data, eliminating the need for contextual information.

2.2 Trajectory Data Mining

2.2.1 Terminology

This section establishes the foundational definitions that will be consistently referenced throughout this thesis.

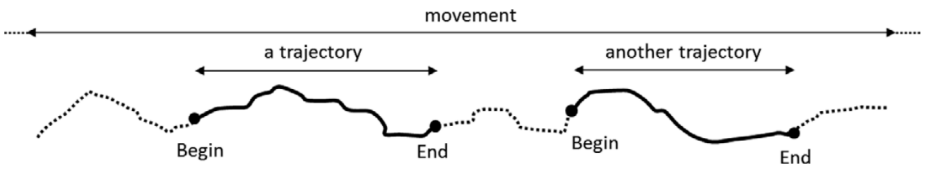


Figure 2.3: Trajectory segments derived from the overall movement pattern of an object [125].

Definition 2.1 (Point). A point P_i is a tuple *(time, latitude, longitude)* representing a specific spatio-temporal location. It signifies the position of an object at a given timestamp.

Definition 2.2 (Movement track). The *movement track* of a moving object is the temporally ordered sequence of spatio-temporal data point records as captured by a positioning device.

Each data point record holds the exact instant of the capture and may include additional attributes captured by the device, such as instantaneous speed, altitude, and direction [125]. No two records share the exact instant value.

Definition 2.3 (Trajectory). A trajectory denotes a portion in the path traversed by a moving object within a defined time interval. It constitutes a sequence of spatio-temporal positions, each uniquely timestamped, thereby providing a discrete record of the object's successive locations throughout its movement.

Formally, a trajectory T can be represented as an ordered sequence $T = \{\langle p_1,t_1\rangle, \langle p_2,t_2\rangle, \ldots, \langle p_n,t_n\rangle\}$, where each element $\langle p_i,t_i\rangle$ indicates the object's presence at location p_i at time t_i [51]. The timestamps are sequentially ordered (i.e., $t_j < t_k \quad \text{for} \quad 1 \leq j < k \leq n$), capturing the continuous movement of the object over the interval from the start time to the end time.

Definition 2.4 (trajectory segment). A *trajectory segment* refers to a specific portion or subsection of a trajectory. It represents a continuous part of this path, usually defined by specific time intervals $[t_{\text{begin}}, t_{\text{end}}]$.

In transportation or movement analysis, a trajectory typically represents the path or course taken by an object or individual over time. Figure. 2.3 illustrates a portion of an object's movement trajectory, highlighting two specific segments deemed relevant for further analysis.

2.3 Mode Detection Approaches

Over the past two decades [103, 128, 160], researchers have shown increasing interest in identifying transportation modes used by commuters, employing a diverse range of techniques, from statistical and expert system methods to more recent machine learning approaches. This section provides a brief overview of these approaches.

2.3.1 Ruled-based Approaches

Rule-based expert systems constitute a significant portion of TMD research. These systems rely on a predefined set of rules, often expressed in an "*if-then*" format. By analysing motion attributes such as average and maximum speed, proximity to network infrastructure (e.g., bus stops, train stations), or deviation from major roads, these systems effectively detect and classify commuters' modes of transportation. As exemplified by Bohte and Maat [19] and Shin et al.[119], early research in this area focused on applying classical rule-based systems. More recent work, such as that of Xiao et al. [145], has extended these techniques to address complex urban scenarios.

Researchers have incorporated fuzzy logic and membership functions to enhance the flexibility and adaptability of rule-based systems. This approach, as demonstrated in [18, 34, 117], allows for more nuanced decision-making by considering the degree to which a trajectory exhibits anomalous characteristics. Combining the precision of rule-based systems with the flexibility of fuzzy logic provides a robust framework for TMD and analysis.

2.3.2 Statistical Methods

Statistical methods have been employed to infer transportation modes based on probabilistic models. Stopher et al. [127] proposed a probability matrix method that leverages speed characteristics to assign weights to different modes of transportation. However, this method struggles distinguishing between bus and car trips, especially under inconsistent speed data. A study by Xu et al. [149] introduced a probabilistic approach that combines a Hidden Markov Model (HMM) with Speed Distribution Law (SDL) and Cumulative Prospect Theory (CPT) to identify transportation modes using mobile phone data. This method aims to improve accuracy across various traffic conditions.

More recently, the authors of [142] proposed a Bayesian Network Learning Framework to identify travel modes using cellular signaling data. This framework considers travel behavior, personal attributes, and traffic environment to classify travel modes accurately. Despite the potential of these statistical methods, their widespread adoption remains limited.

2.3.3 Machine Learning Methods

In recent years, machine learning models have emerged as the dominant approach in TMD research. These methods are broadly categorised into three primary branches: unsupervised and supervised learning. The following sections provide a brief overview of these approaches.

2.3.3.1 Unsupervised Learning Methods

Unsupervised learning [53] is a machine learning technique that trains models on unlabelled data. By analysing patterns within the data, unsupervised learning algorithms can perform tasks such as clustering, dimensionality reduction, and anomaly detection. In the context of TMD, unsupervised learning has been employed to identify transportation modes from mobility data.

Bachir et al. [13] utilised Bayesian inference combined with clustering techniques to classify transportation modes. Wang et al. [138] employed k-means clustering to categorise trips based on their duration. This method divided trips into two clusters: one representing private vehicle trips and the other representing public transport trips. The cluster with a shorter average trip duration, aligning with Google Maps driving time estimates, was assigned to private vehicle trips, while the other cluster was assigned to public transport.

Similarly, k-means clustering of cellular data is the primary technique used in [71] to discriminate between walking, public transport, and car modes. The authors aggregated clusters of neighbouring cells exhibiting data fluctuations and replaced these clusters with a weighted centre point.

Lin et al. [82] proposed a method based on the Kolmogorov-Smirnov test, utilising only instantaneous, average, and maximum speed metrics. The authors assumed that speed distributions across street segments remain consistent for vehicles of the same transport mode but differ significantly between modes, particularly in higher-speed ranges. While achieving high accuracy, the algorithm's performance was not benchmarked against other methods.

More recently, Markos and Yu [87] applied an unsupervised deep learning technique, DeepCAE, to identify transportation modes. By combining a pre-trained convolutional autoencoder with a clustering layer, their model learns meaningful representations from GPS trajectory data, enabling the discovery of underlying patterns and the classification of different transportation modes without requiring labelled training data.

2.3.3.2 Semi-supervised Learning Methods

Semi-supervised learning methods solve the challenge of limited labelled data availability in machine learning. Unlike unsupervised methods, which operate solely on unlabelled data, semi-supervised methods leverage labelled and unlabelled data to enhance model performance. Semi-supervised methods can extract valuable information by combining these two data sources, thereby improving model accuracy and robustness.

There has been a growing interest in semi-supervised approaches to TMD research in recent years. Dabiri et al. [32] proposed SECA, a novel semi-supervised deep learning approach for TMD utilising GPS trajectory data. This method combines a convolutional autoencoder with a convolutional neural network to extract relevant features from GPS segments. Their model outperformed state-of-the-art methods. However, a limitation of this study is the small number of users with labeled data, which was restricted to only two users. This limitation suggests that the algorithm's performance may deteriorate when labeled data is scarce.

Building upon these advancements, James et al. [68] proposed a semi-supervised deep ensemble learning approach for TMD that requires only a small amount of annotated data. This method leverages GPS trajectories of variable lengths. It employs a tailored feature engineering process to construct a robust deep neural network architecture that accurately maps latent information to travel modes.

Furthering exploring semi-supervised learning for TMD, Zhu et al. [166] proposed a CNN-GRU model that leverages labeled and unlabeled GPS trajectory data. The model effectively extracts features from GPS trajectories by employing a pseudo-labeling technique. Additionally, a grouping-based aggregation scheme and a data-flipping augmentation scheme are introduced to enhance the convergence and robustness of the framework. However, the model is susceptible to overfitting.

2.3.3.3 Supervised Learning Methods

Most machine learning methods applied to TMD are rooted in supervised learning. In this paradigm, models are trained on labelled data, where both input features (e.g., GPS trajectories or sensor data) and corresponding output labels (i.e., transportation mode) are provided. TMD is thus considered a supervised classification problem, with the labelled data serving as the ground truth for training and subsequent model evaluation.

Pioneering studies exploring mobility patterns from data and subsequent transportation mode inference based on supervised learning were conducted by Zheng et al. [159, 160] on the Geolife project. This project, which made the data openly available to enhance research reproducibility, focused on detecting four different transportation modes. Since then, a growing body of literature has emerged, with the vast majority utilising the Geolife dataset. Subsequent studies [162, 163] also employed a supervised learning approach.

Building upon these initial studies, Stenneth et al. [126] proposed a method to identify a user's mode of transportation using GPS data and real-time transportation network information, achieving high accuracy compared to previous methods. However, their approach relies on continuous input of transportation network data, limiting its applicability in certain scenarios. Bolbol et al. [20] further demonstrated the applicability of supervised machine learning for TMD by segmenting trajectories, extracting features, and applying SVM for classification.

Several studies, [35, 67, 148] have employed supervised learning approaches for TMD. Jahangiri [67] conducted a comparative study of various supervised learning methods, but the study was limited to specific sensors and transportation modes. Xiao et al. [148] proposed a hybrid approach using GPS data, segmenting trajectories, and extracting global and local features. To enhance model performance, they opted for tree-based ensemble models. Typically, these studies rely on hand-crafted features engineered from raw data.

Recent research has increasingly turned to deep learning architectures for TMD, leveraging their success in fields like computer vision and natural language processing. These models can automatically extract features and learn representations directly from raw data. For example, DeepTransport [123] employs a four-layer LSTM neural network to learn complex spatio-temporal patterns in human movement from large-scale heterogeneous data. The LSTM architecture is well-suited for this task because it captures temporal dependencies. By jointly learning from two sets of features, DeepTransport effectively models the spatio-temporal nature of human mobility.

To further leverage the potential of deep learning, some studies have explored the transformation of trajectory data into image-like representation, from which deep features are extracted.

The proposal outlined in [44] attempts to overcome the limitations of hand-crafted features by employing deep feature extraction from these image-like representations. The final model is then trained in a supervised manner, utilising both deep and hand-crafted features. Similarly, Ribeiro et al. [104] utilise vision transformer-based neural networks to predict transportation modes directly from image representations of trajectory data without using hand-crafted features.

In contrast, other studies have focused on deep learning models that directly process raw sensor data without relying on hand-crafted features or image representations. Dabiri and Heaslip [31] represented trajectory data in a 3D-like structure suitable for CNN-based models. Drosouli et al. [41] introduce TMD-BERT, a transformer-based model for TMD from sensor data. This model leverages the transformer architecture to process the entire sequence of sensor data, understanding the importance of each data point and assigning appropriate weights using attention mechanisms. This enables the model to capture global dependencies within the sequence, leading to more accurate predictions.

2.4 Transportation Mode Classification

This section briefly overviews the most common machine learning classification algorithms employed for TMD. As the primary focus of this thesis is on a classification-based approach to TMD, our subsequent discussion will concentrate on ML classification algorithms.

2.4.1 k-Nearest Neighbours

The k-Nearest Neighbours (k-NN) is a straightforward, distance-based classification algorithm that operates on a lazy learning paradigm [88]. The k-NN lacks a dedicated training phase and memorises all training data, retaining it in memory. To predict the class of a new object, k-NN identifies the classes of the k most similar objects. Using the class information from these closest neighbours, k-NN thus applies a local-learning approach. The parameter k specifies the number of neighbouring labelled objects that the algorithm considers. A notable drawback of k-NN is its potentially slow classification of new objects, which requires computing the distance between the new object and every object in the training set. This algorithm is used for mode detection in [67, 111].

2.4.2 Decision Tree Algorithms

Decision trees (DTs) are a standard supervised learning algorithm for classification and regression tasks [88]. They have been employed in TMD research, as evidenced by studies such as [67, 77, 126, 160]. DTs construct a tree-like model where each internal node represents a “test” on an attribute, each branch represents the outcome of that test, and each leaf node represents a class label. The algorithm recursively partitions the dataset based on the attribute that best separates the classes, using metrics such as information gain or Gini impurity to determine the optimal split.

While DTs are intuitive to interpret and can model complex, non-linear relationships, they are prone to overfitting [109].

2.4.3 Support Vector Machines

Support Vector Machines (SVMs) are a family of supervised learning algorithms widely used for classification and regression tasks [88]. They are particularly effective in high-dimensional spaces and are known for their ability to handle complex nonlinear relationships between features and labels. The *Support Vector Classifier (SVC)* is a popular variant of SVMs designed explicitly for classification problems. It aims to identify a hyperplane that effectively separates data points belonging to different classes while maximising the margin between the hyperplane and the nearest data points (called the *support vectors*). This margin maximisation principle helps SVMs to achieve high classification accuracy and generalisation capabilities, even when dealing with noisy or imbalanced datasets [137]. This algorithm is used in transportation mode classification in [20, 27, 159]. The computational cost of SVMs can be significant, especially for large datasets, as it depends on the number of support vectors. Additionally, SVM performance is susceptible to the choice of hyper-parameters.

2.4.4 Logistic Regression

Logistic Regression (LR) is a probability-based [88] supervised learning algorithm used for classification tasks. In LR, the goal is to model the probability that a given input (e.g., a set of features) belongs to a particular class. The model uses the logistic function [74] (also known as the sigmoid function) to map predicted values to probabilities between 0 and 1:

$$P(Y = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}} \tag{2.1}$$

where:

- $P(Y = 1 \mid X)$ is the probability of the positive class given the input features $X = (X_1, X_2, \ldots, X_n)$,
- β_0 is the intercept term, and
- $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients for each feature.

LR fits these coefficients by maximising the likelihood function and estimating parameters that best explain the observed data. Due to its interpretability and computational efficiency, it is commonly used in fields like medical research, social sciences, and machine learning for binary and multiclass classification tasks. In the context of TMD, studies such as [44] have employed LR for classification.

2.4.5 Bayesian Belief Network

The Bayesian Belief Network (BN) is a probability-based graphical model that represents a domain variable (i.e., input features) and its probabilistic dependency using a directed acyclic graph

(DAG). In this model, the variables are represented by nodes, and edges represent the probabilistic relationships between them [30]. BN is widely used due to its ability to model complex dependencies and uncertainty in data.

Each node within a BN is associated with a conditional probability distribution (CPD) that specifies the probability of its state given the states of its parent nodes. The BN learning process involves two main steps: structure learning and parameter learning. In structure learning, the network's structure is inferred by identifying conditional independence relationships between variables [21]. Once the structure is learned, the parameters of the CPDs are estimated from the training data using maximum likelihood estimation or Bayesian methods.

In a classification problem, given a set of input features X, a BN model can predict the target variable Y. The model computes the posterior probability $P(Y|X)$ using Bayes' theorem:

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)} \tag{2.2}$$

where:

- $P(Y)$ is the prior probability of the target class.
- $P(X|Y)$ is the likelihood of observing the data given the class.
- $P(X)$ is the marginal probability of the data, which can be ignored in classification as it remains constant across all classes.

The class with the highest posterior probability is assigned to the instance. Several TMD studies have employed BN models for mode identification, including [49, 126, 160]. A recent study by Wang et al. [142] also utilised BN for TMD.

2.4.6 Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) is a powerful ensemble learning technique that builds on the principles of the gradient boosting algorithm [29]. XGBoost constructs an ensemble of decision trees sequentially, adding trees that focus on correcting the errors made by previous models. This process is guided by a gradient descent-like optimisation algorithm, minimising a specific loss function. To enhance the model's generalisation performance, XGBoost incorporates regularisation techniques, tree pruning, and parallel computing. This distributed architecture allows for efficient training of large models on multiple machines. XGBoost has been successfully applied to TMD in studies such as [4, 77, 148].

2.4.7 Random Forest

Random Forest (RF) is a powerful ensemble learning technique that constructs multiple decision trees and aggregates their predictions. Each tree is trained on a different bootstrap sample of the training data [16, 88], introducing randomness and reducing overfitting. Additionally, at each node

of a tree, a random subset of features is considered for the split, further enhancing diversity. This feature selection process helps to prevent individual trees from becoming overly reliant on a single feature, improving the overall model's robustness.

By combining multiple diverse trees, RF can often achieve higher accuracy and better generalisation performance than individual decision trees. The final prediction is determined through a majority vote for classification tasks. This ensemble approach leads to more robust and stable predictions, making RF a popular choice for various machine learning applications, including TMD [4, 57, 91, 136].

2.4.8 Deep Learning Algorithms

Deep learning algorithms have achieved significant breakthroughs in various fields, including computer vision, image processing, and natural language processing [39, 143]. As a subfield of machine learning, deep learning utilizes artificial neural networks to learn complex patterns from large datasets. This capability has enabled advancements in image recognition, natural language processing, and speech recognition.

Recent advancements in data availability, particularly the proliferation of sensor data and computational power, such as the development of GPUs, have contributed to the widespread adoption of deep learning techniques [73]. Inspired by these developments, researchers have begun to apply deep learning techniques to spatio-temporal data mining, specifically in the context of TMD. Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks are the most common deep learning architectures used in this area.

2.4.8.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are deep learning models designed explicitly for processing structured grid data, such as images [1]. CNN utilises convolutional layers to extract spatial hierarchies of features, starting from simple patterns like edges and progressing to more complex structures. A typical CNN architecture comprises convolutional, pooling, and fully connected layers. Convolutional layers apply filters to the input data (e.g., an image) to detect local patterns like edges, textures, and shapes. Pooling layers reduce the spatial dimensions of the input features, preserving essential features and reducing computational complexity. Finally, fully connected layers connect the extracted features to the output layer, making predictions about the input data.

While CNN are primarily associated with image data, they are also well-suited for processing spatio-temporal data [1]. Several TMD classification models, such as those in [31, 47, 80, 86], are primarily built using CNN architectures.

2.4.8.2 Long Short-Term Memory

Long Short-Term Memory (LSTM) networks are a specialised type of recurrent neural network (RNN) designed to process sequential data and address challenges associated with long-term dependencies. By incorporating advanced gating mechanisms and memory cells, LSTM regulates

the flow of information within the network [116]. These gates enable the selective retention or forgetting of past data, allowing the network to focus on relevant patterns over extended sequences. This design equips LSTM to effectively capture temporal patterns in data, such as time-series data while mitigating the issue of vanishing gradients.

LSTM networks have been widely adopted in TMD research due to their ability to capture temporal dependencies in sequential data. Studies such as [10, 38, 40, 83, 123] have successfully employed LSTM networks for TMD tasks, leveraging their capability to model complex temporal patterns in GPS trajectories and other sensor data.

2.5 Features for Mode Detection

Feature engineering is a critical step in TMD, as it involves extracting relevant information from raw data to enhance model performance. A diverse range of features have been used. The features proven effective for TMD tasks can be primarily categorised as kinematic, spatial, and contextual [124].

Kinematic features, derived from the spatial and temporal information embedded in trajectory data, relate to the subject's motion. The majority of previous studies have primarily relied on kinematic features. Features like speed [14, 18, 71], acceleration [80, 81, 145], jerk [31, 32, 79], gyroscope's angular velocity [3, 59], magnetometer [130, 150] and rotation vector data [11], along with their statistical derivatives (e.g. mean and standard deviation of speed), are the dominant features in the literature.

Spatial features focus on the spatial characteristics of the data, such as geographic coordinates. This category includes features like total distance travelled [32, 136], heading change, movement direction [98, 148] and stop rate [45, 159]. These features provide insights into the spatial patterns of movement and can help distinguish between different transportation modes. For instance, the total distance travelled can help differentiate between walking and driving, while heading changes and movement direction can provide clues about turning manoeuvrers and changes in speed.

Contextual features provide additional information about the environment and conditions surrounding the trip. These can include features such as the day of the week [95], time of day [165], and trip duration [9]. External data sources such as road and rail network data [13], proximity to stations or bus stops [154, 155], GIS layers [63, 110, 111], and transportation network data [123, 142].

The careful selection and engineering of features significantly impact the performance of TMD models. By considering the specific characteristics of the dataset and the desired level of model accuracy, researchers can optimise feature engineering strategies to achieve superior results. Table 2.1 summarizes previous studies on this subject.

Table 2.1: Overview of approaches used in transport mode detection research

Study	Approach	Data source	Participants	Duration	Granularity	External data	Features	Classifier	No. of modes	Accuracy (%)
[160]	Supervised	GPS data	45	6 months	1-5 s	No	distance, velocity, & acceleration statistics	DT	4	72.8
[117]	Rule-based	GPS data	4,882	–	5-10 m	No	speed and acceleration statistics	fuzzy engine	5	–
[126]	Supervised	GPS data	6	3 weeks	15 s	GIS	speed, acceleration, travel direction, closeness to station	RF	6	93.7
[20]	Supervised	GPS data	81	2 weeks	60 s	No	velocity and acceleration statistics	SVM	6	88
[18]	Rule-based	GPS data	17M points	7 days	5-6.5 s	GIS	speed statistics	fuzzy engine	10	91.6
[157]	Semi-supervised	GPS data	69	> 5 years	1-5 s	No	speed, acceleration, & jerk.	DNN	5	91.4
[65]	Statistical	GPS data	12	–	3.5-5 m	GIS	speed acceleration, rate-of-change	NPDA†	5	0.95‡
[119]	Rule-based	IMU sensor	30	–	1 s	No	acceleration	–	4	82
[67]	Supervised	IMU sensor	10	–	–	No	acceleration, gyroscope, rotation vector	RF	5	93
[44]	Supervised	GPS data	69	> 5 years	1-5 s	No	distance, velociy, deep features	LR	7	67.9
[71]	Unsupervised	Mobile data	300,000	3 days	5 min	No	speed statistics	K-means	3	–
[80]	Supervised	IMU sensor	–	–	–	No	acceleration	CNN	7	94.5
[14]	Unsupervised	GPS data	2	3 days	2 min	No	speed statistics	HMM	4	78
[33]	Ruled-based	Mobile data	3	–	–	TN⋆	speed statistics	heuristic	6	–
[148]	Supervised	GPS data	69	> 5 years	1-5 s	No	velocity, acceleration turning angle	XGB	6	90.8
[31]	Supervised	GPS data	69	> 5 years	1-5 s	No	speed, acceleration, jerk, bearing rate	CNN	5	84.8
[32]	Semi-supervised	GPS data	69	> 5 years	1-5 s	No	speed, acceleration, jerk, distance	convolutional autoencoder	5	76.8
[13]	Unsupervised	Mobile data	–	1 month	–	GIS	distance, number of roads/rail lines, number of station	bayesian inference	2	0.96*
[149]	Statistical	Mobile data	500	20 days	–	No	speed statistics	HMM	3	91.7

– -not available NPDA†- non-parametric discriminant analysis ‡- Cohen's kappa TN⋆ - transportation network * - pearson correlation

Study	Approach	Data source	Participants	Duration	Granularity	External data	Features	Classifier	No. of modes	Accuracy (%)
[142]	Statistical	Mobile data	–	–	–	GIS	distance, speed, personal attributes (age, gender)	bayesian network	5	82.9
[87]	Unsupervised	GPS data	69	> 5 years	1-5 s	No	speed, acceleration, jerk, bearing rate	convolutional autoEncoder	5	80.5
[127]	Statistical	GPS data	72	56 days	15-40 s	GIS	speed statistics	probability matrix	5	95

2.6 Concluding Remark

This chapter has comprehensively reviewed state-of-the-art research on transportation mode detection. We have discussed various data types employed, different approaches to mode detection, and the application of supervised machine-learning techniques. Additionally, we have explored the features used to enhance the accuracy of TMD models.

Chapter 3

Datasets and Preprocessing Tasks

The previous chapter reviewed the state-of-the-art in transportation mode detection, including discussions of various data types, approaches, and algorithms employed in the literature. This chapter delves into the datasets utilised in this thesis research, providing detailed characterisations of each.

3.1 Datasets Description

Two real-world GPS trajectory datasets are employed in this thesis, which are detailed.

3.1.1 The SenseMyFEUP Dataset

The *SenseMyFEUP (SMF2016)* dataset is a GPS trajectory dataset of 227 participants collected in Porto, Portugal, in April 2016 [108]. It was collected using an Android-based mobile application installed on participants' smartphones. The application was designed to automatically record location data whenever user movement was detected and to stop recording when the user ceased moving. SMF2016 is recorded at approximately one sample per second or every 5 meters. The location data is structured as a tuple containing the following attributes: timestamp, longitude, latitude, altitude, speed, bearing, and GPS accuracy, thus $L_i =< time[t], lon, lat, alt, speed, bearing, gps\_acc >$.

Moreover, the application includes an end-of-trip survey administered after each trip completion, prompting users to indicate their travel mode. (Figure. 3.1). This self-reported data serves as the ground truth for our analysis. In all, *SMF2016* contains travel mode information for five different transportation modes: *bike, bus, car, foot* and *metro*.

The SMF2016 dataset additionally includes self-reported information on transportation modes. After each trip, the application prompts participants to complete a survey indicating the transportation mode used for their trip (see Figure. 3.1). This self-reported data serves as the ground truth for our research. The SMF2016 contains transportation mode information for five different modes: bike, bus, car, foot, and metro.

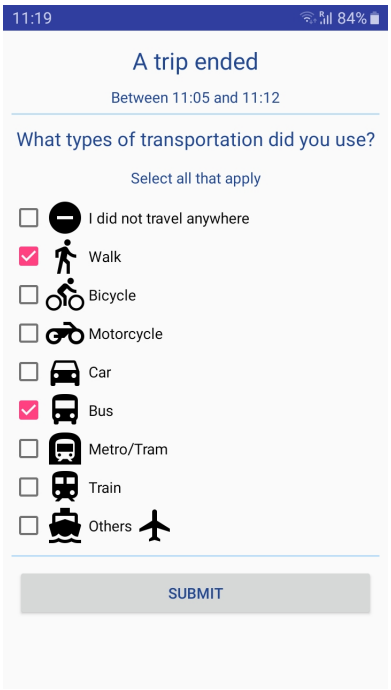


Figure 3.1: SMF2016 end of trip survey.

3.1.2 The Geolife Dataset

We also utilise the publicly accessible *Geolife* dataset, developed and published by the Microsoft Research Asia [160, 162, 163]. Collected from 182 participants, primarily in Beijing, China, over a period spanning nearly six years (April 2007–February 2012), the dataset integrates movement information recorded through various GPS loggers and GPS-enabled phones. Data points feature a spatio-temporal granularity of approximately 1–5 seconds or 5–10 meters. Each trajectory consists of a temporally ordered sequence of latitude and longitude coordinates, represented as $< lat, lon, alt, timestamp >$.

According to the Geolife user guide [164], 73 users have labelled their trajectories with travel mode information. However, upon examining the raw data, we found that only 69 users have recorded transportation mode information. The dataset encompasses ten different transportation modes (8 land): walk, bike, bus, car, taxi, train, subway, motorcycle, airplane, and boat. The Geolife user guide further recommends combining car and taxi modes into a single category.

Table 3.1 summarises the key attributes of the two datasets. To further understand user behaviour, Figures. 3.2 and 3.3 illustrate the distribution of the number of trips per user in both SMF2016 and Geolife, respectively.

The SMF2016 dataset reveals a wide range of trip frequencies among users. Approximately 25% of users have fewer than 10 trips, while 15% have more than 100 trips. Similarly, the Geolife dataset exhibits a skewed distribution, with 25% of users having fewer than 10 trips and 15% having more than 150 trips.

Table 3.1: Key Characteristics of the Datasets

	SMF2016	**Geolife**
Period of data collection:	April 2016	April 2007 - February 2012
Number of participants:	227	182
Travel mode information:	5 modes	10 modes
Sampling rate:	1 sample per second	1 sample every 3-5 seconds
Number of trajectories:	13, 212	18,670
Number of data points:	9,657,405	24,876,978
Distance coverage (km):	229,564	1,292,951
Total duration (hours):	5,082	50,176
Effective days:	30	11,129
Accessibility:	*private*	*public*

3.2 Data Preprocessing

This Section outlines the process of chaining GPS traces into trips in the datasets.

3.2.1 Trip Preprocessing

Both GPS trajectory datasets are chronologically ordered. We define a trip as a sequence of user GPS points in session, separated by a stop of at least 30 minutes. A trip is composed of multiple GPS points, each represented as a tuple containing latitude, longitude, and timestamp information. Our SMF2016 trip chaining algorithm identifies and joins two GPS traces within a session into a single trip based on their timestamps and location, according to the following criteria [108]:

- The GPS traces are associated with a particular user,
- The time interval (Δt) between a trace and its predecessor is no more than 30 minutes,
- The distance between a previous trace and the start of a new one is below 200m.

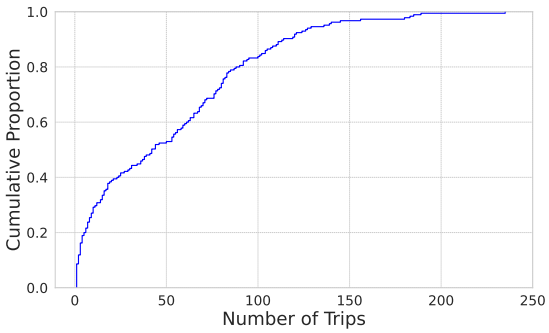


Figure 3.2: SMF2016 - Distribution of trips per user.

