MOBILE RECOMMENDER SYSTEMS WITH BUSINESS EFFECTIVE STRATEGIES

by

MENG QU

A dissertation submitted to the

Graduate School-Newark

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Management

written under the direction of

Dr. Hui Xiong

and approved by

_____

_____

_____

_____

Newark, New Jersey

October 2017

© Copyright 2017

Meng Qu

All Rights Reserved

ABSTRACT OF THE DISSERTATION

Mobile Recommender Systems With Business Effective Strategies

By MENG QU

Dissertation Director: Dr. Hui Xiong

Recommender systems aim to provide personalized suggestions to users based on their backgrounds and interests. The suggestions can be made in a variety of application areas, such as movies, music, news, books, and products. Recommender systems are primarily developed for individuals who lack of the sufficient personal experiences or competence to evaluate an overwhelming number of alternatives. Therefore, recommender systems are usually personalized, and face substantial challenges in coping with information overloaded environments.

This dissertation focuses on building mobile recommender systems with business effective strategies. Due to the explosive growth of GPS trajectory and urban geographical data, mobile recommender systems have been extensively utilized to offer various types of recommendation services. Indeed, recent efforts have been made to develop mobile recommender systems for taxi drivers based on the analysis of taxi GPS traces. In general, there are three ways to provide such recommendation services. The first is to focus on choosing the fastest driving route from the current location to the destination. The second is to provide a sequence of pick-up points for taxi drivers. The goal of this approach is to allow the taxi driver to find a customer within the shortest driving distance. Finally, the third method attempts to strike a balance between the needs of taxi drivers and passengers. However, in the real world,

the income of a taxi driver is strongly related to effective driving hours than to the actual driving distance.

To this end, in this dissertation, we aim to address the challenges involved in providing business effective recommendations in mobile environments from both theoretical and practical perspectives. Specifically, we first develop a cost-effective mobile recommender system that is capable of recommending an entire driving route for taxi drivers and helping them to find a passenger with the highest possible net profit. Experiments based on real-world data demonstrate the efficiency and effectiveness of our systems. Moreover, we develop a virtual station waiting strategy which suggests the right waiting time and locations for taxi drivers in a business effective way. Then, we design an enhanced recommender system by combining the virtual waiting and driving route search strategies. In this enhanced system, we provide a joint learning framework to evaluate the potential profits derived from different strategies and find the optimal solution. Also, we exploit a recursive algorithm to efficiently generate optimal driving route recommendations. Meanwhile, we introduce Top-K route recommendations and a dynamic maximum Net Profit strategy to provide better load balance for recommendations happened at the same location. Finally, the experimental results clearly validate the effectiveness of the enhanced recommender system for taxi drivers, and show that our recommender system can help to substantially increase the income of inexperienced taxi drivers.

ACKNOWLEDGEMENTS

I would like to express my great appreciation to all the people who provided me tremendous support and help during my Ph.D. study.

First, my gratitude goes to my advisor, Prof. Hui Xiong, for his continuous support, guidance and encouragement, which are necessary to survive and thrive the graduate school and the beyond. I am very thankful to him for his insightful comments and advice; for teaching me how to identify key problems with impact, present and evaluate the ideas. I have been learning from him as a student, also as a human being.

I also sincerely grateful to those who served in my committees: Prof. Jian Yang, Prof. Papadimitriou and Prof. Weili Wu . I deeply appreciate their time and effort in reading my depth report, dissertation, and attending my dissertation defense. All of them not only provide inspiring and constructive suggestions on my work and this thesis, but also offer numerous support and help in my job hunting process. Prof. Jian Yang has been a great professor to me over the past six years. His experience and vision in optimization and operation research has inspired me a lot to solve the challenging problems in my research. Prof. Spiros Papadimitriou is a great mentor and also a good friend. He is not only provided lots of useful feedback and suggestions during my Ph.D study but also offered career and personal guidance. Prof. Weili Wu

has also provided many useful feedback and discussions for my research. I thank her particularly for her time rehearsing me for this dissertation.

Special thanks are due to Prof. Micheal Katehakis, Prof. Periklis Papakonstantinou, Prof. Weiwei Chen and Prof. Xiaodong Lin at Rutgers University, Prof. Wenjun Zhou at University of Tennessee, Prof. Yong Ge at Uniersity of Arizona, Prof. Guannan Liu at Beihang University, Prof. Keli Xiao at SUNY-Stony Brook, Prof Yanjie Fu at Missouri University of Science and Technology and Prof. Chuanren Liu at Drexel University for their help with my job search and career development. Thanks are also due to Dr.Hengshu Zhu, Dr. Yanchi Liu, Dr. Liyang Tang, and Dr.Qi Liu. It was a great pleasure working with all of them. The research supporting this dissertation also greatly benefited from open discussions with the faculty and student members in the Data Mining group of Rutgers Business School: Junming Liu, Jingyuan Yang, Zijun Yao, Konstantin Patev, Farid Razzak, Hao Zhong, Zhongmou Li, Bin Liu, Can Chen, Qingxin Meng, Mingfei Teng, Shuying Liu and Huang Xu. It is an honor to be affiliated with such a great group.

