6-2-2017

Travel Mode Identification with Smartphone Sensors

Xing Su
The Graduate Center, City University of New York

Travel Mode Identification with Smartphone Sensors

by

Xing Su

A dissertation submitted to the Graduate Faculty in Computer Science department in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

2017

This manuscript has been read and accepted for the Graduate Faculty in Computer Science department in satisfaction of the dissertation requirements for the degree of Doctor of Philosophy.

__________	____________________
Date	Hanghang Tong Chair of Examining Committee
__________	____________________
Date	Robert Haralick Executive Office

Supervisory Committee:

Hanghang Tong

Ted Brown

Qing He

Ping Ji

Zhigang Zhu

THE CITY UNIVERSITY OF NEW YORK

Abstract

Travel Mode Identification with Smartphone Sensors

by

Xing Su

Advisor: Hanghang Tong

Personal trips in a modern urban society typically involve multiple travel modes. Recognizing a traveller's transportation mode is not only critical to personal context-awareness in related applications, but also essential to urban traffic operations, transportation planning, and facility design. While the state of the art in travel mode recognition mainly relies on large-scale infrastructure-based fixed sensors or on individuals' GPS devices, the emergence of the smartphone provides a promising alternative with its ever-growing computing, networking, and sensing powers.

In this thesis, we propose new algorithms for travel mode identification using smartphone sensors. The prototype system is built upon the latest Android and iOS platforms with multimodality sensors. It takes smartphone sensor data as the input, and aims to identify six travel modes: *walking, jogging, bicycling, driving a car, riding a bus, taking a subway.* The methods and algorithms presented in our work are guided by two key design principles. First, careful consideration of smartphones' limited computing resources and batteries should be taken. Second, careful balancing of the following dimensions (i) user-adaptability, (ii) energy efficiency, and (iii) computation speed.

There are three key challenges in travel mode identification with smartphone sensors, stemming from the three steps in a typical mobile mining procedure. They are (C1) data capturing and preprocessing, (C2) feature engineering, and (C3) model training and adaptation. This thesis is our response to the challenges above.

To address the first challenge (C1), in Chapter 4 we develop a smartphone app that collects a multitude of smartphone sensor measurement data, and showcase a comprehensive set of de-noising techniques. To tackle challenge (C2), in Chapter 5 we design feature extraction methods that carefully balance prediction accuracy, computation time, and battery consumption. And to answer challenge (C3), in Chapters 6, 7, and 8, we design different learning models to accommodate different situations in model training. A hierarchical model with dynamic sensor selection is designed to address the energy consumption issue. We propose a personalized model that adapts to each traveller's specific travel behavior using limited labeled data. We also propose an online model for the purpose of addressing the model updating problem with large scaled data. In addressing the challenges and proposing solutions, this thesis provides an comprehensive study and gives a systematic solution for travel mode detection with smartphone sensors.

Acknowledgements

First I would like to express my sincere gratitude to my advisor Prof. Hanghang Tong for the continuous support of my PhD study and related research, for his immense knowledge and experience in guiding the work. His guidance helped me in all the time of research and writing of this thesis. I am grateful for his patience, motivation and encouragement throughtout the PhD study. I am very thankful for all of the time he spent on writing, editing and revising papers and his help in putting together this dissertation.

Besides my advisor, I would like to thank the rest of my thesis committee: Professor Ted Brown, Professor Qing He, Professor Ping Ji and Professor Zhigang Zhu, for their great help in working on this thesis with insightful advise, professional suggestions and comments.

I would like to also give my sincere thanks to Professor Robert Haralick, for his generous help in spending time work on problems with me, patiently explain the methodology and related knowledge to me whenever I needed. I learnt a lot while working with him.

Many thanks to my department Assistant Program Officer Ms. Dilvania Rodriguez and former APO Ms. Lina Garcia. They are kindhearted colleagues who were always there and offer help, support and advise for us. Their beautiful personality makes the computer science department like a big family.

Many people provided me critical assistant in this thesis work in help collecting data. I must thank Hernan Caceres, Yu Cui, Ming Ni, Zhenghua Zhang, Xiaogong Gong, Cindy

Wang, Jessi Chen, Li Zhong, Tingting Chen, and many anonymous volunteers for taking time out of their busy schedules and traveled over places in the city to collect useful data.

I thank Jonathan Ben-benjamin, Arthur Parzygnat, Suman Bhunia, Xiannian Fan and George Poppe, for the long inspirational discussions, for the weekends we worked together, and for all the fun we have had in the past few years. It was a great honor for me to spend PhD study with all of you.

I thank Dara Pir, Ali Ahmend, Anura Abeywickrama, Allan Zelener, Steve Vayl, Jamie Lennox for the emotional, mental and moral support during my PhD whenever I needed.

I thank all the friends I meet in computer science department and science center. To get to know all of you and have your company in this journey is one of the luckiest things that happened in my life. All of you makes my New York life wonderful and full experience. Whenever I think of the time I have spent together with you in the past years, the memory is full of joy and happiness.

Finally I would like to thank my family: my father, my mother and my younger sister, for their constant support and belief in me through thick and thin.

List of Figures

2.1 A Typical Procedure of Mobile Mining 8

3.1 Flowchart of Travel Mode Identification with Smartphones 18

4.1 iPhone App for Data Collection 24

4.2 x, y and z axis Acceleration Measure of iPhone [1] 25

4.3 x, y and z axis Rotation Measure of iPhone [1] 26

4.4 5% Winsorization of Acceleration Data along y axis 31

4.5 Gaussian Smoothed Rotation Rate 33

5.1 Learning Accuracy by Different Segment Size and Sampling Frequency . . . 36

5.2 Using Slide Window for Data Segmentation 37

5.3 Solution Path of Linear Model with Group-lasso Applied. The vertical dash lines are the moments where new group(s) of features join the whole features used for classification. Each line of red letter(s) are the total sensor group(s) in use with a blue dash arrows pointing to the dashline showing the moments when these group(s) of features start in use. 41

6.1 Acceleration along the Y-Axis: Walking vs. Taking a Bus 43

6.2 Hierarchical Categorization of Travel Modes 44

6.3 Hierarchical Online Learning Workflow 45

6.4 Comparison of Hierarchical Model and General Model 47

7.1 Overview of the Proposed *PerTMoD* 51

7.2 The effectiveness comparisons of *PerTMoD*. *PerTMoD* generally outperforms the compared methods. 60

7.3 The comparisons among *PerTMoD*, *Generic*, and *Base* methods with α varies. *PerTMoD* is consistently better. 61

8.1 Online VS Offline Learning Process 68

8.2 Performance Comparison of Online and Offline Model 69

8.3 The Time Cost of Online and Offline Updating 70

8.4 Prediction Performance with Initial Training Sets of Different Size 71

9.1 Driving Speed at Local and Freeway 73

9.2 Pressure at Summer and Winter on Bike 74

List of Algorithms

1 Hierarchical Framework with Dynamic Sensor Selection 46

2 The *PerTMoD* algorithm. 57

3 Online Learning with Pegasos 65

Contents

Acknowledgements **v**

List of Figures **viii**

List of Algorithms **ix**

1 Introduction **1**

1.1 Motivation 1

1.2 Contributions 4

1.3 Thesis Outlines 5

2 Literature Review **7**

2.1 Mobile Mining 7

2.1.1 Data Capturing with Smartphone Sensors 8

2.1.2 Algorithms 10

2.1.3 Applications 11

2.2 Travel Mode Identification System 13

2.3 Key Challenges 15

3 Thesis Overview **18**

4 Data Capturing and Processing **22**

4.1 Problem Description 22

4.2 Data Collection 23

4.2.1 Data Collection App 23

4.2.2 Data Description 23

4.3 Data de-noising 28

5 Feature Engineering **34**

5.1 Problem Description 34

5.2 Data Segmentation 35

5.3 Feature Extraction 36

5.4 Feature Analysis and Sensor Selection 38

6 Hierarchical Model **42**

6.1 Problem Description 42

6.2 A Hierarchical Framework 42

6.3 Dynamic Sensor Selection 44

6.3.1 Experiment 45

7 Personalized Model **48**

7.1 Problem Description 49

7.2 The Proposed *PerTMoD*: Overview 50

7.3 The Proposed *PerTMoD*: Details 52

7.3.1 Similarity Computation 52

7.3.2 Distribution Estimation 54

7.3.3 Sample Selection 55

7.3.4 Sample Reweighting 56

7.3.5 Algorithm Analysis 56

7.4 Experiment 58

7.4.1 Experiment Setup 59

7.4.2 Experiment Results 60

8 Online Model **62**

8.1 Problem Description 62

8.2 SVM Model 63

8.3 Online Learning 64

8.4 Experiment 65

8.4.1 Online Learning VS Offline Learning 67

8.4.2 How much data is enough to train a initial model? 68

8.5 A Summary of Performance 70

9 Conclusion and Future Work **72**

9.1 Discussion 72

9.1.1 Local Street or Freeway? 72

9.1.2 What's the weather like along the trip? 73

9.1.3 Daytime or Night? Sunny or Cloudy? 74

9.2 Contributions 75

9.3 Future Work 76

Chapter 1

Introduction

Smartphones are ubiquitous and becoming more and more sophisticated, with ever-growing computing, networking, and sensing powers. This has been changing the landscape of people's daily life and has opened the doors for many interesting data mining applications, ranging from health and fitness monitoring, personal biometric signature, assistive technology and elder-care, indoor localization and navigation, to urban computing and smart transportation, etc. A core building block behind these applications is the study of the raw sensor readings of the smartphones in order to learn the facts of activities, events, and situations where the users are involved together with smartphones. In this thesis we focus on the scenarios of identifying users' travel modes in urban transportation with smartphone sensors.

1.1 Motivation

Personal trips in a modern urban society usually involve multiple travel modes, including passenger cars, buses, subway, pedestrian, bicycles, etc. Different travel modes have their own specific characteristics, ranging from the travel speed, the volume, the fuel consumption, the emission usage, the priority level, to the vulnerability. Not only is recognizing transportation modes critical to understand people's travel behaviors [2], but also such information helps improve transportation planning, management and operations. For example, the personal mobility accounts for about two-thirds of the total transportation energy usage [3]. Understanding and assessing an individual's personal contribution to the emissions of a city requires personal travel diaries, including accurate information of travel modes. By leveraging travel modes information, traffic signal control systems are able to treat the travel modes in an integrated way to achieve optimal multi-modal traffic signal control [4–7]. Moreover, travel mode information constitutes an essential part of household travel diary data, which are indispensable for regional transportation planning. Recently, travel behavior has also become a focus for public health research. Studies on adults' and children's travel data have shown that individuals who walk or bike for transportation, or use public transportation, accumulate more physical activities and are more likely to meet public health recommendations [8].

Travel mode identification is a natural extension of vehicle classification, which only targets motorized transportation. There are many existing technologies for vehicle classification. The state of practice mostly leverages infrastructure based fixed sensors, including pneumatic tubes, inductive loop detectors, piezoelectric sensors, Weigh-in-motion (WIM) systems, radar sensors, infrared sensors, acoustic sensors, and computer vision-based sensors [9]. However, traditional fixed sensors bear the following limitations: (i) high installation and maintenance costs, (ii) inapplicability under specific situations (e.g. inclement weather) and (iii) inadequate to obtain travel mode information in a complete trip rather than at a few locations. In the past decade, given the reduced cost in Global Positioning System (GPS), more and more studies have focused on collecting personal travel data using GPS loggers [10].

GPS based floating sensors are more appealing by providing individual trip-chain data with extremely low costs. The drawback is that GPS only provides the location and speed data, and becomes inapplicable under certain scenarios (e.g., underground subways).

The emerging of the smartphones, as an integral part of the wearable devices with increasingly sophisticated technology and ever-growing computing, networking and sensing powers, quickly catches people's eyes. Nowadays, smartphones can provide much more than GPS location information. Indeed, modern smatrphones contain a myriad of sensors which are yet to be fully utilized. Such sensors include accelerometer, gravity sensor, barometer, light sensor, gyroscope, compass, etc. These advanced sensors, used in concert, bring a high potential to enable a variety of smartphone data mining applications such as users' activity recognition, including travel activities, from simple locomotion (e.g. walking, jogging, climbing stairs, taking elevator. etc.) [11] to complex activities (dining, shopping, etc.). Therefore, the smartphone has become one of the best sources for crowdsourcing real time dynamics while travelling.

In this thesis, we are particularly interested in exploring smartphones to automatically classify six different travel modes: ***driving a car, walking, jogging, bicycling, taking a bus*** and ***taking a subway***. Smartphones, as a special form of wearable devices, travel with the user and continuously collect personal travel activity information. Although the collection and sharing of massive mobile data needs to overcome institutional (e.g., who should collect the data), political (e.g., privacy issues), and technical challenges (e.g., bias of the collected samples), they do provide comprehensive information (e.g., vehicle traces) that promise great opportunities in many science and engineering fields [9]. As mentioned above, a smartphone could provide a variety of sensor data which can be conveniently processed to further obtain information concerning the motion of the device (e.g. acceleration, rotation) and the ambient environment (e.g. air pressure, ambient light). Different travel modes tend to have different characteristic speed variations, acceleration rates, and even environmental pressure, and magnetic field variations. This motivates us to use smartphone data for automatic travel mode identification.

1.2 Contributions

There are three key challenges in travel mode identification with smartphone sensors, stemming from the three steps in a typical mobile mining procedure, ranging from (C1) data capturing and preprocessing, (C2) feature engineering and (C3) model training and adaptation. In data capturing and preprocessing, the accelerometer sensor, as the exclusive data source in most existing smartphone-based activity recognition research, is inadequate to differentiate some travel modes. And the data collected from multimodality sensing often accompanies with various noise might overweigh its discriminative power. In feature engineering, the feature extraction needs to balance a number of potentially conflicting factors such as the recognition accuracy, the computational time, the response time as well as the battery consumption. In model training, the main challenges are battery consumption and model adaption. On the battery consumption, continuously sensing would drain the battery quickly. On the model adaption, how to adapt a travel mode recognition model trained on a generic population to a specific user? How to update/adapt a traveller-specific model efficiently over time to accommodate newly collected labelled data for her/him?

In response to these challenges, this thesis is developed to the following three research tasks. First (Task 1. Data Collection and Preprocessing), we develop a smartphone app which allows to collect a multitude of sensor measurement data from smartphones, followed by a comprehensive set of de-noising techniques. Second (Task 2. Feature Engineering), we design the feature extraction methods that carefully balance between the prediction accuracy, the on-line response time and the battery consumption. Third (Task 3. Learning Models), we design different learning models to accommodate different situations in model training. A hierarchical model with dynamic sensor selection is designed to address the energy consumption issue. A personalized model is proposed to adapt to each traveller's specific travel behavior with limited labeled data. An online model is proposed to address the model updating problem with large scaled data. This thesis, in addressing the challenges and proposing solutions, provides an comprehensive study and gives a systematic solution in travel mode detection with smartphone sensors.

Challenges	**Proposed Solutions**	**Publications**
C1.1: Exclusively accelerometer -based solution doesn't work.	PS1.1: Multi-modality sensing	[12, 13]
C1.2: Data noise introduced by the smartphone's mobility	PS1.2: Apply different de-noising techniques	[14]
C2.1: Data segmentation	PS2.1:Slide-window segmentation	[14, 15]
C2.2: Feature extraction	PS2.2: Dynamic sensor selection	[15]
C3.1: Continuous sensing are battery draining on smartphones	S3.1: Hierarchical learning model	[14, 15]
C3.2: User adaption: adapt generic trained model to a specific user.	PS3.2: Personalized learning model	[16]
C3.3: Temporal adaption: adapt the model trained from a limited data to future scenarios.	PS3.3 Online learning model	[14, 14]

Table 1.1: Thesis work and Publications

1.3 Thesis Outlines

The rest of the thesis organizes as follows. Chapter 2 introduces the details of smartphone sensors and conducts a brief literature review of the state of the art of smartphone related data mining research, followed by the open challenges. Chapter 3 gives an overview of the smartphone based travel mode identification system. Chapter 4 - 8 are the main thesis works following the three steps of a typical mobile mining procedure. Each section of the thesis work includes the statement of the problem, proposed solution and corresponding analysis. Chapter 9 concludes the thesis.

Chapter 2

Literature Review

In this section, we review the related work. We start with a generic review of the moblie mining, and then present the state of the art in travel mode recognition, followed by some key challenges for travel mode identification based on smartphone sensors.

2.1 Mobile Mining

As an integral part of our wearable devices, smartphones become increasingly sophisticated, with ever-growing computing, networking and sensing powers. They are usually equipped with multiple sensors including accelerometer, gravity sensor, barometer, light sensor, gyroscope and compass. The role of smartphones in our daily life is no longer limited to the texting-calling device, it is a body-worn sensor device with multi-modality sensing platform and powerful abilities in computing, mobility and connectivity. A smartphone can be used as an unobtrusive sensor in most applications where sensor-generated data are needed. It is very convenient in the case where the person already owns a smartphone that it is not necessary for them to carry additional devices for sensor data collection.

The development of smartphone techniques enables a rich variety of smartphone data mining applications. As is mentioned above, a smartphone can provide a variety of sensor data which can be easily processed to further obtain information concerning the motion of the device and the ambient environment. As we know, different activities tend to have different characteristics of motion patterns(e.g. speed variations, acceleration rates, rotation) and even the environmental patterns (e.g.magnetic field change and ambient air pressure). The smartphone provides us a great chance to explore various possibilities with it.

A typical mobile mining procedure includes 3 parts: data capturing, feature engineering and modeling. The Application sometimes provide feedback to the data capturing and modeling process to update the model. The procedure is shown in Figure. 7.1.

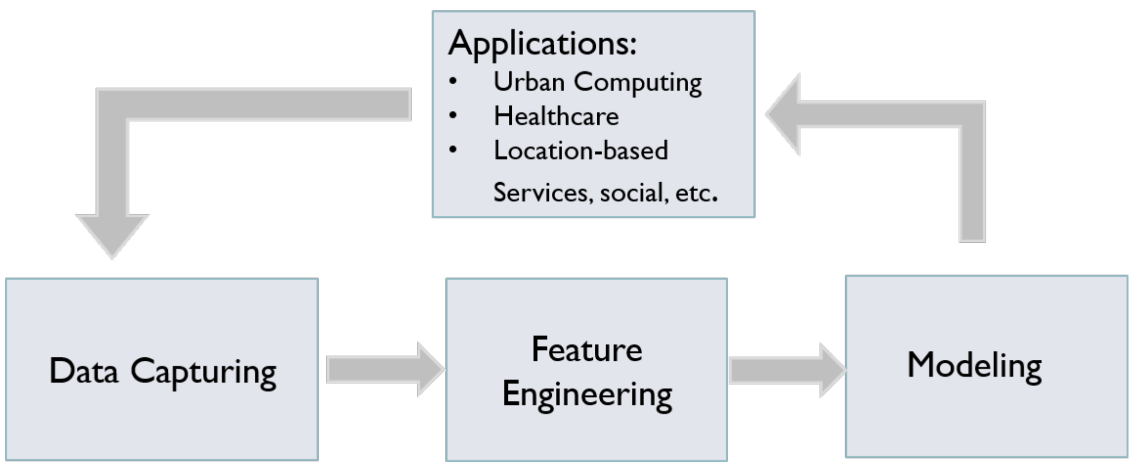


Figure 2.1: A Typical Procedure of Mobile Mining

2.1.1 Data Capturing with Smartphone Sensors

Data capturing is the process of using the smartphone to record data. These data can be the smartphone sensors' reading, the activity log by certain smartphone apps that record how the user is interactive with his/her phone, the network connection log from certain APs or Cellular towers, etc. The users' interaction with smartphones includes how often s/he uses the smartphone and at each interaction what is the activity (e.g. having a phone call, using an app, etc). Analyzing these data can help to understand the user's social life as well as the function the smartphone fulfill in everyday life. The connection log provides the information in the time and space perspective and is helpful in localization and other related scenarios. In this work, we mainly focus on the smartphone sensors' reading.