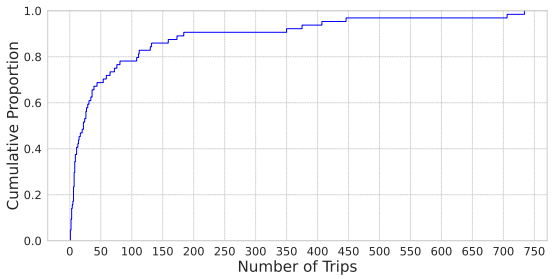


Figure 3.3: Geolife - Distribution of trips per user.

3.2.2 Trip Filtering

However, we observed instances where trips exhibited a significant difference in timestamp between successive GPS points beyond the established threshold. Further investigation revealed the large Δt between consecutive traces was caused by "unstopped sessions" where data collection continued without recorded stops. We thoroughly searched through all sessions in the dataset and documented the affected sessions in Table 3.2.

We observed instances where trips exhibited a significant difference in timestamp between successive GPS points, exceeding the established threshold. Further investigation revealed that these large time gaps were due to "unstopped sessions," where data collection continued without recorded stops. We conducted a thorough search of all sessions in the dataset and documented the affected sessions in Table 3.2.

Table 3.2: Sessions having $\Delta t > 1hr$ between consecutive GPS traces.

Number of cases found	58
Number of sessions affected	26
Total number of sessions in the dataset	5368
Number of trips in unstopped sessions	25

Notably, the trips exhibiting unstopped user sessions originated from trips surpassing the Porto region's boundaries. After excluding these problematic trips and focusing solely on trips within the Porto area, defined by the coordinates (41.38786, -8.77034) and (41,04928, -8.4253), we still observed some foot trips covering over 10 kilometers (Figure. 5.2a). This deviation from the typical pattern of foot trips in urban areas prompted us to cap foot trips at 8 kilometers or less. This resulted in trip distance distribution shown in Figure. 5.2b. In Table 5.3 we present the statistics of filtered trips.

Table 3.3: Trips filtering statistics.

Number of trips before filtering	3730
Trips with large Δt between consecutive traces	15
Number of trips outside the study area	764
Number of trips within Porto area (after filtering)	2951
Total trips (foot mode capped to 8km)	2909

3.2.3 Trip Segmentation

To capture the dynamic nature of human movement patterns as closely as possible and enhance the accuracy of the TMD algorithms, we split the trips into alternating sequences of motion and stationary segments [90, 108]. This granular representation allows for a more detailed analysis of movement behaviour and facilitates the identification of transportation mode transitions. However, this segmentation process can introduce a significant number of short segments, potentially

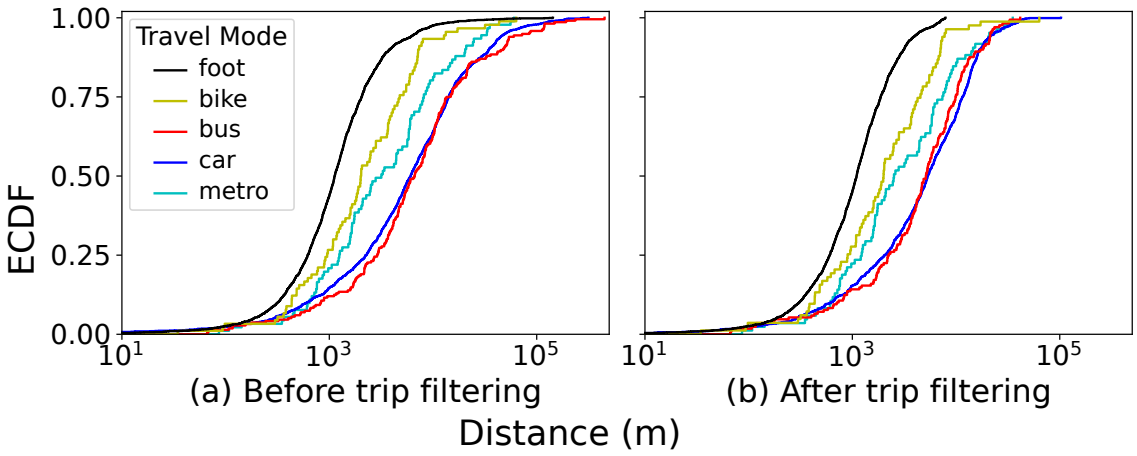


Figure 3.4: Trip distance distribution by travel mode (a) before trip filtering; (b) after filtering to trips covered in Porto region.

degrading the performance of the TMD classifiers [108]. We impose a minimum segment length threshold of 50m to mitigate this issue. This ensures only meaningful and informative segments are considered, improving the classifier's ability to distinguish between different transportation modes. The segmentation process begins by identifying stationary segments, representing periods of inactivity. We employ a threshold-based method using 85^{th} percentile of instantaneous GPS speed. Specifically, a stationary segment is established if the user's average speed falls below 0.5m/s for 5 seconds.

In the Geolife dataset, user GPS trajectories are partitioned into separate trips based on the time interval between two successive GPS points exceeding the predetermined threshold of 30 minutes [31]. Each trip is then further divided into single-mode segments based on transportation mode, with the begin and end timestamps of each segment annotated.

3.2.4 Point-level Attributes

To extract meaningful insights, we compute point-level motion attributes for each point in every trajectory, including speed, acceleration, jerk (rate of change of acceleration), and bearing rate. While the SMF2016 dataset provides instantaneous GPS speed for each location point, the Geolife dataset lacks this information. To address this limitation, we estimate speed for Geolife trajectories by calculating the relative geographical distance between consecutive GPS points using the using the Vincenty Formula [134] and dividing it by the time elapsed.

Consider two consecutive GPS points, P_m, P_n, as illustrated in Figure. 5.3, we calculate the speed of P_m using eq. (5.1)

$$S_{P_m} = \frac{Vincenty(P_m, P_n)}{\Delta t} \tag{3.1}$$

Figure. 3.6 illustrates the speed distribution for each transportation mode in the SMF2016 dataset. While foot and metro speeds exhibit similar distributions in the lower range (0 - 5 m/s, about 55% of the trips), a clear divergence is observed at higher speeds. Bus and car modes show

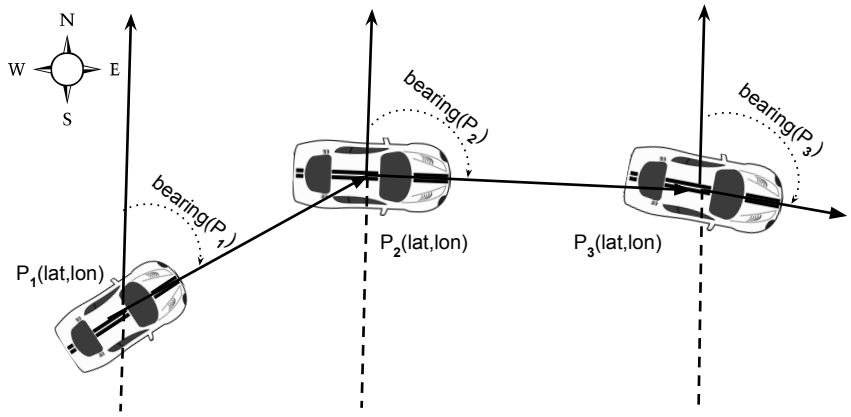


Figure 3.5: Visual representation of bearing between successive GPS points in a trajectory.

similar speed distributions. The distribution of bike speeds is significantly different from that of other modes, particularly at higher speeds.

Similarly, in the speed distribution of Geolife in Figure. 3.7, bus and car modes exhibit similar speed distributions, particularly in the lower speed range below 10 m/s. The distributions of foot and bike modes also share similarities, particularly in the lower speed range. However, it is evident that there is a noticeable difference in the distribution of metro compared to other transportation modes.

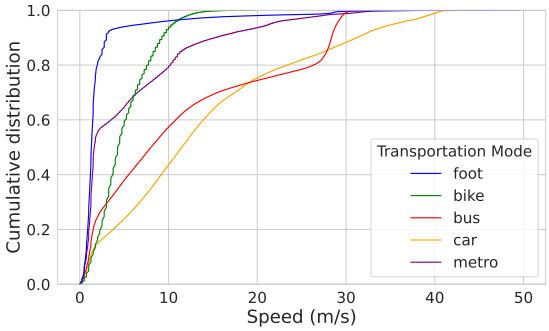


Figure 3.6: SMF2016 speed distribution per transportation mode.

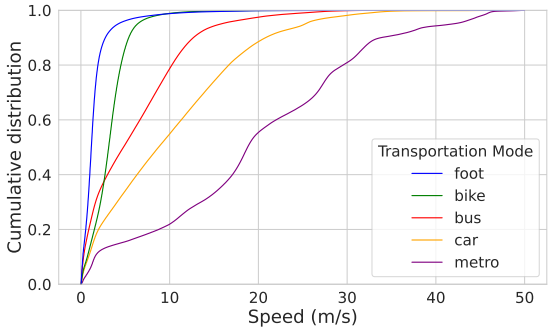


Figure 3.7: Geolife speed distribution per transportation mode.

To calculate point-level acceleration (A_{P_m}) and jerk (J_{P_m}) at point P_m, we utilise eqs. (5.2) and (5.3) respectively as follows:

$$A_{P_m} = \frac{S_{P_n} - S_{P_m}}{\Delta t} \tag{3.2}$$

$$J_{P_m} = \frac{A_{P_n} - A_{P_m}}{\Delta t} \tag{3.3}$$

Different transportation modes exhibit varying rates of change in direction. For example, pedestrians and cyclists frequently change their direction, while cars and metro trains tend to follow a more straight (linear) paths. The bearing, which quantifies the angular deviation from the true north [31], of the straight line connecting two consecutive GPS points (as illustrated in Figure

5.3), provides a measure of the directional change. We calculate the bearing rate between adjacent GPS points by determining the absolute difference between their respective bearings.

The SMF2016 dataset provides point-level bearing information. To calculate the bearing rate, we use the following equation (5.4):

$$BR_{P_m} = |bearing(P_n) - bearing(P_m)| \tag{3.4}$$

It is noteworthy to mention also that the Geolife dataset does not include information on the bearing of location points. Therefore, in order to use eq. (5.4), we first derive point's bearing using eq. (5.5) through eq. (5.7) to calculate this missing information. The values of *lat* and *lon* are first transformed to radian before applying these equation. The result, converted to degrees, is then used in Equation 5.4 to calculate the bearing rate.

$$y = sin[P_n(lon) - P_m(lon)] \times cos[P_n(lat)] \tag{3.5}$$

$$x = cos[P_m(lat)] \times sin[P_n(lat)] - sin[P_m(lat)] \times cos[P_n(lat)] \times cos[P_n(lon) - P_m(lon)] \tag{3.6}$$

$$bearing(P_m) = arctan(y,x) \tag{3.7}$$

3.2.5 Outlier Removal

Data quality is of paramount importance for accurate and meaningful analysis. To ensure data quality, we define and apply outlier detection criteria for point-level motion characteristics. Recognising that GPS data can be susceptible to noise and errors, we carefully examined each data point within each trip segment for evidence of physically unrealistic movement in urban scenarios. Building upon the literature [66, 135], we defined metric thresholds to identify and remove anomalous data points. Specifically, we considered data points exhibiting speeds higher than 50 m/s [31, 117] as outliers. Furthermore, we removed data points with acceleration values falling outside the range of -10 m/s² (deceleration) to 10 m/s² as these values are considered improbable [54, 139] for typical movement in urban settings.

Following the calculation of motion characteristics, each location point P_i within a segment is now represented by a vector comprising four attributes: speed (S_{P_i}), acceleration (A_{P_i}), jerk (J_{P_i}), and bearing rate, thus $L_i =< S_{P_i}, A_{P_i}, J_{P_i}, BR_{P_i} >$.

3.3 Benchmark Study

This section outlines the methodology employed to establish a baseline for this study. To this end, we experiment with the two widely used algorithms for TMD: RF and CNN, as detailed in Section 2.4. We conducted the experiments using the SMF2016, the primary dataset used in this study. Additionally, recognising the imbalance distribution of the dataset, we employed the use of

an ensemble of autoencoders classifier. This algorithm is detailed in Section 3.3.1. To comprehensively evaluate these models, twelve (12) experiments were conducted. These experiments are categorised by the type of features used (raw features vs. feature statistics) and the data splitting method employed: conventional random split versus a split based on the period of data collection. Figure. 3.8 illustrates the conceptual framework of of the proposed methodology.

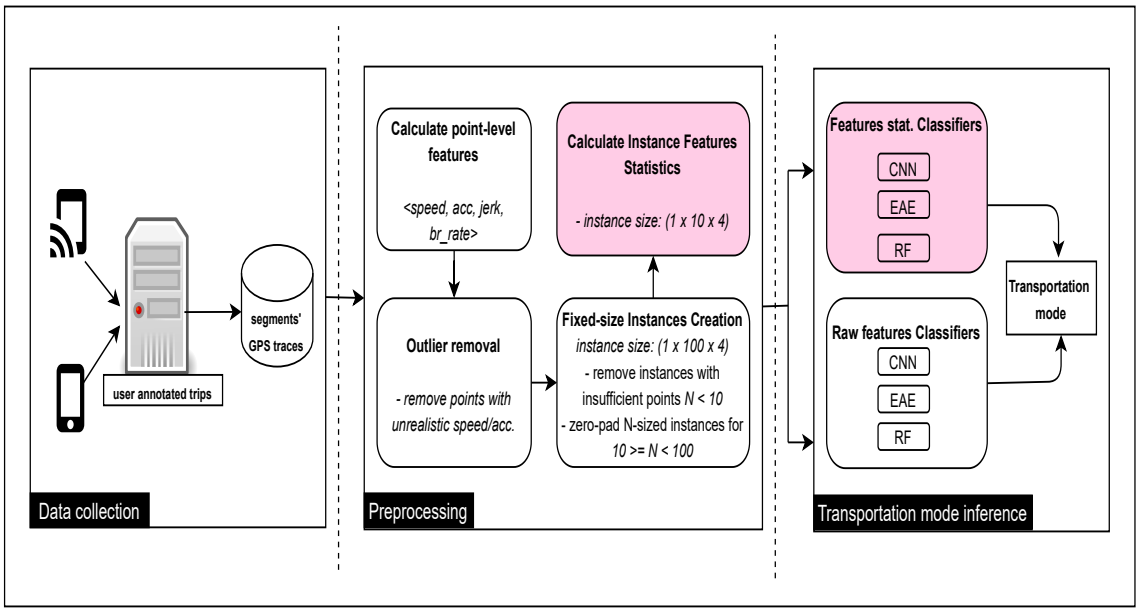


Figure 3.8: Conceptual framework of the proposed methodology.

Inspired by the studies in [31], and and recognising the requirement of fixed-size inputs for CNN, we create split each segment into fixed-size instances of size N=100 timesteps. This value was chosen as it approximates the median number of data points within all segments, which is 123.

To ensure consistent input dimensions, the following instance data handling procedures were implemented:

- Instances with fewer than 10 GPS points: Instances with fewer than 10 GPS points, typically occurring at the end of segments, were discarded due to insufficient data points.
- Instances with 10 or more but not up to 100 GPS points: Instances with between 10 and 100 GPS points were padded with zeros to achieve the desired instance size of $N = 100$.

This approach ensures that all input instances to the CNN have a consistent length as is required.

Each instance comprises 100 data points, where each data point is represented by a four-dimensional vector containing speed, acceleration, jerk, and bearing rate. This constitutes the raw point-level feature set, which is used in experiments conducted on raw data.

For the features statistics experiments, we calculated ten descriptive statistics for each of the four dimensions (speed, acceleration, jerk, and bearing rate) within an instance [108]. These statistics include the mean, median, standard deviation, minimum, maximum, median absolute

deviation (MAD), 25th percentile, 75th percentile, 85th percentile, and inter-quartile range. This resulted in 40 features (10 statistics x 4 dimensions) for each instance.

3.3.1 Ensemble of Autoencoders

We employ the Ensemble of Autoencoders (EAE) classifier [56]. Autoencoders are a type of artificial neural network used in unsupervised learning. An autoencoder comprises two main components:

1. Encoder: This component processes the input data and transforms it into a compressed representation known as a latent space or bottleneck.

2. Decoder: This component receives the compressed representation from the encoder and attempts to reconstruct the original input data.

Designed to learn efficient data representations (encodings), they aim to minimise the difference between the input data and its subsequent reconstruction.

The EAE utilises a modular design consisting of an ensemble of several autonomous autoencoders. This approach leverages the power of multiple autoencoders, each trained to effectively represent data instances belonging to a single class in the dataset.

Specifically, five independent autoencoders are trained, each dedicated to learning the underlying representation of a specific transportation mode: bike, bus, car, foot, and metro. Our argument is that each autoencoder learns at its best to reconstruct the input of its own class. During prediction on unseen data, each autoencoder makes its own prediction by reconstructing the sample.

Given an unseen example s_i, we compute the reconstruction loss l, the mean squared error difference from the reconstructed instance $\bar{s_i}$. Our goal is to minimise the objective function in eq. (3.8) such that the AE with minimum loss is the predicted class. The motivation to use this algorithm is the expected ability to deal with imbalanced data due to its modular capacity.

$$l = \frac{1}{n}\sum_{i=1}^{n}(s_i - \bar{s_i})^2 \tag{3.8}$$

where s_i and $\bar{s_i}$ denote the original input and the reconstructed respectively.

3.4 Performance Evaluation

To comprehensively evaluate model performance, twelve experiments were conducted. These experiments were designed to investigate the impact of different feature sets and data splitting strategies on model accuracy.

- **Classifiers:** Three classifiers were employed: Random Forest, Convolutional Neural Networks, and an Ensemble of Autoencoders.

  - *RF Classifier:* As a well-established machine learning algorithm, random forest was selected as the baseline classifier. This choice is supported by prior research demonstrating its strong performance in mode detection tasks [77, 108].
  - *CNN Classifier:* Inspired by the studies in [31], we created a CNN model using the fixed-length instances as far the CNN requirement. Since the CNN input layer can accept an input sample in 3 dimensions [31]: length, width and depth (channels), our input sample is made-up of the fixed-length GPS instances. Each instance comprises of 4 channels of speed, acceleration, jerk and bearing rate stacked-up. the individual channel size has shape 1 x N. The CNN model therefore takes input instances, where each instance has the shape of 1 x N x 4 ($N = 100$ for raw point-level features, and $N = 10$ for feature statistics models).
  - *EAE Classifier:* The ensemble of autoencoders (EAE) model consists of an ensemble of autonomous autoencoders [56], where each autoencoder (AE) model is trained per class, in this case, 5 AEs, one for each mode.

- **Feature Sets:** Two feature sets were utilised:
  - *Raw Features:* This set comprises raw point-level features, including speed, acceleration, jerk, and bearing rate, for each of the 100 timesteps within an instance.
  - *Feature Statistics:* This set consists of ten descriptive statistics (mean, median, standard deviation, minimum, maximum, median absolute deviation, 25th percentile, 75th percentile, 85th percentile, and inter-quartile range) calculated for each of the four dimensions (speed, acceleration, jerk, and bearing rate) within each instance, resulting in 40 features per instance.

- **Data Splits:** Two data splitting strategies were employed:
  - *Temporal Split:* Data was split based on the period of data collection, using a 3-week trip data for training and 1 week for testing.
  - *Conventional Split:* A conventional 80:20 split was used to divide the data into training and testing sets.

The distribution of instances resulting from the temporal data split is depicted in Table 3.4, while that of the conventional split is presented in Table 3.5.

This experimental design resulted in a total of twelve experiments, allowing for a thorough investigation of the impact of different classifiers, feature sets, and data splitting strategies on model performance.

Model performance was evaluated using several well-known metrics, including precision, recall, F1-score, and the Area under the Receiver Operating Characteristic curve (ROC-AUC). ROC-AUC is a recommended evaluation metric for multi-class classification problems, as it is relatively insensitive to class imbalance and unequal classification error costs [48].

Table 3.4: Distribution of instances in the temporal data split

Travel mode	Train set		Test set		Total
	count	*percentage*	*count*	*percentage*	
foot	18,749	22.01	14,601	32.08	33,350
bike	3,482	4.09	1,398	3.07	4,880
bus	9,566	11.23	8,317	18.27	17,883
car	49,741	58.40	20,301	44.60	70,042
metro	3,634	4.27	904	1.99	4,538

3.5 Experimental Results

3.5.1 Cohort 1: Temporal Split (3-weeks/1-week)

This subsection presents the results of the set of experiments conducted using a temporal data split, where three weeks of data were utilised for model training and one week for testing. Tables 3.6 to 3.8 present the confusion matrices and corresponding classification reports, including recall and precision scores, for each of the models trained and evaluated on this temporal data split.

Despite having similar proportions of bike and metro modes in the training set, the models exhibited better predictive performance for bike mode compared to metro mode, as evidenced by the confusion matrices.

3.5.2 Cohort 2: Conventional Split (80/20)

This subsection presents the results of the experiments conducted using a conventional 80/20 data split, a common approach in machine learning research. This split divides the dataset into training and testing sets, with 80% of the data allocated for model training and 20% for model evaluation. This allows for a direct comparison of model performance with the results obtained using the temporal split (Cohort 1).

The results for all Cohorts 1 and 2 are summarised by different weighted classification metrics in Table 3.12. The decision regarding the train-test data split strategy has a significant impact, as

Table 3.5: Distribution of instances in conventional 80:20 split

Travel mode	Train set		Test set		Total
	count	*percentage*	*count*	*percentage*	
foot	26,712	25.55	6,638	25.40	33,350
bike	3,922	3.75	958	3.67	4,880
bus	14,284	13.66	3,599	13.77	17,883
car	56,003	53.56	14,039	53.71	70,042
metro	3,632	3.47	906	3.47	4,538

Table 3.6: Confusion matrix for RF classifier (raw features).

		Predicted class					
		Foot	Bike	Bus	Car	Metro	**Recall**
Actual class	Foot	8,706	37	83	5,746	29	0.60
	Bike	437	297	7	657	0	0.21
	Bus	1,573	20	375	6,295	54	0.05
	Car	1,738	107	359	18,046	51	0.89
	Metro	375	0	0	523	0	0.00
	Precision	0.68	0.64	0.46	0.58	0.00	

Table 3.7: Confusion matrix for CNN classifier (raw features).

		Predicted class					
		Foot	Bike	Bus	Car	Metro	**Recall**
Actual class	Foot	8,618	46	274	5,627	36	0.59
	Bike	586	298	6	496	12	0.21
	Bus	1,563	29	681	6,044	0	0.08
	Car	1,484	115	391	18,148	163	0.89
	Metro	361	8	9	446	80	0.09
	Precision	0.68	0.60	0.50	0.59	0.27	

Table 3.8: Confusion matrix for EAE classifier (raw features).

		Predicted class					
		Foot	Bike	Bus	Car	Metro	**Recall**
Actual class	Foot	11,106	574	677	1,698	546	0.76
	Bike	904	172	106	180	36	0.12
	Bus	4,897	657	530	2,133	100	0.06
	Car	7,668	2,448	1,532	8,301	352	0.41
	Metro	490	36	33	319	26	0.03
	Precision	0.44	0.04	0.18	0.66	0.02	

Table 3.9: Confusion matrix for RF classifier (features statistics).

		Predicted class					
		Foot	Bike	Bus	Car	Metro	**Recall**
Actual class	Foot	8,548	99	916	4,913	125	0.59
	Bike	334	478	67	519	0	0.34
	Bus	1,553	25	1,101	5,638	0	0.13
	Car	1,430	117	614	18,094	46	0.89
	Metro	350	7	12	513	22	0.02
	Precision	0.70	0.66	0.41	0.61	0.11	

Table 3.10: Confusion matrix for CNN classifier (features statistics).

		Predicted class					
		Foot	Bike	Bus	Car	Metro	**Recall**
Actual class	Foot	9,186	1	295	5,049	70	0.63
	Bike	676	253	0	469	0	0.18
	Bus	1,878	11	1,532	4,880	16	0.18
	Car	1,789	55	981	17,322	154	0.85
	Metro	357	0	4	389	154	0.17
	Precision	0.66	0.79	0.54	0.62	0.39	

Table 3.11: Confusion matrix for EAE classifier (Features statistics).

		Predicted class					
		Foot	Bike	Bus	Car	Metro	**Recall**
Actual class	Foot	6,510	0	131	7,397	563	0.45
	Bike	448	0	0	926	24	0.00
	Bus	857	0	49	7,296	115	0.01
	Car	989	0	61	19,170	81	0.94
	Metro	237	0	0	617	50	0.06
	Precision	0.72	0.00	0.20	0.54	0.06	

evident from the table. Specifically, the 80:20 split obtains better results. This improvement can be attributed to the way instances are distributed between the train and test sets. For instance, if a trip contains 10 instances, a random split might allocate 8 instances to the training set and 2 to the test set. Such a distribution can be counterintuitive for time series data, as the objective is to predict future trips based on past trip data.

To address this potential issue, we recommend employing a temporal split, where data is divided based on time periods. This approach ensures that the model is trained on past data and evaluated on future data, providing a more realistic assessment of generalisation performance and mitigating the potential for overfitting observed in the 80:20 split.

Table 3.12: Summary of the experimental results.

	Metric	CNN	EAE	RF
Cohort 1: Temporal data split	Accuracy	0.61	0.44	0.60
	Precision	0.60	0.47	0.58
	Recall	0.61	0.44	0.60
	F1-score	0.56	0.42	0.54
	ROC-AUC	0.72	0.60	0.76
	Accuracy	0.62	0.57	0.62
	Precision	0.62	0.51	0.59
	Recall	0.62	0.57	0.62
	F1-score	0.59	0.49	0.58
	ROC-AUC	0.76	0.63	0.78
Cohort 2: Conventional data split	Accuracy	0.97	0.74	0.79
	Precision	0.97	0.80	0.82
	Recall	0.97	0.74	0.79
	F1-score	0.97	0.74	0.77
	ROC-AUC	0.99	0.83	0.94
	Accuracy	0.80	0.57	0.82
	Precision	0.80	0.63	0.83
	Recall	0.80	0.57	0.82
	F1-score	0.79	0.45	0.80
	ROC-AUC	0.93	0.55	0.93

Precision, Recall, F1 and ROC-AUC scores are weighted averages

3.6 Summary

In this chapter, we comprehensively describe the datasets utilised in this research. We begin by describing the data collection process and then proceed to examine their key characteristics.

Building upon this foundation, we investigated the effectiveness of different machine learning models in inferring transportation modes from smartphone GPS data. Our methodology involved extracting key motion characteristics (speed, acceleration, jerk, and bearing change) from GPS trajectories and engineering two distinct feature sets: raw point-level features and descriptive statistics derived from these raw features. We then established a baseline model by evaluating the performance of various classifiers, including Random Forest, Convolutional Neural Networks, and an Ensemble of Autoencoders.

The study highlights the impact of the train-test data split strategy. While the conventional 80:20 split initially yielded higher accuracy, it can be potential to model overfit. The temporal split, which divides data based on time periods, provides a more robust evaluation by ensuring that the model is trained on past data and evaluated on future data, mitigating the risk of overfitting.

Chapter 4

Class-subspace Feature Selection

4.1 Introduction

The distribution of transportation modes within a city significantly influences large-scale urban transportation planning. To make informed decisions about transportation systems, authorities and urban planners require a comprehensive understanding of the diverse modes utilised by citizens. This knowledge is essential for developing effective policies that reduce travel time [31], alleviate traffic congestion, or promote the use of public transport in areas where it is most needed [43]. It can be leveraged to refine trajectory recommendation systems by tailoring recommendations based on identified transportation modes used on specific paths or routes.

However, conventional tools such as household surveys and telephone interviews have fallen short of capturing the intricate patterns of human mobility. These techniques are often resource-intensive and inherently susceptible to biases or under-reporting. This limitation accentuates the urgent need for automated methods to acquire travel data accurately. Fortunately, mobile crowd sensing emerges as a promising solution for large-scale, automated data collection [2, 107, 108, 114]. This technology can be leveraged to monitor diverse human activities, including transportation, with unprecedented detail and granularity.

Fuelled by the widespread adoption of smartphones equipped with sophisticated sensor capabilities and advancements in machine learning, transportation mode detection research has gained significant traction in recent years. This burgeoning field focuses on harnessing data from various sources, such as GPS, accelerometers, and gyroscopes, to accurately identify individuals' travel modes. A plethora of TMD approaches have been developed, ranging from statistical models [12, 146] and rule-based expert systems [18, 115, 147] to more recent innovations in machine learning and deep learning architectures [31, 44, 108].

This study makes several significant contributions to the field of transportation mode detection solely using GPS trajectory data:

- Trip pre-processing and data quality handling: we conduct a thorough trip pre-processing and data quality handling to clean and prepare the GPS trajectory data. This contribution was presented in sections 3.2.1 and 3.2.2 of Chapter 3.