I would also like to acknowledge the Department of Management Science and Information Systems (MSIS) for supplying me with the best equipment and working environment that helped me to accomplish much of this work. I want to particularly thank Mrs. Luz Kosar and Mr. Goncalo Filipe for all of their assistance.

Finally, I would like to thank my husband and my son for their love, support and understanding. Without their encouragement and help, I can never live such a wonderful life. Thank you for making my life so special and full of joy.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Preliminaries

Our world has never been changed so much by the evolution of technology. Around the globe, there are hundreds of millions of people taking photos, creating videos, and sending texts every single day, and the deluge of data is growing rapidly. About 2.5 quintillion bytes of data are created every day, which is equivalent to 530,000,000 million songs, 250,000 libraries of congress, and 90 years of HD videos. Indeed, 90% of data in the world today has been created in the last two years alone. These data come from everywhere- sensors used to obtain public transportation information, pictures posted on Facebook, and location traces on Twitter. There were 4.4 zettabytes of data in the world in 2013, and this is expected increase tenfold, to 44 zettabytes, by 2020. There is no doubt that the increasing speed at which data is being created is due to the increasing popularity of Internet usage via various mobile devices as well as the increasing number of individuals who wants to join the digital world.

Fortunately, with recent developments in information technology, it is now easier to collect, retain, and analyze enormous amounts of data. More and more organizations and companies have begun to realize the importance of data and thus to store retail transaction records, stock price histories, credit card information, search logs,

mobile service trajectories, on-line browsing history data, and so on. We use the term "big data" to describe those extremely large data sets that can be analyzed computationally to reveal patterns, trends, and associations, especially those data related to human behaviors and interactions. Those data sets are so overwhelming and complex that commonly used software can not capture, store, analyze, visualize, and process them within a tolerable time period.

Therefore, data mining aims to discover hidden but often useful information to help in decision making. Data mining involves artificial intelligence, machine learning, statistics, and database systems (Chakrabarti et al., 2006). The main goal of data mining is to investigate data and extract understandable information from it for further uses. For example, once we have a good understandings of the users who are looking at a website, we could display the most relevant advertisements on these pages and encourage the users to click on those links. In general, data mining is the process of the "knowledge discovery in databases" process (Fayyad, Piatetsky-Shapiro, & Smyth, 1996).

1.2 Traditional Recommender Systems

Based on the information we collect and analyze using data mining techniques, we were able to understand our users more clearly and develop recommender systems. In traditional recommender systems, there are two classes of entities *users* and *items*. A recommender system is an information system which is capable of providing the "rating" or "preference" that a user would assign to an item. If we use $\mathbf{C}$ to represent the set of all users and let S be the set of all possible recommendable items, we can

then use u as an utility function measuring the usefulness of item s to user c. For each user $c \in \mathbf{C}$, we want to choose the item $s \in \mathbf{S}$ that maximizes u. The utility is usually represented by a rating, but it could also be any function. The relationship can be represented by the following equation:

$$\forall c \in C, s'_c = argmax(u(c, s)), s \in S \tag{1.1}$$

For example, Amazon is one of the most famous companies to use recommender systems to provide web pages with advertisements for products that are geared toward specific users. Other companies, such as Netflix, Facebook and Apple, also use similar recommender systems to suggest movies, music, and other products to their users. The recommended items are retrieved according to the rules set by computer algorithms. Depending on the requirements, those systems usually suggest the top K items relevant to users. Recently, recommender systems have been widely adopted in industry, and helping to increase sales, include a diversity of items, and increase user satisfaction and loyalty

1.3 Mobile Recommender Systems

While traditional recommender systems are widely used in several fields, such as, health care, market basket analysis, education, manufacturing, engineering, customer relationship management, fraud detection, customer segmentation, and criminal investigation, my research aims to address the unique challenges involved in providing recommendations in mobile environments from both theoretical and practical perspectives and design recommender systems that can provide mobile users access to

personalized recommendations anytime and anywhere.

With the rapid development in mobile technology, mobile recommender systems have been a growing area of research. There are 4.6 billion mobile-phone subscriptions worldwide, and between one and two billion people have access to the Internet through other mobile devices. Vavious methods of collecting all useful mobility data have been developed. In general terms, there are two types of mobile data, human mobility data and urban geography data.

- Human mobility data represents people's movement trajectories, which can be phone traces, trajectories of driving routes, or a sequence of posts including geographical information, such as geo-tweets, geo-tagged photos,or check-ins. These data can be indoor traces or outdoor traces.