Smartphone sensors is classified into four categories [17]: motion sensors, environmental sensors, position sensors and connection sensors.

- **Motion Sensors.** These sensors measure acceleration forces and rotational forces along three axes of the phone's coordinates. This category includes accelerometers, gravity sensors, gyroscopes, and rotational vector sensors.

- **Environmental Sensors.** These sensors measure various environmental parameters, such as ambient air temperature and pressure, illumination, and humidity. This category includes barometers, photometers, and thermometers.

- **Position Sensors.** These sensors measure the physical position of a device. This category includes orientation sensors and magnetometers.

- **Connection Sensors.** These sensors provide the solution for smartphones to connect and interact with other devices with various protocols. This category includes Bluetooth, GPS sensors, Wireless sensors, standard cellular connection modulars.

A summary of the smartphone sensors and their sensing data is shown in Table 4.1. The data range and resolution are different from phone to phone. Some have very high resolution sensors and cost more. Some others return a estimated result indicating the level, indicating the exact number (e.g. some proximity sensor returns binary results indicating far or near to user's face).

Sensor Name	**Data Collected**	**Dimensions**	**Unit**
Accelerometer	Acceleration	x, y, z	g-force
Gravity Sensor	Gravity	x, y, z	m/s^2
Gyroscope	Rotation Rate	x, y, z, x_calibration, y_calibration, z_calibration	rad/s
Magnetometer	Magnetic Field	x, y, z, x_calibration, y_calibration, z_calibration	μT
Barometer	Ambient Air Pressure	1	hPa
Rotation Sensor	Rotation Degree (y axis pointing to magnetic north as the default)	Azimuch: Rotation around z Axis Pitch: Rotation around x Axis Roll: Rotation around y Axis	Degree
Proximity Senosr	Relative distance from an object to the view screen of a device	1	cm
Light Sensor	the ambient light level	1	lx
Humidity Sensor	the relative ambient humidity	1	Percentage
Temperature Sensor	Ambient temperature	1	Celcuis
GPS Sensor	Geographical description of current location and estimated speed	Latitude, Longitude, Speed	Degree, m/s

Table 2.1: Sensors' Data Details

In mobile sensing the critical challenge is energy efficiency. What sensors should be used and how to optimize the sensing (e.g. sampling frequency) so as to leverage the battery consumption and system performance is one of the main focus in many mobile sensing related researches and industrial development.

2.1.2 Algorithms

Smartphones, as the small, cheap, always connected devices with powerful sensing capabilities open up countless possibilities for data collection and analysis for a broad range of research. By analyzing the data from smartphones, handful informations can be discovered, such as where is this person, who is accompanying him, what is he doing, what resources are nearby, etc. The existing mobile mining literatures focuses on four areas [18]: Location-aware analytics, context aware analytics and social analytics. Location-awareness with smartphones are through smartphone's connectivity or interaction with ambient environment. Outdoor location information are obtained through GPS, cellular tower connection, etc. Indoor localization are usually under the constrain of no GPS and weak cellular reception. Thus other source of data are utilized such as the wifi connections and signal strength information [19–24], RSS-space tasks of proximity detection with acustic sensors [25–27], bluetooth connectivities [28], etc. Context awareness are the analytics of users' activities that is inferred from smartphones's data. Activity recognition with smartphones is the core technique of context awareness and is currently a hot topic. A low level activity recognition focuses on the task of detecting what is the user's current activity: walking on stairs, sitting, sleeping, driving or jogging? A high level activity recognition related research usually aggregate the activity information in the time manner and find some pattern. For example, the detected walking speed may reveals the mood of a person [29], the accelerations in driving may reveals the person's driving style [30], and the distribution of activities may indicate the user's living style [31]. In the aggregation level, the activities detected by smartphones is used for further mining of informations such as urban traffic volume [32, 33], events goers [32], carbon emission [34], etc. Social analytics research focuses on the aggregation of users' mobility and activities and it involves the social media network. A lot of research are using location based twitter data [35]. The interplay of human's mobility and social ties are modelled using smartphone data [36–38].

2.1.3 Applications

Mobile mining solutions can be applied either as the core method or an assistive technology in many applications. For example, learning the pattern of daily smartphone locations via cellular tower connection records in a city could help estimate the transportation volume and predict the traffic of the city. Smartphone based activity recognition could help the public health agency on family based outdoor activity survey. Based on the fields towards which mobile mining is applied, the main applications fall in the following four categories.

- **Urban computing.** Urban computing is an emerging field, which uses the data that is generated in cities from different sources such as traffic flow, human mobility, geographical data to help modernization of people's lives and tackle the urban issues such as traffic congestion, energy consumption, etc [39]. Smartphones play an important role in urban computing. On one hand, the interaction between humans and the urban environment is reflected by the smartphone and its connection to cellular towers and environment WiFis. The analysis of these data could help urban planning such as gleaning the underlying problems in transportation networks [40], discovering the functional regions and the city boundary [41,42], etc. One the other hand, smartphone sensors data are used to identify users' transportation modes. This lead to applications in urban transportation survey [34], daily traffic monitor [43], public transportation system design [44,45], etc. A full survey of urban computing can be found in [39].

- **Healthcare.** Mobile mining solution is also used in healthcare related fields. In the individual level, the activity recognition and travel mode identification using smartphones could reveal the user's living style such as the working hours of the user, the active status of the user, the main transportation tools s/he is using, etc. The activity recognition can also help in elders' care such as fall detection [46], rehabilitation assistant. More and more health related smartphone apps are developed and launched with latest technologies that assist users in monitoring their sleep quality [47], log the walking steps [48], etc. In the aggregation level, the mobile mining techniques provide health related research useful information such public physical activities [8].

- **Location-based search, social and advertising.** Mobile mining are used heavily in location based services such as maps, navigation and social search (e.g. waze, yelp, foursquare). Social medias learn the social patterns by mining the interaction and locations among social networks. Digital advertisements uses mobile mining techniques to target audiences by learning the patterns of users' behaviors such as transportation behavior, social behavior and social networks [49–53].

- **Privacy and security.** Unfortunately, mobile mining are also used in malicious ways such as probing users' privacy, or stealing information. For example, through the vast data of some smartphone usage, one can obtain the quite accurate activity tracking or inferences of the user's locations [54, 55]. Some mobile malware uses mobile mining technology to "learn" the user's keyboard input [56]. This poses challenges in technology related legislation.

2.2 Travel Mode Identification System

The problem of travel mode identification is tackled with both traditional approaches and the emerging mobile related solutions. Traditional methods involve fixed sensors. Recently, more and more studies focus on travel mode identification with floating sensors, due to their various advantages over fixed sensors. Therefore, in this paper, we only consider floating sensor based approaches (see [11] for a detailed review for fixed sensor based methods). According to the types of sensors adopted, we further categorize the previous work into GPS-based and smartphone-based classification methods.

The vast majority of the early literature in travel survey, which leverages only GPS information (location, speeds, and derived acceleration data), belong to GPS-based classification methods [53, 57–60]. Support Vector Machine (SVM) is one of the most popular methods for classification. Zhang et al. (2011) performed a two-stage classification with SVMs. In the first stage they identify three main travel-mode classes: pedestrian, bicycle, and motorized vehicles. And in the second stage they further classify different classes of vehicles into cars, buses, trains and trams [61]. Bolbol et al. (2012) developed a moving window SVM to classify six travel modes from sparse GPS data [62]. Another study used SVMs with quadratic kernel functions for binary classification, which only considered passenger cars and trucks [9]. Several studies leveraged Geographic Information Systems (GIS) for better detection accuracy. GIS and GPS data were combined to detect five travel modes (walk, car, bus, subway, and commuter rail) in New York City [10]. Moreover, another study proposed a combined fuzzy logic and GIS-based algorithm to process raw GPS data. The algorithm was applied to GPS data collected in the highly complex Greater Copenhagen Area network in Denmark and detected trip legs and distinguished between five modes of transportation [63]. A similar study with fuzzy pattern recognition was conducted in Shanghai, China [64]. Many algorithms presented in this category usually involve heavy data processing and transmission load on mobile devices that may exceed its capacity.

Emerging trends in smartphone-based methods are observed in recent literature. Manzoni et al. (2010) developed an algorithm that automatically classifies the traveller's transportation mode into eight classes using a decision tree. The input features were computed from the Fast Fourier Transform (FFT) coefficients of the total acceleration measured by the accelerometer [34]. A trip analysis system that consists of mobile apps and a centralized analyzer was developed to identify the travel mode and the purpose of the trips sensed by smartphones, using the GPS and accelerometer [65]. It was deployed to the smartphones of the volunteers in Dubuque, IA, to serve both the volunteers and the transit agencies. Another study leveraged the same two types of sensors to classify six different travel modes in the region of Vienna, Austria [66]. Authors proposed multivariate parametric models which are fitted to the distribution of feature vectors extracted from the training set.

Very few studies employ a complete list of smartphone sensors for better classification results. Frendberg (2011) designed a smartphone app to detect transportation modes by applying a Boosted Naive Bayes classifier to the data collected from GPS, accelerometer, orientation, and magnetic sensors [67]. However, the data were collected from a single user and only two travel modes, walk and automobile, were considered in that study. Another recent work collected multimodality travel data in New Delhi, India, from a variety of sensors, including accelerometer, gyroscope, magnetometer, light intensity meter, proximity, sound level and GPS [68]. They focused on two-wheeler and three-wheeler classification with a threshold-based heuristic. However, no pedestrian, cyclist, or subway is considered in that work. In [69], Jahangiri and Rakha explored the solutions with a different combination of sensors such as accelerometer, gyroscope and GPS. They used SVM with Gaussian kernel as the learning model and obtained high accuracy. However they did not include subway as a travel mode. Even though much progress has been made, several key challenges remain open such as dynamic model update, battery consumption reduction. It is discussed in the following section.

2.3 Key Challenges

Although remarkable progress has been made, travel mode recognition using smartphones is still in its infancy stage. This is largely due to the following three key challenges.

Challenge # 1: Data Capturing and Preprocessing. The existing smartphone-based travel mode recognition is almost exclusively based on the accelerometer sensor [11, 70]. While being effective in distinguishing between certain travel modes (e.g., walking, biking, jogging), the accelerometer alone is often inadequate to differentiate other travel modes. For example, the accelerometer reading of a traveller who is driving a car or is taking a bus/subway might be very similar with each other, since the traveller is likely to sit on the seat most of the time. This naturally motivates us to also collect other types of sensors (e.g., magnetometers, barometers). However, the raw data collected from such sensors often accompanies with various noises, due to a number of reasons, ranging from the way a traveller holds the smartphone (e.g., in a pocket vs. in the backpack vs. in hands), to the environmental conditions (e.g. the magnetometers' reading will be distorted by environment circus layout.) Without being appropriately denoised, the negative impact of such noise on the recognition accuracy might outweigh the additional discriminative power embedded in these additional sensors.

Challenge #2: Feature Engineering. As in almost any learning model, designing the "right" feature is often the key to the success of the travel mode recognition algorithm. In our setting, the feature extraction needs to balance a number of potentially conflicting factors, e.g., the recognition accuracy (the discriminative power of the extracted feature), the computational time, the response time as well as the battery consumption. For example, in order to segment the raw sensor reading (which is essentially a time series data), we need to decide the appropriate sampling rate and segmentation length. A longer segment might lead to more powerful feature (e.g., the discriminative FFT coefficients), yet it also requires more computational time to extract such features (by FFT) and/or might delay the on-line response. In general, it is desirable to examine *all* the available sensors to have more discriminative features. However, it inevitably comes with a cost in terms of the battery consumption.

Challenge #3: Learning Models. Different travellers often exhibit dramatically different travel behaviors. For example, some drives cautiously, while others might drive more aggressively with frequent and sudden accelerations and decelerations. Some people walk as fast as others jog. Ideally, we need to collect sufficient labelled data for *each* traveller in order to train a learning algorithm that is tailored for that specific traveller, which is almost impossible in reality. How can we adapt a travel mode recognition model trained on a generic population to a specific user, with only limited labelled data for her/him? How can we update/adapt such a traveller-specific model efficiently over time to accommodate newly collected labelled data for her/him?

Chapter 3

Thesis Overview

In response to the three key challenges, we develop new methods for a smartphone-based, real-time travel mode identification system in this thesis. Our prototype system takes smartphone sensors' data as the input and the identified travel modes as the output. An important feature that differentiates our prototype from the vast majority of the existing work is that it does not use any location information (e.g. GPS, GSM data). It consists of three major tasks/components, as summarized in Figure 3.1.

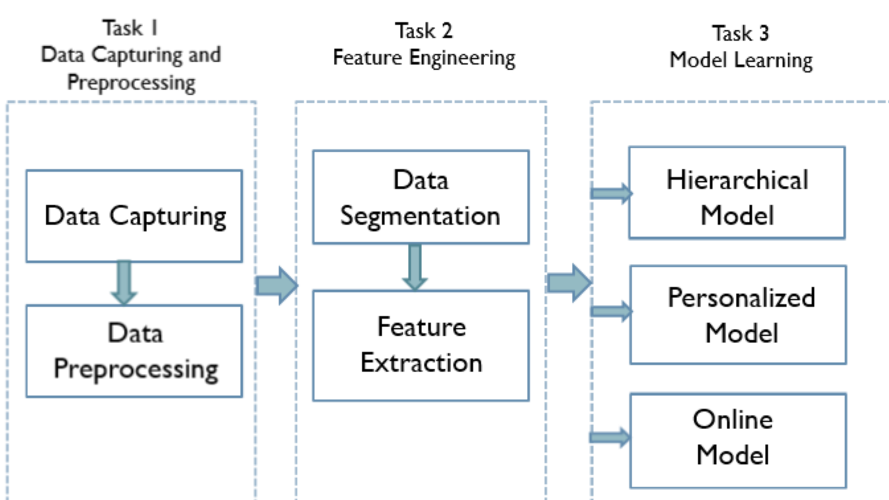


Figure 3.1: Flowchart of Travel Mode Identification with Smartphones

In the first task, we will develop smartphone Apps based on the latest iOS and Android systems for data collection. Volunteers travel with the App installed on their smartphones to record the sensors' data (e.g. accelerometers' data, barometers' data, etc.) during travel. They also record their modes of transportation (e.g. bus, walk, etc) at the same time. The data collected with correspondent travel mode are referred to as labelled data. In order to be able to identify the travel mode from the smartphone sensors' data, we need to first understand their typical patterns from the above labelled data: For example, the accelerometer data collected during jogging fluctuate more heavily than in other travel modes; the air pressure detected during travelling with buses has a smaller mean value than the pressure while travelling with cars; the magnetic field readings show some oscillatory pattern while travelling on subways; and so forth. Some key questions we need to answer include: What are the characteristic patterns behind these data, and how can we combine this information with multimodality sensing to achieve real-time travel mode identification? Given the multiple noise sources during such data collection process, we hypothesize that a single de-noising strategy might be inadequate. Having this in mind, we design a set of comprehensive de-noising to remove as much noise as possible.

In the second task we design the data segmentation and feature extraction methods that carefully balance between the prediction accuracy, the on-line response time and the battery consumption. The data collected preprocessed in step 1 is time series data of different sensors' readings. The goal in this step is first slice the time series data into data segments and then extract the most discriminative features from the segments. A proper segment length is important to both feature extraction and energy consumption. A data segment of short time span may cut off the data inside its cycle of the certain pattern that would make the learning less effective, and a segment of long time span may not provide us extra information but requires more resources for calculation. Besides in a real-time system, it delays the prediction since it needs more data to come in to construct a segment. The feature extraction is a critical step in any model training. In our thesis, it also affects the energy consumption. While the multi-modality sensing empowers us with discriminative information, the high dimensional features from multiple data sources are both battery draining in sensing and calculation. To leverage the model accuracy, the fast response and energy efficiency, we design comprehensive methods on data segmentation and feature extraction.

In the third step, we build a learning model which recognizes the travel mode based on extracted features in task 2. In this thesis we focus on the following six travel modes: *driving a car, walking, jogging, bicycling, taking a bus* and *taking a subway*. The heart of this task is the learning mechanism that could adapt to a specific user's travel behavior, online update with new labeled data, while maintaining a lower cost in energy as well as computational time and resources. We tackle the problem in three angles each of which is a learning model. The first model is a hierarchical learning model. It is based on the results of group feature analysis in feature engineering that not all modes requires features extracted from the data of the 5 sensors. In chapter 7 we first divide the six modes into wheeled mode and unwheeled mode. The wheeled mode includes outdoor mode: biking, and the indoor mode: taking a subway, driving a car and taking a bus. The unwheeled mode includes walking and jogging. Thus the hierarchical model consists three layers. At first layer we train a model that classify wheeled/unwheeled travel mode. In second layer we classify the travel modes within each subcategory. Finally, we classify the car, bus or subway mode within the indoor mode. A dynamic sensor selection mechanism is installed inside the hierarchical model that only wheeled mode requires the full sensors data while majority of the sensors are turned off except the accelerometer and gyroscope. The second model is a personalized learning model. The main idea is that for a user who has limited labeled data we “borrow” data from users whose data is pre-collected and labeled, and their travel behavior is similar to current user. In chapter 8 we first propose a method to calculate the user similarity scores. Then we select data from the users that have higher similarity scores. Then we estimate the data distribution in feature space, and uses it to reweight the borrowed data so as to minimize the model loss with respect to the target user. Experimental evaluations on real travel data show that the proposed method outperforms the generic method in terms of prediction accuracy. The third model is an online learning model. It corporates with the hierarchical model and personalized learning. The model adopts the stochastic sub-gradient descent method and updates the learning models with small portion of the data.

In the next 5-8 chapters, we present the details of these three research tasks. For each task, we present the problem description, the detailed solutions as well as experiment design and results.

Chapter 4

Data Capturing and Processing

4.1 Problem Description

The first step towards a travel mode identification is collecting smartphone sensors' readings during travel. The raw data is then preprocessed in order to remove various noise.

In data capturing, the main question to answer is how to collect data and what to collect. The accelerometer data and GPS/GSM data are the main data sources in most of the existing smartphone-based travel mode identification systems [69]. The GPS and GSM data are used for location information. However, such data is not only sensitive as it reveals the users' locations, but also unstable and even inapplicable in certain scenarios such as the underground transportation in New York City. Accelerometers alone is often inadequate to effectively distinguish all travel modes. For example, in wheeled travel modes (i.e., buses, cars, subways), most of the time travellers are sitting in their seats, resulting very similar accelerometer readings. This naturally motivates us to also collect other types of sensors. However, the raw data collected from such sensors often accompanies with various noises, due to a number of reasons, as outlined in Section 2.3. Without being appropriately de-noised, the negative impact of such noise on the recognition accuracy might outweigh the additional discriminative power embedded in these additional sensors. Therefore, the main task in data preprocessing is to minimize the various noise in the raw sensor readings.

4.2 Data Collection

4.2.1 Data Collection App

We have developed a smartphone app based on iOS and Android systems to collect raw sensor data. Figure. 4.1 shows a screenshot of the iPhone App we developed. This App allows user to collect data both in online mode and offline mode. At online mode, the App is connected to our data server and transfer sensor data in realtime. At offline mode the sensor data is saved on the smartphone local memory and user can choose to send it to server at the availability of Internet connection. For each user the App generates a unique user ID (here we assume that a smartphone belongs to a single user and only record one user's activity). It also provides the user an option to set the sensing frequency.

4.2.2 Data Description

Acceleration data is the most frequenctly used sensor data in smartphone based activity recognition project. In this thesis, besides the acceleration data, we also collect data from the environment sensors, including barometers, magnetometers and light sensors. The ambient air pressure detected by barometers largely depends on the space of the transportation tools, and other factors such as the air conditioner, door opening and closing, number of passengers, etc. The magnetic field measurements detected by magnetometer of smartphones, on the other hand, provide a good indicator of different patterns related to the environment's magnetic field and its changes during travel. For example, a subway system uses magnet to brake. There are more mobile devices on a bus than in a car which would also cause the change in the detected magnetic field. All these differences are revealed in the magnetometer

Figure 4.1: iPhone App for Data Collection

readings and thus can be leveraged to identify the travel mode.