- we research using time-to-frequency feature transformation to extract more informative features from GPS trajectory data.
- We propose the *Class-Subspace* feature selection method that utilises Shapley values to identify and retain feature subsets based on attribution to class prediction.
- We evaluate our work using two real-world GPS trajectory datasets.

This chapter is organized as follows. We begin in Section 5.2 by the reviewing related work. Next, Section 4.3 details the experimental procedures employed in this study. In Section 4.4, we outline the classification algorithms used, followed by a discussion of the evaluation metrics employed to assess the performance of the different models in Section 5.4. Section 5.5 introduces sequential feature selection as a baseline approach. Subsequently, Section 4.7 introduces our proposed Class-Subspace Feature Selection method. The results obtained from our experiments are presented in Section 4.8 and analyzed in Section 4.9. Finally, Section 4.10 concludes the chapter by summarizing the study's main findings and outlining potential directions for future research.

4.2 Related Work

GPS trajectory data comprises a sequence of location points, each represented as a tuple containing spatial and temporal information, typically including longitude, latitude, and timestamp. Data preprocessing is essential for extracting meaningful motion features such as speed, acceleration, and direction changes from the raw data. Trip segmentation is a crucial step in the preprocessing process, involving dividing the trip into segments.

Zheng et al. [160] introduced a change-point-based approach, utilising a change-point algorithm to discriminate between walking and non-walking segments. They established thresholds for maximum velocity ($1.8\,\mathrm{m/s}$) and acceleration ($0.6\,\mathrm{m/s^2}$) for this purpose. This method is based on the following two key considerations: (1) individuals typically walk when transitioning between different modes of transport (e.g., walking from their home to a bus stop, or from a train station to their workplace); and (2) each trip begins and ends with a walking segment. This approach has been widely adopted in subsequent studies [162, 166, 68].

Conversely, a substantial body of research has employed the stop-detection approach. This methodology focuses on identifying stationary locations. By identifying and labelling these points as "stops", the method effectively distinguishes them from locations where the user is in motion, which are labelled as "moving". Two primary categories of stop-detection-based methods are prevalent in the literature [152]: rule-based and clustering-based. Rule-based methods actively determine "stops" by analysing the duration a user spends within a specific area ([31, 9, 111] or by evaluating their movement speed within a defined time interval [108, 90, 111]. In contrast, clustering-based methods ([70, 42] identify stop points by analysing the spatial coordinates of location points. Despite the growing use of these methods, no study has explicitly examined the direct impact of data segmentation methods.

When identifying features for mode detection, two primary categories emerge. Kinematic features, which describe the inherent motion characteristics, encompass attributes such as speed, acceleration, and their statistical derivatives. These features are widely utilised in mode detection, as evident in Table 5.1, [157, 151, 20, 14]. Conversely, geospatial context features characterise the interplay between the movement and its surrounding environment, encompassing spatial and temporal relationships [63]. These features often leverage data derived from GIS and transportation networks, such as proximity to bus stations. These features, which capture the unique characteristics of different travel modes [152], are subsequently utilised in various downstream tasks, including data cleaning and subsequent analysis.

Feature selection constitutes a critical step in the development of effective learning models. This crucial task can be broadly categorised into three distinct approaches [78]: filter methods, wrapper-based methods, and embedded methods. Filter methods are independent of any learning algorithms. evaluate the relevance of features based on their statistical relationship with the target variable. Wrapper methods rely on the performance of the learning algorithm to evaluate the relevance of features by iteratively selecting subsets of features and training a model on each subset. Embedded methods integrate feature selection directly into the structure of a learning algorithm, utilising its inherent properties (e.g., regularisation models) [78].

While prior research has recognised the potential of feature selection in TMD, its specific significance remains largely unexplored. This gap necessitates further investigation. The study in [46] explored the efficacy of wrapper and information retrieval methods for selecting optimal feature subsets, demonstrating improved performance. Nevertheless, a key limitation of their work lies in the failure to conduct a comparative analysis with other prominent feature selection techniques. The proposed framework in [129] incorporates feature selection based on intra/inter-class feature distances. However, the description of this selection approach lacks clarity and specificity. Furthermore, the study fails to provide empirical evidence demonstrating an improvement in results attributable to the feature selection module.

Alazeb et al. [6] employed a recursive feature elimination method to select the most informative features. However, a significant limitation arises from the inability to definitively attribute the observed performance gains solely to the feature selection process, owing to the concomitant implementation of feature selection and imbalance learning techniques within their proposed framework. Isolating the impact of feature selection is crucial for a thorough understanding of its effectiveness and for establishing best practices in TMD.

Despite the growing importance of feature selection in machine learning, its specific impact on TMD accuracy remains largely unexplored. Existing studies have either not explicitly investigated feature selection or have not adequately isolated its contribution to performance gains. This gap in knowledge motivates the need for further research to understand the role of feature selection in optimising TMD models.

Table 4.1: Summary of TMD studies from GPS data.

Study	Population	Duration	Features	Classifier	Modes	Accuracy (%)
[160]	45	6 months	Distance, velocity & acceleration stats.	DT	4	72.8
[157]	69	> 5 years	Speed, acceleration & jerk.	DNN	5	91.4
[44]	69	> 5 years	Distance, velocity, & deep extracted features	LR	7	67.9
[148]	69	> 5 years	Velocity, acceleration turning angle.	XGB	6	90.8
[14]	2	3 days	Speed statistics.	HMM	4	78
[20]	81	2 weeks	Velocity & acceler-ation stats.	SVM	6	88
[31]	69	> 5 years	Speed, acceleration jerk, bearing rate.	CNN	5	84.8
[123]	1.6m users	3 years	Auto-extracted (deep) features.	LSTM	5	81.37
[145]	203	10 months	Spped, acceleration & heading change.	Gaussian	6	92
[108]	227	1 month	Spped, acceleration statistics.	RF	5	85
[151]	410	10 months	Spped statistics & location data.	RF	5	93
[63]	139	12 months	Distance, acceleration, geospatial context	RF	7	93.0
[111]	20	5 months	Distance, speed & acceleration stats.	RF	6	91.2
[112]	91	12 months	Speed, acceleration bearing rate	RF	5	93.94
[156]	69	> 5 years	longitude, latitude, velocity, acceleration frequency–inverse.	Autoencoder	7	68.26
[98]	69	> 5 years	Speed, acceleration, jerk, bearing rate.	CNN	5	83.3
[47]	69	> 5 years	Velocity,acceleration, distance, heading, and curvature.	CNN	5	95.16

4.3 Data Preprocessing

This chapter utilises the two GPS trajectory datasets detailed in Section 3.1. Building upon the rigorous data preprocessing steps outlined in sections 3.2.1 and 3.2.2, which includes trip preprocessing and the creation of fixed-size instances, this chapter utilises the resulting instances. The distribution of instances by transportation mode is depicted in Figure. 4.1.

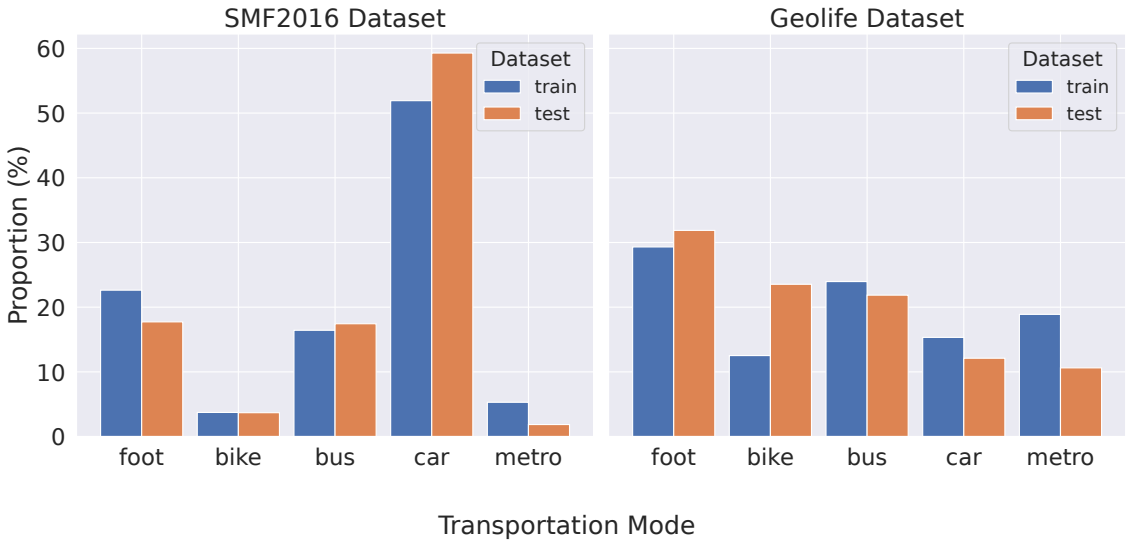


Figure 4.1: Instance distribution by transportation mode.

4.3.1 Time Domain Features

Each instance comprises four dimensions of point-level motion characteristics: speed, acceleration, jerk, and bearing rate, observed over 100 consecutive time steps. This standardised representation provides a consistent and structured input format for subsequent feature engineering, encompassing both time-domain and frequency-domain features. This enables the extraction of spectral characteristics of the trajectories as described in the subsection that follows.

4.3.2 Frequency Domain Features

To capture the spectral characteristics of the trajectories, we employ the Fast Fourier Transform (FFT) algorithm [101] to transform all feature dimensions of instances from the time domain to the corresponding frequency spectrum in the frequency domain. For real-valued input, the FFT produces a symmetric spectrum in which the values for positive frequencies are the complex conjugates of the values for negative frequencies. In this case we considered only the positive frequency values for the frequency features to avoid feature duplication.

Figure 5.4 illustrates an example of a single instance in both the time and frequency domains, highlighting the transformation of the time-series data into its corresponding frequency spectrum.

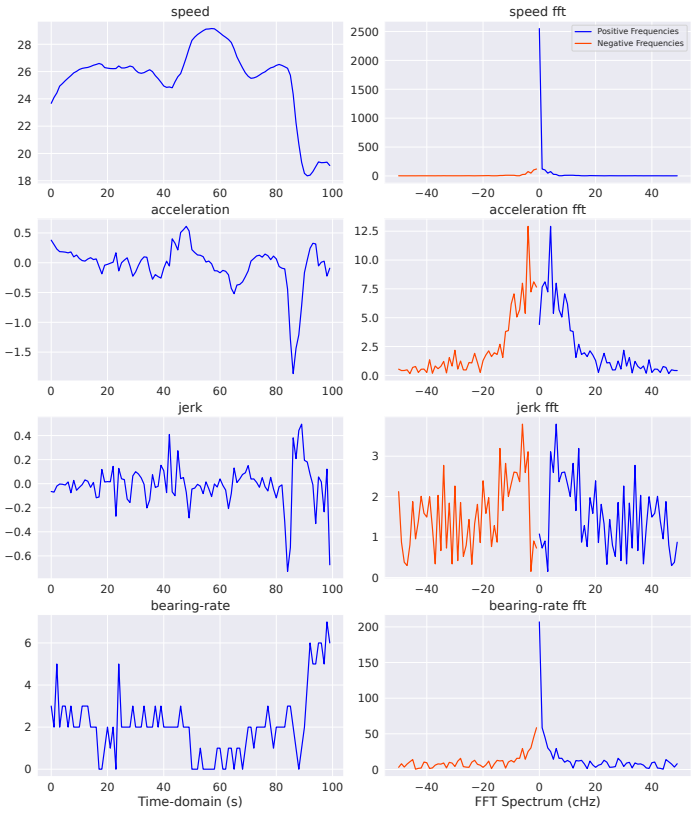


Figure 4.2: An instance's time and frequency domains.

4.4 Classification Algorithms

This study employs four widely-used machine learning classifiers: Decision Trees, Random Forest, Support Vector Machines, and XGBoost. For a detailed overview of these algorithms, please refer to Section 2.4.

4.5 Evaluation Metrics

Evaluation metrics are quantitative measures used to assess the performance and effectiveness of a machine learning model. They are typically used to compare the performance of different models or to evaluate the performance of a model over time. Within the classification domain, the widely prevalent metrics for performance evaluation are *accuracy* and *error rate.* In this section, we describe the most common performance metrics in classification.

4.5.1 Confusion Matrix

Each example data sample can yield one of four potential outcomes in a binary classification task involving two classes. If the sample is genuinely positive and the classification algorithm correctly predicts it, it is termed a *true positive (TP)*. Conversely, if the algorithm incorrectly predicts it as negative, it is referred to as a *false negative (FN)*. Similarly, if the sample is truly negative and

the classifier accurately predicts it as negative, it is labelled as a *true negative (TN)*. If, however, the algorithm erroneously classifies it as positive, it is designated as a *false positive (FP)*. These four distinct outcomes are precisely captured within a structure known as a *contingency table*, or *confusion matrix*.

The confusion matrix is a table summarising a classification algorithm's performance on the test set [48]. It is typically divided into four cells, corresponding to the four possible outcomes.

Table 4.2: Confusion Matrix.

		Predicted Class		
		Positive	*Negative*	*Class Support*
Actual Class	*Positive*	True Positive (TP)	False Negative (FN)	**P**
	Negative	False Positive (FP)	True Negative (TN)	**N**

4.5.2 Accuracy

Accuracy is the proportion of test instances correctly classified by the model. The accuracy score can be calculated from the confusion matrix following eq. (5.8).

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{4.1}$$

4.5.3 Precision

Precision is the proportion of test instances that are classified as positive by the model that are positive. Using the entries of the confusion matrix, this can be calculated as:

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

4.5.4 Recall

Recall evaluates the proportion of test instances that are actually positive and are classified as positive by the model. This is calculated using eq. (5.10).

$$Recall = \frac{TP}{(TP+FN)} \tag{4.3}$$

4.5.5 F1 score

There is always a trade-off between a machine learning model's precision and recall scores. An *F1 score* combines the precision and recall scores into a single value, making it a useful measure for evaluating the performance of a classification model. It is the harmonic mean of the precision and the recall scores, calculated as:

$$F1 = \frac{2 \cdot (Precision \times Recall)}{Precision + Recall} \tag{4.4}$$

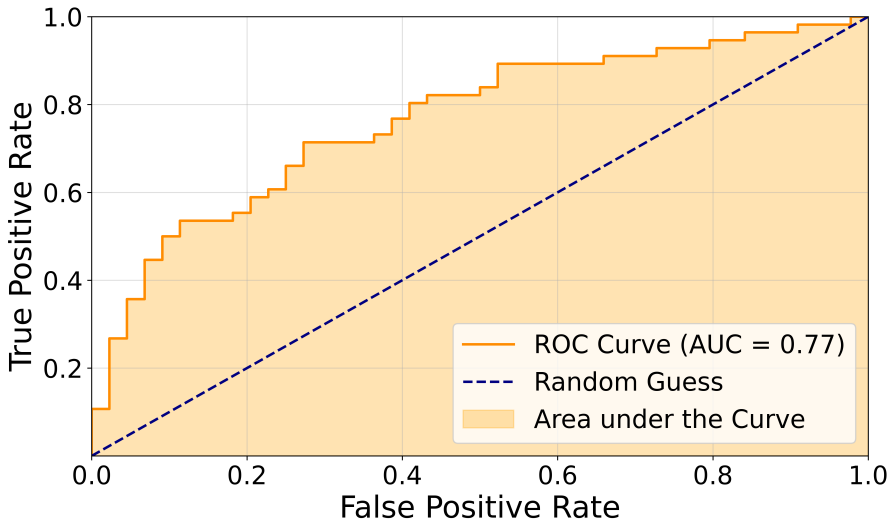


Figure 4.3: Area under the Receiver Operating Characteristic (ROC) Curve.

4.5.6 Receiver Operating Characteristics

So far, all the evaluation metrics mentioned above rely on the value of the possible outcomes from the confusion matrix. They are, therefore, sensitive to the class distribution of positive and negative classes. We can see from the confusion matrix (Table 5.5) that the class distribution is the relationship of the positive class (upper) row to the negative (lower) row [48] (see the "Class Support" column in the confusion matrix Table 5.5). A somewhat better metric insensitive to class distribution is the Receiver Operating Characteristics (ROC). ROC is a graphical representation and a performance measurement tool used in binary classification problems.

The ROC curves are plotted using two performance metrics: the *true positive rate (TPR)* - proportion of positive instances that are correctly classified, and *false positive rate (FPR)* - the proportion of negative instances that are incorrectly classified. Figure 5.5 illustrates a typical ROC curve, showing the relationship between TPR and FPR at various classification thresholds. The Area Under the Curve (AUC) is a common metric for summarising the performance of a classifier based on its ROC curve.

These curves are created by varying the classification threshold and plotting the TPR against the FPR at each threshold. The classification threshold is a value used to decide whether an instance is classified as positive or negative. Lowering the classification threshold will classify more instances as positive, thus increasing both TPR and FPR. A higher area under the ROC curve (AUC) generally indicates a better-performing model, with 1 representing perfect discrimination and an AUC of 0.5 representing just a random classifier.

4.6 Feature Selection

Feature selection is a critical step in machine learning pipelines aimed at identifying and retaining only relevant features while discarding extraneous ones. The primary goal of feature selection is to identify and select the most relevant and informative subset of features from a more extensive set of variables, aiming to improve model performance and interpretability and reduce computational complexity [72]. Previous studies in TMD from GPS data primarily focused on constructing features in the time domain. While this approach has yielded effective learning models, there is a tendency to overlook the potential of frequency-domain features [11, 68].

We use the ten (10) descriptive feature statistics [90] calculated for every feature of an instance for the time-domain features (mean, median, standard deviation, minimum, maximum, MAD, 25^{th}, 75^{th}, 85^{th} percentiles, and inter-quartile range). For the frequency-domain features, we consider the indexes of the top 10 frequency components for each feature of an instance. Thus, each instance consists of 80 features: time-domain 10 x 4 features and frequency-domain 10 x 4 features.

4.6.0.1 Sequential Forward Floating Selection

After generating the pool of features, we applied the *Sequential Forward Floating Selection (SFFS)* [52, 102] technique, a wrapper-based method for feature selection. SFFS begins with an empty feature subset. In the forward selection phase, features are iteratively added to the subset one at a time, with each addition leading to model training and evaluation. The floating operation in SFFS involves removing features from the currently chosen subset if they no longer contribute significantly to the model's performance, even if they were previously selected. The process continues until no further improvement in the evaluation metric, such as accuracy or error rate, is observed on a validation set. This study uses the area under the curve - receiver operating characteristics (ROC-AUC) as the evaluation criterion.

4.7 Proposed Method

This section introduces the proposed feature selection method rooted in Shapley values. We firstly provide a concise overview of Shapley values and their relevance to feature importance assessment.

4.7.1 The Shapley Values

Drawing upon the principles of cooperative game theory, the *Shapley values* emerges as a powerful tool for assessing the contributions of individual features towards a machine learning model predictions [84]. The Shapley value of a feature quantifies its average marginal contribution to the overall prediction across all possible combinations of feature values. It measures the significance

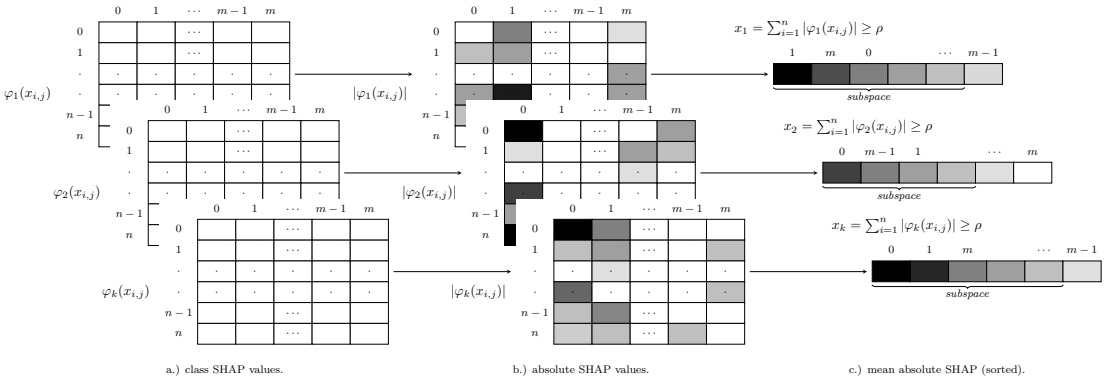


Figure 4.4: An illustration of the Class-Subspace selection process.

of each feature by evaluating its impact on the model prediction across all possible combinations of feature values.

Lundberg, Erion, and Lee (2018) [85] introduced a computationally efficient and theoretically sound approach called SHAP (short for *SHapley Additive exPlanations*) for explaining the output of a machine learning model using insights from game theory. SHAP values is a well-established feature importance technique that quantifies the attribution of each feature to the model prediction by considering all possible marginal contributions of the features. The SHAP value of a feature i, denoted ϕ_i, is calculated following equation 5.12, where N represents the set of all input features and S denotes the feature subset. $E[f(X)]$ represent the conditional expectations of the models with ($X_{S\cup i}$) and without (X_S) feature i, respectively. Despite its computational cost, SHAP has demonstrated commendable properties, including fairness and consistency [158], in assigning importance scores to individual features.

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!} \{E[f(X)|X_{S\cup i} = x_{S\cup i}] - E[f(X)|X_S = x_S]\} \tag{4.5}$$

4.7.2 Class-Subspace Selection

Traditional wrapper-based feature selection algorithms, such as SFFS, do not evaluate each feature in isolation. Instead, they train a model using different combinations of features and determine which combinations yield the best performance (e.g., highest accuracy). This approach prioritises features that collectively enhance the model's predictive power. To address the challenge of identifying a minimal subset of features that adequately capture the classification boundaries for each unique class, we propose the Class-Subspace selection technique.

Our approach involves training an initial classifier and computing the SHAP values $\phi(x_{i,j})$ for all features i and instances j in the training set D. Subsequently, for each unique class k, we evaluate the contribution of each feature to the model prediction of that class by computing the mean absolute SHAP values. These values are then sorted in non-increasing order of magnitude

to identify the most influential features for predicting the class. The minimum subset of features comprising the top contributing values whose sum is larger than the contribution threshold (ρ) is selected.

Figure. 5.6 visually depicts the steps involved in this process. The chosen subset trains a base model specialised in detecting instances belonging to that class. We create k such base models, one for each unique class. Finally, we construct an ensemble of these base models. Algorithm 1 provides the pseudo-code for the Class-Subspace selection technique. In this case, we used $\rho = 0.6$ for features contributing to each class prediction.

Algorithm 1: Class-Subspace

Input: Dataset $D = (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), .., (\mathbf{x}_m, y_m)$;
Number of classes K;
Base learning algorithm T;
Subspace proportion ρ.

Process:

Compute $\phi(x_{i,j}), \quad \forall x_{i,j} \in D$ `// SHAP of input features for all instances`

for $k = 1$ *to* K **do**

  Compute $\mathbf{x}_k = \sum_{i=1}^{n} |\phi(x_{i,j})|$ `// feature vector`

  Sort $\mathbf{x}_k$ in non-increasing order

  Select subspace x_i for which $\mathbf{x}_k$ adds up to ρ

  Fit T using x_i in 5

end

Output: $H(x)$, an ensemble of K base learners

4.8 Experiments and Results

We train the four classification algorithms mentioned in Section 4.4. A comparative analysis of the classification results shows that RF and XGB demonstrated superior performance on both datasets when used with the 80 features initially generated. This is presented in Table 4.3. RF and XGB achieved higher ROC-AUC scores of 75% and 88%, respectively.

We further investigate the impact of different feature combinations to identify the most relevant features through training RF and XGB models with various feature subsets in the following settings:

- *Time-domain:* These models used the 40 features derived from the temporal characteristics of the GPS trajectories (e.g., speed, acceleration, bearing rate) only.
- *Frequency-domain:* These models used the 40 frequency-domain features obtained from the frequency spectrum of the Fourier transformation of time features.
- *Combined:* these models are trained using entire 80 features (40 time + 40 frequency).

Table 4.3: Classification Results for SMF2016 and Geolife Datasets

SMF2016 Classification Results												
Transportation	Precision				Recall				F1-score			
Mode	RF	SVC	DT	XGB	RF	SVC	DT	XGB	RF	SVC	DT	XGB
Foot	0.60	0.57	0.59	0.61	0.93	0.94	0.73	0.93	0.73	0.71	0.65	0.73
Bike	0.97	0.02	0.39	0.88	0.22	0.03	0.23	0.36	0.36	0.02	0.28	0.51
Bus	0.13	0.16	0.18	0.18	0.06	0.07	0.19	0.11	0.08	0.10	0.18	0.14
Car	0.74	0.73	0.71	0.75	0.80	0.69	0.64	0.77	0.77	0.71	0.67	0.76
Metro	0.00	0.28	0.04	0.00	0.00	0.02	0.09	0.00	0.00	0.03	0.06	0.00
		RF				SVC				DT				XGB	
ROC-AUC **		0.75				0.67				0.64				0.75	
Geolife Classification Results												
Transportation	Precision				Recall				F1-score			
Mode	RF	SVC	DT	XGB	RF	SVC	DT	XGB	RF	SVC	DT	XGB
Foot	0.68	0.63	0.65	0.69	0.93	0.83	0.69	0.91	0.79	0.72	0.67	0.79
Bike	0.84	0.61	0.67	0.84	0.68	0.53	0.53	0.69	0.75	0.57	0.59	0.75
Bus	0.71	0.66	0.51	0.71	0.70	0.66	0.56	0.70	0.70	0.66	0.53	0.70
Car	0.67	0.57	0.37	0.65	0.46	0.34	0.46	0.51	0.55	0.42	0.41	0.57
Metro	0.76	0.51	0.51	0.76	0.53	0.40	0.42	0.54	0.63	0.45	0.46	0.63
		RF				SVC				DT				XGB	
ROC-AUC **		0.88				0.83				0.72				0.88	

**metric is a weighted average (global).

- *SFFS-chosen:* These models are trained using the feature subset selected by the SFFS selector for each classifier and dataset.
- *SFFS-common:* These models are trained using the SFFS-selected features by each classifier standard to both datasets.
- *Class-Subspace:* models created using our proposed feature selection subsets.

In Table 4.4 is the summary of the number of SFFS-chosen and common features for each algorithm and dataset. Table 4.5 presents the results obtained for each feature subset and dataset. While yielding only marginal performance gains, our proposed method consistently outperforms other techniques. The only exceptions are the RF model's ROC-AUC score on SMF2016 and the precision and ROC-AUC scores on Geolife. This demonstrates the effectiveness of our method in identifying the most relevant features, ultimately leading to improved classification performance.

Table 4.4: SFFS feature Selection by each classifier across the datasets.

Classifier	Number of SFFS chosen Features SMF2016	Geolife	Number of Features in Common
RF	24	30	10
XGB	55	42	30

Table 4.5: Model evaluations using different feature combinations.

		Precision*		Recall*		F1-score*		ROC-AUC*		Accuracy	
		RF	XGB	RF	XGB	RF	XGB	RF	XGB	RF	XGB
SMF2016	*Time-domain*	0.54	0.61	0.56	0.63	0.55	0.61	0.68	0.70	0.56	0.63
	Freq-domain	0.50	0.50	0.55	0.57	0.52	0.52	0.64	0.74	0.55	0.57
	Combined	0.57	0.62	0.60	0.65	0.58	0.62	0.75	0.75	0.60	0.65
	SFFS-chosen	0.59	0.61	0.62	0.64	0.60	0.61	**0.79**	**0.78**	0.62	0.64
	SFFS-common	0.58	0.62	0.61	0.64	0.58	0.62	0.75	0.76	0.61	0.64
	*Class-Subspace***	**0.61**	**0.64**	**0.68**	**0.68**	**0.62**	**0.64**	0.76	**0.78**	**0.68**	**0.68**
Geolife	*Time-domain*	0.72	**0.73**	0.71	**0.72**	0.70	**0.72**	0.88	0.88	0.71	**0.72**
	Freq-domain	0.43	0.41	0.40	0.40	0.35	0.38	0.71	0.69	0.40	0.40
	Combined	0.72	**0.73**	**0.72**	**0.72**	0.71	**0.72**	0.88	**0.89**	**0.72**	**0.72**
	SFFS-chosen	**0.73**	0.72	**0.72**	0.71	**0.72**	0.71	**0.89**	0.88	**0.72**	0.71
	SFFS-common	0.70	0.71	0.69	0.70	0.68	0.70	0.87	0.88	0.69	0.70
	*Class-Subspace***	0.72	0.72	**0.72**	**0.72**	**0.72**	**0.72**	0.88	0.88	**0.72**	**0.72**

*Weighted average. **Our proposed feature selection method.

Finally, we determined the predicted segments and trips by combining instances and segments through a majority voting strategy. The resulting segment-trip predictions are shown in Table 4.6.