- Urban geography data is data sets including geographic characteristics of a city. Those data may include but not limited to city road networks data, public transportations data, places of interest (POIs) data and regional functions data.

Because of the abundant availability of these data, the advanced development of mobile devices, and the urgent demand for mobile applications, such as driving route recommendation, mobile tourist guides and personal location-based shopping, mobile recommender systems have become a promising field. Given the availability of the GPS, Wi-Fi, and mobile phone data, we can collect Human Mobility and Urban Geography data and meet the urgent demands for better business applications.

Existing researches have already developed many applications related to mobile recommendations in both industry and academics. For example, a mobile restaurant

recommender system can analyze customers' previous dining records and the current time of a day; and then make personal recommendations for a particular user at a certain location.

1.4 Research Motivation

Traditional recommendation systems have existed for many years (Hofmann, 1999; Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994). For instance, Netflix had a famous $1,000,000 competition from 2006 to 2009 attracted over 45,000 contestants from 180 countries. In both industry and academics, many projects have already developed new approaches to recommender systems in the past. Most traditional recommender systems take either of two basic approaches: collaborative filtering or content-based filtering. Collaborative filtering is based on a model of prior user behavior. This can involve only a single user's behavior or the behaviors of many users who have similar traits.

For instance, suppose there is a website that recommends movies. By using the information gathered from the many users who subscribe to this website to watch movies, we can group those users based on their movie preferences. Then, we are able to identify the most popular movies in various categories, such as, romance, adventure, mystery, comedy and adventure, and recommend the most popular movies in a specific category to the users who have not yet watched them. In table 1.1, a set of movies forms the rows, and the columns define the users.

By clustering the users based on their movie preferences, we can see two clusters of two users each-Mary and Ada are in Cluster 1 and Bob and Michael are in Cluster

Table 1.1. An Example of Item-User Movie Watching Matrix

	Mary	*Bob*	*Ada*	*Micheal*
Romance	13	0	11	1
Adventure	0	8	0	13
Mystery	0	6	0	24
Comedy	18	0	8	0

[a] Note: 0 indicates no movie has been watched in this category.

2. The users in Cluster 1 prefer romance and comedy movies, while the users in Cluster 2 prefer adventure and mystery movies. Then, whenever we discover that there is a high rated movie in one of these categories, we can make recommendations to other users in the appropriate cluster. This method requires minimal knowledge engineering efforts. However, it requires a large number of reliable user feedback data to begin with.

Content-based filtering chooses a recommendation based on potential items to recommend rather than other users' opinions. The content to be filtered may be the explicit attributes or characteristics of the items. These recommender systems attempt to recommend items that are similar to those a given user has rated highly in the past. For example, we may use the user's online shopping history, such as which shopping websites the user has logged on to and which products he or she has purchased. If a user buys baby products regularly, content-based filtering can use this history to identify and recommend other popular baby products. Instead of relying on the behaviors of other users in the system, this approach solely relies on

the content that a single user can access. Therefore, the content-based approach does not require data on other users and can make recommendations to users with unique tastes. However, it requires the content to be encoded so as to identify products' meaningful features, and the users' tastes must also be represented as a learnable function of these content features.

In addition to collaborative filtering and content-based filtering, there is also a hybrid approach that combines these two methods. Having the advantages of both methods, this hybrid system can provide more accurate recommendations. For example, Netflix makes recommendations by comparing the watching and the searching habits of a group of similar users and also by offering movies that share characteristics with the films that a given user has rated highly.

However, mobile recommender systems are different than traditional recommender systems because of their unique location-aware capabilities. The development of personalized mobile recommender systems is also much more challenging than developing traditional recommender systems. Indeed, the challenge of developing a mobile recommendation system is inherit in mobile data. These mobile data are usually spatial, with unclear roles for context-aware information and lack of user rating information.

Mobile data are typically heterogeneous and exhibit a spatial and temporal autocorrelation, which means that nearby things have more impact than distant things and their noise levels are also high. For validation, unlike traditional recommender systems such as movie recommendation systems, which typically used previously provided ratings for the movie, we do not have this kind of rating data available in the mobile domain. Therefore, when we develop this kind of recommender systems, the

validation may be a problem because the rating data is not available. In general, if we already developed a recommender system for movie recommendation, the technique is suitable for the movie recommendation is most likely not suitable in the mobile recommendation domain. Because of this general problem, we can not easily adapt the techniques developed in traditional application domain to the mobile environment.

Moreover, mobile recommendation has cost constraints in terms of both time and price. For example, when we recommend a travel package to potential customers, we must consider the length of time they want to stay on a vacation and how much they can afford. If the customers only have three days of vacation, it is senseless to recommend a seven days vacation with a good price and a great itinerary. Mobile recommendation systems also facing a life cycle problem. They are unlike a traditional movie recommender, which likely has more long term value. In a mobile scenario, if you recommend a driving route today, the same recommendation might not work after three month, due to road constructions or changes in driving patterns. Moreover, regarding travel recommendations, travel patterns during the summer and winter are also quite different due to changing weather conditions. Thus, the life cycle for such travel recommender systems may be very short.