Accelerometer. Accelerometer readings measures the changes in velocity along the x, y, and z axes (x,y and z) of the cellphone, as is shown in Firgure. 4.2. Acceleration data is an important reference to detect the pattern of a user's body movement.

Gravity Sensor Gravity sensor readings return the gravity as measured along each axis of the cell phone. If the phone is put on the table with Y axis facing the sky, the reading on Y axis would be roughly $-9.8m/s^2$ while the readings on the other 2 axes would be around

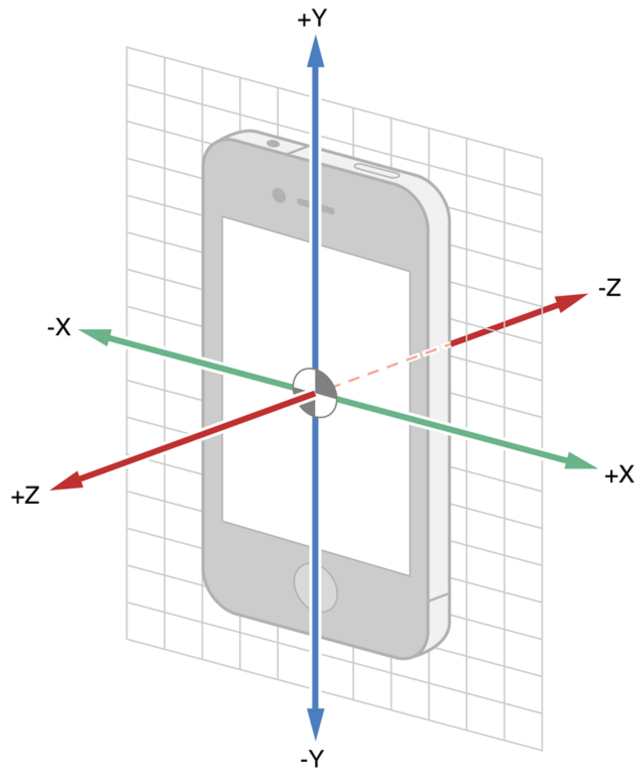


Figure 4.2: x, y and z axis Acceleration Measure of iPhone [1]

$0.0 m/s^2$.

Barometer. Barometer readings return the detected ambient air pressure. In [71], Muralidharan et al. conducted an experiment showing that the pressure detected by the smartphone barometer would change with the building structure and type, and such a pattern is able to be learned. In our experiment, we also verified that the barometer reading is discriminative with different transportation modes.

Gyroscope. The gyroscope measures the rate of rotation around the three axes. Figure. 4.3 shows the standard direction of measures for rotation along x, y and z axis.

Light Sensor. It measures the ambient light level in SI lux units. In Android phones this value is able to obtained directly [72]. However, in iPhone it is discouraged to use in

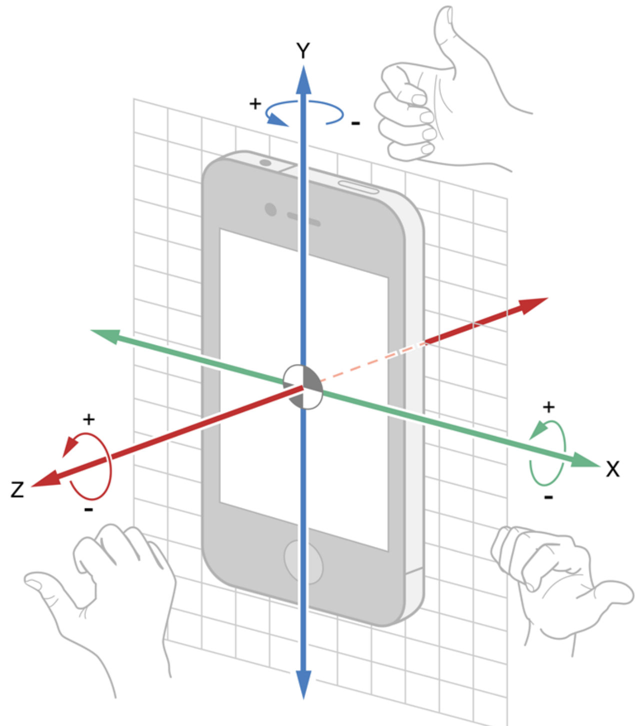


Figure 4.3: x, y and z axis Rotation Measure of iPhone [1]

Apps since it is a low level framework [73]. Therefore in iPhone instead of using the light sensor, we use screen brightness reading which is the brightness level of the screen [74]. This reading is adjusted by the ambient light level when phone is unlocked and thus it is similar to the light senso.

Magnetometer. It measures the earth's geomagnetic field plus bias introduced from the device itself and its surroundings. The smartphones provide both raw readings of the magnetic field as well as the calibrated readings. The calibrated magnetic field usually filters out the bias introduced by the device and, in some cases, its surrounding fields [75, 76]. The magnetometer reading also returns the hard iron bias values separately for customized calibration.

In this thesis, the data collection is conducted 3 times in the past 3 years. The first time is in the Jul of 2014, the second time data collection happened in Dec 2014 and the last one

Table 4.1: Sensors' Setting

Sensor Used	Data Name	Freq.	Dimensions
Accelerometer	Acceleration (m/s^2)	16 Hz	x, y, z
Gravity Sensor	Gravity (m/s^2)	16 Hz	x, y, z
Gyroscope	Rotation Rate (rad/s)	16 Hz	x, y, z, x_{calib}, y_{calib} z_{calib}
Magnetometer	Magnetic Field (μT)	1 Hz	x, y, z x_{calib}, y_{calib}, z_{calib}
Barometer	Ambient Air Pressure (hPa)	1 Hz	1
Light Sensor	Ambient Light Level (lx)	1 Hz	1

Table 4.2: Data Collection Details

Mode	#Samples	Duration(Min.)	Gender Dist.	City Dist.
Bike	20606	72.1	1 Female, 2 Males	NYC, Buffalo.
Bus	44080	147	1 Female, 3 Males	NYC, Buffalo
Car	227967	760	6 Females, 6 Males	NYC, Buffalo, CS
Jog	16838	56.2	2 Females, 5 Males	NYC, Buffalo, CS
Subway	72643	242.1	6 Females, 3 Males	NYC, Buffalo
Walk	95701	319	7 Females, 5 Males	NYC, Buffalo, CS

happened in Jul - Aug 2016. In total, we have 16 volunteers collecting data in Buffalo, NY, New York City (NYC), NY and College Station (CS) TX while travelling with the 6 major travel modes. The collected data has variation in user's gender, travel city, travel season, as well as the phone models. All the collected data used in this thesis are de-identified. The sensors and their sensing settings are shown in Table 4.1 and the data details for each mode is shows in Table 4.2.

De-noising Technique	Noises to Remove
Data Rotation	Noise caused by phones' different positions and heading directions.
Winsorization	The outliers in sensor reading such as spikes in data, or noise caused by other sudden events (e.g. phone dropped to the floor)
Gaussian Smoothing	The high frequency noise (e.g. white noise naturally exists in sensors)
Normalization	Difference in data scale of different data sources

Table 4.3: De-noising Techniques and the Targetted Noises

4.3 Data de-noising

In data preprocessing, we apply a comprehensive set of techniques to remove various noise in the raw data.

De-noising #1: Data Rotation. Among all the sensors' readings we collected, acceleration is measured along the phone axes whose coordinates are determined by the phone's position and heading direction. Since it is unrealistic to coordinate all the travellers to have the same phone position and heading direction, the acceleration data is measured in a different phone coordinate system and needs to rotate back to a standard coordinate system before any further calculation. Here we define the standard coordinate system to be y axis perpendicular to the earth pointing toward the sky and z axis pointing to the magnetic north. The x axis is determined in a right-handed coordinate system. Magnetic field and gravity are used to rotate the readings from the phone's coordinates to the standard coordinates. First we rotate the smartphone's z axis to point to the magnetic north direction and then rotate y axis to be vertical to the earth and pointing up to the sky.

The readings of both calibrated and uncalibrated ambient magnetic field are accessible. According to [77], the differences between calibrated and uncalibrated magnetic field data are as follows. The hard iron calibration is reported separately in uncalibrated magnetic field,

Sensor Field	**Timestamp**	**Value Name**	**Value(μT)**
TYPE_MAGNETIC_FIELD_UNCALIBRATED	633	$x_{uncalib}$	100.7734
TYPE_MAGNETIC_FIELD_UNCALIBRATED	633	$y_{uncalib}$	-53.8651
TYPE_MAGNETIC_FIELD_UNCALIBRATED	633	$z_{uncalib}$	461.7340
TYPE_MAGNETIC_FIELD_UNCALIBRATED	633	x_{bias}	69.1016
TYPE_MAGNETIC_FIELD_UNCALIBRATED	633	y_{bias}	-26.0590
TYPE_MAGNETIC_FIELD_UNCALIBRATED	633	z_{bias}	497.7585
TYPE_MAGNETIC_FIELD	633	x_{calib}	31.6711
TYPE_MAGNETIC_FIELD	633	y_{calib}	-27.8061
TYPE_MAGNETIC_FIELD	633	z_{calib}	-36.0245

Table 4.4: Magnetic Field Reading

while calibrated reading includes the calibration in measurement. For example, the calibrated and uncalibrated magnetic field measures at the same time by one of our smartphones is shown in Table 4.4. The *TYPE_MAGNETIC_FIELD* are the ambient magnetic field measured in 3-dimensional space at time tick 633. The *TYPE_MAGNETIC_FIELD_UNCALIBRATED* are the hard iron calibrated magnetic field in 3 dimensions, as well as the bias that is calibrated along each dimension.

We denote the magnetic field vector by $\mathbf{M}$, the magnetic field magnitude is calculated by

$$\|\mathbf{M}\| = \sqrt{{M_x}^2 + {M_y}^2 + {M_z}^2} \tag{4.1}$$

If we calculate the calibrated magnetic field using x_{calib}, y_{calib}, z_{calib}, the magnitude is 55.44 μT, which is close to the value 53.723 μT measured and recorded by [78] in Buffalo on the day our data was collected. We use this fact for the first step of phone coordinates rotation.

Gravity reading is then used for the rotation. Gravity reading is recorded along 3 axes. For example, a sample of gravity reading in the format of $[t, x, y, z]$ is $[677, -0.850, 5.090, 8.339]$.

We denote the gravity by $\mathbf{G}$, and the gravity magnitude is calculated by

$$\|\mathbf{G}\| = \sqrt{{G_x}^2 + {G_y}^2 + {G_z}^2} \tag{4.2}$$

The gravity magnitude of the sample reading is $9.807m/s^2$, which is close to the gravity $9.804m/s^2$ of Buffalo recorded on [79].

We denote the raw gravity and magnetic field reading by $\mathbf{G_0} = [G_{x0}, G_{y_0}, G_{z0}]$ and $\mathbf{M_0} = [M_{x0}, M_{y_0}, M_{z0}]$. In the standard coordinate system, the earth gravity is $\mathbf{G_s} = [G_{xs}, G_{y_s}, G_{zs}]$ where $G_{xs} = 0,\ G_{y_s} = \sqrt{G_{x0}^2 + G_{y_0}^2 + G_{z0}^2},\ G_{zs} = 0$, and magnetic field is $\mathbf{M_s} = [M_{xs}, M_{y_s}, M_{zs}]$, where $M_{xs} = 0,\ M_{y_s} = 0,\ M_{zs} = \sqrt{M_{x0}^2 + M_{y_0}^2 + M_{z0}^2}$. $\mathbf{R_1}$ and $\mathbf{R_2}$ are the two rotation matrices we use. $\mathbf{R_1}$ rotates the raw gravity reading $\mathbf{G_0}$ to the Earth gravity $\mathbf{G_s}$, and $\mathbf{R_2}$ rotates the raw reading of magnetic field $\mathbf{M_0}$ to $\mathbf{M_s}$

$$\mathbf{R_1 G_0^T} = \mathbf{G_s^T} \tag{4.3}$$

$$\mathbf{R_2 M_0^T} = \mathbf{M_s^T} \tag{4.4}$$

Using equation (3) and (4) we can get $\mathbf{R_1}$ and $\mathbf{R_2}$. Now assume we have the raw reading of acceleration $\mathbf{A_0}$ at the same time. We use equation (5) to rotate the acceleration into the standard coordination value $\mathbf{A_s}$.

$$\mathbf{R_2 R_1 A_0^T} = \mathbf{A_s^T} \tag{4.5}$$

By rotation, the sensor reading will be *position-independent*.

De-noising #2: Winsorization. The winsorization is used to reduce the possible spurious outliers in the data. Figure. 4.4 shows the distribution of acceleration data along y

axis while travelling by a car. The top plot is the sorted acceleration in the ascending order, plotted with red line. The blue dash lines indicate the 2.5% cutting point at both sides. The bottom plot is the unsorted raw acceleration along time plotted with black solid line. The two horizontal red solid lines indicate the upper bound (97.5%) and the lower bound (2.5%) of the data value. We can see that in the plot, the blue dash lines cut at the point where the red data curve on the left side starts to change from very steep to very flat, and on the right side after the blue dash it starts to change from very flat to very steep. Meanwhile, in the unsorted raw data plots, the red lines where the value is cutoff mostly keep the characteristics of the curve and only eliminate some outliers and noise. We have a similar observation in most of our data. Therefore, we propose to chose 5% as the fraction for winsorization.

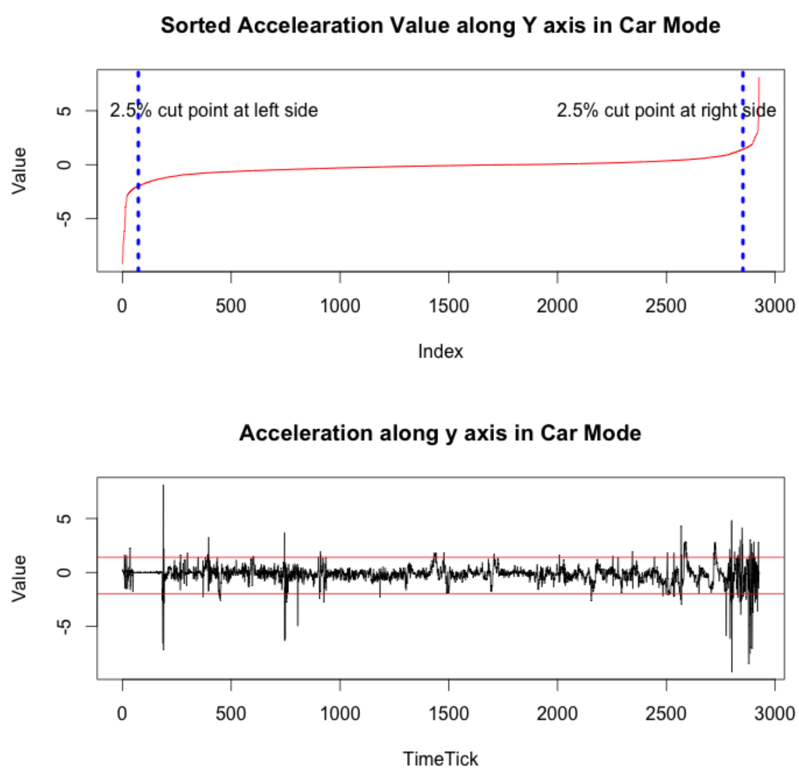


Figure 4.4: 5% Winsorization of Acceleration Data along y axis

De-noising #3: Gaussian Smoothing. In winsorization, we eliminate the overall outliers within the data. To analyzing the data value' change in a relatively longer time period, it is also important to eliminate the white noise and high frequency fluctuation as much as possible. For example, if we want to learn the direction change during moving (e.g. when does the car take a turn, which direction it turns to), we need to analyze the reading from the rotation sensor (the red line in Figure 4.5). Roughly, the reading contains three kinds of changes: the white noise in highest frequency, the fluctuation in second highest frequency which is caused by the phone's movement together with the carrier's body, and the trend change which is as the average value increasing/decreasing and the peak points. While analyzing the reading's trend change, we would like to eliminate the first two kinds of changes while keeping the third one. The solution is to use Gaussian filter to smooth the data. It works as a low-pass filter and attenuates high frequency signal periods in the data. We use the segment length as the smooth parameter σ, which in this case is 64. The blue line in Figure. 4.5 includes the results after smoothing. We can see that the filtration drops all the high frequency oscillations most of which come from noises, and leaves only the main up and down trends. We compared the smoothed line with the recorded actual turning events. All the local extrema (1-5 points marked with a yellow line) in Figure. 4.5 are when the turning events actually happened.

De-noising #4: Normalization Data collected by different sensors measuring different aspects of travelling events are quite diversified in both range and value. For example, the accelerometer data are between $\pm 40m/s^2$, while the magnitude of magnetic field could reach $500mT$. In order to compare the data from different sensors, we normalize them between 0 and 1.

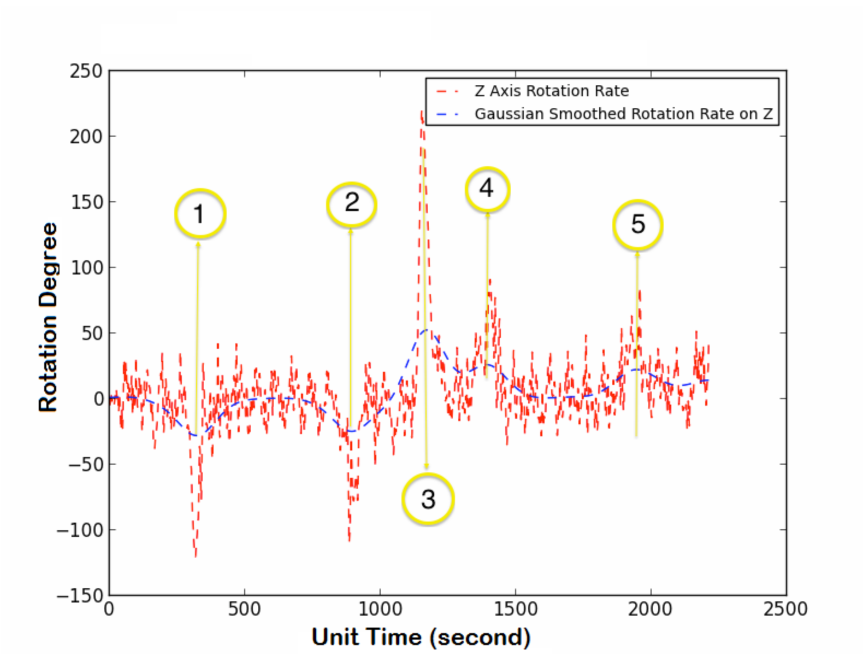


Figure 4.5: Gaussian Smoothed Rotation Rate

Chapter 5

Feature Engineering

5.1 Problem Description

Feature engineering is a crucial step for an effective model. The main purpose is to extract and select features which serve the input of next step of modeling. For travel mode identification, feature engineering has three stages, including data segmentation, feature extraction and feature selection. The preprocessed data is time series of different sensor readings. For convenience, we refer to the reading from all sensors at a single time as "one data sample". It is the unit data that is generated at one time tick. A data segment contains one or more data samples. To extract features, the preprocessed data is sliced into small segments. Features are derived from a segment and form a feature vector, with the principle that the feature vector is informative and discriminative. Feature selection is the process of selecting a subset of the features in order to reduce the computational complexity. It is crucial to the response time.

As is mentioned in Section 2.3, the main challenge in data segmentation is to find a proper way to slice data providing the sensor' sampling rate. If the length of a segment is too short, it might be less informative and result in a less effective model. On the other hand, increasing the length of the data segments means the system requires more time to obtain enough readings. In feature extraction and selection, the constrains are the computational cost and the battery consumption. This is true especially when the features are extract from multimodality sensors, the sensing task for smartphones is battery draining.