4.9 Discussion

The results presented in Table 4.5 underscore the significance of time-domain features for accurate mode detection. This trend is particularly evident in the Geolife dataset, where time-domain features consistently outperform frequency-domain features. While the benefits of frequency-domain features are less pronounced in the SMF2016 dataset, they still contribute to the improvements in overall performance. The combined features models demonstrate improvements in overall performance, further emphasising the value of incorporating both time-domain and frequency-domain features. This highlights the potential benefits of incorporating time-domain and frequency-domain features in mode detection models. We experimented with the influence of standard features selected by SFFS from each dataset, as shown in Table 4.4. However, this did not lead to improved classification outcomes.

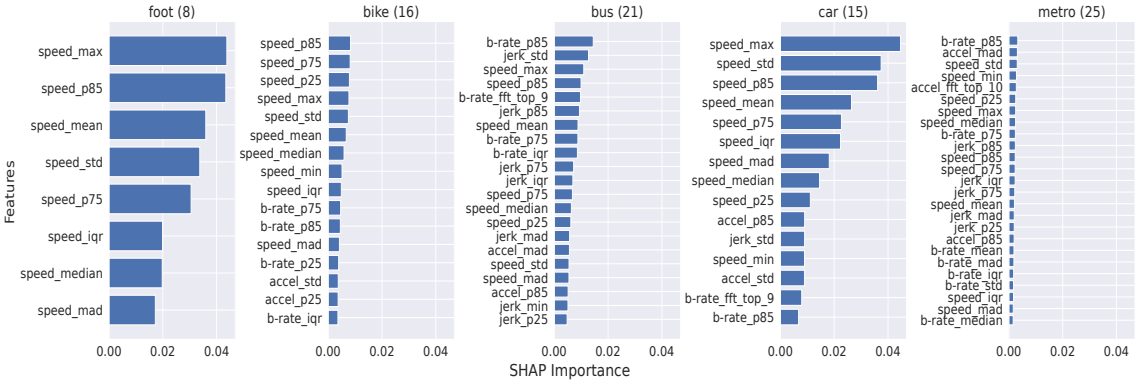
Our proposed method consistently improves classification results, except for the few cases in ROC-AUC and precision scores. Perhaps the most exciting part is its flexibility to select various features per class. This selection is influenced by the individual feature's contribution to the overall model. While some classes, such as foot, can be accurately predicted with only a few features, others, such as the *metro*, require more features but still struggle for correct prediction. One possible explanation is that the models struggle to find the optimal feature subset to correctly discriminate the *metro* class. Nevertheless, our method's ability to tailor feature selection to each class demonstrates its adaptability and potential for enhancing classification performance.

Figure. 4.5 illustrates the feature subsets selected by our proposed method for individual classes using the SMF2016 dataset with both RF and XGB classifiers. The number in parentheses indicates the features selected for each class, using a $\rho = 0.6$ threshold. This significantly reduced the feature set, focusing on the most informative features for each class. We anticipate the proposed feature selection method will improve discrimination between bus and car classes, addressing the higher misclassification rate. This expectation is based on the observation that the most important predictors for buses (jerk and bearing rate features) differ substantially from those for cars (primarily dominated by speed features).

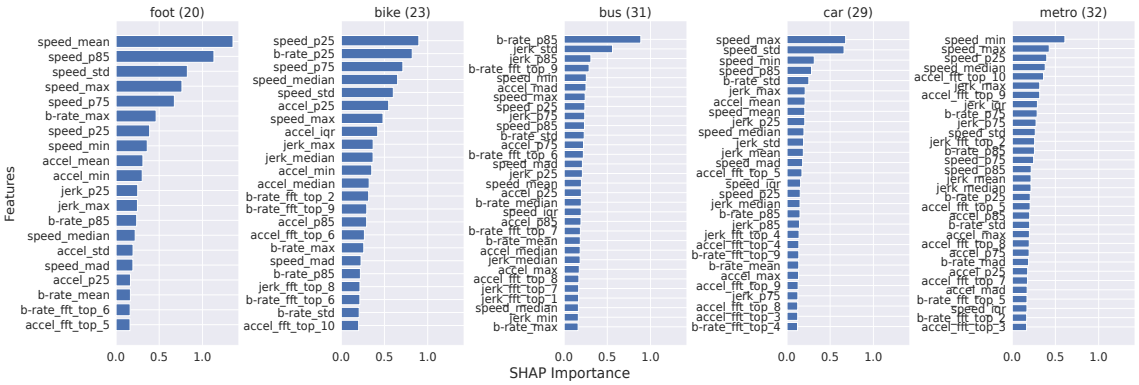
Table 4.6: Segment and trip prediction results (%).

		Precision*		Recall*		F1-Score*		Accuracy	
		RF	XGB	RF	XGB	RF	XGB	RF	XGB
SMF2016	Segment	65.2	71.3	69.4	70.6	60.9	63.6	69.4	70.6
	Trip	75.5	76.1	79.5	79.0	74.1	74.7	79.5	79.0
Geolife	Segment	74.4	75.4	73.9	74.9	73.8	74.8	73.9	74.9
	Trip	79.0	80.5	76.6	78.8	76.9	79.1	76.6	78.8

*weighted averages.



(a) Subspace selection with RF.



(b) Subspace selection with XGB.

Figure 4.5: Features selected by subspace algorithm for SMF2016 using RF **(a)** and XGB **(b)** for $\rho = 0.6$.

4.10 Summary

This paper presents a methodology that exclusively leverages GPS trajectory data to extract motion features for accurate transportation mode classification. We initially constructed time-domain motion features and subsequently enriched the feature set by transforming time-domain features into the frequency domain. To optimise feature selection, we explored the impact of various feature combinations, including the sequential forward selection technique. Finally, we proposed a novel feature selection method that relies on the concept of game theory to select the most relevant predictors for individual classes in the problem. Our method consistently outperforms other methods explored in our work, though improvements are potentially needed. We evaluated our work using two real-world GPS trajectory datasets. Overall, the final segment prediction results achieved 70.6% accuracy in SMF2016 and 78.8% in Geolife for XGB. Trip prediction accuracy achieved at least 79% in SMF2016 for both classifiers and 78.8% in Geolife for XGB.

Chapter 5

Class-Subspace Feature Selection for Transportation Mode Detection

This chapter addresses a primary objective of this thesis: the development of specialised feature selection techniques to resolve the challenges of overlapping characteristics in transportation mode classification. While the preceding chapters established the fundamental datasets and preprocessing requirements for GPS trajectory analysis, this chapter introduces a more sophisticated methodological layer designed to handle the class-similarity inherent in transportation mode data. By shifting the focus from global feature optimisation to class-specific subspaces, this work provides a targeted solution to the classification bottlenecks identified in the broader Research Objectives of this study.

The research presented here introduces a novel Class-Subspace feature selection framework utilising Shapley values to isolate bespoke feature subsets for individual transportation modes. By integrating time-domain and frequency-domain features obtained through Fast Fourier Transform, this approach demonstrates how class-aware selection can capture discriminative patterns that are often masked by conventional techniques. This study, which utilises both the SenseMyFEUP and Geolife datasets, has been published in the *International Journal of Data Science and Analytics* under the title *"From Attribution to Selection: Harnessing SHAP values for Class-Subspace Feature Selection in Transportation Mode Detection"* [89].

5.1 Introduction

Accurate and detailed understanding of complex human mobility patterns, particularly transportation mode choices, is fundamental for effective urban planning and policy development. Traditional data collection approaches, however, face inherent limitations in capturing the necessary scale and granularity for such insights. In this context, mobile crowdsensing [2, 107, 108, 114] has emerged as a promising approach for large-scale, automated data collection. This technology enables researchers and planners to monitor diverse human activities, including transportation, with a higher degree of details and granularity, thereby facilitating informed decision-making.

The capabilities of mobile crowdsensing have been substantially advanced by the widespread adoption of GPS-enabled devices [22, 23], including smartphones with sophisticated sensor technologies, in parallel with significant progress in machine learning methodologies. As a result, transportation mode detection (TMD) has become an increasingly active research field in recent years. TMD leverages data from various sources—such as GPS, accelerometers, and gyroscopes—to accurately identify individuals' travel modes. Researchers have developed diverse TMD approaches, ranging from statistical models [12, 146], rule-based expert systems [18, 147, 115] to more recent innovations incorporating machine learning and deep learning architectures [31, 108, 44].

Despite the exploration of diverse sensor data in existing literature, this study focuses exclusively on TMD using GPS trajectory data. This approach is justified by the practical and technical advantages of GPS for wide-coverage data collection. Specifically, GPS data is considered an efficient sensor modality, available not only through ubiquitous consumer smartphones but also via dedicated low-power logging devices. Furthermore, GPS offers rich spatial and temporal movement information highly effective for mode recognition.

The study addresses the following key research questions within the context of TMD:

- *What preprocessing workflow is necessary to address the quality of crowdsourced GPS trajectory data for transportation mode detection?*
  To address this, we implement a multi-stage preprocessing and data quality handling pipeline to clean and prepare the GPS trajectory data, mitigating common issues such as outliers, noise, and user annotation errors.

- *How effective are frequency-domain features, derived from GPS trajectories, in enhancing transportation mode detection?*
  To address this, we employ Fast Fourier Transform (FFT) to extract frequency-domain features from GPS trajectories, complementing time-domain features.

- *How can the most relevant features for each transportation mode be identified to foster the development of robust and interpretable models?*
  We propose the *Class-Subspace* feature selection method that utilises Shapley values to identify and retain feature subsets based on attribution to class prediction.

We evaluate the performance of our approach on two real-world GPS trajectory datasets collected from smartphones and GPS loggers in two different cities.

This chapter is organised as follows. Section 5.2 provides a comprehensive overview of related literature. Section 5.3 presents a detailed description of the experimental procedures employed in this study. **??** introduces our proposed feature selection method. The results are presented in Section 5.7, followed by a discussion of their implications in Section 5.8. Finally, Section 5.9 concludes the paper by summarising the main findings and outlining directions for future research.

5.2 Related Work

GPS trajectory data comprises a sequence of location points, each represented as a tuple containing spatial and temporal information, typically including longitude, latitude, and timestamp. Data preprocessing is essential for extracting the raw data's meaningful motion features, such as speed, acceleration, and direction changes. Trip segmentation is a crucial step in preprocessing, involving dividing the trip into segments.

Zheng et al. [160] introduced a change-point-based approach, utilising a change-point algorithm to discriminate between walking and non-walking segments. They established thresholds for maximum velocity ($1.8\,\mathrm{m/s}$) and acceleration ($0.6\,\mathrm{m/s^2}$) for this purpose. This method is based on the following two key considerations: (1) individuals typically walk when transitioning between different modes of transport (e.g., walking from home to a bus stop, or from a train station to workplace); and (2) each trip begins and ends with a walking segment. This approach has been widely adopted in subsequent studies [162, 166, 68].

Alternatively, a substantial body of research has employed the stop-detection approach. This methodology focuses on identifying stationary locations. By identifying and labelling these points as "stops", the method effectively distinguishes them from locations where the user is in motion, which are labelled as "moving". Two primary categories of stop-detection-based methods are prevalent in the literature [152]: rule-based and clustering-based. Rule-based methods determine "stops" by analysing the duration a user spends within a specific area ([31, 9, 111] or by evaluating their movement speed within a defined time interval [108, 90, 111]. In contrast, clustering-based methods ([70, 42] identify stop points by analysing the spatial coordinates of location points. These diverse segmentation approaches highlight an inconsistency in preprocessing strategies across the literature, demonstrating that segmentation remains a non-standardised step in TMD pipelines.

Two primary categories emerge when identifying features for mode detection. Kinematic features, which describe the inherent motion characteristics, encompass attributes such as speed, acceleration, and statistical derivatives. These features are widely utilised in mode detection, as evident in Table 5.1, [157, 151, 20, 14]. Conversely, geospatial context features characterise the interplay between the movement and its surrounding environment, encompassing spatial and temporal relationships [63]. These features often leverage data from GIS and transportation networks, such as proximity to bus stations. These features, which capture the characteristics of different travel modes [152], are subsequently utilised in various downstream tasks, including data cleaning and subsequent analysis.

Table 5.1: Summary of transportation mode detection studies from GPS data.

Study	Population	Duration	Features	Modes	Classifier	Accuracy (%)
[108]	227	1 month	speed & acceleration stats.	bike, bus, car, foot, metro.	RF	85
[31]	69	> 5 years	speed, acceleration, jerk, bearing rate.	bike, bus, car, foot, train.	CNN	84.8
[44]	69	> 5 years	distance, velocity & deep extracted features	bike, bus, car, foot, subway, taxi, train.	LR	67.9
[160]	45	6 months	distance, velocity, acceleration stats.	bike, bus, car, foot.	DT	72.8
[111]	20	5 months	distance, speed & acceleration stats.	bike, bus, foot, car, train.	RF	91.2
[157]	69	> 5 years	speed, acceleration, jerk	bike, bus, car, foot, train.	DNN	91.4
[151]	410	10 months	speed stats. & location data.	bike, bus, car, foot, rail.	RF	93
[20]	81	2 weeks	velocity & acceleration stats.	bike, bus, car, foot, train, tube.	SVM	88
[14]	2	3 days	speed stats.	bus & car, foot, stationary, rail.	HMM	78
[63]	139	12 months	distance, acceleration, geospatial context	bike, boat, bus, car, foot, train, tram.	RF	93.0

Continued on next page

Table 5.1 – continued from previous page

Study	Population	Duration	Features	Modes	Classifier	Accuracy (%)
[148]	69	> 5 years	velocity, acceleration, turning angle.	bike, bus & taxi, car, foot, subway, train.	XGB	90.8
[123]	1.6m user	3 years	auto-extracted (deep) features.	bike, foot, car, station-ary, train.	LSTM	81.4
[145]	203	10 months	speed, ac-celeration & heading change.	bike, bus, car, foot, e-bike, subway.	Gaussian	92
[112]	91	12 months	speed, ac-celeration, bearing rate.	bike, bus, foot, car, train.	Autoencoder	93.9
[156]	69	> 5 years	velocity, acceleration, frequency-inverse	bike, bus, car, foot, subway, train, taxi.	Autoencoder	68.3
[98]	69	> 5 years	speed, accel-eration, jerk, bearing rate.	bike, bus, car, foot, train.	CNN	83.3
[47]	69	> 5 years	distance, heading, curvature.	bike, bus, car, foot, subway.	CNN	95.2

Feature selection constitutes a critical step in the development of effective learning models. This crucial task can be broadly categorised into three distinct approaches [78]: filter methods, wrapper-based methods, and embedded methods. Filter methods are independent of any learning algorithms. evaluate the relevance of features based on their statistical relationship with the target variable. Wrapper methods rely on the learning algorithm's performance to evaluate the relevance of features by iteratively selecting subsets of features and training a model on each subset. Embedded methods integrate feature selection directly into the structure of a learning algorithm, utilising its inherent properties (e.g., regularisation models) [78].

While prior research has recognised the potential of feature selection in TMD, its specific significance remains largely unexplored. This gap necessitates further investigation. The study in [46]

explored the efficacy of wrapper and information retrieval methods for selecting optimal feature subsets, demonstrating improved performance. Nevertheless, a key limitation of their work lies in the failure to conduct a comparative analysis with other prominent feature selection techniques. The proposed framework in [129] incorporates feature selection based on intra/inter-class feature distances. However, the description of the selection approach lacks clarity and specificity. Furthermore, the study fails to provide empirical evidence demonstrating an improvement in results attributable to the feature selection module.

Alazeb et al. [6] employed a recursive feature elimination method to select the most informative features. However, a significant limitation arises because it is impossible to attribute the observed performance gains solely to the feature selection process, as their proposed framework integrated feature selection and imbalance learning techniques. Isolating the impact of feature selection is crucial for a thorough understanding of its effectiveness and for establishing best practice in TMD. Researchers recognise feature selection as a crucial step for optimising machine learning models, yet its precise contribution within multi-faceted TMD frameworks, such as that of Alazeb et al., is often difficult to determine reliably. This coupling of techniques obscures the distinct impact of the feature selection process itself, making the identification of the most effective features for individual transportation modes challenging.

To address this gap, the *Class-Subspace* feature selection method is introduced in the current study. The method leverages Shapley values to provide feature interpretability and is explicitly designed to isolate and optimise the relevance of features on a mode-specific basis. The Class-Subspace approach establishes a clear connection between feature subsets and mode accuracy, thereby supporting the design of robust and focused TMD models.

5.3 Materials and Methods

This section details the materials and methods used in this study. The overall conceptual framework is shown in Fig. 5.1.

5.3.1 Terminology

Point: A point refers to a single GPS location point recorded at a specific timestamp, typically at 1 Hz when the GPS device is actively acquiring signals.

Trajectory: A trajectory is a chronologically ordered sequence of GPS points, each point is a pair of spatial coordinates and a timestamp:.

$$T = \{(x_1, y_1, t_1), (x_2, y_2, t_2), \ldots, (x_n, y_n, t_n)\}$$

Session: A session is defined as a continuous period of GPS data recording initiated when a user starts moving and terminated when they stop.

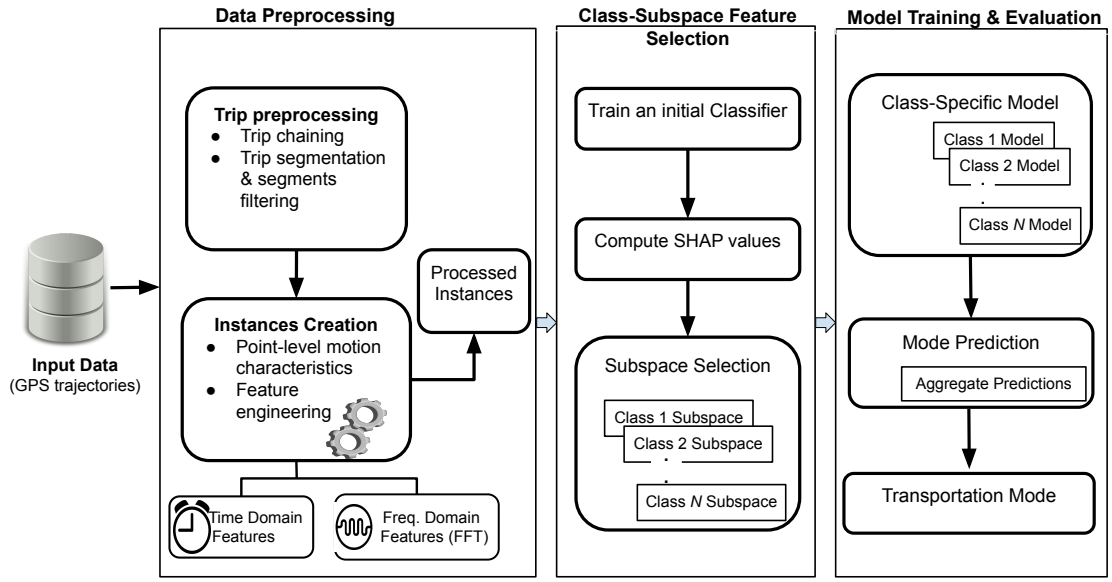


Figure 5.1: Conceptual framework detailing the overall methodology pipeline.

Trip: A trip is defined as a sequence of GPS points representing a single travel episode, characterised by continuous movement with a potential stop of no more than 30 minutes between consecutive segments. A trip can consist of multiple sessions for a user.

Segment: A segment is a continuous subsequence of a trip in which the user travels using a single mode of transportation.

5.3.2 Datasets

This study utilises two real-world GPS trajectory datasets: the *SenseMyFEUP* [108] dataset and the *GeoLife* [161] dataset, which are described below.

5.3.2.1 SenseMyFEUP dataset

The *SenseMyFEUP (SMF2016)* dataset is a GPS trajectory dataset of 227 participants collected in Porto - Portugal in April 2016. It was collected using an Android-based mobile application installed on participants' smartphones. The application was designed to record location data whenever user movement is detected automatically and to stop recording when the user stops moving. *SMF2016* dataset has a sampling frequency of approximately 1 sample per second [108]. The location data is structured as a tuple containing the following attributes: timestamp, longitude, latitude, altitude, speed, bearing, and GPS accuracy, thus $L_i = < time[t], long, lat, alt, speed, bearing, gps\_acc >$. Moreover, the application includes an end-of-trip survey administered after each trip completion, prompting users to indicate their travel mode. This user-annotated data, collected through an end-of-trip survey administered after each trip completion, serves as the ground truth for our analysis. Only trips with a single reported mode of transportation were included in this study. In all,

SMF2016 contains travel mode information for five different transportation modes: *bike, bus, car, foot* and *metro*.

5.3.2.2 Geolife dataset

We utilised the publicly accessible *Geolife* dataset published by Microsoft Research Asia [163, 160, 162] to assess the reliability of our method. This dataset was collected from 182 participants mainly in Beijing - China, primarily over five years (April 2007 - February 2012). The data collection employed various GPS loggers and GPS-enabled phones, capturing movement information at a spatio-temporal granularity of approximately 1-5 seconds or 5-10 meters per data point. According to the *Geolife* user guide [164], 73 users have annotated their trajectories with travel mode information. However, we found only 69 users with transportation mode information upon examining the raw data. There are about ten different transportation modes *(walk, bike, bus, car, taxi, train, subway, motorcycle, airplane, boat)* available in the dataset.

For the purpose of this study, we focus solely on ground transportation modes, even though the dataset includes annotations for other types of transportation. In compliance with the user guide [164], we regard the labels of both *taxi* and *car* as *car mode*. Specifically, we consider five transportation modes that correspond to those available in the SMF2016 dataset.

5.3.3 Trip Preprocessing

This section outlines the process of chaining GPS points into trips in the SMF2016 dataset and subsequently ensuring data quality through a series of preprocessing and filtering steps. For consistency across the datasets used in this study, we adopt the standard methodology for separating user trajectories into trips: a time interval (Δt) between two successive GPS points that exceeds 30 minutes defines the boundary between two distinct trips, following established literature [31].

Our trip chaining algorithm in the SMF2016 dataset identifies and joins two GPS points within a session into a single trip according to the following criteria [108]:

- the points are associated with a particular user,
- the time interval (Δt) between a point and its predecessor is no more than 30 minutes,
- the distance between a previous point and the start of a new one is below 200m.

5.3.3.1 Data Quality Control and Filtering

To ensure the dataset's integrity for urban mobility analysis, we implemented a series of quality control steps following the initial trip chaining. The first step involved a diagnostic check for temporal anomalies, identifying "unstopped sessions" where the interval between successive GPS points (Δt) in a trip exceeded the established threshold. These cases, which resulted from data collection persisting despite a lack of recorded movement, are detailed in Table 5.2. Investigation further revealed that many of these anomalous sessions corresponded to trips exceeding the scope

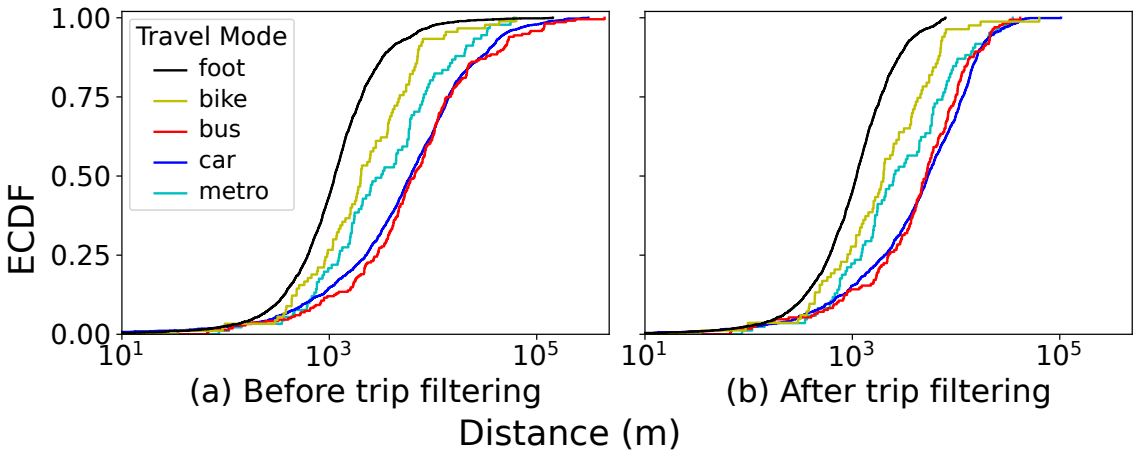


Figure 5.2: Trip distance distribution by travel mode (a) before trip filtering; (b) after filtering to trips covered in Porto region.

of the study area's urban context. To maintain a focus on urban-scale mobility, we implemented a spatial filter using metropolitan coordinates (41.38786, -8.77034) and (41.04928, -8.4253). This filter removed affected trips extending beyond the functional urban area, ensuring the data characterise only typical city-scale transportation.

Table 5.2: Sessions having $\Delta t > 1hr$ between consecutive GPS points.

Number of cases found	58
Number of sessions affected	26
Total number of sessions in the dataset	5368
Number of trips in unstopped sessions	25
Single mode trips affected	15

Second, we addressed mode-specific anomalies. We observed several trips manually annotated as "foot" mode that exceeded 10 km, which is statistically improbable for urban pedestrian movement (Fig. 5.2a). Consequently, we applied a conservative 8 km distance cap on all walking trips. As shown in Fig. 5.2b, this targeted filter produced a more plausible distance distribution. The final filtered dataset, used for all subsequent analysis, is summarised in Table 5.3.

Table 5.3: Summary of data filtering stages.

Initial number of trips	3730
Trips with temporal gaps (large Δt)	15
Non-urban trips (spatial filter)	764
Remaining urban-scale trips	2951
Final dataset (foot mode capped at 8 km)	2909

5.3.4 Instances Creation

To prepare the dataset for mode detection, each trip trajectory was segmented into alternating sequences of motion and stationary instances. The creation of instances is a multi-step process that involves:

5.3.4.1 Trip Segmentation

The trajectory segmentation process begins by identifying stationary segments, which represent periods of user inactivity. We employ a threshold-based method using the 85^{th} percentile of instantaneous GPS speed [108]. A segment is classified as stationary if the user's average speed remains below 0.5 m/s for a minimum duration of 5 seconds. All remaining segments are categorised as motion segments, representing active travel. This process inherently divides each trip into its corresponding single-mode segments. We then finalise the preparation by assigning the ground truth labels to these segments, achieving this by leveraging the manual user annotations of transportation modes.

5.3.4.2 Segments Filtering

Following trip segmentation, the process may generate several short segments that are potentially uninformative and can degrade the performance of the TMD classifiers due to data sparsity [108]. To ensure only meaningful segments are used for model development, we apply a filtering step, imposing a minimum segment length threshold of 50m. This dual-purpose filter mitigates noise from short, insignificant movements and enhances the classifier's ability to distinguish between different transportation modes by focusing on more substantial movement patterns. Applying this 50m length threshold resulted in the removal of 11,060 segments from the SMF2016 dataset (38.7% of the total segments) and 1,476 segments from the Geolife dataset (14.3% of the total segments).

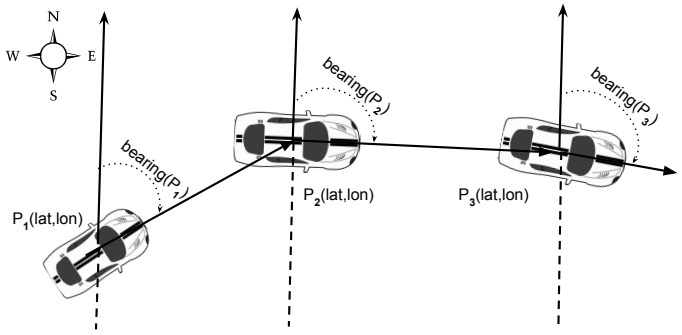


Figure 5.3: Visual representation of bearing between successive GPS points in a trajectory.

5.3.4.3 Time Domain Features

Our approach involves computing four motion attributes for each location point in the time domain. Because the Geolife dataset lacks instantaneous GPS speed information—unlike the SMF2016 dataset—we first calculate the instantaneous speed value of each point using eq. (5.1) for any two consecutive data points (P_m, P_n). Additionally, we calculate point-level acceleration, jerk, and bearing rate (as illustrated in Fig. 5.3) using eq. (5.2), eq. (5.3), and eq. (5.4), respectively.

$$S_{P_m} = \frac{Vincenty(P_m, P_n)}{\Delta t} \tag{5.1}$$

$$A_{P_m} = \frac{S_{P_n} - S_{P_m}}{\Delta t} \tag{5.2}$$

$$J_{P_m} = \frac{A_{P_n} - A_{P_m}}{\Delta t} \tag{5.3}$$

$$BR_{P_m} = |bearing(P_n) - bearing(P_m)| \tag{5.4}$$

$$y = sin[P_n(lon) - P_m(lon)] \times cos[P_n(lat)] \tag{5.5}$$

$$x = cos[P_m(lat)] \times sin[P_n(lat)] - sin[P_m(lat)] \times cos[P_n(lat)] \times cos[P_n(lon) - P_m(lon)] \tag{5.6}$$

$$bearing(P_m) = arctan(y, x) \tag{5.7}$$

We divide each segment into fixed-length instances of $N = 100$ time steps. This approach ensures a consistent input format for our classification models. We selected this specific value to approximate the median number of data points present within all segments of the SMF2016 (123) and Geolife (200) [31] datasets, which allows our separate models to learn from a representative sample of each dataset's characteristics.