Based on the above prospective on mobile data, it will be very difficult for us to apply the traditional techniques to a mobile recommender. That is why there is a need for us to develop mobile recommender systems that use business effective strategies.

1.5 Related Work

In the literature, many efforts have been devoted to building personalized recommender systems, such as content-based recommendation (Mooney & Roy, 1999), collaborative filtering based recommendation (Su & Khoshgoftaar, 2009) as well as the hybrid recommendation (Pazzani, 1999). Furthermore, some recommender systems (Adomavicius & Tuzhilin, 2005) also aim to address the information overloaded problem by identifying user interests and providing personalized suggestions. However, those traditional recommender systems (Bell & Koren, 2007; Deshpande & Karypis, 2004; Koren, 2008) are more focused on recommendation of online information, such as online movie, article, book or webpage. In most of the cases, the research data are based on user ratings, which are very different from the data collected in mobile environment.

Developing personalized recommender systems in mobile and pervasive environments is much more challenging than developing recommender systems in traditional domains due to the complexity of spatial data and intrinsic spatio-temporal relationships, the unclear roles of context-aware information (Zhu et al., 2012), and the increasing availability of environment sensing capabilities. Those unique challenges are actually inherit in the mobile data we have. Indeed, recommender systems in the mobile environments have been studied before (Abowd, Atkeson, & al, 1997; Averjanova, Ricci, & Nguyen, 2008; Cena, Console, & al, 2006; Cheverst, Davies, & al, 2000; Miller, Albert, & al, 2003; Tveit, 2001; Heijden, Kotsis, & Kronsteiner, 2005). For instance, the works in (Abowd et al., 1997; Cena et al., 2006) target at

the development of mobile tourist guides. Zhu *et al.* proposed a uniform framework of personalized context-aware recommendation for mobile users. The framework can discover users' personal context-aware preferences by mining the context logs of many mobile users. Heijden *et al.* have discussed some technological opportunities associated with mobile recommendation systems (Heijden et al., 2005). Averjanova *et al.* have developed a map-based mobile recommender system that can provide users with some personalized recommendations (Averjanova et al., 2008). However, the above prior works are mostly based on user ratings or interactions, and corresponding recommender systems are developed for smart mobile devices, such as mobile phones. Indeed, the problem of building mobile recommender systems for taxi business remains pretty much open.

Recently, the abundant availability of Taxi GPS traces has enabled new ways of doing taxi business. Plenty efforts have been made on developing mobile recommender systems for taxi drivers by using Taxi GPS traces. These systems can extract energy-efficient transportation patterns from historical location traces and recommending potential pick-up points for taxi drivers. For example, Ge *et al.* (Ge, Xiong, Tuzhilin, et al., 2010) defined a novel problem of mobile sequential recommendation by leveraging the historical GPS data from taxi drivers. By solving this problem, a novel energy-efficient mobile recommender system has been developed. This system can provide an optimal sequence of pick-up points for taxi drivers. Also, Powell *et al.* (Powell, Huang, Bastani, & Ji, 2011) proposed a grid-based method to suggest the profit locations for taxi drivers by constructing a spatio-temporal profitability map. In addition, Yuan *et al.* (Yuan, Zheng, Xie, & Sun, 2013; Yuan et al.,

2010; Yuan, Zheng, Zhang, Xie, & Sun, 2011) have carried out a series of studies on mobile intelligence by leveraging taxi trajectories, such as pick-up points detection based on probabilistic models, and location recommendation for both the taxi drivers and customers. Siyuan liu *et al.* (S. Liu, Wang, Liu, & Krishnan, 2015) focus on recommending series of pick-up locations to taxi drivers by proposing a frame work including a series of models to study how a taxi driver gathers and learns information in an uncertain environment through the use of their social network. Moreover, a comprehensive study is carried out by Tong Xu *et al* (T. Xu et al., 2016) to reveal how the social propagation affects for better prediction of cab drivers' future behaviors. Daqing Zhang *et al.* (D. Zhang et al., 2015) predicted and improved the revenue of taxi drivers through analyzing three perspectives of their service strategies. Jianbin Huang *et al.* (Huang et al., 2015) challenged the high computational complexity with mobile sequential recommendation by proposing a dynamic programing based method with off line processing stage and on line searching stage. Huigui Rong *et al* (Rong, Zhou, Yang, Shafiq, & Liu, 2016) investigated how to increase the taxi drivers income in each one-hour time slot by modeling the passenger seeking process as a Markov Decision Process. Yong Ge *et al* also developed a taxi driving fraud detection system, which is able to systematically investigate taxi driving fraud (Ge, Xiong, Liu, & Zhou, 2011). They further worked on taxi business intelligence system and explored the massive taxi location traces from different business perspectives with various data mining functions (Ge, Liu, Xiong, & Chen, 2011). Shiyou Qian *et al* (Qian, Cao, Mouël, Sahel, & Li, 2015) proposed a sharing considered route assignment mechanism and aimed to provide recommendation fairness for a group of

competing taxi drivers, without sacrificing driving efficiency. Fei Miao *et al* presented a receding horizon control framework to dispatch taxis and match spatiotemporal ratio between demand and supply for taxi service quality with minimum current and anticipated future taxi idle driving distance (Miao et al., 2016).