5.2 Data Segmentation

We introduced the slide-window segmentation, which leads to a fast response while preserving the information of each segment. To construct one segment, the slide-window mechanism extends the newly registered data into a segment by including a certain amount of the most recent cached data. The slide-window design is illustrated in Figure. 5.2. The ratio of the new data and cached data in a segment is configurable. The use of cached data will serve to avoid the boundary issue in the data. Without it, some discriminative patterns might be missed simply because a significant amount of consecutive readings would be divided into separate segments.

There are two factors for a proper length of a data segment: the sampling rate and the cached data size. As for the sampling rate, it determines how long it takes to collect enough data for one segment and how much information one segment contains. For example, if smartphone A collects data with sampling rate $200ms/reading$ and smartphone B collects data with sampling rate $100ms/reading$, a segment of length 10 contains information within 2 *sec* of phone A while it contains information within 1 *sec* of phone B. Besides, the sampling rate also determines how detail the information in one segment is. The cached data size, on the other side, determines much new information is included in one segment. The larger the ratio of cached data is in one segment, the shorter it takes to form a data segment. However, it also put more delay for event detection with larger ratio of cached data.

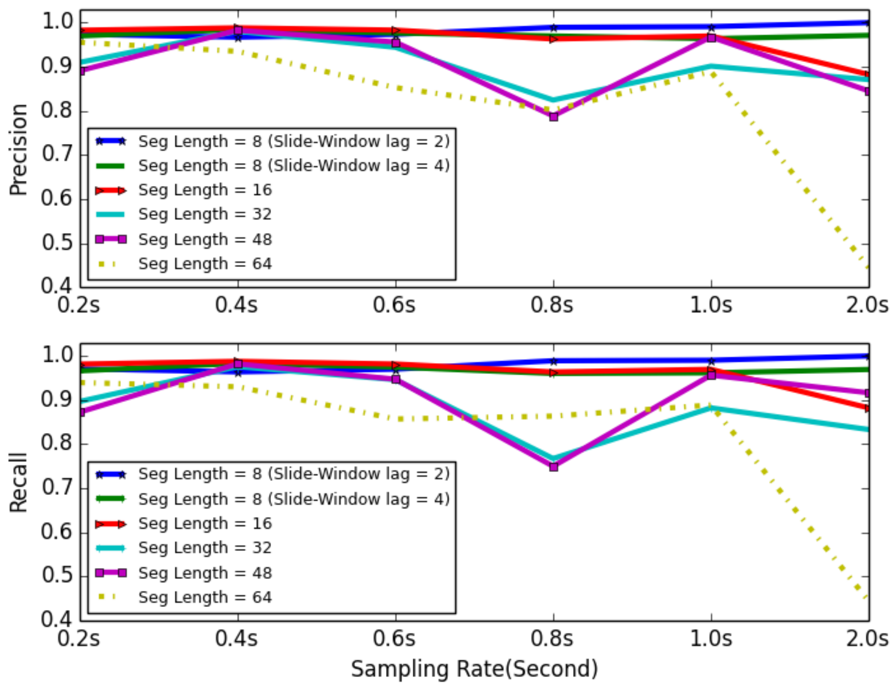


Figure 5.1: Learning Accuracy by Different Segment Size and Sampling Frequency

We've done comparative studies on how the sampling rate, segment size, as well as the cached data size would affect the model's classification accuracy, as is shown in Figure. 5.1. As for the denotation, for example, if a segment contains 2 new data readings and 3 cached data readings, we say the segment length is 8 with slide-window of which lag is equal to 2.

5.3 Feature Extraction

Following data segmentation, we extract features in both time and frequency domains. Table. 5.1 summarizes the most used features in smartphone sensors mining literature.

Recall that the raw data include acceleration, magnetic field and air pressure data. Acceleration and magnetic field data are analyzed in both the time domain and the frequency domain. Pressure is analyzed in the time domain only since the main change in ambient air pressure largely depends on the space of the transportation tools and the statistical features

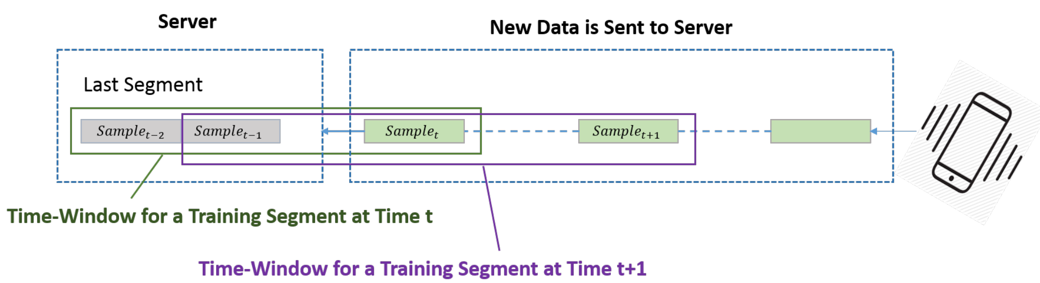


Figure 5.2: Using Slide-Window for Data Segmentation. If the segment size consists of three data samples and is configured to take two historical data samples, at time t new data sample "$Sample_t$" is combined with the historical data "$Sample_{t-2}$" and "$Sample_{t-1}$". And the segmentation window slides forward as the "$Sample_{t+1}$" comes in at time $t+1$. "$Sample_{t-1}$" and "$Sample_t$" are segmented with "$Sample_{t+1}$" as a new segment.

Category	Features	Description
Time Domain Features	Mean, Max, Min, Std	The mean, max, minimum value and the standard deviation of the data segment
	Correlation	The Correlation of data of each axis pair.
	Signal-Magnitude Area (SMA)	SMA is calculated as the sum of the magnitude of the three axes acceleration within the segment window [80].
	Average Resultant Acceleration	ARA is the average of the square root of the sum of the values of each axis.
Frequency Domain Features	Energy	Calculated as the sum of the squared Discrete Fourier transform (DFT) component magnitudes. Ravi in [81] uses the normalized energy divided by the window length.
	Entropy	Calculated as the normalized information entropy of the DFT components, and it helps in discriminating the activities with the similar energy features [82].
	Time between Peak	The time between the peaks in the sinusoidal waves [83].
	Binned Distribution	This feature is essentially the historgram of the FFT

Table 5.1: Features and Description in Smartphone Sensors Data Mining

such as max, min, standard deviation, mean would be enough to describe the characteristics. In total, we have extracted 161, as is summarized in Table 5.2.

5.4 Feature Analysis and Sensor Selection

While all the 161 extracted features seems to be effective in distinguishing different travel modes, it comes with a cost in terms of both the computation and battery consumption. Intuitively not all the travel modes need all the features in learning. For example, the travel modes that involve more motion activities might rely on motion based features more than others modes in classification. In this section, we analyze the features importance by applying group lasso in the classification model to explore the possibilities of eliminating the sensor usage.

The lasso is a very popular technique for variable selection for high dimensional data. Group-lasso [84] is a generalization of the lasso for doing group-wise variable selection. For a given dataset that contains the data samples, denoted by $\mathbf{X}$ and corresponding label vectors $\mathbf{y}$, group-lasso try to solve the penalized least squares:

$$\min_{\boldsymbol{\beta}} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \sum_{l=1}^{L} \sqrt{p_k}\|\boldsymbol{\beta}^{(k)}\|_2, \lambda > 0,$$

where p_k is the cardinality of $k-th$ index set, $\|\beta^{(k)}\| = \sqrt{\sum_{j\in I_k} \beta_j}$, I_k is the index set.

To apply the group lasso technique, we categorize the features by their source sensors and get 5 feature groups: accelerometer based features, gyroscope based features, light sensor based features, barometer based features and magnetometer based features. Among the five groups, accelerometer based group and gyroscope based feature groups are motion based groups. The rest feature groups are environmental based groups.

In order to verify our hypothesis that the travel modes that involve more motion activities might rely on motion based features more than others modes, we first categorize the 6 travel modes into two classes: wheeled travel mode and unwheeled travel mode. Wheeled travel mode includes biking, taking a bus, taking a subway and driving a car. Unwheeled travel mode includes walking and jogging. We further split the wheeled mode into two subcategories: indoor mode (taking a bus, taking a subway, taking a bus) and outdoor mode (biking). The reason for the wheeled/unwheeled categorization is that while walking and jogging the user's body are moving more drastically. This would reflect on the smartphone sensor readings such as acceleration and rotation. As for indoor/outdoor categorization, the intuition is that the readings from environmental sensors (e.g. magnetometers, light sensor, barometers) might show different patterns due to the ambient environment difference between indoor travel tools and at outdoors. We uses *gglasso* package [85] in R. The results are shown in Figure. 5.3.

From the results of feature analysis using group-lass, we can conclude that 1) unwheeled travel mode (walking and jogging) can read a higher classification accuracy with features only drawn from accelerometer (A) readings; 2) Inside wheeled travel mode, features drawn from gyroscope (R) and magnetometer (M) are important to classify indoor/outdoor modes. This gives us a hint in dynamically selecting the sensors in the travel mode detection. We expand this conclusion in next section.

Name	Description
$x_{\max}, x_{\min}, x_{\text{std}}, x_{\text{avg}}$ $y_{\max}, y_{\min}, y_{\text{std}}, y_{\text{avg}}$ $z_{\max}, z_{\min}, z_{\text{std}}, z_{\text{avg}}$	The statistical description of acceleration data segments along x, y and z axes, respectively.
$x_{\text{offset}}, y_{\text{offset}}, z_{\text{offset}}$	The offset of the DFT results of the x, y and z axes of acceleration data segment, respectively.
$x_{\text{freq}}, y_{\text{freq}}, z_{\text{freq}}$	The principle frequency of the acceleration along x, y and z axis, respectively.
$xe_{\text{std}}, ye_{\text{std}}, ze_{\text{std}}$	The standard deviation of energy vector calculated from x, y and z axes acceleration of the data segments, respectively.
$x_1, x_2, ..., x_{10}$ $y_1, y_2, ..., y_{10}$ $z_1, z_2, ..., z_{10}$	Histogram of normalized energy of the acceleration along x, y and z axes of the data segment, respectively.
$xpk_1, xpk_2, xpk_3, xpk_4,$ $ypk_1, ypk_2, ypk_3, ypk_4,$ $zpk_1, zpk_2, zpk_3, zpk_4$	The time zone (indexes of time tick) of the top four data peaks of x, y and z axes acceleration, respectively.
$a_{\max}, a_{\min}, a_{\text{std}}, a_{\text{avg}}$	The statistical description of total acceleration.
$a_{\text{offset}}, a_{\text{freq}}, ae_{\text{std}}$	The offset,principle frequency and standard deviation of the DFT analysis on total acceleration.
$a_1, a_2, ..., a_{10}$	Histogram of normalized energy of the total acceleration
$apk_1, apk_2, apk_3, apk_4$	The time zone (indexes of time tick) of the top four data peaks of total acceleration.
$rx_{\max}, rx_{\min}, rx_{\text{std}}, rx_{\text{avg}}$ $ry_{\max}, ry_{\min}, ry_{\text{std}}, ry_{\text{avg}}$ $rz_{\max}, rz_{\min}, rz_{\text{std}}, rz_{\text{avg}}$	The statistical description of rotation data segments along x, y and z axes, respectively.
$rx_{\text{offset}}, ry_{\text{offset}}, rz_{\text{offset}}$	The offset of the DFT results of the x, y and z axes of rotation data segment, respectively.
$rx_{\text{freq}}, ry_{\text{freq}}, rz_{\text{freq}}$	The principle frequency of the rotation along x, y and z axis, respectively.
$rxe_{\text{std}}, rye_{\text{std}}, rze_{\text{std}}$	The standard deviation of energy vector calculated from x, y and z axes rotation of the data segments, respectively.
$rx_1, rx_2, ..., rx_{10}$ $ry_1, ry_2, ..., ry_{10}$ $rz_1, rz_2, ..., rz_{10}$	Histogram of normalized energy of the rotation along x, y and z axes of the data segment, respectively.
$rxpk_1, rxpk_2, rxpk_3, rxpk_4,$ $rypk_1, rypk_2, rypk_3, rypk_4,$ $rzpk_1, rzpk_2, rzpk_3, rzpk_4$	The time zone (indexes of time tick) of the top four data peaks of x, y and z axes rotation data, respectively.
$p_{\max}, p_{\min}, p_{\text{std}}, p_{\text{avg}}$	The statistical description of the ambient air pressure data segment from the smartphone's barometer
$me_{\max}, me_{\min}, me_{\text{std}}, me_{\text{avg}}$	The statistical description of the magnetic field data segment from the smartphone's magnetometer.
mag_{offset}	The offset of the DFT result of the magnetic field data segment.
$l_{\max}, l_{\min}, l_{\text{std}}, l_{\text{avg}}$	The statistical description of the environment brightness from the smartphone's light sensor.

Table 5.2: Extracted Features and Descriptions

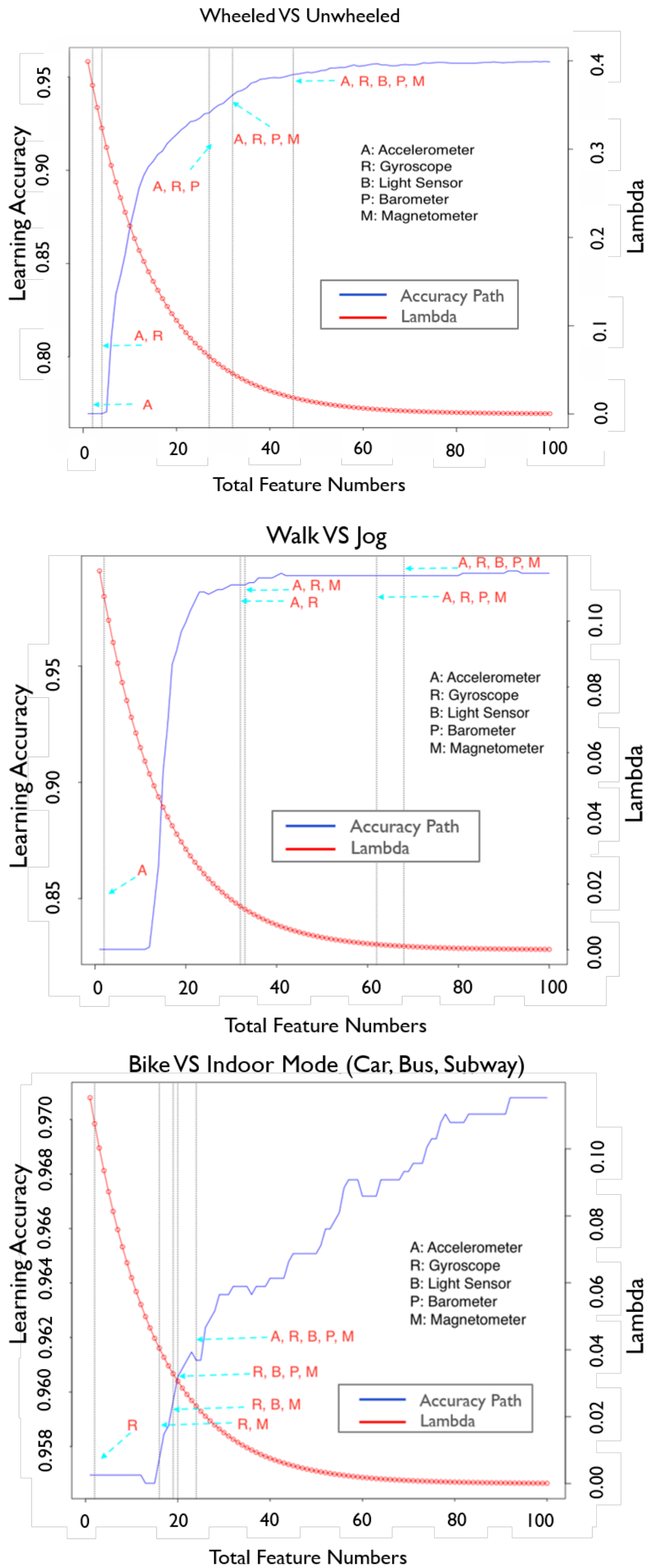


Figure 5.3: Solution Path of Linear Model with Group-lasso Applied. The vertical dash lines are the moments where new group(s) of features join the whole features used for classification. Each line of red letter(s) are the total sensor group(s) in use with a blue dash arrows pointing to the dashline showing the moments when these group(s) of features start in use.

Chapter 6

Hierarchical Model

6.1 Problem Description

Model learning is the critical step in classification problems. In this thesis the model learning process takes the data from smartphone sensors as the input, and learns a classification model that can predict the travel mode. The continuous operation with multiple sensors working could lead to the smartphone's battery discharge after a few hours, making this approach unfeasible. Thus despite the general goal of high classification accuracy, learning with smartphone sensor data also demands the model to be less battery draining on the smartphone side. In this chapter we explore the solution of a hierarchical model that provides a dynamic sensor selection mechanism to address this problem.

6.2 A Hierarchical Framework

Intuitively, during a wheeled travelling (e.g., bicycling or on a car, bus, subway), the body movement is less drastic than the unwheeled travelling (e.g., walking and jogging). This observation is reflect on the accelerometer reading. Figure 6.1 shows the comparison of the accelerometer readings when a user is taking a bus and when s/he is walking. We can see that the acceleration is more fluctuated with much higher magnitude in walking in this time period. We further conclude in Sec. 5.4 through feature analysis that not all travel modes need all the features drawn from the readings of all sensors. This conclusion enables the possibility for the App to dynamically select sensors based on current predicted travel mode. For example, if currently the predicted travel mode is walking or jogging, we could keep the accelerometer for sensing and turn off the rest sensors.

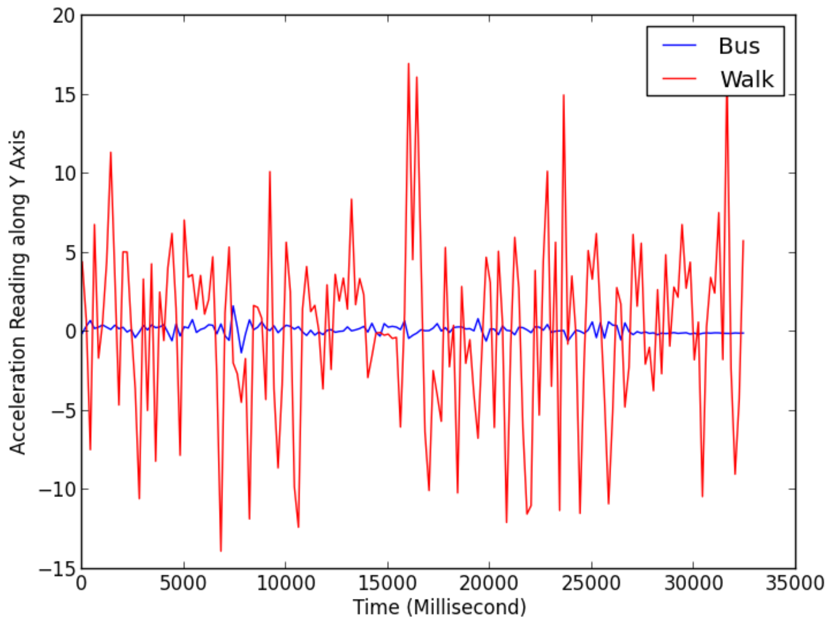


Figure 6.1: Acceleration along the Y-Axis: Walking vs. Taking a Bus

First we propose a hierarchical categorization of the 6 travel modes based on the results of group feature analysis. The hierarchical structure separates the travel mode into layers of subcategories. It contains three layers in total. In The first layer, the travel modes is categorized into wheeled travel mode (bus, train, bike and car) and unwheeled travel mode (walking, jogging). Then in second layer of wheeled travel mode, it splits into two subcategories: indoor mode (bus, car, subway) and outdoor mode (bike). And the third layer classification is for bus, car and subway. Each layer has its own classifier. The hierarchical structure is shown in Firgure. 6.2.