5.3.4.4 Frequency Domain Features

While most GPS-based TMD methodologies focus exclusively on time-domain features, we incorporate frequency domain features to gain a deeper insight into the periodic characteristics of the motion signals. This is because different travel modes possess distinct frequency signatures: for instance, cyclic motion patterns present in modes such as walking or cycling (due to human gait or pedalling) could introduce clear, higher-frequency components in acceleration and jerk. Conversely, modes involving steady mechanical travel, such as car or bus trips, are typically characterised by dominant low-frequency components. We leverage the frequency domain to isolate and quantify these patterns, which we hypothesised could enhance the classifier's ability to discriminate between modes.

To achieve this, we utilised the Fast Fourier Transform (FFT) algorithm [101] to transform all feature dimensions of each instance from the time domain to the corresponding frequency spectrum. For real-valued input, the FFT produces a symmetric spectrum where the values for positive frequencies are the complex conjugates of the values for negative frequencies. We considered only the absolute value (magnitude) of the positive frequency components for feature creation, which

effectively extracts the signal's energy distribution across frequencies while avoiding feature duplication and the complexity of using the raw imaginary values. Fig. 5.4 shows an example of an instance in the time and frequency domains.

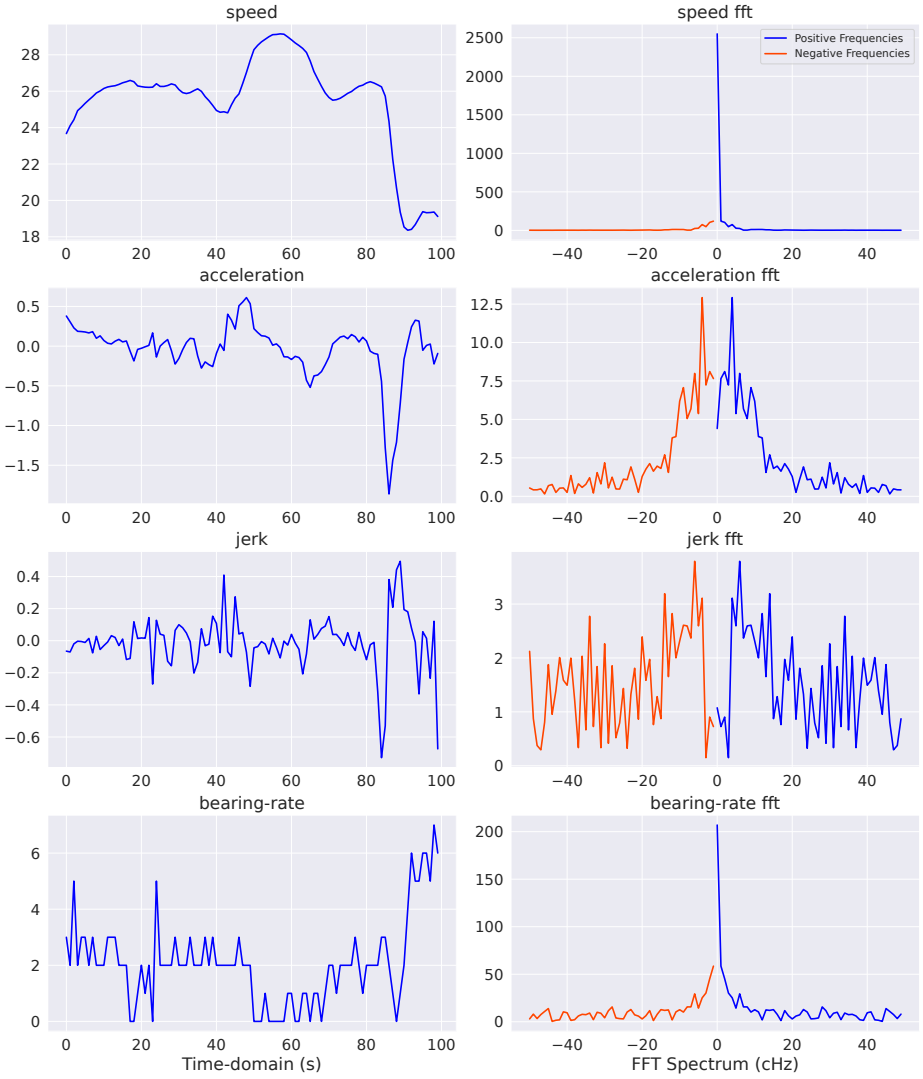


Figure 5.4: An instance's time and frequency domains.

5.3.4.5 Dataset Splitting

To prepare the resulting instances for model training and evaluation, we implemented a temporal data split, separating the train and test sets based on the data collection period. This strategy is crucial because the datasets represent a time series of user GPS records. We opted to deviate from the conventional random data split to mitigate the potential risk of data leakage and avoid optimistic performance estimates. The conventional random split does not preserve the chronological order of the trip data, which can lead to models performing unrealistically well on the test set because they have inadvertently seen patterns from the test period during training [90]. As demonstrated

in previous work [90], failing to preserve the chronological order can result in accuracy overestimation. Therefore, the temporal data split provided a realistic evaluation of the models' predictive capabilities on future, unseen data by ensuring that the model learned only from previous trip data.

For the SMF2016 dataset, we allocated the first three weeks of data for model training and reserved the subsequent week for the test set. Similarly, for the Geolife dataset, we utilised data from April 2007 to November 2008 for model training and reserved all data beyond November 2008 for the test set. The distribution of instances by transportation mode is provided in Table 5.4.

Table 5.4: Instance distribution by transportation mode in both datasets.

Mode	**SMF2016 Train**		**SMF2016 Test**		**Geolife Train**		**Geolife Test**	
	Count	**(%)**	**Count**	**(%)**	**Count**	**(%)**	**Count**	**(%)**
Foot	15,181	22.63	7,211	17.73	9,581	29.31	8,169	31.67
Bike	2,502	3.73	1,502	3.69	4,092	12.52	6,010	23.30
Bus	11,024	16.43	7,095	17.44	7,831	23.96	5,720	22.18
Car	34,832	51.92	24,116	59.28	5,011	15.33	3,108	12.05
Metro	3,553	5.30	758	1.86	6,175	18.89	2,784	10.79
Total	**67,092**	**100.00**	**40,682**	**100.00**	**32,690**	**100.00**	**25,791**	**100.00**

5.3.5 Classification Algorithms

This section provides an overview of the classification algorithms used in the study.

5.3.5.1 Decision Trees

Decision Trees (DT) are a supervised learning method commonly used for classification and regression purposes [88]. Decision trees construct a directed graph with a tree-like structure, where each internal node represents a feature, and each leaf node represents a class label. A DT classifier partitions the input space recursively based on the training input and label vector such that instances with the same labels are clustered together. An impurity measure, such as *entropy* or *Gini* impurity, determines the splitting criterion, which seeks to minimise classification error.

5.3.5.2 Support Vector Machines (SVMs)

SVMs are a family of supervised learning algorithms widely used for classification and regression tasks [88]. A key strength of the SVMs lies in the ability to handle non-linear data. SVMs achieve this by employing the *kernel* function [**?**], which increases the problem dimensionality, effectively transforming a non-linear problem into a linearly separable one. The *Support Vector Classifier (SVC)* is a variant of the SVMs designed explicitly for classification problems. SVC seeks to identify an optimal hyperplane that effectively separates data points belonging to different classes while maximizing the margin between the hyperplane and the nearest data points (also

called the *support vectors*). This margin maximization principle, made possible by the kernel-induced increase in dimensionality, contributes significantly to the high classification accuracy and generalization capabilities of SVMs, even in the presence of noise or imbalanced data [137].

5.3.5.3 Ensemble Methods

Ensemble methods combine multiple base models to produce a single, more robust predictive model. The following ensemble methods were utilised in this study.

Random Forest Random forests (RF) are an ensemble method that combines multiple decision trees to enhance predictive accuracy. Each decision tree within the forest is trained on a different bootstrap sample [88, 16], similar to bagging; however, at each node, the split rule is selected from a random subset of predictive features rather than the complete set. The random feature selection at each split can be advantageous in datasets with many predictive variables by reducing correlation among trees and enhancing generalisation performance [16]. Nevertheless, its practical benefit depends on the proportion of informative features. Although computationally intensive, random forests leverage decision trees as base learners and are versatile for classification and regression tasks. The prediction variability decreases as more trees are added to the ensemble, improving result stability.

Extreme Gradient Boosting Extreme Gradient Boosting or *XGBoost* is a powerful open-source software library that follows the principle of the gradient boosting algorithm [29]. It uses decision trees as base learners, sequentially adding them to an ensemble while focusing on rectifying the errors made by prior models. This sequential refinement is guided by the gradient boosting approach, which minimises a predefined loss function. Furthermore, XGBoost incorporates several techniques such as regularization, tree pruning, and parallel processing to enhance the generalisability of the final model [28]. XGBoost's strength lies in its distributed architecture, enabling parallel tree boosting across machines.

5.3.6 Features per Instance

We use ten descriptive feature statistics [90] calculated for every feature of an instance for the time-domain features (mean, median, standard deviation, minimum, maximum, MAD, 25^{th}, 75^{th}, 85^{th} percentiles, and inter-quartile range). For the frequency-domain features, we consider the indexes of the top 10 frequency components for each feature of an instance. Thus, each instance consists of 80 features: time-domain 10 x 4 features and frequency-domain 10 x 4 features.

5.3.7 Feature Selection

Following the generation of an extensive pool of time-domain and frequency-domain features, we employed the Sequential Forward Floating Selection (SFFS) [52, 102], a wrapper-based feature selection method, to derive the optimal subset. SFFS initiates the selection process with an empty

feature subset and iteratively builds this subset by combining forward addition and backward floating steps.

In the forward step, features are sequentially introduced to the subset based on their contribution to model performance, evaluated against a held-out validation set using the Area Under the Receiver Operating Characteristic Curve (ROC-AUC). Subsequently, a floating step is initiated, wherein previously selected features are assessed for removal. If the model's performance, following the removal of a feature, does not significantly degrade, that feature is permanently discarded from the subset. This iterative cycle of forward introduction and backward evaluation continues until a termination criterion is met, when no further significant gain in ROC-AUC is observed.

5.4 Evaluation Metrics

Evaluation metrics are quantitative measures used to assess the performance and effectiveness of a machine learning model. They are typically used to compare the performance of different models or to evaluate the performance of a model over time. Within the classification domain, the widely prevalent metrics for performance evaluation are *accuracy* and *error rate.* In this section, we describe the most common performance metrics in classification.

5.4.1 Confusion Matrix

Each example data sample can yield one of four potential outcomes in a binary classification task involving two classes. If the sample is genuinely positive and the classification algorithm correctly predicts it, it is termed a *true positive (TP)*. Conversely, if the algorithm incorrectly predicts it as negative, it is referred to as a *false negative (FN)*. Similarly, if the sample is truly negative and the classifier accurately predicts it as negative, it is labelled as a *true negative (TN)*. If, however, the algorithm erroneously classifies it as positive, it is designated as a *false positive (FP)*. These four distinct outcomes are precisely captured within a structure known as a *contingency table*, or *confusion matrix*.

The confusion matrix is a table summarising a classification algorithm's performance on the test set [48]. It is typically divided into four cells, corresponding to the four possible outcomes.

Table 5.5: Confusion Matrix.

		Predicted Class		
		Positive	*Negative*	*Class Support*
Actual Class	*Positive*	True Positive (TP)	False Negative (FN)	**P**
	Negative	False Positive (FP)	True Negative (TN)	**N**

5.4.2 Accuracy

Accuracy is the proportion of test instances correctly classified by the model. The accuracy score can be calculated from the confusion matrix following eq. (5.8).

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{5.8}$$

5.4.3 Precision

Precision is the proportion of test instances that are classified as positive by the model that are positive. Using the entries of the confusion matrix, this can be calculated as:

$$Precision = \frac{TP}{TP+FP} \tag{5.9}$$

5.4.4 Recall

Recall evaluates the proportion of test instances that are actually positive and are classified as positive by the model. This is calculated using eq. (5.10).

$$Recall = \frac{TP}{(TP+FN)} \tag{5.10}$$

5.4.5 F1 score

There is always a trade-off between a machine learning model's precision and recall scores. An *F1 score* combines the precision and recall scores into a single value, making it a useful measure for evaluating the performance of a classification model. It is the harmonic mean of the precision and the recall scores, calculated as:

$$F1 = \frac{2 \cdot (Precision \times Recall)}{Precision + Recall} \tag{5.11}$$

5.4.6 Receiver Operating Characteristics

So far, all the evaluation metrics mentioned above rely on the value of the possible outcomes from the confusion matrix. They are, therefore, sensitive to the class distribution of positive and negative classes. We can see from the confusion matrix (Table 5.5) that the class distribution is the relationship of the positive class (upper) row to the negative (lower) row [48] (see the "Class Support" column in the confusion matrix Table 5.5). A somewhat better metric insensitive to class distribution is the Receiver Operating Characteristics (ROC). ROC is a graphical representation and a performance measurement tool used in binary classification problems.

The ROC curves are plotted using two performance metrics: the *true positive rate (TPR)* - proportion of positive instances that are correctly classified, and *false positive rate (FPR)* - the proportion of negative instances that are incorrectly classified. Figure 5.5 illustrates a typical ROC curve, showing the relationship between TPR and FPR at various classification thresholds. The

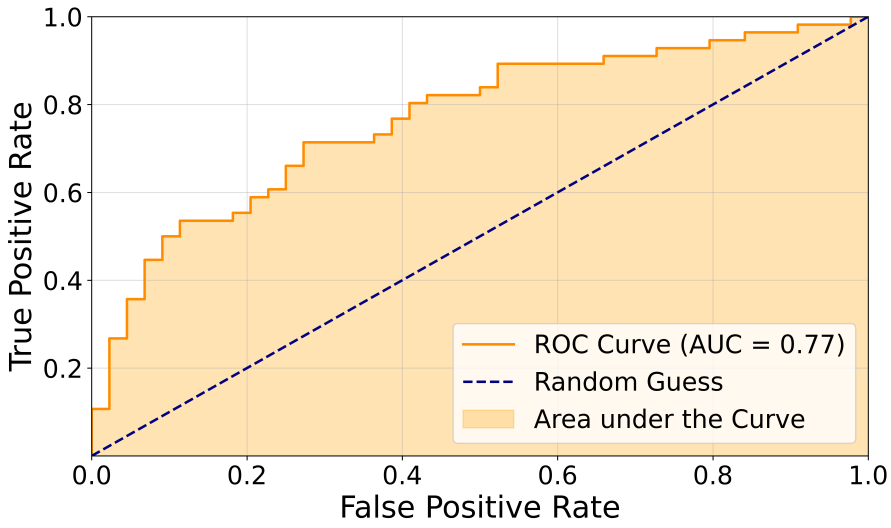


Figure 5.5: Area under the Receiver Operating Characteristic (ROC) Curve.

Area Under the Curve (AUC) is a common metric for summarising the performance of a classifier based on its ROC curve.

These curves are created by varying the classification threshold and plotting the TPR against the FPR at each threshold. The classification threshold is a value used to decide whether an instance is classified as positive or negative. Lowering the classification threshold will classify more instances as positive, thus increasing both TPR and FPR. A higher area under the ROC curve (AUC) generally indicates a better-performing model, with 1 representing perfect discrimination and an AUC of 0.5 representing just a random classifier.

5.5 Feature Selection

Feature selection is a critical step in machine learning pipelines aimed at identifying and retaining only relevant features while discarding extraneous ones. The primary goal of feature selection is to identify and select the most relevant and informative subset of features from a more extensive set of variables, aiming to improve model performance and interpretability and reduce computational complexity [72]. Previous studies in TMD from GPS data primarily focused on constructing features in the time domain. While this approach has yielded effective learning models, there is a tendency to overlook the potential of frequency-domain features [11, 68].

We use the ten (10) descriptive feature statistics [90] calculated for every feature of an instance for the time-domain features (mean, median, standard deviation, minimum, maximum, MAD, 25^{th}, 75^{th}, 85^{th} percentiles, and inter-quartile range). For the frequency-domain features, we consider the indexes of the top 10 frequency components for each feature of an instance. Thus, each instance consists of 80 features: time-domain 10 x 4 features and frequency-domain 10 x 4 features.

5.5.1 Sequential Forward Floating Selection

After generating the pool of features, we applied the *Sequential Forward Floating Selection (SFFS)* [52, 102] technique, a wrapper-based method for feature selection. SFFS begins with an empty feature subset. In the forward selection phase, features are iteratively added to the subset one at a time, with each addition leading to model training and evaluation. The floating operation in SFFS involves removing features from the currently chosen subset if they no longer contribute significantly to the model's performance, even if they were previously selected. The process continues until no further improvement in the evaluation metric, such as accuracy or error rate, is observed on a validation set. This study uses the area under the curve - receiver operating characteristics (ROC-AUC) as the evaluation criterion.

5.6 Proposed Feature Selection Method

This section introduces the proposed feature selection method rooted in Shapley values. We firstly provide a concise overview of Shapley values and their relevance to feature importance assessment.

5.6.1 The Shapley Values

Drawing upon the principles of cooperative game theory, the *Shapley values* emerges as a powerful tool for assessing the contributions of individual features towards a machine learning model predictions [84]. The Shapley value of a feature quantifies its average marginal contribution to the overall prediction across all possible combinations of feature values. It measures the significance of each feature by evaluating its impact on the model prediction across all possible combinations of feature values.

Lundberg, Erion, and Lee (2018) [85] introduced a computationally efficient and theoretically sound approach called SHAP (short for *SHapley Additive exPlanations*) for explaining the output of a machine learning model using insights from game theory. SHAP value is a well-established feature importance technique that quantifies the attribution of each feature to the model prediction by considering all possible marginal contributions of the features. The SHAP value of a feature i, denoted ϕ_i, is calculated following equation 5.12, where N represents the set of all input features and S denotes the feature subset.

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} \{E[f(X)|X_{S \cup i} = x_{S \cup i}] - E[f(X)|X_S = x_S]\} \tag{5.12}$$

The conditional expectation $E[f(X) \mid X_{S \cup i} = x_{S \cup i}]$ refers to the model's expected output when feature i is included along with the features in subset S. In contrast, $E[f(X) \mid X_S = x_S]$ represents the expected output when feature i is excluded, considering only the features in subset S.

Despite its computational cost, SHAP has demonstrated commendable properties, including fairness and consistency [158], in assigning importance scores to individual features.

5.6.2 Class-Subspace Selection

Traditional wrapper-based feature selection algorithms, such as SFFS, do not determine each feature's relevance based solely on its individual merit. Instead, they operate by training a model using different combinations of feature subsets and determine which combination yields the best overall performance (e.g., highest accuracy). This approach fundamentally prioritises features that collectively enhance the model's overall predictive power. However, experimental results (see Figures. A.1 and A.2 in Appendix A) showed that the features most relevant for classifying one transportation mode (e.g., bus) were often sub-optimal for classifying another (e.g., car). This difference implies that relying on a single, globally optimised feature subset necessarily compromises the predictive performance for certain classes.

To address this limitation—the need to capture distinct classification boundaries for each unique mode—we propose the Class-Subspace feature selection technique. This method identifies a minimal, class-specific subset of features that adequately captures the discriminative information required to classify a single target mode against all others.

Our approach involves training an initial classifier and computing the SHAP values $\phi(x_{i,j})$ for all features i and instances j in the training set D. Subsequently, for each unique class k, we evaluate the contribution of each feature to the model prediction of that class by computing the mean absolute SHAP values. These values are then sorted in decreasing order of magnitude to identify the most influential features for predicting the class. The minimum subset of features comprising the contributing values whose sum is larger than the contribution threshold (ρ) is selected. Figure 5.6 visually depicts the steps involved in the process.

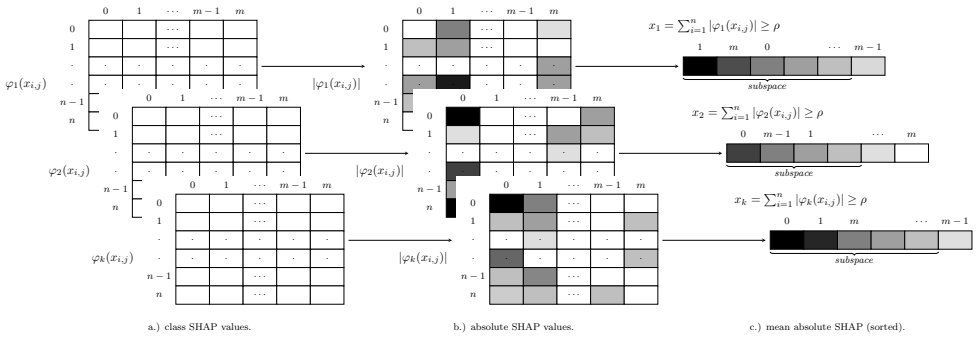


Figure 5.6: An illustration of the selection process for individual class in a dataset.

Once these k discrete feature subsets $\{S_1, S_2, \ldots, S_k\}$ have been identified, they are used to construct the predictive framework. We adopt a modular approach whereby a base learner is trained for each class k using only its corresponding optimal subset S_k. This ensures that each model acts as a specialist discriminator, specifically designed to maximise the detection of instances belonging to its respective class. While the final classification is achieved via a hard-voting aggregation of these specialised learners, this multi-model configuration is employed primarily as a transparent proxy to measure the discriminative utility of the selected features. Unlike traditional ensemble

methods that aim to optimise predictive variance, our objective remains the isolation and characterisation of class-dependent feature relevance. Algorithm 2 provides the pseudocode for the Class-Subspace selection technique. For the experiments, a contribution threshold of $\rho = 0.6$ was employed for features contributing to each class prediction; i.e., only features explaining 60% of cumulative relevance were retained. The impact of the subspace threshold (ρ) on classification performance was evaluated through a sensitivity analysis (see Fig. B.1 in Appendix B).

During the testing phase on unseen data samples, the k base models—each operating exclusively on its respective class-specific feature subset S_k—generate individual predictions. These are subsequently aggregated through hard voting to determine the predicted transportation mode.

Algorithm 2: Class-Subspace Selection

Input: Training data $D = \{(x_j, y_j)\}_{j=1}^{m}$ with features $i \in \{1, ..., n\}$;
Class labels $y_j \in \{1, ..., K\}$;
Base learning algorithm T;
Subspace contribution threshold ρ.
Output: A set of Class-specific feature subsets $\{S_k\}_{k=1}^{K}$
Process:
Train T using D;
Compute SHAP values $\phi(x_{i,j})$ for all features $i \in n$ and instances $j \in m$.
for $k \leftarrow 1$ *to* K **do**
  Initialise an empty list M_k for mean absolute SHAP values;
  for $i \leftarrow 1$ *to* n **do**
    Compute the mean absolute SHAP value for feature i for class k:
    $m_{i,k} \leftarrow \frac{1}{m_k} \sum_{j|y_j=k} \left|\phi(x_{i,j})\right|$;
    Add $(i, m_{i,k})$ to M_k;
  end
  Sort M_k in decreasing order;
  Initialise an empty set S_k for chosen feature subset for class k;
  Initialise a cumulative sum $current\_sum \leftarrow 0$;
  for $(feature\_index, value)$ *in* M_k **do**
    Add $feature\_index$ to S_k;
    $current\_sum \leftarrow current\_sum + value$;
    if $current\_sum \geq \rho$ **then**
      break
    end
  end
end
return $\{S_k\}_{k=1}^{K}$.

5.7 Experiments and Results

We train the four classification algorithms described in Section 5.3.5. A comparative analysis of the classification results shows that RF and XGB demonstrated superior performance on both

datasets when used with the 80 features initially generated. This is presented in Table 4.3. Specifically, for the SMF2016 dataset, both RF and XGB achieved a ROC-AUC score of 75%, while for the Geolife dataset, they both attained an higher ROC-AUC of 88%,

5.8 Discussion

5.9 Conclusion

Chapter 6

Hierarchical Classification Approach to Mode Detection

6.1 Introduction

Recent advancements in machine learning have spurred significant research into extracting insights from large-scale sensor data. Within the field of human activity recognition (HAR), transportation mode detection [91] aims to develop machine learning algorithms that predict the modes of transportation used by individuals.

TMD inherently tackles a *multiclass* classification - a classification problem involving more than two classes. Trips often involve the use of multiple transportation modes; for example, a commute might combine walking to a train station followed by a train journey. Existing research [9, 90, 104] typically does not recognise or exploit the inherent relationships between different transportation modes. This is crucial because some transportation modes, e.g. buses, and cars, share certain characteristics or infrastructure, making them more closely related than others. By not considering these modal relationships, existing research could limit the accuracy of TMD algorithms, potentially leading to misclassification between such modes.

In this study, we propose the hierarchical classification approach to transportation mode detection - *HiClass4MD* algorithm - which addresses this challenge by incorporating the inherent hierarchical relationships between transportation modes. *HiClass4MD* leverages misclassification errors from a standard flat classifier to learn the class hierarchy using agglomerative clustering. This learned class hierarchy is then used to guide the development of a TMD classifier to improve correct prediction, potentially reducing misclassification between transportation modes.

This chapter is structured as follows. Section 6.2 provides a brief overview of existing hierarchical classification approaches. We present our proposed method for transportation mode detection with hierarchical classification, *HiClass4MD*, in Section 6.3. Section 6.4 presents the results obtained, followed by the discussion in Section 6.5. Finally, Section 6.6 summarises the key contributions and outlines future research directions.

6.2 Hierarchical Classification

Conventional flat classification techniques, referring to standard binary or multiclass methods, overlook the structural information between distinct classes. Most common classification methods decompose a multiclass problem into several binary problems to achieve better separation between the classes [8]. However, this approach neglects the hierarchical information that may exist within the classes.

In contrast, a significant proportion of real-world classification tasks lend themselves naturally to a hierarchical classification approach [62, 121]. In these problems, the classes to be predicted form an organised hierarchy. By incorporating hierarchical structures, classification methods can exploit the relationships between classes, potentially leading to improved performance. In what follows, we describe the different approaches to hierarchical classification.

6.2.1 The Big-Bang or Global Approach

This approach builds a single, complex classification model over the entire hierarchy in one go. Unlike other methods, the method does not involve separate training stages for each level in the class hierarchy. Instead, it considers the entire structure of the classes all at once [96]. This *"one-step"* process results in a more complex model compared to other approaches. During testing, the model analyses each new example and assigns it a category within the hierarchy, with the potential to classify it at any level. Compared to other hierarchical approaches, big-bang has the advantage of a potentially smaller final model size [120]. However, the high complexity of this approach is a major drawback, and is probably the reason it is rarely used.

6.2.2 Local Classifiers Approach

This approach utilises a modular structure, employing several local classifiers to discriminate between classes as dictated by the hierarchical structure. Each local classifier focuses on a specific portion of the hierarchy, leveraging local informational perspectives to make classification decisions. This approach can be further subdivided based on the strategies it employs to utilise local information and construct classifiers that exploit it [120].

- *Local Classifier per Node (LCN):* This method involves training a binary classifier at each node within the hierarchy, excluding the root node.
- *Local Classifier per Parent Node (LCPN):* This method trains a multiclass classifier at each parent node (including the root) to discriminate between its child nodes throughout the hierarchy until reaching leaf nodes.
- *Local Classifier per Level(LCL):* This approach involves training a multiclass classifier at every level of the hierarchy.

Algorithm 3: Class Hierarchy Learning

Input: Dataset $D = (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), .., (\mathbf{x}_m, y_m)$;
Number of classes C;
Flat classifier F;

Process:

1 Fit F on D, and evaluate the confusion matrix M.

2 Compute $\bar{M}$, the row-wise normalisation of M: $\bar{M}_{ij} = \frac{M_{ij}}{\sum_{k=1}^{n} M_{ik}}$ $\forall i, j \in C$.

3 Determine the *Overlapping Matrix* $\bar{M}_O$: the degree of overlapping between the classes:

$$Overlap(i,j) = \begin{cases} \frac{\bar{M}_{ij} + \bar{M}_{ji}}{2} & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

$$\forall i, j \in C.$$

4 Calculate the *Distance Matrix* $\bar{M}_S$ applying class *distance function*: $d_S(i,j) = 1 - Overlap(i,j)$; $\forall i, j \in C$.