Different from the above studies, in this dissertation, we propose to develop a novel cost effective recommender system (Qu, Zhu, Liu, Liu, & Xiong, 2014). This recommender system can provide an entire driving route to taxi drivers and the drivers are able to find a customer with the largest potential profit by following this route. A summary of this work has been published in the Twentieth ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD 2014). Then we developed this recommender system with two different strategies and also propose two methods to solve the load unbalance problem. A summary of this work has been submitted to IEEE Transactions on Knowledge and Data Engineering (TKDE). A large amount of researches based on the works in this dissertation have been observed. Such as, Chenyi Zhang *el al*'s work on personalized trip recommendation by incorporating POIs availability, uncertain traveling time and diversity of the POIs constraints (C. Zhang, Liang, Wang, & Sun, 2015; C. Zhang, Liang, & Wang, 2016). Dongxu Shao *el al*'s research on estimating taxi demand supply level by using taxi trajectory data stream (Shao, Wu, Xiang, & Lu, 2015). Guoyang Qin *el al* discover the factors that may affect the income level of taxi drivers and compute the elasticity for the significant factors (Qin et al., 2017). Moreover, Zeyang Ye *el al* investigate the performance of simulated annealing in mobile recommendation problems with a focus on identifying the optimal cooling schedule method (Ye, Xiao, & Deng, 2015).

1.6 Research Contributions

In this dissertation, we investigate the unique characteristics of mobile recommender systems and demonstrate how to develop mobile recommender systems that use effective business strategies. The proposed research has the following major contributions:

- Developed a cost-effective recommender system for taxi drivers. This recommender system can maximize taxi drivers' profits when drivers follow the recommended routes to find passengers. Specifically, instead of recommending a sequence of pick-up points, our recommender system is capable of providing the complete driving route with the largest potential profit.

- Proposed a net profit objective function for evaluating the potential profits of driving routes. This objective function is capable of evaluating the potential net profit of a candidate route based on our road network.

- Developed a graph representation of road networks that contains all possible routes by mining historical taxi GPS traces and generated an optimal driving route to recommend.

- Presented a novel recursion strategy based on the special form of the net profit function for efficiently searching optimal candidate routes. This recursion strategy decreased the computational cost of the graph based approach.

- Developed an enhancing recommender system with business effective strategies for taxi drivers. The design goal is to maximize the taxi drivers' profits when

they adopt the proposed route searching strategy or virtual station waiting strategy.

- Discovered four basic properties of virtual stations, such as, high customer demand, high pick-up earnings, stop & waiting and dynamic properties. Investigated those properties and found potential virtual stations and their active regions during different time slots.

- Proposed a virtual station waiting strategy, which recommends that taxi drivers drive directly to a waiting spot and wait for the next customer in line. Compared the potential net profit from this waiting strategy with the profit from previous route searching strategy and recommended the strategy with highest net profit to the taxi drivers.

- Provided two strategies to create a better load balance for recommendations that occur at the same location, including Top-K route recommendations and dynamic Maximum Net Profit strategy.

- Conducted extensive experiments using real-world data sets collected from Beijing and the San Francisco Bay area. The experimental results clearly validated the effectiveness of the proposed recommender systems.

1.7 Overview

Chapter 2 introduces A Cost-Effective Recommender System for Taxi Drivers. Instead of recommending a sequence of pick-up points and letting the driver decide how to get to those points, this recommender system is capable of providing an entire

driving route, and the drivers are able to find customers that will result in the largest potential profit by following the recommendations. This chapter also addresses the computational challenge embedded in making mobile recommendations using GPS trajectory data. A novel recursion strategy that are capable to efficiently search the optimal drive route and recommend it to users is introduced.

Chapter 3 presents an Enhancing Recommender System for Taxi Drivers with Business Effective Strategies. Because taxi drivers do not always want to drive around to find their next passenger, they may pick up a passenger more quickly by waiting in line at a hot pick-up spot. We call these hot spots virtual stations and take those virtual stations into consideration in developing recommendation strategies that are intended to maximize taxi drivers' profits. This chapter also introduces two strategies intended to create a better load balance for recommendations occurring at the same location.