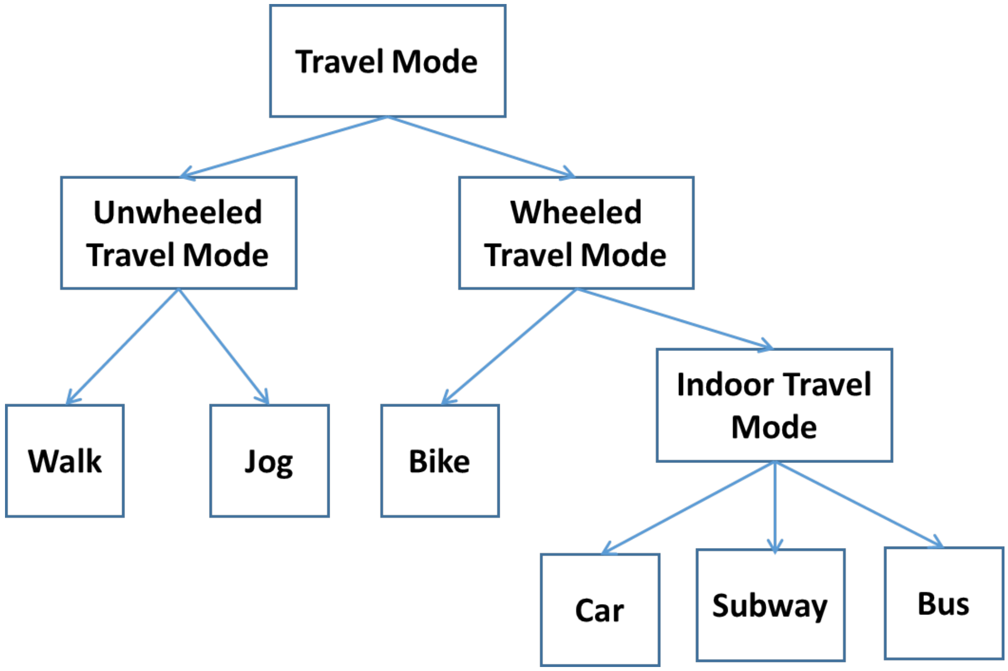


Figure 6.2: Hierarchical Categorization of Travel Modes

6.3 Dynamic Sensor Selection

The dynamic sensor selection is based on the hierarchical framework. That is, the sensors are turned on and off based on current detected travel categories and subcategories. In this hierarchical model, there are four learning models in corresponding to the hierarchical structure shown in Figure. 6.3.

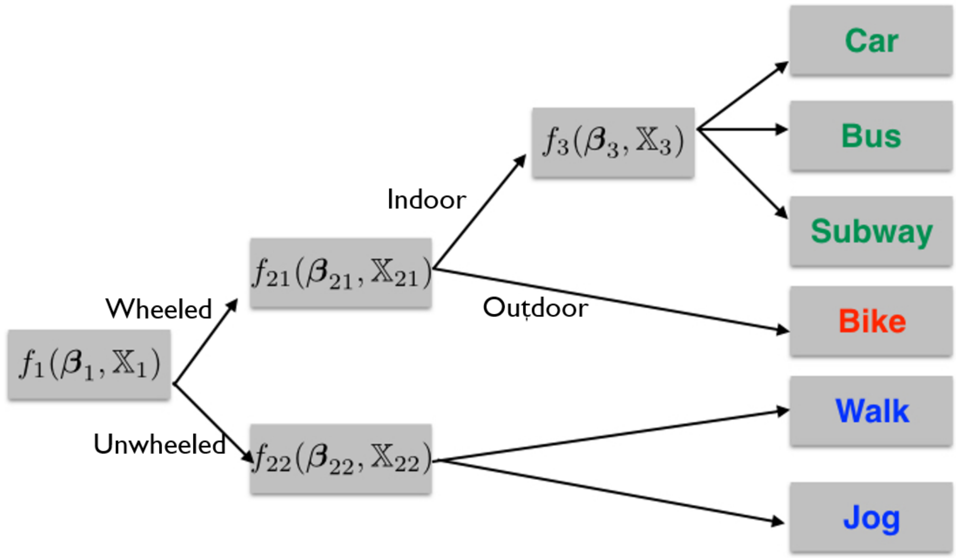


Figure 6.3: Hierarchical Online Learning Workflow

f_1 is the model at wheeled/unwheeled level; f_{21} is the model at indoor/ourdoor level within wheeled subcategory; f_{22} is the model for walking/jogging classification at unwheeled subcategory; f_3 is the model for car/bus/subway classification at the indoor subcategory of wheeled mode. The classification results will be calculated combining all four models:

$$y = \mathbb{1}_{f_1=1} \cdot \mathbb{1}_{f_{21}=1} \cdot f_3 + \mathbb{1}_{f_1=1} \cdot \mathbb{1}_{f_{21}=-1} + \mathbb{1}_{f_1=-1} \cdot \mathbb{1}_{f_{22}}$$

Note that f_{21} and f_{22} are mutually exclusive in learning process, and also f_{22} and f_3. So the algorithm won't update the four models at same time at an on-line updating fashion. The algorithm for hierarchical framework with dynamic sensor selection is shown in Alg.1.

6.3.1 Experiment

The experiment is designed to compare the performance between hierarchical model and general model. The general model is a multiclass SVM classifier with one-against-all method. The hierarchical model works following the procedure shown in Alg. 1. We first conduct some

Algorithm 1 Hierarchical Framework with Dynamic Sensor Selection

Input: The learning models for each level: f_1, f_{21}, f_{22} and f_3.
Input: S_1, S_{21}, S_{22} and S_3, the sensor groups selected with group lasso. S_{all} is the whole sensor set of current smartphone.
for $t = 1 : T$ **do**
  Calculate $y_{1t} = f_1(\mathbf{w}_1, \mathbf{x_t})$
  if $y_{1t} == 1$ **then**
    Turn on sensors: $S_1 \cup S_{21}$
    Calculate $y_{21t} = f_{21}(\mathbf{w}_{21}, \mathbf{x_t})$
    if $y_{21t} == 1$ **then**
      Turn on sensors: $S_1 \cup S_{21} \cup S_3$
    end if
  else
    Turn on sensors: $S_1 \cup S_{22}$
  end if
end for

Table 6.1: Sensors Power Consumption

Sensor in Use	Power Consumption Ratio
Phone Idle	1
Light Sensor(1 Hz)	1
Barometer Sensor(1 Hz)	1.02
Accelerometer(5 Hz)	1.59
Magnetometer(5 Hz)	1.45
Gyroscope(5 Hz)	1.82

experiments to estimate the sensors' battery consumption on several phone models (iPhone 5, 6, Samsung Galaxy S4). The power usage is estimated at different sampling frequencies when the target sensor is turned on and the other sensors and related Apps are all turned off. We combine the results of our experiments with conclusions from other literature [86–88] to give the estimated sensors battery consumption of the 5 major sensors used in this thesis. Instead of giving the exact numbers, which might be different from phone to phone, we use the ratio of battery consumption by comparing to the battery consumption when the phone idles and the result is shown in Table 6.1.

We compare the average sensors in use, the total calculation time and the prediction accuracy. The results are shown in Figure. 6.4. We can see that the hierarchical model is comparable to the general model in prediction accuracy. Meanwhile it outperforms the general model in total sensors in use as well as the time spend for computing. We estimated the power comsumption based on the conclusion in Table 6.1 and the result shows that the hierarchical model works well in improving the battery efficiency on smartphones and uses only 66% of the battery consumption in generic model.

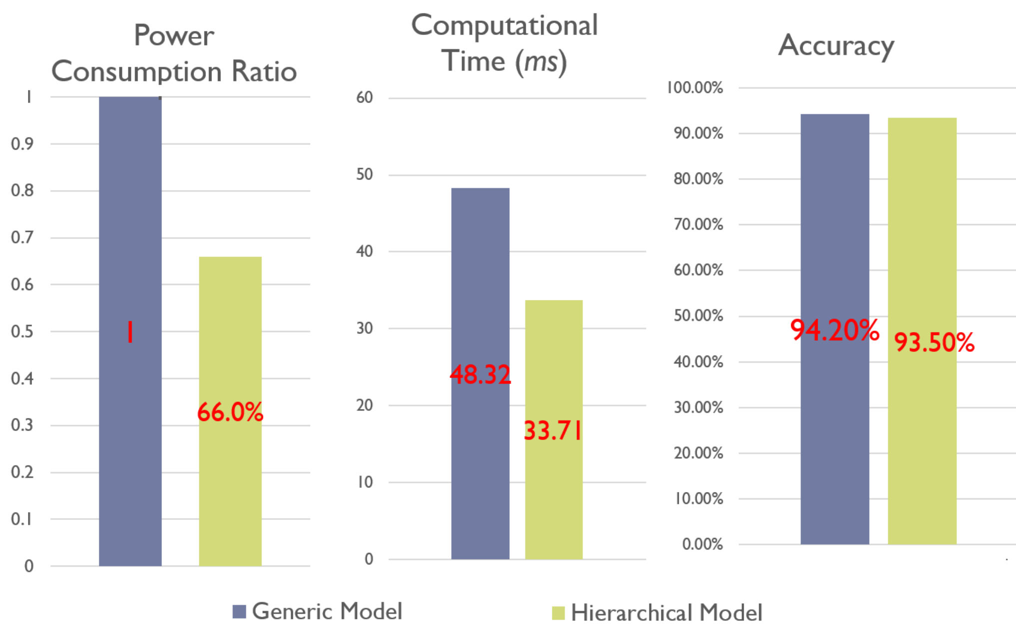


Figure 6.4: Comparison of Hierarchical Model and General Model

Besides the fact that the sensors' usage is decreased with the hierarchical model during the travel mode identification, another benefit of the hierarchical structure is that it provides the users an option to choose the level of the details of the output. For example, some transportation surveys are only interested in distinguishing wheeled travel modes from unwheeled travel modes. In this case, only first-level classification would be applied.

Chapter 7

Personalized Model

Another fundamental limitation in the travel mode detection is the lack of *personalization*. That is, existing methods tend to train a generic classifier for all users, while the behaviors in travel may differ greatly from one user to another. This limitation indicates that a personalized treatment for the travel mode detection may result in better prediction accuracy. However, following personalized treatment, another limitation stems from the *label sparsity*. That is, there may be a very limited set of labels for a certain user, making the trained classifier less effective.

To address the above challenges, in this thesis, we propose a personalized travel mode detection method *PerTMoD* based on the data from smartphone sensors. We focus on the detection of six major travel modes in modern urban society: *driving a car, walking, jogging, bicycling, taking a bus*, and *taking a subway*. The basic idea of *PerTMoD* is to apply Kernel Mean Matching [89] to address the *personalization* limitation, and adopt transfer learning [90] to address the *label sparsity* limitation. In particular, the proposed *PerTMoD* method can be divided into two stages. In the first stage, *PerTMoD borrows* data from the users whose labeled data is already collected in the database for a given target user. Here, we propose a method to calculate the user similarity scores between the target user and the existing users within each travel mode. In the second stage, *PerTMoD reweights* the borrowed data and use it as the training data for the target user. The proposed method is able to estimate the sample distributions, and then reweight the samples based on the estimated distributions so as to minimize the model loss with respect to the target user's data.

7.1 Problem Description

In this section, we first introduce the terminology and notations used in this section, and then present the problem definition.

Base data. We use base data B to indicate the pre-collected smartphone data. Each user in the base data has a collection of data samples, and each data sample is represented as $\{\mathbf{x}, y\}$, where $\mathbf{x} \in \mathbb{R}^p$ is a p-dimensional feature mapped from the raw data and $y \in \{y_1, y_2, ..., y_m\}$ is one of the m travel mode labels.

Target data. The target data T is the data from the user that we aim to predict the travel modes for. We assume that there is a small portion of available labels in T. We further denote the data distribution of the feature space in T as P_T.

Source data. The source data S is a subset of the base data B. It is selected from B with the aim to learn a personalized model for the user in target data T. The data distribution of the feature space in S is denoted with P_S.

Based on the above terminology, we define the personalized travel mode detection problem as follows.

Problem 1. *Personalized Travel Mode Detection Problem*

Given: *(1) a collection of existing users' travel data B where all the data samples are labeled, and (2) the target user's travel data T where only a small portion of data samples are* *labeled;*

Find: *(1) a subset of source data S from B with respect to T; (2) the weight for each data sample in S, and (3) the labels for the unlabeled data in T.*

As we can see from the above definition, we assume that there is a set of labeled travel data B pre-collected from different users. Then, to fulfill the personalization of the target user that we aim to detect the travel modes for, we select a suitable subset of data samples S for this user, reweight them, and then make the predictions based on these reweighted data samples as well as the partially labeled data in T.

7.2 The Proposed *PerTMoD*: Overview

In this section, we present the overview of the proposed solution *PerTMoD* for Problem 1.

Due to the diversity in individuals' travel behaviors, simply learning a generic model from the labeled data may not result in good performance. Additionally, learning a personalized model for a target user may face the issue of insufficient labeled data which would cause a high variance of the trained travel model detector. In this work, we aim to transfer the knowledge from the pre-collected data, and adapt the knowledge to the target user's travel behaviors. The overview of *PerTMoD* is shown in Fig. 7.1.

As we can see from Fig. 7.1, given the input, there are two major steps:

1. *Sample selection*: what kind of data to *borrow* for a given target user in T?

2. *Sample reweighting*: how to *reweight* the borrowed data for the target user?

For the first step, the basic idea is to select samples from B that are most similar to the small portion of labeled data in T. Here, one of the key challenges is to quantify the

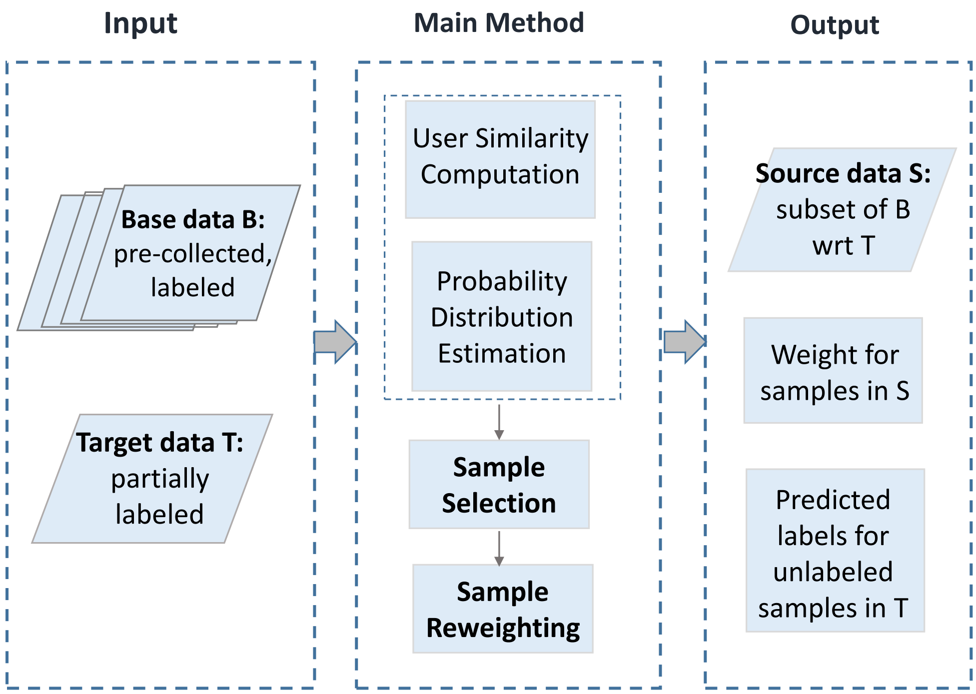


Figure 7.1: Overview of the Proposed *PerTMoD*

similarities between users in terms of their travel behaviors. The similarity reflected in the data would be similar data range and fluctuation in motion related sensor readings, or similar environment parameters like ambient air pressure, etc. Meanwhile, we observed that the non-sensor factors such as gender, city of travel events, and smartphone types also play important roles in the user similarity computation. For example, the phone model as well as the operating system defines how sensitive the sensors are, which directly affects the data readings. We will explore both sensor factors and non-sensor factors in our user similarity computation. Moreover, since it is possible two users having similar jogging patterns might act quite differently in driving, the similarity computation is conducted within each travel mode.

For the second step, the key challenge is to reweight the selected samples (i.e., source data S) from the first step, so as to minimize the mean distance between the data in S and the labeled data in T. In this work, we adopt Kernel Mean Matching (KMM) method [89,91] for this step. KMM is used under the situations when the distributions of training data and test data do not match, and it proposes to minimize the loss of learning by reweighting the samples in the training set. Formally, KMM aims to find the suitable weights $\beta(\mathbf{x})$ for the minimization of the following formulation

$$\begin{aligned} R[P_T, \theta, l(\mathbf{x}, y, \theta)] &= \mathbf{E}_{(\mathbf{x},y)\sim P_T}[l(\mathbf{x}, y, \theta)] \\ &= \mathbf{E}_{(\mathbf{x},y)\sim P_S}[\beta(\mathbf{x})l(\mathbf{x}, y, \theta)] \end{aligned} \tag{7.1}$$

where P_T and P_S are the estimated distributions. The key finding of KMM is that when $\beta(\mathbf{x}) = \frac{P_T(\mathbf{x},y)}{P_S(\mathbf{x},y)}$, the kernel mean difference between the test data and the reweighted training data is the minimum. In our problem setting, we also match the distributions within each travel mode.

The output of *PerTMoD* is the selected and reweighted data S from B with respect to T, as well as the predictions which can be made based on S and T. In the following section, we will present the details of the two major steps in *PerTMoD*.

7.3 The Proposed *PerTMoD*: Details

In this section, we present the details of the proposed *PerTMoD*, including similarity computation, distribution estimation, sample selection, and sample reweighting, followed by some algorithm analysis.

7.3.1 Similarity Computation

To analyze the similarity of two users' data, we first consider the sensor factors/features that are extracted from the sensor data. The used sensors include accelerometers, barometers, magnetometers, and gyroscope (see the experimental section for more details). Based on the data readings, we define the following two metrics.

- **Data Center**. A data center is associated to a specific user and a specific travel mode. We denote a data center associated with user i and travel mode k by $C_i^{(k)}$ $(k = 1, 2, ..., m)$

$$C_i^{(k)} =< \bar{x}_{i,1}^{(k)}, \bar{x}_{i,2}^{(k)}, ..., \bar{x}_{i,p}^{(k)} > \tag{7.2}$$

  where $\bar{x}_{i,j}^{(k)}$ is the mean of the j-th features that is associated to user i and mode k.

  For user i, the data center profile is a vector of length m: $< C_i^{(1)}, C_i^{(2)}, ..., C_i^{(m)} >$. If this user does not have certain mode k in his/her travel data, we will skip this value in calculation.

- **Distance**. The travel data distance is defined as the the Euclidean distance of two data centers of the same mode. We denote it by $D_{i,i'}^{(k)}$.

$$D_{i,i'}^{(k)} = \|C_i^{(k)} - C_{i'}^{(k)}\|_2 \tag{7.3}$$

For non-sensor factors (which is related to a user's personal information), we consider the following four factors: gender, phone model, city of travel, and the season when travel happened. We use A to denote the non-sensor similarity matrix and each entry $A_{i,i'}$ is defined as

$$A_{i,i'} = \sum_l a_{i,i'}^{(l)} \tag{7.4}$$

with

$$a_{i,i'}^{(l)} = \begin{cases} 1 - \xi, & \text{if user } i \text{ and } i' \text{ have the} \\ & \text{same value in } l\text{-th factor,} \\ \xi, & \text{otherwise.} \end{cases}$$

where $0.5 \leq \xi < 1$ is a parameter to control the importance of non-sensor factors.

Based on the Eq. (7.3) and Eq. (7.4), we define the similarity score between two users under the same label k as

$$o_{i,i'}^{(k)} = \frac{A_{i,i'}}{D_{i,i'}^{(k)}} \tag{7.5}$$

As we can see from the above equation, the higher the non-sensor similarity and the smaller the sensor distance, the more similar the two users are. For parameter ξ, smaller value of ξ indicates higher impact of the non-sensor factors.