5 Join classes into cluster H_C bottom-up based on closest distance in $\bar{M}_S$.

Output: *Class Hierarchy* H_C

6.3 Proposed Method

6.3.1 Class Hierarchy Learning

The current study builds upon the research by Silva et al. [122] on hierarchical classification for multiclass problems. This method infers the hierarchical structure of classes from the misclassification errors of a standard flat classifier. The process begins by training a standard flat classifier, denoted as F, on the entire dataset. We then evaluate the resulting confusion matrix, M. This matrix is subsequently normalised row-wise, resulting in $\bar{M}$. Next, we compute the class overlapping matrix, $\bar{M}_O$, which captures the degree of overlap between individual classes i and j in all possible combinations (considering both $M_{i,j}$, $M_{j,i}$) as described by Lango et al. [75]. Following this, the Similarity Matrix $\bar{M}_S$, is obtained. This matrix represents the distance between the classes based on the overlap information. Finally, we use this distance information for agglomerative hierarchical clustering [94] to group classes into a hierarchy. This process is outlined in Algorithm 3. Applying this method with several classification algorithms and the SMF2016 dataset results in the class hierarchies depicted in Figures 6.1 through 6.4.

6.3.2 HiClass4MD for Transportation Mode Detection

The proposed method, *HiClass4MD*, leverages the class hierarchy learning described in Section 6.3.1 to establish a tree structure for transportation mode classification. Training of the *HiClass4MD* involves fitting a classifier using the LCPN approach. During the testing phase on an unseen sample, the process follows the hierarchy until the predicted class is determined at the leaf node. This is illustrated in the proposed method block diagram in Figure. 6.5.

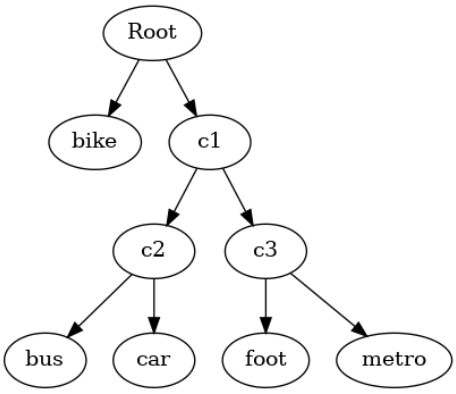


Figure 6.1: DT learned hierarchy.

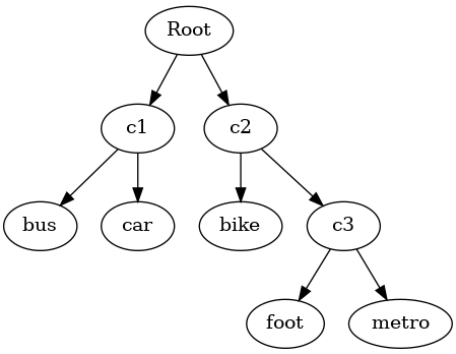


Figure 6.2: RF learned hierarchy.

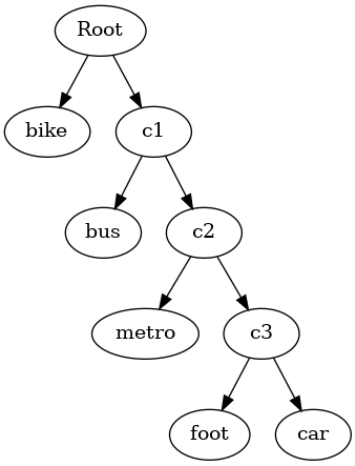


Figure 6.3: SVM learned hierarchy.

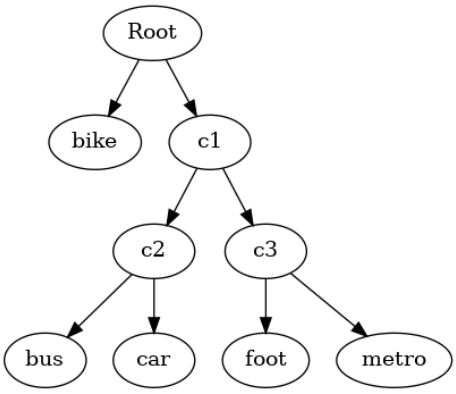


Figure 6.4: XGB learned hierarchy.

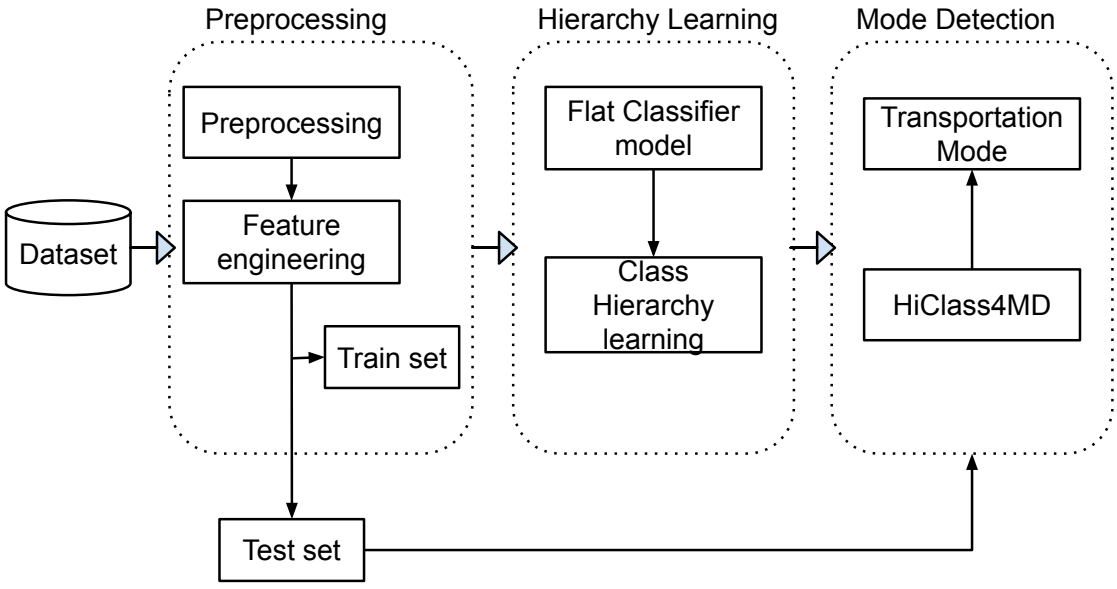


Figure 6.5: Block diagram of the proposed method.

6.3.3 Dataset and Preprocessing

We employed the GPS trajectories datasets described in Section 3.1. The data preprocessing steps and feature engineering, are described in detail in Section 3.2. Together, each instance is composed of the following features:

- *Time-domain features:* 10 descriptive statistics x 4 dimensions, detailed in Section 4.3.1.
- *Frequency-domain features:* top-10 indexes of the FFT frequency spectrum x 4 dimensions, detailed in Section 4.3.2.

By combining the above, each instance in the final datasets comprises 80 features (40 from the time domain and 40 from the frequency domain).

6.3.4 Evaluation Metrics

In the standard flat classification approach, performance metrics such as precision, recall, and F-measure are calculated based on three possible prediction outcomes: a correctly predicted positive outcome (True Positive - TP), an actual negative example incorrectly predicted as positive (False Positive - FP), or a positive example incorrectly predicted as negative (False Negative -FN).

However, performance evaluation in hierarchical classification requires a distinct approach due to the inherent hierarchical relationships within the class structure. As the inference process traverses down the class hierarchy (as illustrated in Figure 6.6), the prediction process must consider not only the predicted class itself but also its ancestor classes, similar to the true class and all its ancestors.

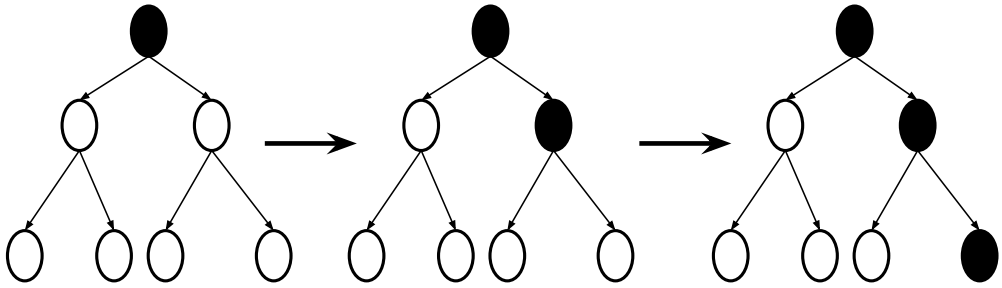


Figure 6.6: An illustration of the prediction process in hierarchical classification.

We therefore adopt the recommendations outlined in [120]. We define $\hat{T}$ as the set comprising the true class (and similarly, $\hat{P}$ for the predicted class) for sample i and all its ancestor classes within the hierarchy. Hierarchical precision (and similarly, recall and F-measure) are then calculated using the equations provided in Table 6.1.

Table 6.1: Evaluation metrics for Flat and Hierarchical Classification

Flat classification	Hierarchical classification
$Precision = \frac{TP}{TP+FP}$	$hP = \frac{\sum_i \lvert\hat{P}_i \cap \hat{T}_i\rvert}{\sum_i \lvert\hat{P}_i\rvert}$
$Recall = \frac{TP}{TP+FN}$	$hR = \frac{\sum_i \lvert\hat{P}_i \cap \hat{T}_i\rvert}{\sum_i \lvert\hat{T}_i\rvert}$
$F1 = 2 \times \frac{Precision \times Recall}{Precision+Recall}$	$hF = 2 \times \frac{hP \times hR}{hP+hR}$

6.4 Experimental Results

This study employs a selection of established machine learning algorithms commonly used for classification tasks: Decision Trees (DT), Random Forests (RF), Support Vector Machines (SVM), and XGBoost. DT recursively partition the data space based on features, while RF build ensembles of DT to improve robustness. SVM identify hyperplanes that effectively separate data points belonging to different classes. XGBoost leverages gradient boosting with decision trees to achieve high accuracy and generalisability. Our primary focus is to evaluate the effectiveness of the proposed *HiClass4MD* against these algorithms and assess the potential benefits of incorporating *HiClass4MD* into the transportation mode classification process.

The experiments were conducted using the sklearn [97] implementation of these algorithms (for DT, RF, and SVM), and XGBoost library [28]. Default hyperparameters of the algorithms were used, other than the number of trees in RF (`n_estimators=5000`) and for XGB (`n_estimators=1000`), for the flat classification experiments. The same settings were used with a wrapper around the algorithms to implement the hierarchical classification scenarios.

6.5 Discussion

To evaluate the proposed method, we utilised the hierarchical metrics developed in the context of hierarchical classification (hP, hR, hF) [120]. These metrics are extended versions of the well known conventional metrics for flat classification, but tailored to incorporate peculiarities of hierarchical classification. Table 6.1 shows the expressions for that. However, given the primary interest in the predicted transportation mode rather than the path to prediction in the hierarchy, we also assessed *HiClass4MD* using the conventional metrics. Consequently, the confusion matrices for *HiClass4MD* are presented in Table 6.3, from which conventional metrics were computed and compared with those of the flat classifiers (Table 6.2). These results are shown in parenthesis for each metrics in Table 6.1.

Counterintuitively, *HiClass4MD* did not consistently outperform flat classifiers. While decision trees (DT) exhibited marginal improvements, random forests (RF) showed no significant differences. This might be attributed to the fact that DT inherently produces more errors than RF

Table 6.2: Confusion matrices of the flat classification models

		Decision Tree (DT)					Random Forest (RF)				
		Predicted class					Predicted class				
		Foot	Bike	Bus	Car	Metro	Foot	Bike	Bus	Car	Metro
Actual class	Foot	5248	36	373	1236	318	6727	0	0	481	3
	Bike	411	338	88	563	102	300	335	0	867	0
	Bus	1232	104	1354	3961	444	1657	0	394	5044	0
	Car	1792	383	5738	15415	788	2247	9	2577	19283	0
	Metro	229	15	46	398	70	321	3	0	434	0

		Support Vector Machines (SVM)					XGBoost (XGB)				
		Predicted class					Predicted class				
		Foot	Bike	Bus	Car	Metro	Foot	Bike	Bus	Car	Metro
Actual class	Foot	6794	0	21	396	0	6672	4	3	527	5
	Bike	782	51	16	653	0	382	535	0	581	4
	Bus	1733	0	530	4819	13	1579	12	794	4695	15
	Car	2267	2529	2644	16656	20	2068	51	3493	18490	14
	Metro	330	0	17	398	13	304	9	26	419	0

Table 6.3: Confusion matrices of the hierarchical classification models

		Decision Tree (DT)					Random Forest (RF)				
		Predicted class					Predicted class				
		Foot	Bike	Bus	Car	Metro	Foot	Bike	Bus	Car	Metro
Actual class	Foot	5238	38	421	1170	344	6704	0	0	503	4
	Bike	334	439	114	579	36	374	150	0	978	0
	Bus	1589	95	1439	3796	176	1656	0	381	5058	0
	Car	2324	482	3614	17051	645	2236	9	2470	19401	0
	Metro	255	30	68	363	42	320	0	0	432	6

		Support Vector Machines (SVM)					XGBoost (XGB)				
		Predicted class					Predicted class				
		Foot	Bike	Bus	Car	Metro	Foot	Bike	Bus	Car	Metro
Actual class	Foot	5976	170	20	784	261	6722	38	45	376	30
	Bike	461	164	4	829	44	388	613	8	469	24
	Bus	1599	172	59	5065	200	1992	0	827	3949	327
	Car	1458	269	8778	13055	556	4678	1875	1222	14973	1368
	Metro	297	5	6	360	90	367	3	30	295	63

in flat classification, providing a richer error distribution for *HiClass4MD* to exploit. SVM and XGB even experienced slight performance degradation.

6.6 Summary

This study introduced the *HiClass4MD*, a hierarchical classification method for transportation mode detection. By leveraging misclassification errors from a typical flat classifier, *HiClass4MD* aims to enhance performance. While hierarchical metrics demonstrated improvements, the evaluation using conventional metrics revealed no significant improvement overall.

Decision trees benefited marginally from the hierarchical approach, but random forests showed no significant improvements. Support vector machines and extreme gradient boosting algorithms even degraded. These findings underscore the need for further investigation into the factors influencing the effectiveness of hierarchical classification in this context. Despite these results, the potential benefits of hierarchical approaches warrant continued exploration.

Chapter 7

Characterising Class Imbalance in Transportation Mode Detection

7.1 Introduction

The majority of research approaches TMD as a supervised machine learning problem, specifically a classification task. Researchers leverage various data sources to train their models. Commonly used datasets include GPS trajectory data (e.g. [31, 91]) and sensor data collected from IMU sensors such as accelerometers and gyroscopes (e.g. [105]). The number of transportation modes that these models aim to identify can range from as few as three to as many as ten [18]. One common challenge persists irrespective of the number of transportation modes or the data type employed: the distribution of data points belonging to individual modes within the dataset. Often, the proportion of transportation modes is unequal, with some modes dominating the data while others are scarce. This unequal distribution leads to a problem known as *class imbalance*.

Class imbalance refers to a situation in supervised learning problems where some classes have significantly more data samples available compared to others. This problem is well-studied in binary classification tasks (e.g., credit card fraud detection [50]), but it remains an active area of research in multi-class scenarios such as TMD [113]. A previous study [90] acknowledges the impact of class imbalance in identifying underrepresented transportation modes. Imbalance handling techniques employed in a later study [91] did not yield significant performance improvement. Furthermore, class imbalance can be exacerbated by other intrinsic data characteristics, such as class overlap. Class overlap refers to a situation where data points belonging to different classes share similar characteristics, making it difficult for algorithms to discriminate between them [113].

This study investigates the combined effects of class imbalance and class overlap on TMD model performance. We analyse two real-world GPS trajectory datasets. The first dataset exhibits significant class imbalance [90]. The second is the Geolife [164] benchmark dataset, a fairly-balanced GPS trajectory data containing several transportation modes. By comparing the performance of TMD models on these datasets, we aim to gain a deeper understanding of how the

combination of class imbalance and class overlap affects TMD model performance, particularly for underrepresented transportation modes.

This chapter is organised as follows. Section 7.2 provides an overview of related work on class imbalance and TMD. This is followed by a description of the materials and methods used in this research in Section 7.3. The experimental results and analysis are presented in Section 7.4. Finally, Section 7.5 concludes the paper by summarising key insights.

7.2 Related Work

Learning from imbalanced data, where one class has significantly fewer examples than others, poses a significant challenge. This difficulty stems not only from the limited samples in the minority class, but also from other data intrinsic characteristics such as class overlap. Class overlap can further obscure the characteristics of the rare class, making it even harder to learn effectively. Denil and Trappenberg [37] argue that class overlap can be even more detrimental than class imbalance. They reason that while imbalance can be mitigated to some extent by acquiring more training data, class overlap inherently makes classification a more difficult task. Even with increasing data, overlap can lead to increasingly complex models.

An empirical study by Lango and Stefanowski [75] explores how various factors affect the classification of data with imbalanced and multiple classes. Using carefully designed synthetic data, the study showed that overlap, imbalance, and number and size of classes all play a significant role. Researchers have further investigated methods to assess class overlap in imbalanced learning. Santos et al. [113] delve into the combined effect of class imbalance and overlap. Their work proposes a categorisation of metrics to evaluate class overlap. These metrics consider various aspects, including feature similarity between classes, data structure indicators of overlap, and the contribution of individual data points. The authors argue for the limitations of a single metric in capturing the complexities of class overlap, particularly in imbalanced data scenarios.

Regarding the methods to learn a classifier in the presence of class imbalance, two approaches are available: data-level and algorithm-level [144]. Data-level solutions modify the class distribution of the training data. Common techniques include oversampling the minority class, undersampling the majority class, or a combination of both. However, these techniques have limitations. Oversampling can introduce bias towards the oversampled class and lead to overfitting if exact copies are used. Undersampling discards potentially useful data from the majority class. Algorithm-level approaches, on the other hand, focus on modifying the learning algorithm itself. These can involve techniques like cost-sensitive learning, which assigns higher penalties for misclassifying instances of the minority class, and boosting methods that adjust the weights of training examples to focus on those that are poorly classified, especially for the minority class.

In the context of TMD, the impact of class imbalance has not been extensively explored. Existing studies that acknowledge class imbalance often resort to eliminating underrepresented classes, potentially discarding valuable data [148, 153, 154]. While the study in [90] observes the impact of imbalance on model prediction, even with imbalance handling techniques, performance

improvements might be marginal [91]. This current work investigates how resampling a balanced dataset can be used to understand the impact of class imbalance on TMD model prediction.

7.3 Materials and Methods

Two real-world datasets containing GPS trajectory data are utilised in this work. These datasets are described in Section 3.1.

7.3.1 Preprocessing and Feature Engineering

We applied the same data processing and feature engineering pipeline as described in [91]. In brief, this pipeline involves the following steps:

1. Extracting basic motion features: This involves calculating four point-level motion attributes like speed, acceleration, jerk, and bearing rate from the GPS data.
2. Segmentation: Trajectories are segmented into sequences of moving and stopped segments to capture human mobility patterns more effectively.
3. Instance creation: Fixed-size instances of size N=100 data points are created from the segmented trajectories.
4. Feature extraction: Statistical and spectral features are calculated from each instance in both the time and frequency domains.

This process results in a set of 80 features: 40 capturing characteristics in the time domain and 40 in the frequency domain.

The resulting distribution of these instances is presented in Table 7.1. It is clear that SMF2016 presents class imbalance. For example, in the training set, the majority classes foot and car, account for nearly 75% while the minority classes bike and metro put together contribute just 9%. In contrast, Geolife exhibits a fairly balanced class distribution, with the majority classes (foot and bus) comprising around 53%, while the minority classes (bike and car) contribute approximately 28%.

Table 7.1: Proportion of instances by transportation mode in each dataset.

		Transportation Mode				
Dataset		Foot	Bike	Bus	Car	Metro
SMF2016	Train set	22.63	3.73	16.43	51.92	5.30
	Test set	17.73	3.69	17.44	59.28	1.86
Geolife	Train set	29.31	12.52	23.96	15.33	18.89
	Test set	31.87	23.54	21.86	12.11	10.62

7.3.2 Imbalanced Handling Techniques

To address the class imbalance observed in the transportation mode distribution of the SMF2016 dataset, four imbalanced handling techniques are employed:

1. SMOTE (Synthetic Minority Oversampling Technique) [24]: This method creates synthetic minority class samples by selecting a random minority sample, finding its k nearest neighbours, and generating a new data point along the line segment between the sample and a randomly chosen neighbour.

2. AdaBoost (Adaptive Boosting) [55]: This boosting technique combats imbalanced data by iteratively training weak learners. AdaBoost focuses on previously misclassified classes during each iteration, ultimately building a stronger ensemble classifier.

3. SMOTEBoost [25]: This hybrid technique combines SMOTE's data generation with AdaBoost's weighting strategy. SMOTEBoost balances the class distribution by creating synthetic minority samples, while AdaBoost utilises weights to train stronger learners for imbalanced data.

4. DECOC (Diversified Error Correcting Output Codes) [15]: This technique tackles multiclass imbalanced data by training a diverse ensemble of algorithms. DECOC selects the best performing learner for each data point, creating a more powerful classifier.

7.4 Experimental Analysis

7.4.1 Baseline Model

This study employs a Random Forest (RF) classifier as the baseline model. Random Forest is a well-established ensemble learning technique that has proven effective in various classification tasks [16]. It operates by combining the predictions from a multitude of randomised decision trees, demonstrably achieving superior overall performance compared to a single decision tree. This ensemble approach capitalises on the principle that an aggregation of weaker models can collectively outperform a single, highly complex model. Random Forest leverages a technique known as bagging, where each tree within the forest is trained on a bootstrapped subset of the training data. The final prediction for a new instance is made by aggregating the votes from all trees in the forest, resulting in a more robust and generalisable model.

Random Forest offers a limited number of hyperparameters requiring tuning, we deviate from the default settings solely for the number of trees in the forest. In this study, we employ a forest comprising 5,000 trees (i.e., `n_estimators=5000`).

7.4.2 Evaluation Metrics

While accuracy and error rate are commonly used metrics for evaluating classification performance, their application in imbalanced datasets can be misleading. This study utilises precision,

recall, and F-measure [131] as the primary evaluation metrics. These metrics are calculated based on the values derived from confusion matrix, as described in Section 5.4. We additionally employ the area under the curve - receiver operating characteristics (AUC-ROC) [48]. The ROC curve is a graphical visualisation that depicts a model's performance across all possible classification thresholds.

7.4.3 Experimental Setup

We conducted the experiments in 4 different setups as detailed below:

7.4.3.1 Experiment 1: Baseline

This experiment establishes a baseline by training the RF classifier described in Section 7.4.1 on both datasets. We employ the data in original form without resampling to preserve the class distribution within each dataset. This allows us to assess the models performance on the datasets using the evaluation metrics defined in Section 7.4.2. Additionally, for subsequent experiments where Geolife is resampled to create imbalanced datasets, we train a new RF classifier just before employing the imbalanced learning technique. This allow us to assess the impact of the technique on the resampled data.

The classification results in Table 7.2 demonstrate significant performance disparity between the SMF2016 and Geolife datasets. This observation underscores the challenges associated with imbalanced datasets. For instance, the bike class, a minority class in both datasets but with a lower proportion in SMF2016, exhibits generally better performance in the Geolife dataset (precision: 84%, recall: 68%, F1: 75%) compared to SMF2016 (precision: 97%, recall: 22%, F1: 36%). Similarly, the model completely misclassifies the metro class instances in SMF2016 (0% for all metrics), whereas the car class, another minority class in Geolife, achieves a reasonable F1-score of 55%. Notably, resampling the Geolife dataset to address imbalance did not significantly impact model performance compared to the original data in either case.

7.4.3.2 Experiment 2: Addressing Imbalance in SMF2016

This experiment uses imbalanced learning techniques to address the imbalance in SMF2016. As shown in Table 7.3, data oversampling using SMOTE and RF classifier did not yield significant improvements compared to the baseline. DECOC showed improvement for bike prediction (recall: 22% to 37%, F1: 36% to 52%) without significantly affecting the majority car and foot class predictions. AdaBoost, on the other hand, exhibited a stability in recall score for bike (22% to 21%) but resulted in a decrease in F1-score (36% to 17%) due to a precision-recall trade-off. Notably, SMOTEBoost emerged as the most effective technique for minority class predictions. It maintained a stable recall score for bike (22% to 24%) while achieving a good F1-score (28%). Considering the weighted average metric values, the overall performance of the models is only marginally better than the baseline.

Table 7.2: Baselines classification results.

	SMF2016			Geolife (original)			Geolife (experiment 3)			Geolife (experiment 4)		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Foot	0.60	0.93	0.73	0.69	0.93	0.79	0.63	0.96	0.76	0.65	0.94	0.77
Bike	0.97	0.22	0.36	0.84	0.68	0.75	0.91	0.49	0.63	0.92	0.51	0.65
Bus	0.13	0.05	0.08	0.71	0.71	0.71	0.64	0.77	0.70	0.70	0.57	0.63
Car	0.74	0.80	0.77	0.67	0.47	0.55	0.85	0.28	0.42	0.38	0.66	0.48
Metro	0.00	0.00	0.00	0.77	0.54	0.63	0.73	0.57	0.64	0.75	0.19	0.31
Weighted avg.	0.57	0.60	0.58	0.73	0.72	0.71	0.73	0.68	0.66	0.70	0.64	0.63
AUC-ROC		75.23			88.14			87.80			87.71	

Table 7.3: Experiment 2: Using imbalanced techniques with the SMF2016.

	AdaBoost			DECOC			RF + SMOTE			SMOTEBoost		
Mode	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Foot	0.48	0.75	0.58	0.60	0.93	0.73	0.58	0.94	0.72	0.57	0.71	0.63
Bike	0.14	0.21	0.17	0.86	0.37	0.52	0.84	0.25	0.39	0.35	0.24	0.28
Bus	0.27	0.32	0.29	0.10	0.06	0.08	0.17	0.07	0.10	0.20	0.20	0.20
Car	0.64	0.21	0.31	0.74	0.75	0.74	0.75	0.80	0.77	0.73	0.66	0.69
Metro	0.01	0.18	0.02	0.02	0.00	0.00	0.00	0.00	0.00	0.06	0.11	0.07
Weighted avg.	0.52	0.32	0.35	0.59	0.63	0.60	0.61	0.66	0.62	0.58	0.56	0.57

7.4.3.3 Experiment 3: Gelife Resampled

This experiment investigates the impact of class imbalance in the Geolife dataset. We hypothesise the high misclassification error in SMF2016 is primarily caused by class imbalance. We therefore resample the Geolife dataset to a majority-minority class distribution similar to SMF2016. We anticipate substantial decrease on the model's performance.

Specifically, we oversampled the majority class (foot) using SMOTE to match the proportion of the highest majority (car) in SMF2016. The second most frequent class (bus), exceeds the proportion of the second-highest class in SMF2016 (foot). We therefore employ random undersampling (RUS) to downsample it. Furthermore, RUS is applied to downsample all other classes to achieve a distribution similar to SMF2016. The specific resampling proportions used are detailed in Table 7.4.

Table 7.4: Geolife class distribution in the resampled experiments

	Original	Experiment 3		Experiment 4	
	(%)	Proportion	Method	Proportion	Method
Foot	29.31	51.92	SMOTE	22.63	RUS
Bike	12.52	3.73	RUS	3.73	RUS
Bus	23.96	22.63	RUS	16.43	RUS
Car	15.33	5.30	RUS	51.92	SMOTE
Metro	18.89	16.43	RUS	5.30	RUS

The classification results using imbalanced learning techniques for this experiment are presented in Table 7.5. While the performance metrics exhibit a decrease compared to the baseline (precision: 73%, recall: 68%, F1-score: 66%), the extent of this degradation varies across techniques. For instance, AdaBoost demonstrates a 26% decrease in F1-score (40% vs. 66%). DECOC performs even worse. However, when considering individual class prediction compared to the SMF2016 results in Table 7.3, the degradation is not as severe as initially anticipated.

Table 7.5: Experiment 3: Geolife resampled.

	AdaBoost			DECOC			RF + SMOTE			SMOTEBoost		
Mode	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Foot	0.64	0.38	0.48	0.59	0.78	0.67	0.70	0.92	0.80	0.69	0.89	0.78
Bike	0.73	0.51	0.60	0.20	0.24	0.22	0.85	0.70	0.77	0.78	0.67	0.72
Bus	0.25	0.12	0.16	0.24	0.02	0.04	0.69	0.73	0.71	0.54	0.67	0.60
Car	0.28	0.61	0.39	0.41	0.03	0.06	0.73	0.42	0.54	0.50	0.25	0.33
Metro	0.20	0.52	0.28	0.01	0.45	0.02	0.74	0.58	0.65	0.43	0.22	0.29
Weighted avg.	0.48	0.40	0.40	0.40	0.18	0.17	0.74	0.73	0.72	0.63	0.64	0.62

7.4.3.4 Experiment 4: Geolife to SMF2016 Class Equivalence

This experiment is similar to Experiment 3, but we resample each class to match the exact proportion of the corresponding class in SMF2016 (Exp. 4 in Table 7.4). The same resampling techniques (SMOTE or RUS) are applied appropriately to achieve the distribution. The corresponding classification results for all the classifiers is presented in Table 7.6.