CHAPTER 2

A COST-EFFECTIVE RECOMMENDER SYSTEM FOR TAXI DRIVERS

The GPS technology and new forms of urban geography have changed the paradigm for mobile services. As such, the abundant availability of GPS traces has enabled new ways of doing taxi business. Indeed, recent efforts have been made on developing mobile recommender systems for taxi drivers using Taxi GPS traces. These systems can recommend a sequence of pick-up points for the purpose of maximizing the probability of identifying a customer with the shortest driving distance. However, in the real world, the income of taxi drivers is strongly correlated with the effective driving hours. In other words, it is more critical for taxi drivers to know the actual driving routes to minimize the driving time before finding a customer. To this end, in this chapter, we propose to develop a cost-effective recommender system for taxi drivers. The design goal is to maximize their profits when following the recommended routes for finding passengers. Specifically, we first design a net profit objective function for evaluating the potential profits of the driving routes. Then, we develop a graph representation of road networks by mining the historical taxi GPS traces and provide a Brute-Force strategy to generate optimal driving route for recommendation. However, a critical challenge along this line is the high computational cost of the graph based approach. Therefore, we develop a novel recursion strategy based on the special form of the net profit function for searching optimal candidate routes efficiently. Particularly, instead

of recommending a sequence of pick-up points and letting the driver decide how to get to those points, our recommender system is capable of providing an entire driving route, and the drivers are able to find a customer for the largest potential profit by following the recommendations. This makes our recommender system more practical and profitable than other existing recommender systems. Finally, we carry out extensive experiments on a real-world data set collected from the San Francisco Bay area and the experimental results clearly validate the effectiveness of the proposed recommender system.

2.1 Introduction

Recent years have witnessed the rapid development of wireless sensor technologies in mobile environments, such as GPS, Wi-Fi and RFID. The advances of such technologies indicate the possibility to change radically the existing methods of doing taxi business. Indeed, recent efforts have been made on providing personalized mobile services to taxi drivers through the analysis of Taxi GPS traces. In general, there are three existing ways to provide such services. The first way is to focus on the development of the fastest driving route (Yuan et al., 2010, 2011; Zheng, Yuan, Xie, Xie, & Sun, 2010; Zheng, Liu, Yuan, & Xie, 2011; Nagy & Salhi, 2005), which shows the fastest driving route from the current location to the destination. The second way is to provide a sequence of pick-up points for taxi drivers. The goal is to allow the taxi driver to find a customer within the shortest driving distance (Ge, Xiong, Tuzhilin, et al., 2010). Finally, an alternative service is to strike a balance between the needs of taxi drivers and passengers (Yuan et al., 2013).

Indeed, most of the existing mobile recommender systems for taxi business are focused on extracting energy-efficient transportation patterns from historical location traces and recommending a sequence of potential pick-up points for taxi drivers (Yuan et al., 2010, 2011; Ge, Xiong, Tuzhilin, et al., 2010). However, in the real world, the income of taxi drivers is strongly correlated with the effective driving hours which may not necessarily lead to energy-efficiency. In other words, it is more critical for taxi drivers to know the actual driving routes to minimize the driving time before finding a customer. Taxi drivers usually rent their cabs from taxi companies for a fixed time period. There is a fixed per-hour cost associated with gas usage and the rental fee. The profit of a taxi driver really depends on how much money the driver can make per hour; that is, how effectively the drivers can make use of their driving time.

To that end, in this chapter, we propose to develop a cost-effective recommender system for taxi drivers. The design goal is to maximize their profits when following the recommended routes for finding passages. In particular, the proposed system can provide an entire driving route rather than just recommending a sequence of discrete pick-up points and letting the driver decide how to get to those points, and the drivers are able to find a customer with the largest potential profit by following the recommended route. This makes our recommender system more practical and profitable than other existing mobile recommender systems (Ge, Xiong, Tuzhilin, et al., 2010).

To achieve the design goal and recommend an entire driving route which allows the taxi drivers to maximize their profits by following the recommended route, there are

several factors to be considered. First, it is necessary to know the pick-up probabilities along the route. Second, it should be able to compute the profit that drivers can make after picking up a customer somewhere on the route. Third, the potential driving time on the route should be estimated. Indeed, all these issues can be solved by mining the historical Taxi GPS traces. However, a key challenge is how to combine the impact of all these factors. Indeed, in this research, we develop a net profit objective function to collectively integrate the impact of the above factors. The net profit objective function can be used for evaluating the potential profit of the driving routes. Then, we develop a graph representation of road networks and provide a Brute-Force strategy to generate optimal driving route for finding passengers. In addition, the search for candidate driving routes is essential a combinatorial search problem. The computational cost is prohibited. Therefore, we further develop a pruning strategy to reduce the search space and improve the computational performances. In particular, we first change the graph representation of road networks to a new structure, namely a recursion tree, based on the special form of the net profit function. Then, we design a novel recursion strategy based on the recursion tree for searching optimal candidate routes in an efficient way.