For a given label k, we can generate its similarity matrix $O^{(k)}$ with its entries as defined in Eq (7.5). The data samples from the top-n similar users form the candidate set S_C for the source data S.

7.3.2 Distribution Estimation

Another essential component of our method is the distribution estimation. For any data sample $\mathbf{x}_i$ in the feature space, we define its potential to another data point $\mathbf{x}_j$ in the same space as $V(\mathbf{x}_i, \mathbf{x}_j)$,

$$V(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 + \gamma} \tag{7.6}$$

where γ is a variable that defines the sharpness of the potential distribution. The more close two data points are, the higher the potential is between them. Based on the potential, we can define the probability distribution of any data S under a certain label by the density function:

$$\begin{aligned} P_S(\mathbf{x}|y) &= N_S \sum_{\mathbf{x}_j \in S} V(\mathbf{x}, \mathbf{x}_j) \\ &= N_S \sum_{\mathbf{x}_j \in S} \frac{1}{\|\mathbf{x} - \mathbf{x}_j\|_2^2 + \gamma}. \end{aligned} \tag{7.7}$$

where N_S is the normalizer for data S so that $\sum_{\mathbf{x}_i \in S} P_S(\mathbf{x}_i) = 1$. With this estimation, we can use the mean matching process to assign weights such that the distributions of two sets of data samples are close to each other.

7.3.3 Sample Selection

Simply using the data samples from most similar users (i.e., the output of Subsection 4.1) may cause the label imbalance problem. Here, we tackle this problem by defining a filter function.

The data from top-n similar users will form a ball such that all the data samples fall inside the sphere of the ball. We define a ball G whose center is $\mathbf{c}$ and whose radius is r. Using all the data from top-n similar users means that r should satisfy the following inequality

$$r \geq \sup_{\mathbf{x} \in T} \{\|\mathbf{x} - \mathbf{c}\|_2\}.$$

In order to balance the data between S and T, we choose a smaller r to serve as a filter. That is, we define the following filter function $F_G(\mathbf{x}) : S \cup T \mapsto \mathbb{R}$

$$F_G(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in G \\ 0, & \text{otherwise.} \end{cases} \tag{7.8}$$

The filter depends on the choice of r and it filters out the points outside the ball G in the feature space. We apply the filter on the candidate set S_C to obtain S. By doing this, we make sure that the points in T are contained in the range of S. The updated P_S is given by

$$P_S'(\mathbf{x}|y) = \frac{P_S(\mathbf{x}|y) \cdot F_G(\mathbf{x})}{\sum_{\mathbf{x}' \in S} P_S(\mathbf{x}'|y) \cdot F_G(\mathbf{x}')} \tag{7.9}$$

We do not need to apply F_G on P_T because T is already inside the scope of G by definition.

7.3.4 Sample Reweighting

So far, we have the source data S selected. We next move to the sample reweighting via distribution matching. A prerequisite of distribution matching is that the two distributions are overlapped in their domain of definition, which is guaranteed by our sample selection method. Basically, the weight is assigned towards each $\mathbf{x}$ in S according to its relationship to the samples in T. Here, we directly show the computation of weights in the following equation and leave the proof for the next subsection.

$$\begin{aligned}\beta_y(\mathbf{x}) &= \frac{P_T(\mathbf{x}|y)}{P'_S(\mathbf{x}|y)} \\ &= \frac{\sum_{\mathbf{x}_j\in T}\frac{N_T}{\|\mathbf{x}-\mathbf{x}_j\|_2^2+\gamma}}{\sum_{\mathbf{x}_j\in S}\frac{N_S}{\|\mathbf{x}-\mathbf{x}_j\|_2^2+\gamma}}\end{aligned} \tag{7.10}$$

If T and S are the same set, we have $\beta_y(\mathbf{x}) = 1$ for any $\mathbf{x}$.

The overall algorithm for our *PerTMoD* is summarized in Alg. 2. In the algorithm, we compute the similarities between data samples in B and T, select the samples from B, and then compute the weights of the selected samples, respectively. The above steps are conducted in each travel mode. We apply the output of the algorithm on the weighted SVM as shown in Eq. (7.11) to predict the labels for unlabeled data samples in T.

$$\begin{aligned} \underset{\mathbf{w},\xi}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n_{tr}}\beta(\mathbf{x}_i)\xi_i \\ \text{subject to} \quad & y_i\langle\mathbf{w},\mathbf{x}_i\rangle \ \geq 1-\xi_i \\ & \xi_i \geq 0, i=1,2,... \end{aligned} \tag{7.11}$$

7.3.5 Algorithm Analysis

Here, we briefly analyze the proposed algorithm (Alg. 1).

The proposed algorithm is built upon the Kernel Mean Matching [89]. It is proved that

Algorithm 2 The *PerTMoD* algorithm.

Input: base data B and target data T
Output: source data S and its weight vector β_S
Parameters: top-n most similar users in sample selection, the similarity score parameter ξ in Eq. (7.5), and the potential parameter γ in Eq. (7.6)

$S \leftarrow \{\}$;
$\beta_S \leftarrow \{\}$;
for all $k \in 1 : K$ **do**
    Calculate the similarity matrix $O^{(k)}$ between B and labeled data in T by Eq. (7.5);
    Select n most similar users in current travel mode k to form S_C;
    Apply the filter in Eq. (7.8) on S_C to obtain S_k;
    Calculate the weight $\beta_{S_k}(\mathbf{x})$ for $\mathbf{x} \in S_k$ by Eq. (7.10);
    $S \leftarrow S \cup S_k$;
    $\beta_S \leftarrow \beta_S \cup \beta_{S_k}$;
end for
return S and β_S;

the loss function in Eq. (7.1) is bounded given that the solution $\beta(\mathbf{x})$ minimizes the following problem

$$\begin{aligned} \min_{\beta} \quad & \|\mu(P_T) - \mathbf{E}_{x\sim P_S(\mathbf{x})}[\beta(\mathbf{x}) \cdot \mathbf{x}]\| \\ \text{subject to} \quad & \beta(\mathbf{x}) > 0, \\ & \mathbf{E}_{x\sim P_S(\mathbf{x})}[\beta(\mathbf{x})] = 1. \end{aligned} \tag{7.12}$$

In the following lemmas, we show that the proposed sample weight estimation in Eq. (7.10) also meets the condition in Eq. (7.12).

Lemma 1. *In each labeled class y_i, given the construction of $P_T(\mathbf{x}|y_i)$ and $P_S(\mathbf{x}|y_i)$, $\beta_{y_i}(\mathbf{x}) = \frac{P_T(\mathbf{x}|y_i)}{P_S(\mathbf{x}|y_i)}$ guarantees the minimum of the following problem:*

$$\begin{aligned} \min_{\beta_{y_i}} \quad & ||\mu(P_T|y_i) - \mathbf{E}_{\mathbf{x}\sim P_S(\mathbf{x}|y_i)}[\beta_{y_i}(\mathbf{x}) \cdot \mathbf{x}]|| \\ \textit{subject to} \quad & \beta_{y_i}(\mathbf{x}) > 0, \\ & \mathbf{E}_{x\sim P_S(\mathbf{x}|y_i)}[\beta(\mathbf{x}|y_i)] = 1. \end{aligned} \tag{7.13}$$

Proof. First, since we focus on the importance of samples within each label class y_i with respect to the data samples of the same class in T, for a fixed y_i, we have

$$\beta_{y_i}(\mathbf{x}) = \frac{P_T(\mathbf{x})}{P_S(\mathbf{x})} = \frac{P_T(\mathbf{x}|y_i)}{P_S(\mathbf{x}|y_i)} \tag{7.14}$$

In Section 7.3.3 the filter function applied on P_S in Eq. (7.8) guarantees that range of of data S in the feature space contains T, thus $P_T \ll P_S$. This guarantees the validity of Eq. (7.13). Then, by the definitions of P_S and P_T, the calculation of $\beta_{y_i}(\mathbf{x})$ in Eq. (7.14) satisfies the constraints and gives Eq. (7.13) the value of 0, which completes the proof. □

Based on Lemma 1, we have that the minimization within each class label guarantees the overall minimization of the distribution matching, as shown in the following lemma.

Lemma 2. $\beta_y(\mathbf{x})$ *as defined in Eq.* (7.10) *guarantees the minimization of Eq.* (7.12).

Proof. By Lemma 1, the following term

$$\begin{aligned}&\min_{\beta} ||\mu(P_T) - \mathbf{E}_{x\sim P_S(\mathbf{x})}[\beta(\mathbf{x})\cdot \mathbf{x}]|| \\ &= \sum_{y_i} \min_{\beta_{y_i}} ||\mu(P_T|y_i) - \mathbf{E}_{x\sim P_S(\mathbf{x}|y_i)}[\beta_{y_i}(\mathbf{x})\cdot \mathbf{x}]||\end{aligned}$$

is guaranteed to be the minimum. The constraint on β such that $\mathbf{E}_{x\sim P_S(\mathbf{x})}[\beta(\mathbf{x})] = 1$ is guaranteed by the definition of P_S. Thus, with the probability estimation in Alg. 1, the sum of the minimization of distribution discrepancy within each class leads to the minimization of the discrepancy of the mean of the two distributions, which completes the proof. □

7.4 Experiment

In this section, we show the data collection, present the experimental setup, and discuss the experimental results.

7.4.1 Experiment Setup

In our experiments, we select 10 users as the target users and use their data as target data T. For this user, we hide $(1 - \alpha)$ of its labels. We assume that there are a small portion of available labels, which means α is usually very small (e.g., 1%). We use the data from other users as base data B. The proposed method is applied on the target data and base data. Then, we report the prediction accuracy of each user when this user is selected as the target user.

To evaluate the effectiveness of *PerTMoD*, we compare the following methods:

- *Generic*. The generic method ignores user personalization and trains the classifier on all available labeled data.

- *Basic*. The basic method indicates the case when only the labeled data in T is used as training data.

- *Random*. Personalized prediction with transferred data randomly selected.

- *Selection*. Personalized prediction where we use the transferred data samples in S without reweighting.

- *R-Selection*. Personalized prediction where we add the data samples that are not in S as training data.

- *R-Reweighting*. Personalized prediction when we add sample reweighting on the *R-Selection* method.

- *PerTMoD*. The proposed personalized travel mode detection method.

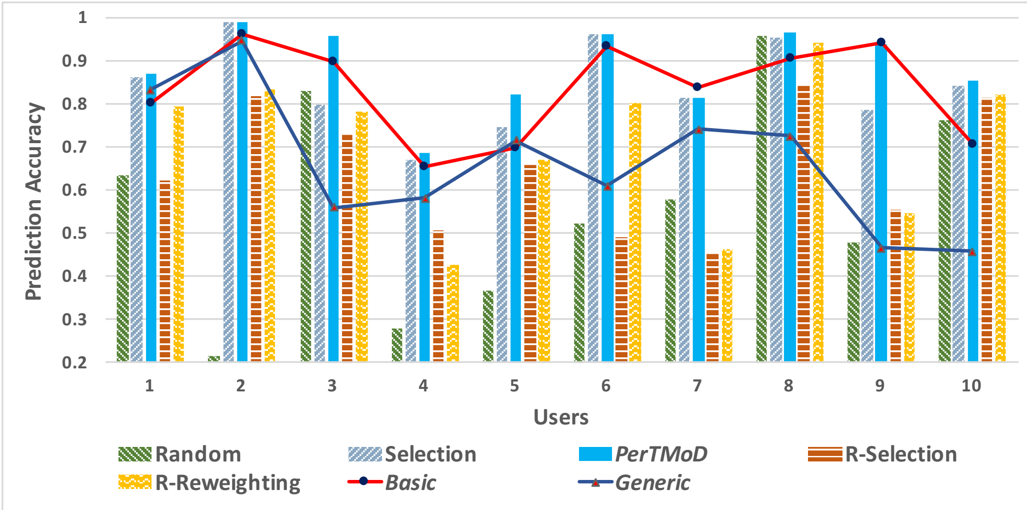


Figure 7.2: The effectiveness comparisons of *PerTMoD*. *PerTMoD* generally outperforms the compared methods.

7.4.2 Experiment Results

First, we compare the effectiveness of different methods, and the result are shown in Fig. 7.2 where the x-axis is the user id, and the y-axis is the prediction accuracy. The label portion is set as 2% (i.e., $\alpha = 2\%$) in the figure. The three parameters of the *PerTMoD* method are tuned via cross validation. In particular, the reported results are under the parameter setting $n = 3$, $\xi = 0.2$, $\gamma = 0.003$.

We can first observe from the figure that, the *PerTMoD* method generally outperforms all the compared methods. For example, averaging over the 10 users, *PerTMoD* improves the *Generic* method by 22.3% in terms of absolute prediction accuracy. This result validates the effectiveness of the personalization treatment in *PerTMoD* for the travel mode detection problem. Additionally, *PerTMoD* is averagely 5.2% better than the *Basic* method. This result indicates the usefulness of transferring knowledge from existing data. Second, we can observe that *Selection* is better than the *Random* method. This result verifies the effectiveness of the proposed data selection method (i.e., the first stage) in *PerTMoD*. Third, *PerTMoD* is better than the *Selection* method. Moreover, we find that applying the reweighting method on the data that is not in S can still improve the prediction accuracy (i.e., *R-Reweighting* is better than *R-Selection*). This result further verifies the usefulness of sample reweighting (i.e, the second stage in *PerTMoD*).

Figure 7.3: The comparisons among *PerTMoD*, *Generic*, and *Base* methods with α varies. *PerTMoD* is consistently better.

Next, the effectiveness results when the portion (α) of available labels in T varies are shown in Fig. 7.3. In the figure, we vary α from 1% to 10%, and report the average prediction accuracy over all users. As we can see, *PerTMoD* consistently outperforms the *Generic* method and the *Basic* method, indicating the usefulness of both the proposed personalization treatment and the proposed knowledge transfer method.

Chapter 8

Online Model

8.1 Problem Description

In previous section the proposed method *PerTMoD*solves the problem of model adaption to a specific user. In this section we focus on the problem of temporal model adaption. That is, given a user-specific model, how do we update it with new labeled data? The common scenario for this problem is when a user provides the feedback (correct label) to his/her current travel mode. With the new labeled data, we would like to update the pre-trained model. Re-train the model with the whole udpated dataset will be both time and resource consuming, providing the scale of travel data volume. Ideally, the shorter the model updating process is, the better. To meet this criteria, the classification model should satisfy the following two principles:

1. The classification model can process a small batch of samples despite the data used previously. In other words, it should use online updating strategy.

2. The model updating algorithm should not put restrictions on the data scale since each update should only depend on data sensed from the current traveler's behavior.

In the following part we propose an online learning model that meets the above principles and it detects the travel mode with promising accuracy.

8.2 SVM Model

Support vector machines (SVMs) are a supervised classification tool that provides the largest margin between two hyperplanes of the classes in the multi-dimensional feature space. In a binary classification problem, we have a training set $S = \{(\mathbf{x_i}, y_i)\}_{i=1}^{m}$, where $\mathbf{x_i} \in \mathbf{R}^n$ and $y_i \in \{+1, -1\}$. The pair $(\mathbf{x_i}, y_i)$ is composed of an arbitrary input x and the prediction label y. To train a SVM is to find the minimizer of the following problem:

$$\min_{\mathbf{w},\xi} \left(\frac{\lambda}{2} \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{(\mathbf{x},y)\in S} \ell(\mathbf{w}; (\mathbf{x}, y)) \right) \tag{8.1}$$

Where,

$$\ell(\mathbf{w}; (\mathbf{x}, y)) = \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x} \rangle\} \tag{8.2}$$

We denote the objective function in Equation (6) by $f(\mathbf{w})$. Subgradient descent has often been proposed to find a solution for the approximate objective functions like $f(\mathbf{w})$ [92]. A simplification for subgradient descent is stochastic gradient descent (SGD). SGD allows the update on batch gradient descent with randomly picked samples at each iteration [93]. At each iteration, the update is given by:

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w_t} + \eta \nabla_{\mathbf{w}} f \tag{8.3}$$

Since the stochastic algorithm does not need to remember which examples were visited during the previous iterations, it can process examples on the fly in a deployed system [94].

In the travel mode identification problem, on the other hand, different users could behave very differently in travelling. Therefore it is possible that certain user's behavior shows a huge difference from the data we used to train the general model. In the meanwhile, different users may need different size of new data for the model updating in order to reach a stable performance. Thus the model updating requires user-specific samples and the outputs should be user-specific parameters. In our design, we require a model updating algorithm that does not restrict the data scale and can process with a small batch of samples despite the previous data. The SGD algorithm fits our requirements well.

8.3 Online Learning

In [95], Shalev-Shwartz et. al proposed the SGD algorithm: Primal Estimated sub-GrAdient SOlver (Pegasos) for SVM. Pegasos works solely on the primal objective function at each iteration thus its running time does not depend on the entire training set size [95]. The sub-gradient of the approximation in Equation (6) is then given by:

$$\nabla_t = \lambda \mathbf{w}_t - \mathbb{1}[y_{i_t} \langle \mathbf{w}_t, \mathbf{x}_{i_t} \rangle < 1] y_{i_t} \mathbf{x}_{i_t} \tag{8.4}$$

where $\mathbb{1}[y_{i_t} \langle \mathbf{w}_t, \mathbf{x}_{i_t} \rangle < 1]$ is the indicator function that takes a value of one if $\mathbf{w}$ yeilds non-zero loss on the example $(\mathbf{x}, y)$. Substituting ∇_t in Equation (8) with Equation (9), and use learning rate $\eta_t = 1/\lambda t$, the update of $\mathbf{w}$ is

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w_t} + \eta_t \nabla_{\mathbf{w}} \mathbb{1}[y_{i_t} \langle \mathbf{w}_t, \mathbf{x}_{i_t} \rangle < 1] y_{i_t} \mathbf{x}_{i_t} \tag{8.5}$$

In our model updating mechanism, we take the initially trained (user specific) model as the input, and employ the Pegasos method as our online model updating method and update the model with a single training instance. Fig. 3 shows the pseudocode of the online model updating algorithm. Note that the updating process is based on a pre-trained model. The input $\mathbf{w_0}$ in the model updating process is the general model from the first phase training, which is different from [95]. We use OVA (One Versus All) strategy for the multi-class classifier.

Algorithm 3 Online Learning with Pegasos

Input: $\mathbf{w_0}$, the weight vector of the general model.
λ, the regularization parameter.

Output: The updated weight vector of the learning model: $\mathbf{w_{T+1}}$

Parameters: T, the maximum training epoch, $\mathbf{S}$, the training set.

$t \leftarrow 0$

while $t < T$ **do**
    $\eta_t = \frac{1}{\lambda t}$

    choose $i_t \in \{1, 2, ..., |\mathbf{S}|\}$ uniformly at random

    if $y_t < \mathbf{w_t}, \mathbf{x_t} >< 1$ **then**
        Set $\mathbf{w_{t+1}} \leftarrow (1 - \eta_t \lambda)\mathbf{w_t} + \eta_t y_t \mathbf{x_t}$

    else
        Set $\mathbf{w_{t+1}} \leftarrow (1 - \eta_t \lambda)\mathbf{w_t}$

    end if
    $\mathbf{w_{t+1}} \leftarrow \min\{1, \frac{1/\sqrt{\lambda}}{\|\mathbf{w_{t+1}}\|}\}\mathbf{w_{t+1}}$

    $t \leftarrow t + 1$
end while
return $\mathbf{w_{T+1}}$

8.4 Experiment

In this section, we explain the development of experiment methodology, and then analyze the results and discuss the solutions as well as existing problems. We take the best performance of the general learning models in [96] as the baseline for the performance analysis. Our goal is to verify that the online updating solution that follows the principles of short system response time and less calculation, meanwhile it performs as well as the baseline. To begin with, we first raise the following questions that we aim to find the answer to each of them in following part:

1. The baseline in [96] is using Bayes Net learning model. Here we demonstrate the online learning model with Pegasos (SVM). Before developing the online model, we need to fill the comparison gap between the method in [96] and the online learning method: the performance of offline learning with Pegasos (SVM) as the learning algorithm. Would the offline with Pegasos perform as well as the results in [96]? Our expectation is that the Pegasos updating with the whole dataset, as offline learning, should achieve similar performance in classification accuracy as in [96].