Surprisingly, compared to SMF2016, the resampled Geolife classification results have not shown a significant decrease in performance (as reported in Table 7.2) except for AdaBoost and DECOC. The majority class (Car) now shows much worse results on all metrics when compared to SMF2016 or GeoLife unsampled, where it was a minority class. than The minority classes (Bike, Bus) degrade w.r.t the original dataset, but still can be better learnt than on the SMF2016 dataset. This suggests that factors beyond class imbalance might be influencing the predictions on SMF2016. Therefore, we will further investigate the possibility of class overlap.

Table 7.6: Experiment 4: Geolife resampled to SMF2016 class distribution.

	AdaBoost			DECOC			RF + SMOTE			SMOTEBoost		
Mode	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Foot	0.64	0.39	0.49	0.59	0.89	0.71	0.71	0.91	0.80	0.70	0.86	0.77
Bike	0.64	0.58	0.61	0.43	0.21	0.28	0.81	0.73	0.77	0.69	0.69	0.69
Bus	0.25	0.13	0.17	0.37	0.05	0.09	0.73	0.65	0.69	0.63	0.58	0.60
Car	0.19	0.43	0.26	0.46	0.12	0.19	0.57	0.56	0.56	0.48	0.42	0.44
Metro	0.24	0.45	0.31	0.01	0.30	0.02	0.77	0.50	0.61	0.46	0.29	0.35
Weighted avg.	0.46	0.39	0.40	0.46	0.25	0.27	0.73	0.72	0.72	0.63	0.64	0.63

7.4.4 Class Overlap Evaluation

Using `pymfe` [7], a Python meta-feature extractor package, we assess class overlap using two categories of metrics: *structural overlap* and *feature overlap*.

- Structural Overlap: We assess structural overlap, which characterises the complexity of class and information pertaining the internal structure of classes (data morphology). We compute the following metrics:
  - *Fraction of borderline points (N1):* the proportion of data points that lie close to the decision boundary between classes.
  - *Fraction of hyperspheres covering data (T1):* the compactness of class clusters.
  - *Local set average cardinality (LSAvg):* the average number of nearest neighbours for each data point.
- Feature Overlap: characterises the similarity of individual features between classes in the data domain. Here, we calculate the following metrics:

- *Maximum Fisher's discriminant ratio (F1):* measures the separability between classes by projecting data points perpendicularly using all possible features combination.
- *Directional vector maximum Fisher's discriminant ratio (F1v):* this variant of F1 searches for a vector that maximises the separation between classes.

In all metrics, a high value indicates strong overlap. The exception is LSAvg, where a low value suggests strong overlap. The calculated class overlap metrics are presented in Table 7.7.

Table 7.7: Evaluation of class overlap in datasets.

	Metric	SMF2016	Geolife (original)	Geolife (Exp. 3)	Geolife (Exp. 4)
Structural Overlap	N1	0.44	0.46	0.37	0.47
	T1	0.56	0.81	0.88	0.89
	LSCAvg	0.99	0.99	0.99	0.99
Feature Overlap	F1**	0.86	0.85	–	–
	F1v**	0.20	0.16	0.12	0.13

** *mean features value* – *fail to compute for some features (nan)*

While the class overlap metrics in Table 7.7 provide a global view, they don't reveal individual feature behaviour. To gain a deeper understanding, we examine the individual feature distributions of Fisher's discriminant ratio (Figures. 7.1 and 7.2). These distributions show that highly discriminative features (speed features identified in our previous study [91]) have higher values in SMF2016 and lower in Geolife. The higher values observed for these features in SMF2016 suggest greater overlap, and potentially the lower values in Geolife suggest better separability. The combined effect of class imbalance and class overlap are significant contributors to the poor classification performance observed in SMF2016.

7.5 Summary

This study underlines the critical role of balanced class distributions in achieving robust transportation mode classification models. Our work demonstrates that a baseline random forest model, whilst effectively learnt on a relatively balanced dataset, suffers significantly when presented with an imbalanced one. Our exploration of various imbalance-handling techniques to mitigate this effect yielded only marginal improvements in results.

Resampling a fairly balanced dataset to such a degree of imbalance suggests the potential influence of other classification difficulty factors, possibly in conjunction with class imbalance itself. We hypothesise that class overlap may be such a factor, and therefore assessed the degree of class overlap from both data domain structural and feature perspectives. Maximum Fisher's discriminant ratio analysis indicates that feature overlap, particularly amongst the most discriminative features, may be a significant source of the observed weak separability between classes.

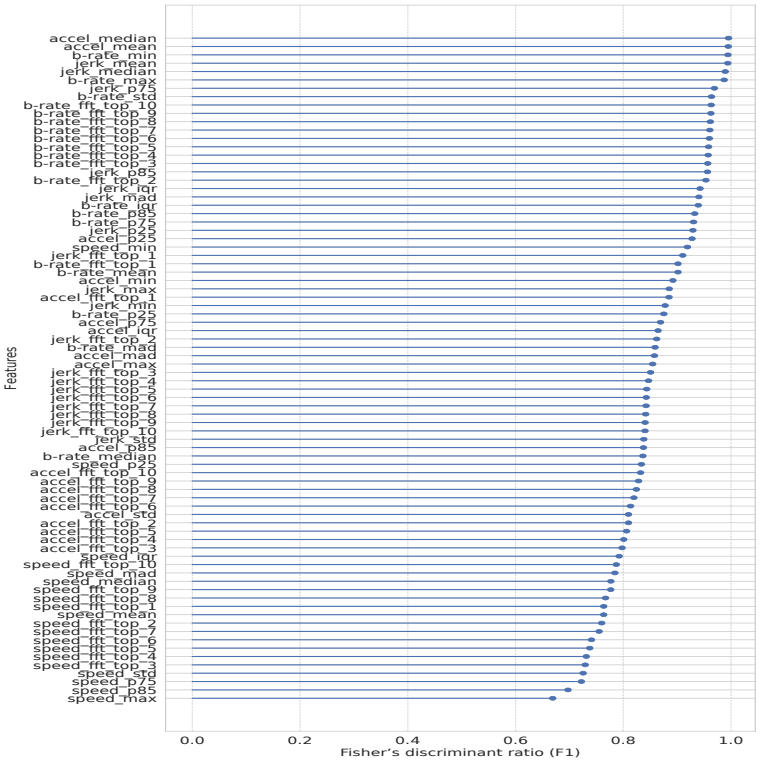


Figure 7.1: SMF2016 Fisher discriminant scores of features.

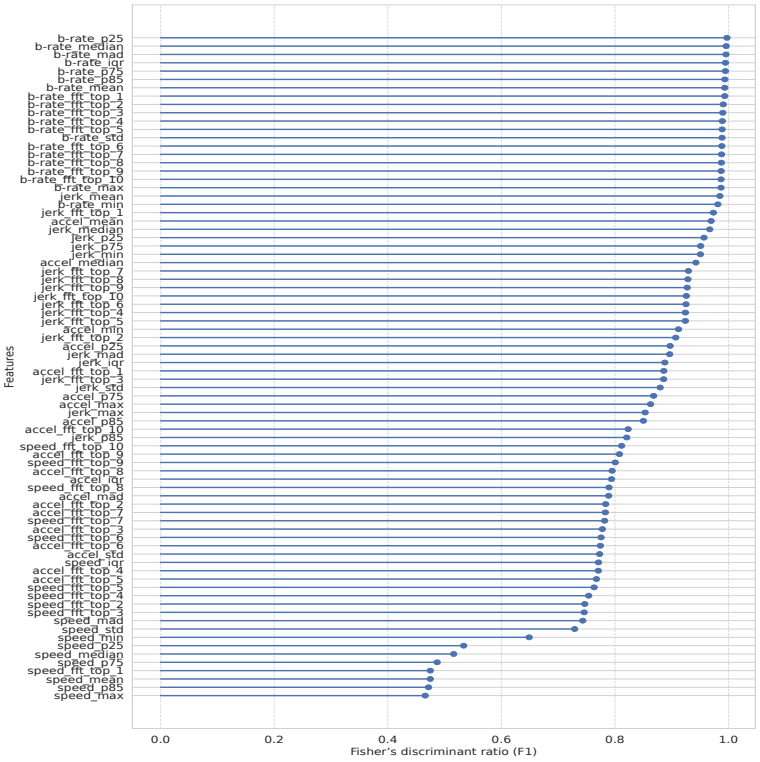


Figure 7.2: Geolfe Fisher discriminant scores of features.

In conclusion, we emphasise the importance of exploring a broader range of classification difficulty factors, in addition to class imbalance, when building transportation mode classification models. This will ensure the development of more robust and generalisable models.

Chapter 8

Conclusion

In conclusion, this thesis investigated the multifaceted challenges of accurately detecting transportation modes using solely GPS trajectory data. The primary objective was to explore and enhance existing transportation mode detection methodologies by addressing critical issues: optimal feature selection, the impact of class imbalance, and the potential benefits of hierarchical classification. The key contributions of this study are summarised in the following.

8.1 Summary of Findings

- **Class-subspace feature selection**
  This work, detailed in Chapter 5 focused on optimising feature engineering and selection for TMD. The results demonstrated the effectiveness of combining time-domain and frequency-domain features in enhancing classification accuracy. Traditional feature selection methods, such as sequential forward selection, often select a single set of features common for all classes. However, the importance of different features can vary significantly across different transportation modes.

  This research introduced a novel feature selection method, the Class-Subspace feature selection. The proposed method leverages the Shapley values, a game-theoretic concept, to assess the individual contribution of each feature to the classification accuracy of each class. By analysing the Shapley values for each feature and each class, it is possible to identify the most relevant features for each specific transportation mode. This class-specific feature selection approach can effectively tailor the feature set to the unique characteristics of each mode, leading to significant improvements in classification accuracy compared to traditional methods.

- **Hierarchical classification approach to mode detection**
  This work, presented in Chapter 6, delved into the potential advantages of employing a hierarchical classification approach for transportation mode detection. The existing body of

TMD literature primarily focuses on standard flat classification models. However, transportation modes often exhibit inherent hierarchical relationships or infrastucture. For instance, "car" mode is closely related to a "bus," class than it is to a "metro" or "bike". Recognising and exploiting these hierarchical relationships can potentially enhance classification accuracy and provide more nuanced insights into travel patterns.

To this end, this research proposed a novel hierarchical classification framework, termed *HiClass4MD*. *HiClass4MD* leverages the inherent hierarchical structure within the transportation mode space. It begins by training a flat classifier on the available dataset. Subsequently, the misclassification errors of this initial classifier are analysed to identify patterns and relationships between misclassified instances. This analysis informs the construction of a hierarchical structure. Finally, a series of specialised classifiers are trained at each level (internal node) of the hierarchy, effectively refining the classification process and mitigating the impact of common misclassification errors.

This hierarchical approach offers several potential benefits. By decomposing the complex TMD problem into a series of simpler sub-problems, *HiClass4MD* can potentially improve overall classification accuracy. Furthermore, the hierarchical structure provides a valuable framework for understanding the underlying relationships between different transportation modes and identifying the specific challenges associated with distinguishing between certain modes.

References

[1] Charu C. Aggarwal. *An Introduction to Neural Networks*, pages 1–52. Springer International Publishing, Cham, 2018. ISBN 978-3-319-94463-0. doi: 10.1007/978-3-319-94463-0_1. URL https://doi.org/10.1007/978-3-319-94463-0_1.

[2] Ana Aguiar and João Rodrigues. Sensemycity: a mobile iot tool for researching intelligent urban mobility. In *2022 14th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 725–733. IEEE, 2022.

[3] Md Golam Rabiul Alam, 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*, 24(4):4671–4681, 2023.

[4] Md Mustakin Alam, Tanjim Ahmed, Meraz Hossain, Mehedi Hasan Emo, Md Kausar Islam Bidhan, Md Tanzim Reza, Md Golam Rabiul Alam, Mohammad Mehedi Hassan, Francesco Pupo, and Giancarlo Fortino. Federated ensemble-learning for transport mode detection in vehicular edge network. *Future Generation Computer Systems*, 149:89–104, 2023.

[5] F Taia Alaoui, Hassen Fourati, Alain Kibangou, Bogdan Robu, and N Vuillerme. Urban transportation mode detection from inertial and barometric data in pedestrian mobility. *IEEE Sensors Journal*, 22(6):4772–4780, 2021.

[6] Abdulwahab Alazeb, Danyal Khan, and Ahmad Jalal. Intelligent transportation activity recognition using deep belief network. In *2024 4th Interdisciplinary Conference on Electrics and Computer (INTCEC)*, pages 1–6. IEEE, 2024.

[7] Edesio Alcobaça, Felipe Siqueira, Adriano Rivolli, Luís PF Garcia, Jefferson T Oliva, and André CPLF De Carvalho. Mfe: Towards reproducible meta-feature extraction. *Journal of Machine Learning Research*, 21(111):1–5, 2020.

[8] Esra'a Alshdaifat, Frans Coenen, and Keith Dures. Hierarchical classification for solving multi-class problems: A new approach using naive bayesian classification. In *Advanced Data Mining and Applications: 9th International Conference, ADMA 2013, Hangzhou, China, December 14-16, 2013, Proceedings, Part I 9*, pages 493–504. Springer, 2013.

[9] Thiago Andrade and João Gama. How are you riding? transportation mode identification from raw gps data. In *EPIA Conference on Artificial Intelligence*, pages 648–659. Springer, 2022.

[10] Guven Asci and M Amac Guvensan. A novel input set for lstm-based transport mode detection. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 107–112. IEEE, 2019.

[11] Huthaifa I Ashqar, Mohammed H Almannaa, Mohammed Elhenawy, Hesham A Rakha, and Leanna House. Smartphone transportation mode recognition using a hierarchical machine learning classifier and pooled features from time and frequency domains. *IEEE Transactions on Intelligent Transportation Systems*, 20(1):244–252, 2018.

[12] Behrang Assemi, Hamid Safi, Mahmoud Mesbah, and Luis Ferreira. Developing and validating a statistical model for travel mode identification on smartphones. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):1920–1931, 2016.

[13] Danya Bachir, Ghazaleh Khodabandelou, Vincent Gauthier, Mounim El Yacoubi, and Eric Vachon. Combining bayesian inference and clustering for transport mode detection from sparse and noisy geolocation data. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 569–584. Springer, 2018.

[14] Thanos Bantis and James Haworth. Who you are is how you travel: A framework for transportation mode detection using individual and environmental characteristics. *Transportation Research Part C: Emerging Technologies*, 80:286–309, 2017.

[15] Jingjun Bi and Chongsheng Zhang. An empirical comparison on state-of-the-art multi-class imbalance learning algorithms and a new diversified ensemble learning scheme. *Knowledge-Based Systems*, 158:81–93, 2018.

[16] Gérard Biau and Erwan Scornet. A random forest guided tour. *Test*, 25:197–227, 2016.

[17] Filip Biljecki. Automatic segmentation and classification of movement trajectories for transportation modes. In *Workshop on Modelling Moving Objects. Informatics Institute, University of Amsterdam, Science Park*, 2010.

[18] Filip Biljecki, Hugo Ledoux, and Peter Van Oosterom. Transportation mode-based segmentation and classification of movement trajectories. *International Journal of Geographical Information Science*, 27(2):385–407, 2013.

[19] Wendy Bohte and Kees Maat. Deriving and validating trip purposes and travel modes for multi-day gps-based travel surveys: A large-scale application in the netherlands. *Transportation Research Part C: Emerging Technologies*, 17(3):285–297, 2009.

[20] Adel Bolbol, Tao Cheng, Ioannis Tsapakis, and James Haworth. Inferring hybrid transportation modes from sparse gps data using a moving window svm classification. *Computers, Environment and Urban Systems*, 36(6):526–537, 2012.

[21] Remco R Bouckaert. Properties of bayesian belief network learning algorithms. In *Uncertainty in Artificial Intelligence*, pages 102–109. Elsevier, 1994.

[22] Andrea Capponi, Claudio Fiandrino, Burak Kantarci, Luca Foschini, Dzmitry Kliazovich, and Pascal Bouvry. A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities. *IEEE communications surveys & tutorials*, 21(3):2419–2465, 2019.

[23] Giuseppe Cardone, Luca Foschini, Paolo Bellavista, Antonio Corradi, Cristian Borcea, Manoop Talasila, and Reza Curtmola. Fostering participaction in smart cities: a geo-social crowdsensing platform. *IEEE Communications Magazine*, 51(6):112–119, 2013.

[24] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

[25] Nitesh V Chawla, Aleksandar Lazarevic, Lawrence O Hall, and Kevin W Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases: PKDD 2003: 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003. Proceedings 7*, pages 107–119. Springer, 2003.

[26] Cynthia Chen, Jingtao Ma, Yusak Susilo, Yu Liu, and Menglin Wang. The promises of big data and small data for travel behavior (aka human mobility) analysis. *Transportation research part C: emerging technologies*, 68:285–299, 2016.

[27] Huey-Kuo Chen, Hsiao-Ching Ho, Luo-Yu Wu, Ian Lee, and Huey-Wen Chou. Two-stage procedure for transportation mode detection based on sighting data. *Transportmetrica A: transport science*, 20(1):2118558, 2024.

[28] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[29] Tianqi Chen and Tong He. Higgs boson discovery with boosted trees. In *NIPS 2014 workshop on high-energy physics and machine learning*, pages 69–80. PMLR, 2015.

[30] Jie Cheng and Russell Greiner. Learning bayesian belief network classifiers: Algorithms and system. In *Advances in Artificial Intelligence: 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2001 Ottawa, Canada, June 7–9, 2001 Proceedings 14*, pages 141–151. Springer, 2001.

[31] Sina Dabiri and Kevin Heaslip. Inferring transportation modes from GPS trajectories using a convolutional neural network. *Transportation research part C: emerging technologies*, 86:360–371, 2018.

[32] Sina Dabiri, Chang-Tien Lu, Kevin Heaslip, and Chandan K Reddy. Semi-supervised deep learning approach for transportation mode identification using gps trajectory data. *IEEE Transactions on Knowledge and Data Engineering*, 32(5):1010–1023, 2019.

[33] Somayeh Danafar, Michal Piorkowski, and Krzysztof Krysczcuk. Bayesian framework for mobility pattern discovery using mobile network events. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 1070–1074. IEEE, 2017.

[34] Rahul Deb Das and Stephan Winter. A fuzzy logic based transport mode detection framework in urban environment. *Journal of Intelligent Transportation Systems*, 22(6):478–489, 2018.

[35] Francisco González de Cossío and Guillaume Sabiron. A robust approach for transportation mode detection using smartphone-based gps sensors and road network information. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 959–965. IEEE, 2023.

[36] Merkebe Getachew Demissie, Santi Phithakkitnukoon, Titipat Sukhvibul, Francisco Antunes, Rui Gomes, and Carlos Bento. Inferring passenger travel demand to improve urban mobility in developing countries using cell phone data: a case study of senegal. *IEEE Transactions on intelligent transportation systems*, 17(9):2466–2478, 2016.

[37] Misha Denil and Thomas Trappenberg. Overlap versus imbalance. In *Advances in Artificial Intelligence: 23rd Canadian Conference on Artificial Intelligence, Canadian AI 2010, Ottawa, Canada, May 31–June 2, 2010. Proceedings 23*, pages 220–231. Springer, 2010.

[38] Fan Ding, Yongyi Zhang, Jiankun Peng, Yuming Ge, Tao Qu, Xingyuan Tao, and Jun Chen. A hybrid method for intercity transport mode identification based on mobility features and sequential relations mined from cellular signaling data. *Computer-Aided Civil and Infrastructure Engineering*, 2024.

[39] Shi Dong, Ping Wang, and Khushnood Abbas. A survey on deep learning and its applications. *Computer Science Review*, 40:100379, 2021.

[40] Ifigenia Drosouli, Athanasios Voulodimos, Georgios Miaoulis, Paris Mastorocostas, and Djamchid Ghazanfarpour. Transportation mode detection using an optimized long short-term memory model on multimodal sensor data. *Entropy*, 23(11):1457, 2021.

[41] Ifigenia Drosouli, Athanasios Voulodimos, Paris Mastorocostas, Georgios Miaoulis, and Djamchid Ghazanfarpour. Tmd-bert: A transformer-based model for transportation mode detection. *Electronics*, 12(3):581, 2023.

[42] Sumanto Dutta and Bidyut Kr Patra. Inferencing transportation mode using unsupervised deep learning approach exploiting gps point-level characteristics. *Applied Intelligence*, 53 (10):12489–12503, 2023.

[43] Naveen Eluru, Vincent Chakour, and Ahmed M El-Geneidy. Travel mode choice and transit route choice behavior in montreal: insights from mcgill university members commute patterns. *Public Transport*, 4:129–149, 2012.

[44] Yuki Endo, Hiroyuki Toda, Kyosuke Nishida, and Akihisa Kawanobe. Deep feature extraction from trajectories for transportation mode estimation. In *Advances in Knowledge Discovery and Data Mining: 20th Pacific-Asia Conference, PAKDD 2016, Auckland, New Zealand, April 19-22, 2016, Proceedings, Part II 20*, pages 54–66. Springer, 2016.

[45] Martina Erdelić, Tonči Carić, Edouard Ivanjko, and Niko Jelušić. Classification of travel modes using streaming gnss data. *Transportation Research Procedia*, 40:209–216, 2019.

[46] Mohammad Etemad, Amilcar Soares Junior, and Stan Matwin. On feature selection and evaluation of transportation mode prediction strategies. *arXiv preprint arXiv:1808.03096*, 2018.

[47] Kunkun Fan, Daichao Li, Xinlei Jin, and Sheng Wu. A multi-scale attributes fusion model for travel mode identification using gps trajectories. *Geo-spatial Information Science*, pages 1–15, 2024.

[48] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[49] Tao Feng and Harry JP Timmermans. Transportation mode recognition using gps and accelerometer data. *Transportation Research Part C: Emerging Technologies*, 37:118–130, 2013.

[50] Yang Feng, Min Zhou, and Xin Tong. Imbalanced classification: A paradigm-based review. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 14(5):383–406, 2021.

[51] Zhenni Feng and Yanmin Zhu. A survey on trajectory data mining: Techniques and applications. *IEEE Access*, 4:2056–2067, 2016.

[52] Francesc J Ferri, Pavel Pudil, Mohamad Hatef, and Josef Kittler. Comparative study of techniques for large-scale feature selection. In *Machine intelligence and pattern recognition*, volume 16, pages 403–413. Elsevier, 1994.

[53] Peter Flach. *Machine learning: the art and science of algorithms that make sense of data*. Cambridge university press, 2012.

[54] Damian Frej, Paweł Grabski, Rafał S Jurecki, and Emilia M Szumska. Experimental study on longitudinal acceleration of urban buses and coaches in different road maneuvers. *Sensors*, 23(6):3125, 2023.

[55] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[56] Kemilly Dearo Garcia, Cláudio Rebelo de Sá, Mannes Poel, Tiago Carvalho, João Mendes-Moreira, João MP Cardoso, André CPLF de Carvalho, and Joost N Kok. An ensemble of autonomous auto-encoders for human activity recognition. *Neurocomputing*, 439:271–280, 2021.

[57] Santosh Giri, Ruben Brondeel, Tarik El Aarbaoui, and Basile Chaix. Application of machine learning to predict transport modes from gps, accelerometer, and heart rate data. *International Journal of Health Geographics*, 21(1):19, 2022.

[58] Hristijan Gjoreski, Mathias Ciliberto, Lin Wang, Francisco Javier Ordonez Morales, Sami Mekki, Stefan Valentin, and Daniel Roggen. The university of sussex-huawei locomotion and transportation dataset for multimodal analytics with mobile devices. *IEEE Access*, 6: 42592–42604, 2018.

[59] Martin Gjoreski, Vito Janko, Gašper Slapničar, Miha Mlakar, Nina Reščič, Jani Bizjak, Vid Drobnič, Matej Marinko, Nejc Mlakar, Mitja Luštrek, et al. Classical and deep learning methods for recognizing human activities and modes of transportation with smartphone sensors. *Information Fusion*, 62:47–62, 2020.

[60] Raed Abdullah Hasan, Hafez Irshaid, Fadi Alhomaidat, Sangwoo Lee, and Jun-Seok Oh. Transportation mode detection by using smartphones and smartwatches with machine learning. *KSCE Journal of Civil Engineering*, 26(8):3578–3589, 2022.

[61] Guenter W Hein. Status, perspectives and trends of satellite navigation. *Satellite Navigation*, 1(1):22, 2020.

[62] Hayden S Helm, Weiwei Yang, Sujeeth Bharadwaj, Kate Lytvynets, Oriana Riva, Christopher White, Ali Geisa, and Carey E Priebe. Inducing a hierarchy for multi-class classification problems. *arXiv preprint arXiv:2102.10263*, 2021.

[63] Ye Hong, Emanuel Stüdeli, and Martin Raubal. Evaluating geospatial context information for travel mode detection. *Journal of Transport Geography*, 113:103736, 2023.

[64] Haosheng Huang, Yi Cheng, and Robert Weibel. Transport mode detection based on mobile phone network data: A systematic review. *Transportation Research Part C: Emerging Technologies*, 101:297–312, 2019.

[65] Anke Huss, Johan Beekhuizen, Hans Kromhout, and Roel Vermeulen. Using gps-derived speed patterns for recognition of transport modes in adults. *International journal of health geographics*, 13:1–8, 2014.

[66] John N Ivan, Thomas Jonsson, and Attila Borsos. Motor vehicle speeds: Recommendations for urban sustainability. *Transportation research record*, 2301(1):1–8, 2012.

[67] Arash Jahangiri and Hesham A Rakha. Applying machine learning techniques to transportation mode recognition using mobile phone sensor data. *IEEE transactions on intelligent transportation systems*, 16(5):2406–2417, 2015.

[68] JQ James. Semi-supervised deep ensemble learning for travel mode identification. *Transportation Research Part C: Emerging Technologies*, 112:120–135, 2020.

[69] JQ James. Travel mode identification with gps trajectories using wavelet transform and deep learning. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):1093–1103, 2020.

[70] Zhihuan Jiang, Ailing Huang, Geqi Qi, and Wei Guan. A framework of travel mode identification fusing deep learning and map-matching algorithm. *IEEE Transactions on Intelligent Transportation Systems*, 24(6):6401–6415, 2023.

[71] A Kalatian and Y Shafahi. Travel mode detection exploiting cellular network data. matec web conf. 81, 03008, 2016.

[72] Utkarsh Mahadeo Khaire and R Dhanalakshmi. Stability of feature selection algorithm: A review. *Journal of King Saud University-Computer and Information Sciences*, 34(4): 1060–1073, 2022.

[73] Murad Khan, Bilal Jan, Haleem Farman, Fasee Ullah, Ihtesham Ul Islam, Abdul Hanan Abdullah, and Atif Khan. Future of big data and deep learning for wireless body area networks. *Deep Learning: Convergence to Big Data Analytics*, pages 53–77, 2019.

[74] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. *Logistic regression*. Springer, 2002.

[75] Mateusz Lango and Jerzy Stefanowski. What makes multi-class imbalanced problems difficult? an experimental study. *Expert Systems with Applications*, 199:116962, 2022.

[76] Guanyao Li, Chun-Jie Chen, Sheng-Yun Huang, Ai-Jou Chou, Xiaochuan Gou, Wen-Chih Peng, and Chih-Wei Yi. Public transportation mode detection from cellular data. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 2499–2502, 2017.

[77] Ji Li, Xin Pei, Xuejiao Wang, Danya Yao, Yi Zhang, and Yun Yue. Transportation mode identification with gps trajectory data and gis information. *Tsinghua Science and Technology*, 26(4):403–416, 2021.

[78] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50 (6):1–45, 2017.

[79] Linchao Li, Jiasong Zhu, Hailong Zhang, Huachun Tan, Bowen Du, and Bin Ran. Coupled application of generative adversarial networks and conventional neural networks for travel mode detection using gps data. *Transportation Research Part A: Policy and Practice*, 136: 282–292, 2020.

[80] Xiaoyuan Liang and Guiling 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)*, pages 338–342. IEEE, 2017.

[81] Xiaoyuan Liang, Yuchuan Zhang, Guiling Wang, and Songhua Xu. A deep learning model for transportation mode detection based on smartphone sensing data. *IEEE Transactions on Intelligent Transportation Systems*, 21(12):5223–5235, 2019.

[82] Miao Lin, Wen-Jing Hsu, and Zhuo Qi Lee. Detecting modes of transport from unlabelled positioning sensor data. *Journal of Location Based Services*, 7(4):272–290, 2013.

[83] Hongbin Liu and Ickjai Lee. End-to-end trajectory transportation mode classification using bi-lstm recurrent neural network. In *2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, pages 1–5. IEEE, 2017.

[84] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4766–4777, December 4–9 2017.

[85] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

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

[87] Christos Markos and JQ James. Unsupervised deep learning for gps-based transportation mode identification. In *2020 IEEE 23rd international conference on intelligent transportation systems (ITSC)*, pages 1–6. IEEE, 2020.

[88] João Moreira, Andre Carvalho, and Tomás Horvath. *A general introduction to data analytics*. John Wiley & Sons, 2018.