When recommending the driving routes to the taxi drivers, we also provide a strategy for making a better load balance for the recommendations happening at the same location. Specifically, we exploit a minimum redundant strategy. For each target location, we transform each candidate route in the recommended list associated with this location into a direction vector. Then, we are able to calculate the correlations among this candidate route in terms of their directions. If there are several requests

happening at the same location within a short time period, this minimum redundant strategy can provide recommendations in a load balanced way.

Finally, we carry out extensive experiments on a real-world data set collected from the San Francisco Bay area and the experimental results clearly validate both the effectiveness and efficiency of the proposed recommender system.

Overview. The remainder of this chapter is organized as follows. Section 2.2 formulates the problem of cost-effective recommendations for taxi drivers and introduce some preliminaries. Section 2.3 provides a detailed description of our recommender system. In Section 2.4, we report the experimental results. Finally, Section 2.5 concludes this work.

2.2 Problem Formulation

In this section, we first introduce some preliminaries, and then formally define the problem of Maximum Net Profit (MNP) recommendation for taxi drivers.

2.2.1 Preliminaries

Here, we first introduce some basic concepts used throughout this dissertation.

Road Network Formulation

Definition 1 (Road Segment) *A long street can be separated into several road segments r by its crossroads. Specifically, each segment r is associated with a start point $r.s$ and an end point $r.e$. Moreover, each segment r also has several adjacent segments forming a set $r.next[]$, which satisfies $\forall r_i \in r.next[]$ iff. $r.e = r_i.s$.*

Definition 2 (Route) *A route R is a sequence of connected road segments, i.e.,*

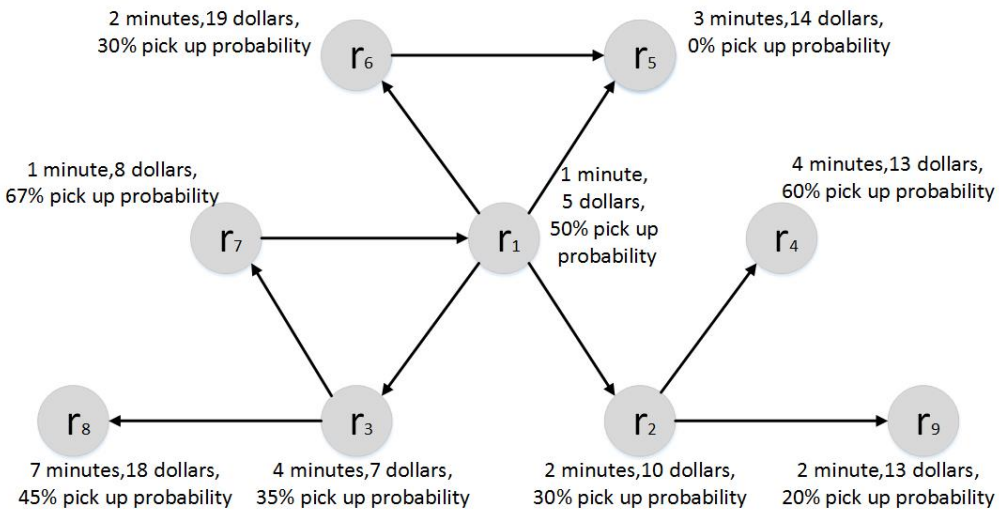


Figure 2.1. An example of a route segment network.

$R = (r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_M)$, where $r_{k+1}.s = r_k.e$ $(1 \leq k < M)$. The start point and the end point of a route R can be represented as $R.s = r_1.s$ and $R.e = r_n.e$.

Definition 3 (Road Segment Network) *The road segment network G can be represented by a graph $G =< V, E >$, where $V = \{r_i\}$ is the node set that consists of all road segments and E is the edge set, which satisfies $\exists e_{ij} \in E$ iff. $r_j \in r_i.next[]$.*

Figure 2.1 demonstrates an example of the road segment network. In this graph, each node represents for a road segment. Note that, each edge only has one direction. This is because we do not allow taxi drivers to drive back and forth in the same single road segment, which is not recommended in real life and has a high potential to result in traffic jam and accidents. However, taxi drivers can take a loop through three road segments, such as nodes r_1, r_3 and r_7, which can form a loop for drivers.