2. We would further promote the learning algorithm to online mode that update the pre-trained classification model with a single instance. Based on the expected results of the first question, would the online updating give promising results compared to the offline method?

3. Since the online model updates itself using a single instance, would it excel the offline training mode in computation time?

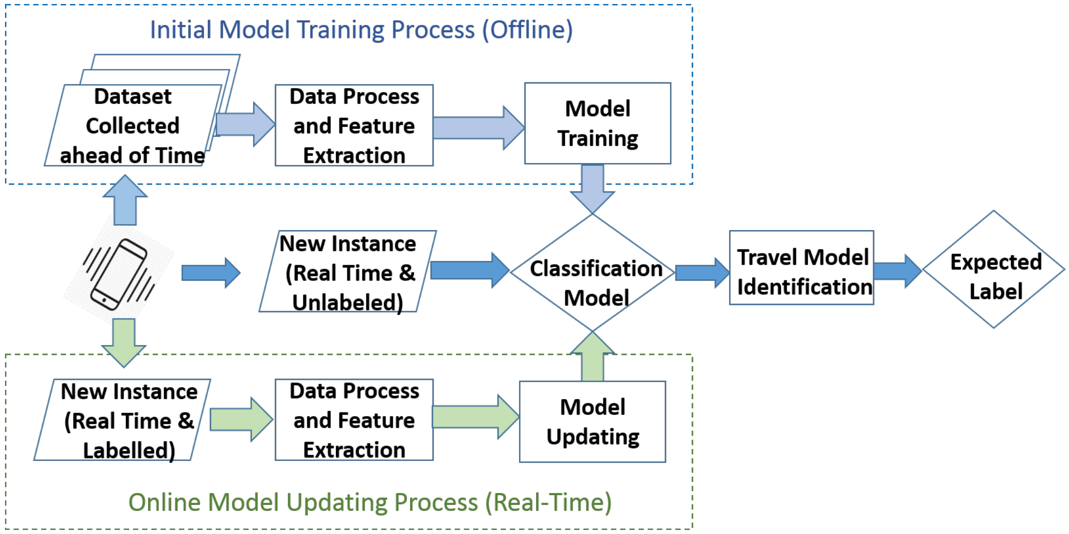
First we compare the baseline with both online and offline updating models. Then we describe the method to improve the system's performance in terms of (i) energy consumption, (ii) response time, (iii) the amount of data needed for training an adaptive model for smartphones users.

8.4.1 Online Learning VS Offline Learning

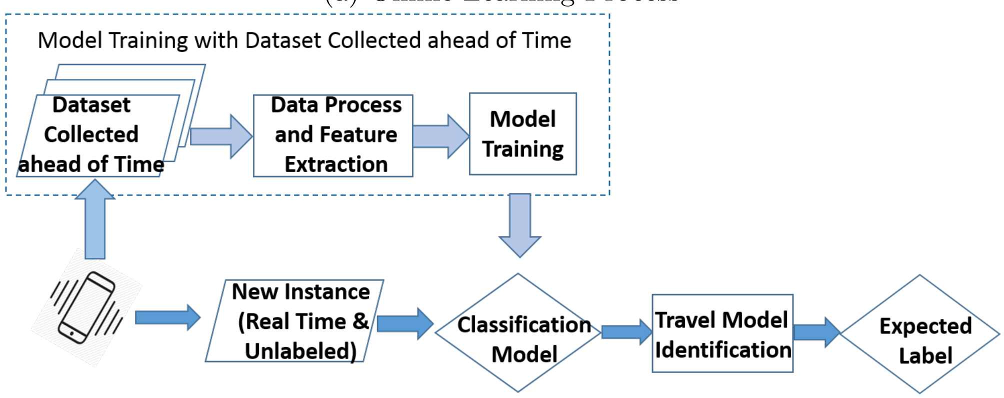
To compare the online updating strategy with the traditional offline training method, we use Pegasos batch updating method in the offline training and take the whole training set as the batch size. In the online mode, we update the model with each single instance. In the offline mode, the dataset used to train a new model is the combination of the existing dataset and the new instance (the old model is abandoned). Fig. 8.1 shows the process of online updating and offline updating. In order to compare the performance of online learning with offline learning, we repeat the offline learning with the whole updated dataset every time after the new sample data sample comes in. By doing so, we guarantee the offline model we compare with is trained with same data as our online model.

We compared the two updating process in prediction performance and time cost. In both experiments, we use the data collected with Android phones in winter.

The baseline we compare to is the best results in [96] using Bayes Net Model. Fig. 8.2 shows that both online updating and offline updating using SVM with subgradient descent solver achieve promising results. The accuracy of the offline learning model is as good as the baseline, this answers our first question. The online updating model begins with lower accuracy: 65% recall and 75% precision in prediction. The performance improves significantly after about 50 iterations. And the accuracy of online and offline updating model converge as more data is used for training. This answers the second question. Figure 8.3 shows the time cost of online and offline updating. The time cost of online updating is relatively stable and accumulated to less than $0.01s$ for nearly 250 iterations (data generated in about 53 minutes) of updating, while the time cost of offline updating increases faster and it reaches $85s$ around the 250th iteration. So the online updating is superior to the offline



(a) Online Learning Process



(b) Offline Learning Process

Figure 8.1: Online VS Offline Learning Process

model. So far we've answered all three questions from the beginning of this section, and the principle 1 of online updating strategy is guaranteed.

8.4.2 How much data is enough to train a initial model?

In machine learning problems in general, the size of training dataset is also critical. Smaller dataset usually results in less accurate models. In this paper, since the model is updated with new data samples, the whole system performance should less depend on the initial training set. In order to see how much data is necessary to train the initial dataset, we did

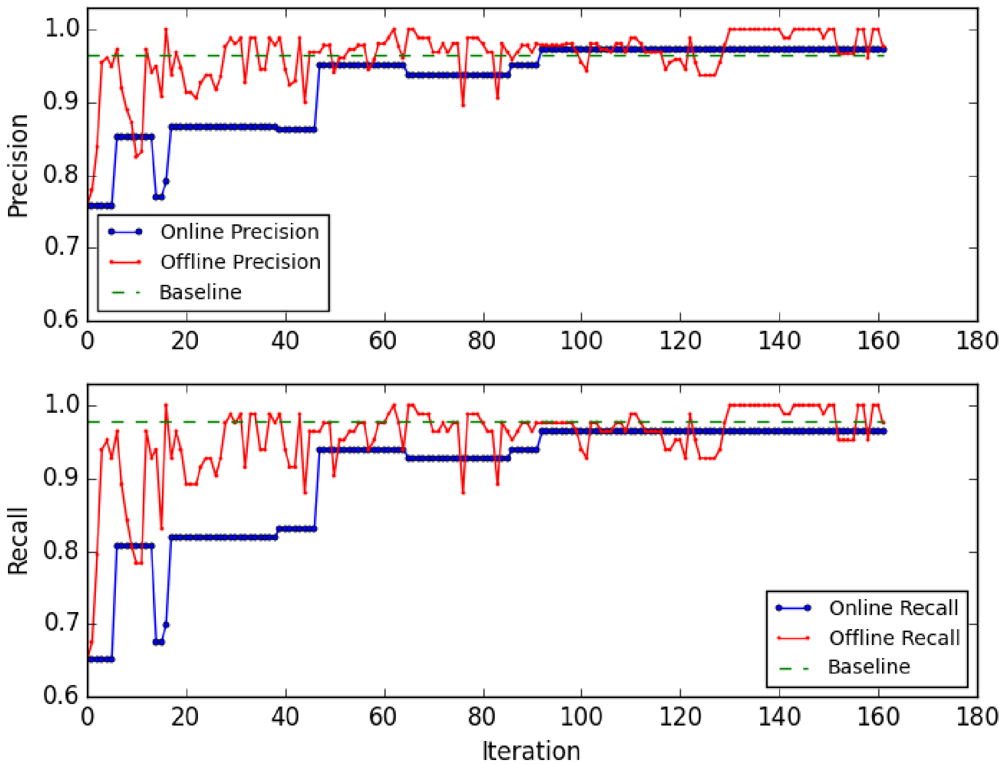


Figure 8.2: Performance Comparison of Online and Offline Model

experiment with different initial training sets (we take 20%, 30%, 40%, 50% of the whole training set as initial training set and the rest is the add-up training set). Figure 8.4 shows the performance comparison with different initial training sets. It is found that smaller initial training datasets have lower accuracies at the beginning. However, as the model is updated with new coming instances, the accuracies increase and tend to converge. The experiment result reaches as high as 90% predicting precision even if only 20% of the data is used as initial training set. This result means the system meets the 2nd principle so far as it promises a quick start for the online learning model.

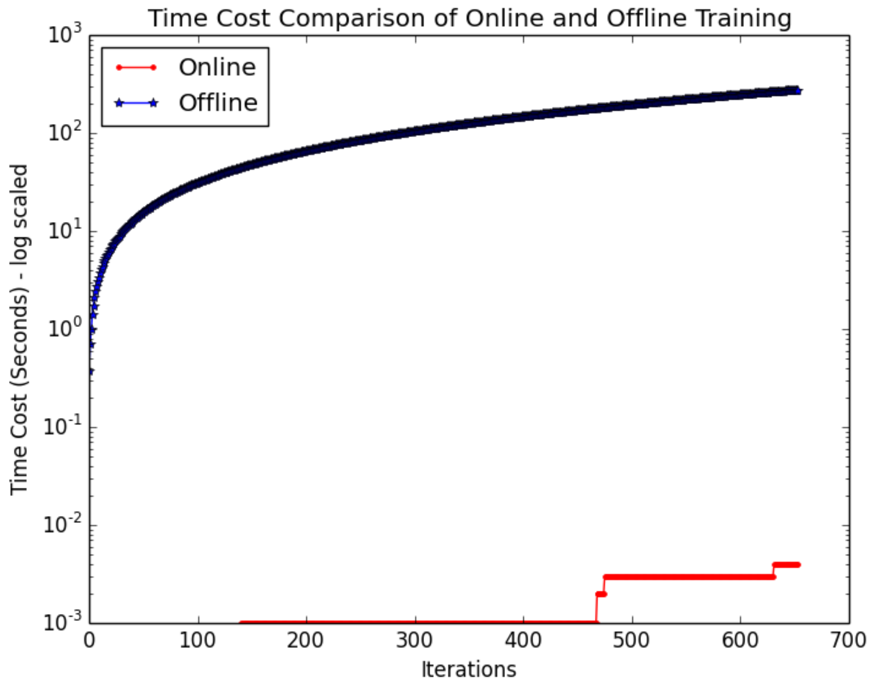


Figure 8.3: The Time Cost of Online and Offline Updating

8.5 A Summary of Performance

We present the development of methodology and how it meets the outlined design principles in previous sections. We now demonstrate the system performance using our experiment results. The confusion matrix of the experiment result is in Table 8.1. The first layer classification accuracy reaches 96%. Second layer classification accuracy is 96.7% for unwheeled travel mode and 95.4%.

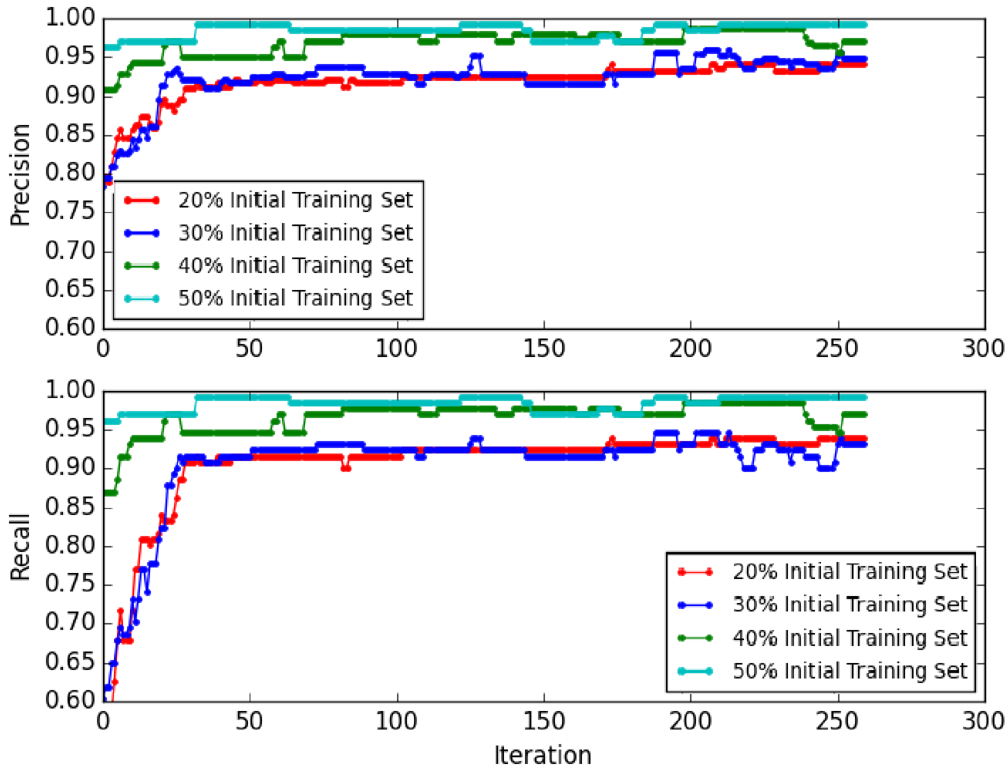


Figure 8.4: Prediction Performance with Initial Training Sets of Different Size

Table 8.1: Classification Result

First Layer	Accu -racy	Second Layer	Confusion Matrix			
Unwheeled Modes	96%	Classified as	Walk	Jog		
		Walk	2523	39		
		Jog	54	260		
Wheeled Modes		Classified as	Bike	Bus	Subway	Car
		Bike	1241	0	0	1
		Bus	2	159	23	621
		Subway	49	42	4862	115
		Car	170	54	18	16877

Chapter 9

Conclusion and Future Work

9.1 Discussion

In previous sections we analyze the performance of the online learning model with different configurations. In this section, we will show some other interesting and useful aspects of the data we collected.

9.1.1 Local Street or Freeway?

Because the smartphone GPS sensor is quite weak in subways, inaccurate in urban areas and is energy consuming, we did collect but did not use GPS data for the model training and prediction. However by analyzing the GPS data on its own, it reveals to us whether the drive is on local or highway. In Android phones, the GPS modular provides the estimated speed [77] besides the longitude and latitude of the current position. Among all of the differences between driving in freeway and local streets, a simple fact is that the driving speed can drop to 0 for many times when driving on local roads. This is merely happening during freeway driving unless there's a traffic jam, which is rare and not considered in our discussion. Fig. 9.1 shows the GPS estimated speed for a vehicle's local and freeway drive, respectively. We can see that besides the fact that freeway drive has a higher speed most of the time, the speed of local drive reaches 0 for several times while the freeway drive does not show such aspects. This information does not need a very accurate speed estimation or strong GPS signal, nor does it require a high sampling rate. Therefore, we can use GPS at very low frequency to detect whether the vehicle is on freeway or not.

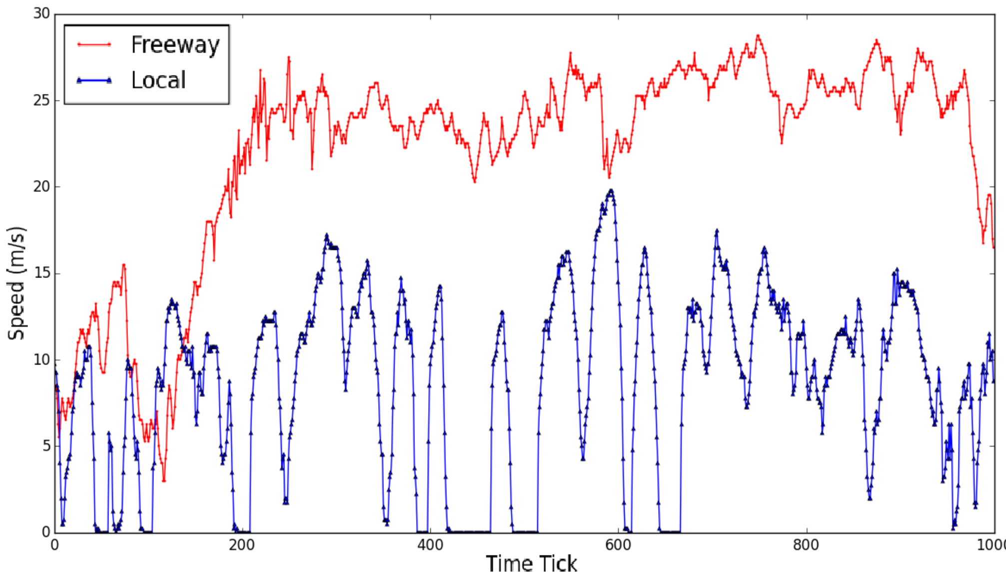


Figure 9.1: Driving Speed at Local and Freeway

9.1.2 What's the weather like along the trip?

We use the barometer data to get the information of ambient pressure, which is an important feature in identifying the different transportation modes. Besides, the air pressure also reveals some weather information such as raining, sunny, hot or cold. For example winter days have higher pressure than summer days, the barometer readings recorded in different weather situations show this pattern. Fig. 9.2 shows the barometer reading while riding a bike in summer and winter. This could be used as a reference for transportation survey record, as an indication of weather. Further research is needed in order to learn more weather information from the barometer readings.

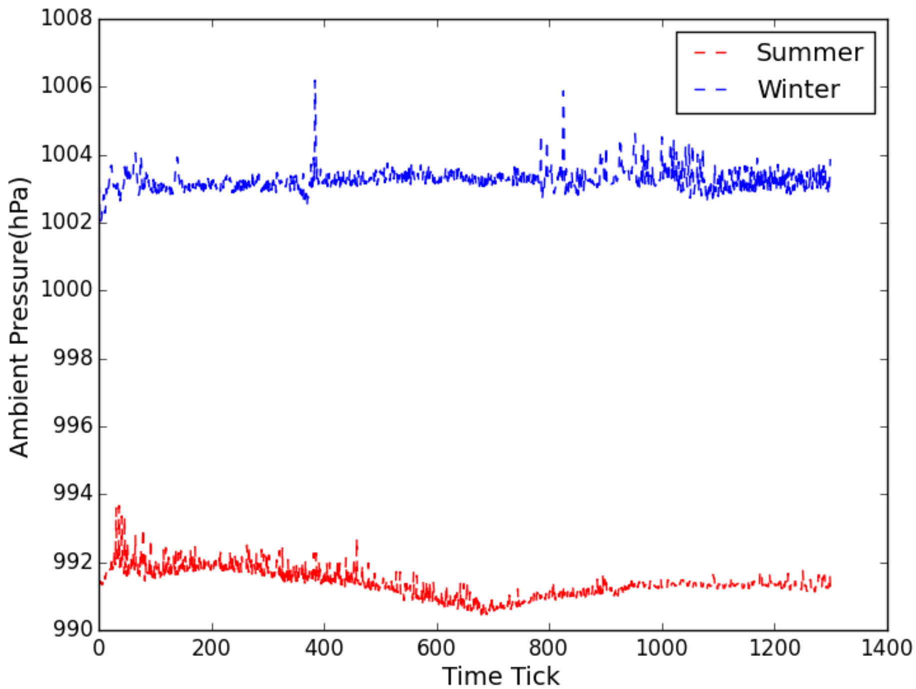


Figure 9.2: Pressure at Summer and Winter on Bike

9.1.3 Daytime or Night? Sunny or Cloudy?

The light sensor could sense the ambient illumination. A smartphone provides some default value for different environment conditions such as night without light, cloudy or sunny day [77]. This can also be a reference in transportation survey. However, one may argue that the sensed ambient light also heavily depends on where the user put the phone: inside a pocket or in hands, and the two ways could result in totally different readings even at the same place. It needs further work and other information to draw the conclusion. We will not expand the discussion here.