[89] Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. From attribution to selection: Harnessing SHAP values for class-subspace feature selection in transportation mode detection. Submitted to International Journal of Data Science and Analytics.

[90] Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. Transportation mode detection from gps data: A data science benchmark study. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 3726–3731. IEEE, 2021.

[91] Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. Inferring transportation mode using pooled features from time and frequency domains. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3985–3990. IEEE, 2023.

[92] Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. *HiClass4MD*: A hierarchical classifier for transportation mode detection. In *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2024.

[93] Akilu Rilwan Muhammad, Ana Aguiar, and João Mendes-Moreira. Characterising class imbalance in transportation mode detection: An experimental study. In Vicente Julian et al., editors, *Intelligent Data Engineering and Automated Learning – IDEAL 2024*, volume 15347 of *Lecture Notes in Computer Science*. Springer, Cham, 2025. doi: 10.1007/978-3-031-77738-7_6. URL https://doi.org/10.1007/978-3-031-77738-7_6.

[94] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

[95] Asif Nawaz, Huang Zhiqiu, Wang Senzhang, Yasir Hussain, Amara Naseer, Muhammad Izhar, and Zaheer Khan. Mode inference using enhanced segmentation and pre-processing on raw global positioning system data. *Measurement and Control*, 53(7-8):1144–1158, 2020.

[96] Ju-Youn Park and Jong-Hwan Kim. Incremental class learning for hierarchical classification. *IEEE transactions on cybernetics*, 50(1):178–189, 2018.

[97] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[98] Xiaohui Pei, Xianjun Yang, Tao Wang, Zenghui Ding, Yang Xu, Lin Jia, and Yining Sun. Travel-mode inference based on gps-trajectory data through multi-scale mixed attention mechanism. *Heliyon*, 10(15), 2024.

[99] Santi Phithakkitnukoon, Titipat Sukhvibul, Merkebe Demissie, Zbigniew Smoreda, Juggapong Natwichai, and Carlos Bento. Inferring social influence in transport mode choice using mobile phone data. *EPJ Data Science*, 6:1–29, 2017.

[100] Yingchun Qu, Hang Gong, and Pu Wang. Transportation mode split with mobile phone data. In *2015 IEEE 18th international conference on intelligent transportation systems*, pages 285–289. IEEE, 2015.

[101] Elias Rajaby and Sayed Masoud Sayedi. A structured review of sparse fast fourier transform algorithms. *Digital Signal Processing*, 123:103403, 2022.

[102] Sebastian Raschka. Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *Journal of open source software*, 3(24): 638, 2018.

[103] Sasank Reddy, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani Srivastava. Determining transportation mode on mobile phones. In *2008 12th IEEE international symposium on wearable computers*, pages 25–28. IEEE, 2008.

[104] Ricardo Ribeiro, Alina Trifan, and António JR Neves. A deep learning approach for transportation mode identification using a transformation of gps trajectory data features into an image representation. *International Journal of Data Science and Analytics*, pages 1–10, 2024.

[105] Sebastien Richoz, Lin Wang, Philip Birch, and Daniel Roggen. Transportation mode recognition fusing wearable motion, sound, and vision sensors. *IEEE Sensors Journal*, 20(16): 9314–9328, 2020.

[106] Joao GP Rodrigues, Ana Aguiar, Fausto Vieira, Joao Barros, and Joao P Silva Cunha. A mobile sensing architecture for massive urban scanning. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1132–1137. IEEE, 2011.

[107] João GP Rodrigues, Ana Aguiar, and João Barros. Sensemycity: Crowdsourcing an urban sensor. *arXiv preprint arXiv:1412.2070*, pages 1–10, 12 2014.

[108] João GP Rodrigues, João P Pereira, and Ana Aguiar. Impact of crowdsourced data quality on travel pattern estimation. In *Proceedings of the First ACM Workshop on Mobile Crowdsensing Systems and Applications*, pages 38–43, 2017.

[109] Lior Rokach and Oded Maimon. Decision trees. *Data mining and knowledge discovery handbook*, pages 165–192, 2005.

[110] Avipsa Roy, Daniel Fuller, Trisalyn Nelson, and Peter Kedron. Assessing the role of geographic context in transportation mode detection from gps data. *Journal of transport geography*, 100:103330, 2022.

[111] Paria Sadeghian, Xiaoyun Zhao, Arman Golshan, and Johan Håkansson. A stepwise methodology for transport mode detection in gps tracking data. *Travel Behaviour and Society*, 26:159–167, 2022.

[112] Paria Sadeghian, Arman Golshan, Mia Xiaoyun Zhao, and Johan Håkansson. A deep semi-supervised machine learning algorithm for detecting transportation modes based on gps tracking data. *Transportation*, pages 1–21, 2024.

[113] Miriam Seoane Santos, Pedro Henriques Abreu, Nathalie Japkowicz, Alberto Fernández, Carlos Soares, Szymon Wilk, and João Santos. On the joint-effect of class imbalance and overlap: a critical review. *Artificial Intelligence Review*, 55(8):6207–6275, 2022.

[114] Pedro M Santos, João GP Rodrigues, Susana B Cruz, Tiago Lourenço, Pedro M d'Orey, Yunior Luis, Cecília Rocha, Sofia Sousa, Sérgio Crisóstomo, Cristina Queirós, et al. Portolivinglab: An iot-based sensing platform for smart cities. *IEEE Internet of Things Journal*, 5(2):523–532, 2018.

[115] Anke Sauerländer-Biebl, Elmar Brockfeld, David Suske, and Eric Melde. Evaluation of a transport mode detection using fuzzy rules. *Transportation research procedia*, 25:591–602, 2017.

[116] Jürgen Schmidhuber and Sepp Hochreiter. Long short-term memory. *Neural Comput*, 9(8): 1735–1780, 1997.

[117] Nadine Schuessler and Kay W Axhausen. Processing raw data from global positioning systems without additional information. *Transportation Research Record*, 2105(1):28–36, 2009.

[118] Li Shen and Peter R Stopher. Review of gps travel survey and gps data-processing methods. *Transport reviews*, 34(3):316–334, 2014.

[119] Dongyoun Shin, Daniel Aliaga, Bige Tunçer, Stefan Müller Arisona, Sungah Kim, Dani Zünd, and Gerhard Schmitt. Urban sensing: Using smartphones for transportation mode classification. *Computers, Environment and Urban Systems*, 53:76–86, 2015.

[120] Carlos N Silla and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data mining and knowledge discovery*, 22:31–72, 2011.

[121] Daniel Silva-Palacios, Cesar Ferri, and María José Ramírez-Quintana. Improving performance of multiclass classification by inducing class hierarchies. *Procedia Computer Science*, 108:1692–1701, 2017.

[122] Daniel Silva-Palacios, Cèsar Ferri, and María José Ramírez-Quintana. Probabilistic class hierarchies for multiclass classification. *Journal of computational science*, 26:254–263, 2018.

[123] Xuan Song, Hiroshi Kanasugi, and Ryosuke Shibasaki. Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level. In *Proceedings of the twenty-fifth international joint conference on artificial intelligence*, pages 2618–2624, 2016.

[124] Sajjad Sowlati, Rahim Ali Abbaspour, and Alireza Chehreghan. An approach to assess the role of features in detection of transportation modes. *Transportation*, pages 1–36, 2024.

[125] S. Spaccapietra, C. Parent, and L. Spinsanti. Trajectories and their representations. In Chiara Renso, Stefano Spaccapietra, and Esteban Zimányi, editors, *Mobility Data: Modeling, Management, and Understanding*, pages 3–22. Cambridge University Press, 2013. doi: 10.1017/CBO9781139128926.002.

[126] Leon Stenneth, Ouri Wolfson, Philip S Yu, and Bo Xu. Transportation mode detection using mobile phones and gis information. In *Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 54–63, 2011.

[127] Peter Stopher, Camden FitzGerald, and Jun Zhang. Search for a global positioning system device to measure person travel. *Transportation Research Part C: Emerging Technologies*, 16(3):350–369, 2008.

[128] Peter R. Stopher, Q. Jiang, and C. FitzGerald. Processing GPS data from travel surveys. In *2nd International Colloquium on Behavioral Foundations of Integrated Land-Use and Transportation Models: Frameworks, Models and Applications*, Toronto, Ontario, Canada, June 2005.

[129] Melania Susi, Valérie Renaudin, and Gérard Lachapelle. Motion mode recognition and step detection algorithms for mobile phone users. *Sensors*, 13(2):1539–1562, 2013.

[130] Siavash Taherinavid, Seyed Vahid Moravvej, Yen-Lin Chen, Jing Yang, Chin Soon Ku, and Lip Yee Por. Automatic transportation mode classification using a deep reinforcement learning approach with smartphone sensors. *IEEE Access*, 2023.

[131] Jafar Tanha, Yousef Abdi, Negin Samadi, Nazila Razzaghi, and Mohammad Asadpour. Boosting methods for multi-class imbalanced data classification: an experimental review. *Journal of Big data*, 7:1–47, 2020.

[132] Department of Economic United Nations and Population Division Social Affairs. *World Population Prospects 2019: Highlights*. ST/ESA/SER.A/423. United Nations, New York, 2019.

[133] U.S. Government. GPS.gov – official u.s. government information about the global positioning system (gps) and related topics. http://www.gps.gov/, 2006. [Online; accessed September 2023].

[134] Thaddeus Vincenty. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey review*, 23(176):88–93, 1975.

[135] Katja Vogel. What characterizes a "free vehicle" in an urban area? *Transportation research part F: traffic psychology and behaviour*, 5(1):15–29, 2002.

[136] Bao Wang, Linjie Gao, and Zhicai Juan. Travel mode detection using gps data and socioeconomic attributes based on a random forest classifier. *IEEE Transactions on Intelligent Transportation Systems*, 19(5):1547–1558, 2017.

[137] Benjamin X Wang and Nathalie Japkowicz. Boosting support vector machines for imbalanced data sets. *Knowledge and information systems*, 25:1–20, 2010.

[138] Huayong Wang, Francesco Calabrese, Giusy Di Lorenzo, and Carlo Ratti. Transportation mode inference from anonymized and aggregated mobile phone call detail records. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 318–323. IEEE, 2010.

[139] Jun Wang, Karen K Dixon, Hainan Li, and Jennifer Ogle. Normal acceleration behavior of passenger vehicles starting from rest at all-way stop-controlled intersections. *Transportation Research Record*, 1883(1):158–166, 2004.

[140] Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Sami Mekki, Stefan Valentin, and Daniel Roggen. Enabling reproducible research in sensor-based transportation mode recognition with the sussex-huawei dataset. *IEEE Access*, 7:10870–10891, 2019.

[141] Lin Wang, Hristijan Gjoreski, Mathias Ciliberto, Paula Lago, Kazuya Murao, Tsuyoshi Okita, and Daniel Roggen. Summary of the sussex-huawei locomotion-transportation recognition challenge 2020. In *Adjunct proceedings of the 2020 ACM international joint conference on pervasive and ubiquitous computing and proceedings of the 2020 ACM international symposium on wearable computers*, pages 351–358, 2020.

[142] Min Wang, Huan Liu, Jing He, Chengchuan An, Jingxin Xia, and Zhenbo Lu. Bayesian network learning framework for travel mode identification based on cellular signaling data. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2991–2997. IEEE, 2023.

[143] Senzhang Wang, Jiannong Cao, and S Yu Philip. Deep learning for spatio-temporal data mining: A survey. *IEEE transactions on knowledge and data engineering*, 34(8):3681–3700, 2020.

[144] Gary M Weiss. Foundations of imbalanced learning. *Imbalanced learning: Foundations, algorithms, and applications*, pages 13–41, 2013.

[145] Guangnian Xiao, Qin Cheng, and Chunqin Zhang. Detecting travel modes using rule-based classification system and gaussian process classifier. *IEEE Access*, 7:116741–116752, 2019.

[146] Guangnian Xiao, Qin Cheng, and Chunqin Zhang. Detecting travel modes from smartphone-based travel surveys with continuous hidden markov models. *International Journal of Distributed Sensor Networks*, 15(4):1550147719844156, 2019.

[147] Guangnian Xiao, Qin Cheng, and Chunqin Zhang. Detecting travel modes using rule-based classification system and gaussian process classifier. *IEEE Access*, 7:116741–116752, 2019.

[148] Zhibin Xiao, Yang Wang, Kun Fu, and Fan Wu. Identifying different transportation modes from trajectory data using tree-based ensemble classifiers. *ISPRS International Journal of Geo-Information*, 6(2):57, 2017.

[149] Dafeng Xu, Guojie Song, Peng Gao, Rongzeng Cao, Xinwei Nie, and Kunqing Xie. Transportation modes identification from mobile phone data using probabilistic models. In *Advanced Data Mining and Applications: 7th International Conference, ADMA 2011, Beijing, China, December 17-19, 2011, Proceedings, Part II 7*, pages 359–371. Springer, 2011.

[150] Xiaoyu Xu. Automatic classification of transportation modes using smartphone sensors: addressing imbalanced data and enhancing training with focal loss and artificial bee colony algorithm. *Journal of Optics*, pages 1–15, 2024.

[151] Mofeng Yang, Yixuan Pan, Aref Darzi, Sepehr Ghader, Chenfeng Xiong, and Lei Zhang. A data-driven travel mode share estimation framework based on mobile device location data. *Transportation*, 49(5):1339–1383, 2022.

[152] Ningkang Yang, Christelle Al Haddad, Iuliia Yamnenko, and Constantinos Antoniou. Machine learning for data-centric transport mode detection: A systematic review. *Available at SSRN 4960556*.

[153] Wenhao Yu and Guanwen Wang. Graph based embedding learning of trajectory data for transportation mode recognition by fusing sequence and dependency relations. *International Journal of Geographical Information Science*, 37(12):2514–2537, 2023.

[154] Jiaqi Zeng, Yi Yu, Yong Chen, Di Yang, Lei Zhang, and Dianhai Wang. Trajectory-as-a-sequence: A novel travel mode identification framework. *Transportation Research Part C: Emerging Technologies*, 146:103957, 2023.

[155] Jiaqi Zeng, Yulang Huang, Guozheng Zhang, Zhengyi Cai, and Dianhai Wang. Travel mode identification for non-uniform passive mobile phone data. *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[156] Ziyi Zeng, Guanwen Wang, Yifan Zhang, Qingfeng Guan, and Wenhao Yu. Masked graph autoencoder for self-supervised transportation mode recognition. *Transactions in GIS*, 2024.

[157] Chenhan Zhang, Yuanshao Zhu, Christos Markos, Shui Yu, and JQ James. Toward crowdsourced transportation mode identification: A semisupervised federated learning approach. *IEEE Internet of Things Journal*, 9(14):11868–11882, 2021.

[158] Wei Zhao, Tarun Joshi, Vijayan N Nair, and Agus Sudjianto. Shap values for explaining cnn-based text classification models. *arXiv preprint arXiv:2008.11825*, 2020.

[159] Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, and Wei-Ying Ma. Understanding mobility based on gps data. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 312–321, 2008.

[160] Yu Zheng, Like Liu, Longhao Wang, and Xing Xie. Learning transportation mode from raw gps data for geographic applications on the web. In *Proc. 17th Int. Conf. World Wide Web*, pages 247–256, 2008.

[161] Yu Zheng, Longhao Wang, Ruochi Zhang, Xing Xie, and Wei-Ying Ma. Geolife: Managing and understanding your past life over maps. In *The Ninth International Conference on Mobile Data Management (mdm 2008)*, pages 211–212. IEEE, 2008.

[162] Yu Zheng, Yukun Chen, Quannan Li, Xing Xie, and Wei-Ying Ma. Understanding transportation modes based on gps data for web applications. *ACM Transactions on the Web (TWEB)*, 4(1):1–36, 2010.

[163] Yu Zheng, Xing Xie, Wei-Ying Ma, et al. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.

[164] Yu Zheng, Hao Fu, Xing Xie, Wei-Ying Ma, and Quannan Li. *Geolife GPS trajectory dataset - User Guide*, geolife gps trajectories 1.1 edition, July 2011. URL https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/.

[165] Qiuhui Zhu, Min Zhu, Mingzhao Li, Min Fu, Zhibiao Huang, Qihong Gan, and Zhenghao Zhou. Identifying transportation modes from raw gps data. In *Social Computing: Second International Conference of Young Computer Scientists, Engineers and Educators, ICYCSEE 2016, Harbin, China, August 20-22, 2016, Proceedings, Part I 2*, pages 395–409. Springer, 2016.

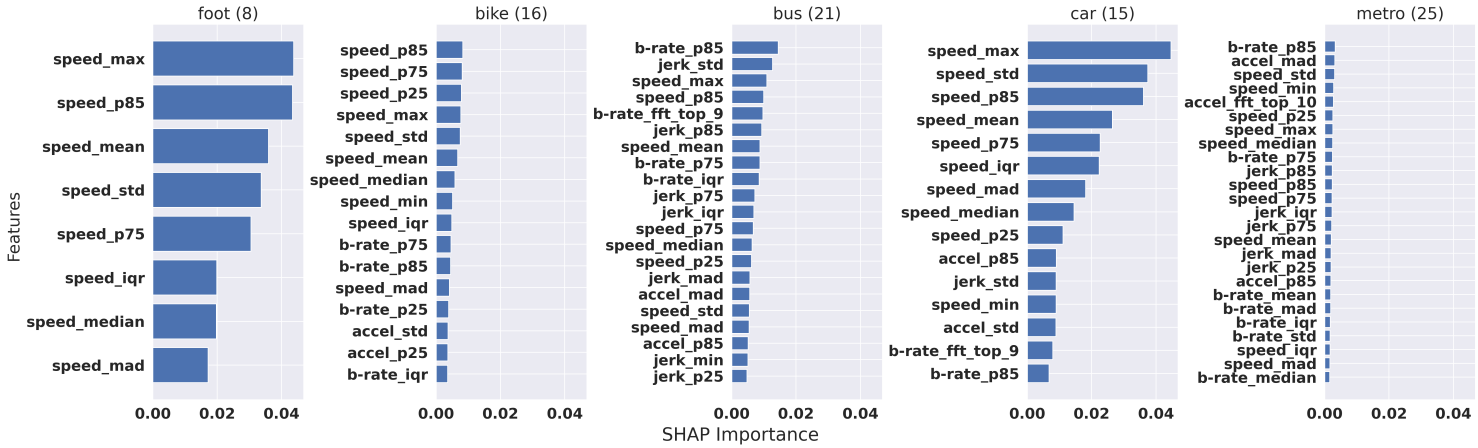
[166] Yuanshao Zhu, Yi Liu, JQ James, and Xingliang Yuan. Semi-supervised federated learning for travel mode identification from gps trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 23(3):2380–2391, 2021.

Appendix A

Features Selected with Class-subspace Selection

Subspace Selected Features (SMF2016 dataset)

(a) Subspace feature selection using Random Forest.



(b) Subspace feature selection using XGBoost.

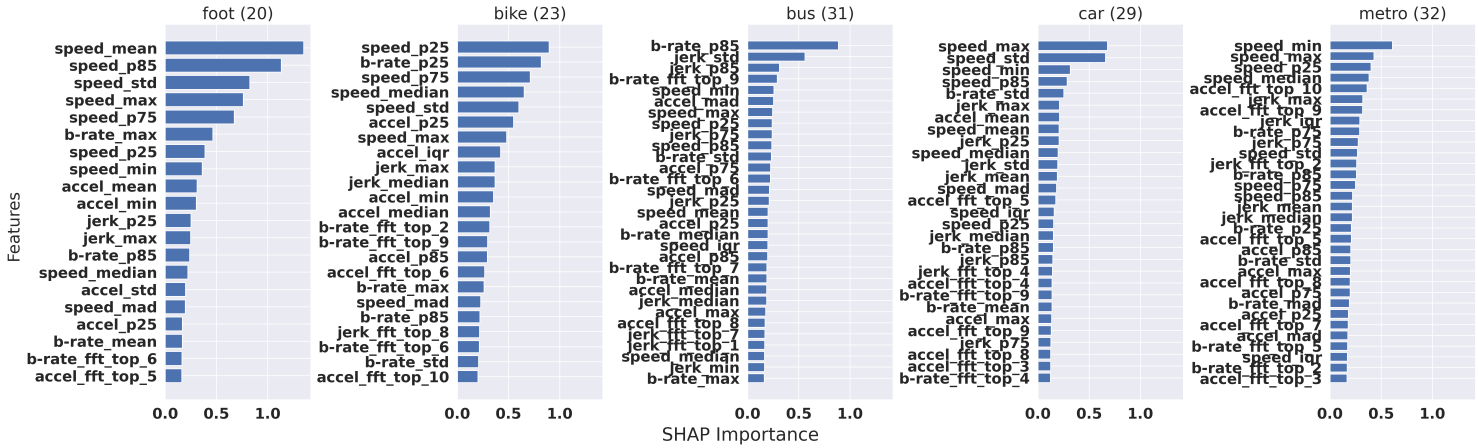
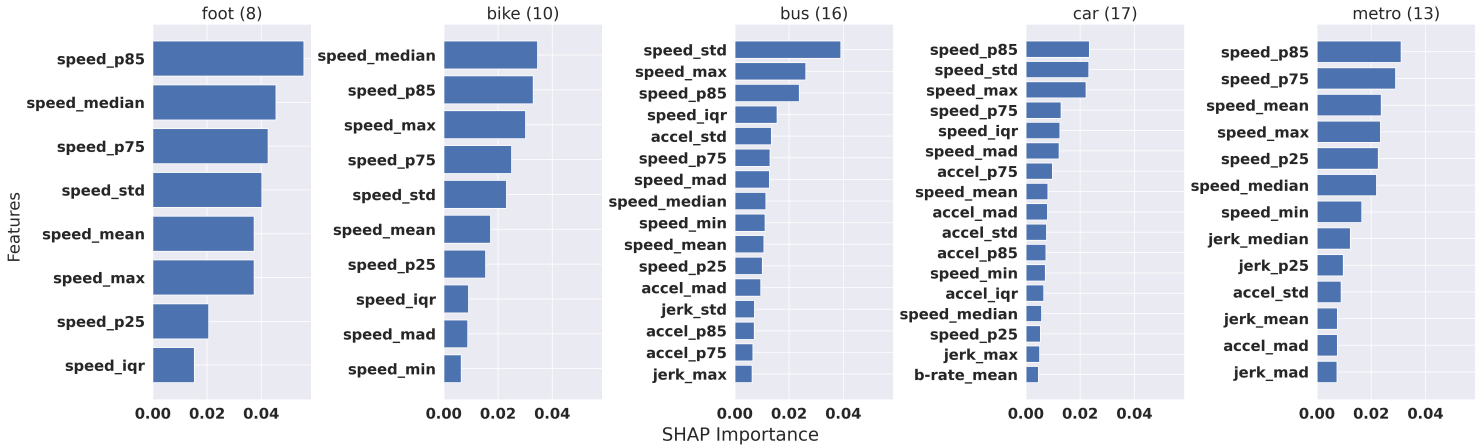


Figure A.1: Features selected by the Subspace algorithm for the SMF2016 dataset using **(a)** Random Forest and **(b)** XGBoost for $\rho = 0.6$.

Subspace Selected Features (Geolife dataset)

(a) Subspace feature selection using Random Forest.



(b) Subspace feature selection using XGBoost.

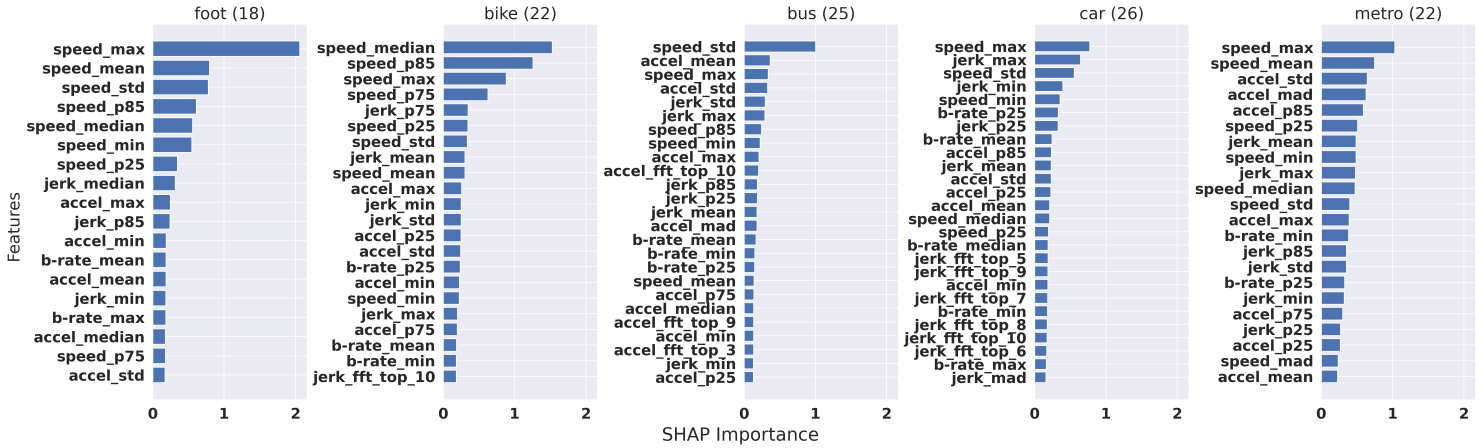


Figure A.2: Features selected by the Subspace algorithm for the Geolife dataset using **(a)** Random Forest and **(b)** XGBoost for $\rho = 0.6$.

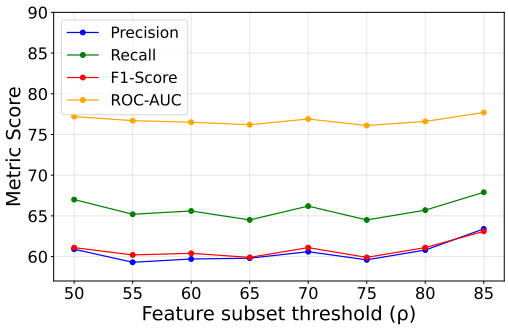
Appendix B

Sensitivity of the Subspace threshold (ρ)

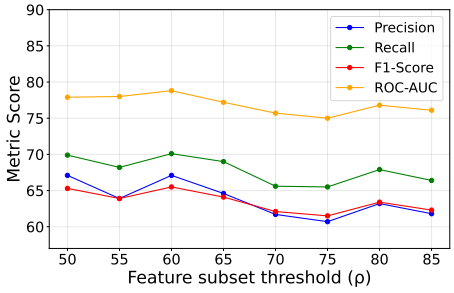
This analysis compares RF and XGB classifier performance across both datasets as we varied ρ between 50% and 85% of the total feature contribution. The Geolife dataset demonstrates remarkable robustness, with all metrics remaining highly stable across the entire range of ρ for both RF and XGB. This suggests that the feature representation in Geolife is inherently less sensitive to the specific threshold used. Conversely, the SMF dataset exhibits greater sensitivity, particularly with the XGB classifier.

While both RF and XGB maintain stability on the robustnessGeolife dataset, the SMF dataset highlights divergent behaviours. XGB exhibits greater sensitivity to ρ adjustments, characterised by noticeable fluctuations in precision, recall, and F1-score. Conversely, RF demonstrates relatively stable performance on SMF, despite minor variations in recall.

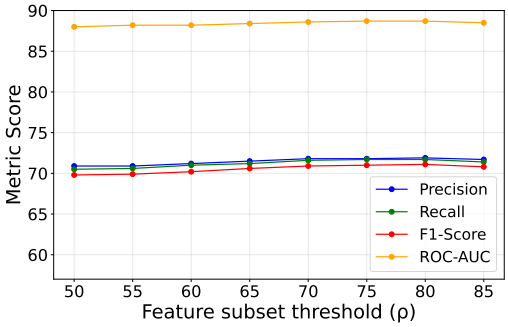
Overall, the analysis identifies potential areas for optimisation for the SMF dataset, e.g., the slight dips in performance observed around ρ values of 70-75% for XGB. These fluctuations suggest that the choice of ρ in this range can significantly impact the model's ability to balance precision and recall. The consistently high performance in the Geolife dataset, particularly the high and stable ROC-AUC scores around 89%, indicates that the subspace threshold does not significantly affect the model's ability to distinguish between classes.



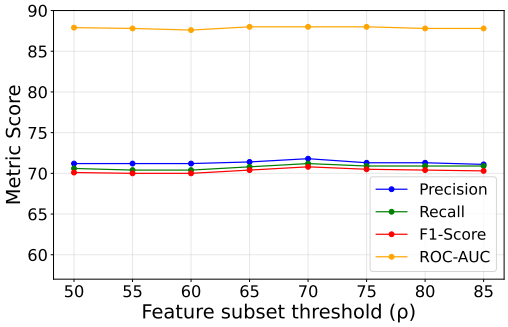
(a) SFM2016 with RF classifier.



(b) SMF2016 with XGB classifier.



(c) Geolife with RF classifier.



(d) Geolife with XGB classifier.

Figure B.1: Sensitivity of the subspace threshold (ρ).