9.2 Contributions

In this thesis, we propose new algorithms for travel mode identification using smartphone sensors. The prototype system is built upon the latest Android and iOS platforms with multimodality sensors. It takes the sensors' data as the input and aims to identify six travel modes: *walking, jogging, bicycling, driving a car, riding a bus, taking a subway.* This thesis is devoted to three complementary research tasks. First (Task 1. Data Collection and Preprocessing), we develop an app that alows the collection of a multitude of sensor measurement data from smartphones, and present a comprehensive set of de-noising techniques. Second (Task 2. Feature Engineering), we design feature extraction methods that carefully balance between prediction accuracy, calculation time, and battery consumption. Third (Task 3. Learning Models), we develop new learning models that employ a hierarchical framework with dynamic sensor selection mechanisms, and adapt to each traveller's specific travel behavior as well as to newly collected data for a given traveller. Using carefully designed experiments, we validate our proposed methods and examine their effectiveness in addressing the aforementioned challenges. The results in comparison studies show that our methods are successful in tackling the problems, and that the performance of the travel mode detection system we propose in this thesis is promising. The design of the prototype system is operational, with optimizations that make it energy efficient, user-friendly, and noise resilient. This thesis provides useful inputs for the multi-modal traffic signal control, and could benefit other application domains, such as urban design, public health, emission detection, personal travel plan, etc. Besides, the methods used in this thesis are also applicable in many other smartphone sensor based machine learning applications such as indoor localization, user activity recognition, biometric information identification, etc.

9.3 Future Work

In this thesis we have focused on feasible solutions (dynamic sensor selection, fast response, model adaption) in using smartphone sensors for travel mode detection. Our future work of this thesis will keep the same principle in feasibility and it includes three main parts. First, in energy consumption the existing work have considered the dynamic sensor selection based on the group feature analysis. The next step of this work is on feature selection within each sensor's feature group to further decrease the dimensions of the feature space. Second, in user similarity computing in this thesis we give the non-sensor factors the same weight in calculation. In reality some factors are more important that others and there might be a dominating factor in certain travel modes. For example in some of our experiment study it shows that the seasonal factor is dominating in the mode of taking a bus. We will focus on how the different non-sensor factors affect the users' travel behavior at different scenarios in future research. Third, in this thesis we limit the information to only use the data collected by the 5 sensors. In future work we will introduce other augmented information such as map information, low frequency GPS data, travel speed, etc. to improve the classification accuracy.

Bibliography

[1] Apple, "Motion events," https://developer.apple.com/library/content/documentation/EventHandling/Conceptual/EventHandlingiPhoneOS/motion_event_basics/motion_event_basics.html, March 2016, (Accessed on 03/13/2017).

[2] S. Bamberg, I. Ajzen, and P. Schmidt, "Choice of travel mode in the theory of planned behavior: The roles of past behavior, habit, and reasoned action," *Basic and applied social psychology*, vol. 25, no. 3, pp. 175–187, 2003.

[3] I. E. Agency, apr 2004. [Online]. Available: http://www.iea.org/impagr/cip/archived_bulletins/issue_no23.htm

[4] Q. He, K. L. Head, and J. Ding, "Pamscod: Platoon-based arterial multi-modal signal control with online data," *Transportation Research Part C: Emerging Technologies*, vol. 20, no. 1, pp. 164–184, 2012.

[5] ——, "Multi-modal traffic signal control with priority, signal actuation and coordination," *Transportation Research Part C: Emerging Technologies*, vol. 46, pp. 65–82, 2014.

[6] N. Ding, Q. He, and C. Wu, "Performance measures of manual multimodal traffic signal control," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2438, pp. 55–63, 2014.

[7] N. Ding, Q. He, C. Wu, and J. Fetzer, "Modeling traffic control agency decision behavior for multimodal manual signal control under event occurrences," *to appear in IEEE Transactions on Intelligent Transportations System.*

[8] C. Rissel, N. Curac, M. Greenaway, and A. Bauman, "Physical activity associated with public transport use: a review and modelling of potential benefits," *International journal of environmental research and public health*, vol. 9, no. 7, pp. 2454–2478, 2012.

[9] Z. Sun and X. J. Ban, "Vehicle classification using gps data," *Transportation Research Part C: Emerging Technologies*, vol. 37, pp. 102–117, 2013.

[10] H. Gong, C. Chen, E. Bialostozky, and C. T. Lawson, "A gps/gis method for travel mode detection in new york city," *Computers, Environment and Urban Systems*, vol. 36, no. 2, pp. 131–139, 2012.

[11] X. Su, H. Tong, and P. Ji, "Activity recognition with smartphone sensors," *Tsinghua Science and Technology*, vol. 19, no. 3, pp. 235–249, 2014.

[12] ——, "Accelerometer-based activity recognition on smartphone," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 2014, pp. 2021–2023.

[13] H. X. Su and H. Caceres, "Travel mode identification with smartphones," *Sensors*, vol. 15, p. 16.

[14] X. Su, H. Caceres, H. Tong, and Q. He, "Fast online travel mode identification using smartphone sensors," in *Transportation Research Board Annual Meeting 2016*. TRB, 2016.

[15] ——, "Online travel mode identification using smartphones with battery saving considerations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2921–2934, 2016.

[16] Q. H. J. L. H. T. Xing Su, Yuan Yao, "Personalized travel mode detection with smartphone sensors," *Submitted to IJCAI*, 2017.

[17] "Sensors overview — android developers," http://developer.android.com/guide/topics/sensors/sensors_overview.html, (Visited on 09/02/2015).

[18] S. Papadimitriou and T. Eliassi-Rad, "Mining mobility data," in *Proceedings of the 24th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 2015, pp. 1541–1542.

[19] A. M. Ladd, K. E. Bekris, A. Rudys, L. E. Kavraki, and D. S. Wallach, "Robotics-based location sensing using wireless ethernet," *Wireless Networks*, vol. 11, no. 1-2, pp. 189–204, 2005.

[20] B. Ferris, D. Fox, and N. D. Lawrence, "Wifi-slam using gaussian process latent variable models." in *IJCAI*, vol. 7, no. 1, 2007, pp. 2480–2485.

[21] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor wifi graphslam," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1038–1043.

[22] B. Benjamin, G. Erinc, and S. Carpin, "Real-time wifi localization of heterogeneous robot teams using an online random forest," *Autonomous Robots*, vol. 39, no. 2, pp. 155–167, 2015.

[23] H. S. Maghdid, I. A. Lami, K. Z. Ghafoor, and J. Lloret, "Seamless outdoors-indoors localization solutions on smartphones: Implementation and challenges," *ACM Computing Surveys (CSUR)*, vol. 48, no. 4, p. 53, 2016.

[24] C. Wu, Z. Yang, and Y. Liu, "Smartphones based crowdsourcing for indoor localization," *Mobile Computing, IEEE Transactions on*, vol. 14, no. 2, pp. 444–457, 2015.

[25] C. Feng, W. S. A. Au, S. Valaee, and Z. Tan, "Received-signal-strength-based indoor positioning using compressive sensing," *Mobile Computing, IEEE Transactions on*, vol. 11, no. 12, pp. 1983–1993, 2012.

[26] K. Liu, X. Liu, and X. Li, "Guoguo: Enabling fine-grained indoor localization via smartphone," in *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*. ACM, 2013, pp. 235–248.

[27] M. Rossi, J. Seiter, O. Amft, S. Buchmeier, and G. Tröster, "Roomsense: an indoor positioning system for smartphones using active sound probing," in *Proceedings of the 4th Augmented Human International Conference*. ACM, 2013, pp. 89–95.

[28] H. Ye, T. Gu, X. Zhu, J. Xu, X. Tao, J. Lu, and N. Jin, "Ftrack: Infrastructure-free floor localization via mobile phone sensing," in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2–10.

[29] R. LiKamWa, Y. Liu, N. D. Lane, and L. Zhong, "Can your smartphone infer your mood," in *PhoneSense workshop*, 2011, pp. 1–5.

[30] T. Chakravarty, A. Ghose, C. Bhaumik, and A. Chowdhury, "Mobidrivescore - a system for mobile sensor based driving analysis: A risk assessment model for improving one's driving," in *Sensing Technology (ICST), 2013 Seventh International Conference on*. IEEE, 2013, pp. 338–344.

[31] A. M. Khan, Y.-K. Lee, S. Lee, and T.-S. Kim, "Human activity recognition via an accelerometer-enabled-smartphone using kernel discriminant analysis," in *Future Information Technology (FutureTech), 2010 5th International Conference on*. IEEE, 2010, pp. 1–6.

[32] R. Becker, R. Cáceres, K. Hanson, S. Isaacman, J. M. Loh, M. Martonosi, J. Rowland, S. Urbanek, A. Varshavsky, and C. Volinsky, "Human mobility characterization from cellular network data," *Communications of the ACM*, vol. 56, no. 1, pp. 74–82, 2013.

[33] G. De Nunzio, C. C. Wit, P. Moulin, and D. Di Domenico, "Eco-driving in urban traffic networks using traffic signals information," *International Journal of Robust and Nonlinear Control*, 2015.

[34] V. Manzoni, D. Maniloff, K. Kloeckl, and C. Ratti, "Transportation mode identification and real-time co2 emission estimation using smartphones," *SENSEable City Lab, Massachusetts Institute of Technology, nd*, 2010.

[35] A. Abbasi, T. H. Rashidi, M. Maghrebi, and S. T. Waller, "Utilising location based social media in travel survey methods: bringing twitter data into the play," in *Proceedings of the 8th ACM SIGSPATIAL International Workshop on Location-Based Social Networks*. ACM, 2015, p. 1.

[36] P. A. Grabowicz, J. J. Ramasco, B. Gonçalves, and V. M. Eguíluz, "Entangling mobility and interactions in social media," *PLoS One*, vol. 9, no. 3, p. e92196, 2014.

[37] J. L. Toole, C. Herrera-Yaqüe, C. M. Schneider, and M. C. González, "Coupling human mobility and social ties," *Journal of The Royal Society Interface*, vol. 12, no. 105, p. 20141128, 2015.

[38] A. Sadilek, H. Kautz, and J. P. Bigham, "Finding your friends and following them to where you are," in *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 2012, pp. 723–732.

[39] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 3, p. 38, 2014.

[40] Y. Zheng, Y. Liu, J. Yuan, and X. Xie, "Urban computing with taxicabs," in *Proceedings of the 13th international conference on Ubiquitous computing*. ACM, 2011, pp. 89–98.

[41] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and pois," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 186–194.

[42] C. Ratti, S. Sobolevsky, F. Calabrese, C. Andris, J. Reades, M. Martino, R. Claxton, and S. H. Strogatz, "Redrawing the map of great britain from a network of human interactions," *PloS one*, vol. 5, no. 12, p. e14248, 2010.

[43] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang, "T-drive: driving directions based on taxi trajectories," in *Proceedings of the 18th SIGSPATIAL International conference on advances in geographic information systems*. ACM, 2010, pp. 99–108.

[44] S. Shaheen, S. Guzman, and H. Zhang, "Bikesharing in europe, the americas, and asia: past, present, and future," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2143, pp. 159–167, 2010.

[45] N. Lathia and L. Capra, "Mining mobility data to minimise travellers' spending on public transport," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 1181–1189.

[46] M. A. Habib, M. S. Mohktar, S. B. Kamaruzzaman, K. S. Lim, T. M. Pin, and F. Ibrahim, "Smartphone-based solutions for fall detection and prevention: challenges and open issues," *Sensors*, vol. 14, no. 4, pp. 7181–7208, 2014.

[47] 2016. [Online]. Available: https://mysleepbot.com/

[48] 2016. [Online]. Available: http://stepzapp.com/

[49] L. T. Le, T. Eliassi-Rad, F. Provost, and L. Moores, “Hyperlocal: inferring location of ip addresses in real-time bid requests for mobile ads,” in *Proceedings of the 6th ACM SIGSPATIAL International Workshop on Location-Based Social Networks*. ACM, 2013, pp. 24–33.

[50] F. J. Provost, T. Eliassi-Rad, and L. S. Moores, “Methods, systems, and media for determining location information from real-time bid requests,” Apr. 21 2015, uS Patent 9,014,717.

[51] M. Raento, A. Oulasvirta, R. Petit, and H. Toivonen, “Contextphone: A prototyping platform for context-aware mobile applications,” *Pervasive Computing, IEEE*, vol. 4, no. 2, pp. 51–59, 2005.

[52] W. Enck, P. Gilbert, S. Han, V. Tendulkar, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, “Taintdroid: an information-flow tracking system for real-time privacy monitoring on smartphones,” *ACM Transactions on Computer Systems (TOCS)*, vol. 32, no. 2, p. 5, 2014.

[53] C. Chen, H. Gong, C. Lawson, and E. Bialostozky, “Evaluating the feasibility of a passive travel survey collection in a complex urban environment: Lessons learned from the new york city case study,” *Transportation Research Part A: Policy and Practice*, vol. 44, no. 10, pp. 830–840, 2010.

[54] J. Chon and H. Cha, “Lifemap: A smartphone-based context provider for location-based services,” *IEEE Pervasive Computing*, no. 2, pp. 58–67, 2011.

[55] H. Martín, A. M. Bernardos, J. Iglesias, and J. R. Casar, “Activity logging using lightweight classification techniques in mobile devices,” *Personal and ubiquitous computing*, vol. 17, no. 4, pp. 675–695, 2013.

[56] Z. Xu, K. Bai, and S. Zhu, “Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors,” in *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 2012, pp. 113–124.

[57] E.-H. Chung and A. Shalaby, “A trip reconstruction tool for gps-based personal travel surveys,” *Transportation Planning and Technology*, vol. 28, no. 5, pp. 381–401, 2005.

[58] G. Draijer, N. Kalfs, and J. Perdok, “Global positioning system as data collection method for travel research,” *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1719, no. 1, pp. 147–153, 2000.

[59] P. Stopher, E. Clifford, J. Zhang, and C. FitzGerald, *Deducing mode and purpose from GPS data*. Institute of Transport and Logistics Studies, 2008.

[60] W. Bohte and K. 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*, vol. 17, no. 3, pp. 285–297, 2009.

[61] L. Zhang, S. Dalyot, D. Eggert, and M. Sester, "Multi-stage approach to travel-mode segmentation and classification of gps traces," in *ISPRS Workshop on Geospatial Data Infrastructure: from data acquisition and updating to smarter services*, 2011.

[62] A. Bolbol, T. Cheng, I. Tsapakis, and J. Haworth, "Inferring hybrid transportation modes from sparse gps data using a moving window svm classification," *Computers, Environment and Urban Systems*, vol. 36, no. 6, pp. 526–537, 2012.

[63] T. K. Rasmussen, J. B. Ingvardson, K. Halldórsdóttir, and O. A. Nielsen, "Using wearable gps devices in travel surveys: A case study in the greater copenhagen area," in *Proceedings from the Annual Transport Conference at Aalborg University) ISSN*, 2013, pp. 1603–9696.

[64] C. Xu, M. Ji, W. Chen, and Z. Zhang, "Identifying travel mode from gps trajectories through fuzzy pattern recognition," in *Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on*, vol. 2. IEEE, 2010, pp. 889–893.

[65] M. Li, J. Dai, S. Sahu, and M. Naphade, "Trip analyzer through smartphone apps," in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2011, pp. 537–540.

[66] P. Nitsche, P. Widhalm, S. Breuss, and P. Maurer, "A strategy on how to utilize smartphones for automatically reconstructing trips in travel surveys," *Procedia-Social and Behavioral Sciences*, vol. 48, pp. 1033–1046, 2012.

[67] M. Frendberg, "Determining transportation mode through cellphone sensor fusion," Ph.D. dissertation, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, 2011.

[68] S. Garg and P. Singh, "A novel approach for vehicle specific road/traffic congestion," 2014.

[69] A. Jahangiri, H. Rakha *et al.*, "Applying machine learning techniques to transportation mode recognition using mobile phone sensor data."

[70] M. Shoaib, S. Bosch, O. D. Incel, H. Scholten, and P. J. Havinga, "A survey of online activity recognition using mobile phones," *Sensors*, vol. 15, no. 1, pp. 2059–2085, 2015.

[71] K. Muralidharan, A. J. Khan, A. Misra, R. K. Balan, and S. Agarwal, "Barometric phone sensors: more hype than hope!" in *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*. ACM, 2014, p. 12.

[72] Android, "Sensor — android developers," https://developer.android.com/reference/android/hardware/Sensor.html#TYPE_LIGHT, March 2016, (Accessed on 03/13/2017).

[73] Apple, "Iokit.framework - iphone development wiki," http://iphonedevwiki.net/index.php/IOKit.framework, March 2017, (Accessed on 03/13/2017).

[74] “brightness - uiscreen — apple developer documentation,” https://developer.apple.com/reference/uikit/uiscreen/1617830-brightness, (Accessed on 03/13/2017).

[75] Apple, “magneticfield - cmmagnetometerdata — apple developer documentation,” https://developer.apple.com/reference/coremotion/cmmagnetometerdata/1616084-magneticfield, March 2016, (Accessed on 03/13/2017).

[76] Android, “Sensor — android developers,” https://developer.android.com/reference/android/hardware/Sensor.html#TYPE_MAGNETIC_FIELD, March 2016, (Accessed on 03/13/2017).

[77] A. D. Reference, apr 2014. [Online]. Available: http://developer.android.com/reference/android/hardware/SensorEvent.html#values

[78] N. G. D. Center, apr 2014. [Online]. Available: http://www.ngdc.noaa.gov/geomag-web/#igrfwmm

[79] G. Physikalisch-Technische Bundesanstalt, Braunschweig, apr 2014. [Online]. Available: http://www.ptb.de/cartoweb3/SISproject.php

[80] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, “A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer,” *Information Technology in Biomedicine, IEEE Transactions on*, vol. 14, no. 5, pp. 1166–1172, 2010.

[81] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, “Activity recognition from accelerometer data,” in *Proceedings of the national conference on artificial intelligence*, vol. 20, no. 3. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005, p. 1541.

[82] L. Bao and S. S. Intille, “Activity recognition from user-annotated acceleration data,” in *Pervasive computing*. Springer, 2004, pp. 1–17.

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

[84] M. Yuan and Y. Lin, “Model selection and estimation in regression with grouped variables,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[85] Y. Yang and H. Zou, “gglasso: group lasso penalized learning using a unified bmd algorithm,” 2013.

[86] Google, “Coding for battery life,” https://dl.google.com/io/2009/pres/W_0300_CodingforLife-BatteryLifeThatIs.pdf, 2009, (Accessed on 04/25/2017).

[87] K. Katevas, H. Haddadi, and L. Tokarchuk, “Sensingkit: Evaluating the sensor power consumption in ios devices,” in *Intelligent Environments (IE), 2016 12th International Conference on*. IEEE, 2016, pp. 222–225.

[88] K. Sankaran, M. Zhu, X. F. Guo, A. L. Ananda, M. C. Chan, and L.-S. Peh, "Using mobile phone barometer for low-power transportation context detection," in *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. ACM, 2014, pp. 191–205.

[89] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Scholkopf, "Correcting sample selection bias by unlabeled data," in *NIPS*, 2006, pp. 601–608.

[90] S. J. Pan, Q. Yang, and W. Fan, "Tutorial description — ijcai 2013," http://ijcai13.org/program/tutorial/TD2, Aug. 2013, (Accessed on 02/10/2017).

[91] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[92] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," DTIC Document, Tech. Rep., 1985.

[93] D. Saad, "Online algorithms and stochastic approximations," *Online Learning*.

[94] L. Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 421–436.

[95] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for svm," *Mathematical programming*, vol. 127, no. 1, pp. 3–30, 2011.

[96] H. T. Xing Su, Hernan Caceres and Q. He, "Travel mode identification with smartphones," *Transporation Research Board Annual Meeting*, 1 2015.