Calculation of Net Profit

For each segment r, the net profit $g(r)$ consists of two components, namely *potential earning* and *potential cost*. Specifically, we define the potential earnings of segment r as $e(r)$, which can be computed by

$$e(r) = \frac{\sum_{i=1}^{N_r} Fee(i;r)}{N_r} P(r), \tag{2.1}$$

where N_r is the number of picking-up passengers in segment r during a given time period, $Fee(i;r)$ is the earning from the i-th pick-up passenger and $P(r)$ is the pick-up possibility in segment r, which will be introduced in Section 2.3. Meanwhile, the potential cost of segment r, i.e., $c(r)$, can be computed by

$$c(r) = (1 - P(r))(L(r) \cdot Gas + T(r) \cdot CompanyFee), \tag{2.2}$$

where $L(r)$ is the length of segment r, Gas is the price of gas per unite distance (e.g., per mile), $T(r)$ is the traveling time through segment r and $CompanyFee$ is the opportunity cost per unit time (e.g., per minute). Indeed, $T(r)$ is sensitive to the real-time traffic conditions. For example, a traffic jam will result in a high $T(r)$, and thus bring a high cost of $T(r) \cdot CompanyFee$. In this case, the segment will not recommended by our model. Therefore, the net profit of segment r, i.e., $g(r)$, can be computed by

$$g(r) = e(r) - c(r). \tag{2.3}$$

Based on the above, we can further define the net profit for each route R. Specifically, given a route $R = (r_1 \to r_2 \to \cdots \to r_M)$ starting from r_1, its total net profit

can be computed by

$$G(R, r_1, M) = g(r_1) + \sum_{i=2}^{M} g(r_i) \prod_{j=1}^{i-1} \big(1 - P(r_j)\big). \tag{2.4}$$

Intuitively, the net profit of route R is the sum of the net profit of road segments $\{r_i\}$ contained in R, which is weighted by the possibility of not picking up any passenger in previous segments (i.e., r_1 to r_{i-1}).

Indeed, with the possibility weights in net profit, the taxi driver will not consider the segments which are far away from her current location because the expected profit there is very low. To be more specific, we can define the average increasing rate of net profit as $\tau = \frac{<G(R,r_i,M+1)> - <G(R,r_i,M)>}{<G(R,r_i,M+1)>}$ to indicate the profit increase when increasing one more road segment in the route. Figure 2.2 shows the trend of the increasing rate with respect to different numbers of increased road segments and different pick-up possibilities. We can observe that the increasing rate is less than 10% after increasing more than 5 road segments. Indeed, the average pick-up probability of each road segment in our experiments is always less than 0.1, therefore it is possible for us to set an upper bound Λ for route length M in Equation 2.4. Based on the above definitions, we can formally define the MNP recommendation problem as follows.

Definition 4 (Problem Statement) *Given the current location $LCab \in r$ of a taxi driver, a fixed cruising length M, and a set of route candidates $\mathbb{R}$, where $\forall R \in \mathbb{R}$ satisfies R starts from r. The MNP recommendation problem is to recommend a route $R^* \in \mathbb{R}$, which has the maximum net profit, i.e.,*

$$R^* = \arg\max_{R \in \mathbb{R}} \big\{G(R, r, M)\big\}. \tag{2.5}$$

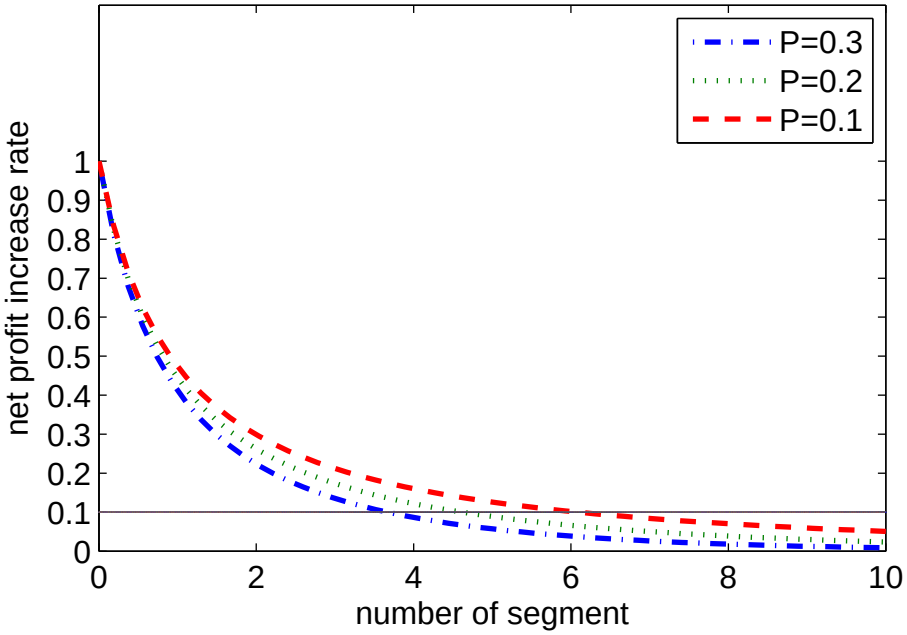


Figure 2.2. The average increase rate of net profit with respect to different number of increased road segment and the different fixed pick up possibility (i.e., $P(r) = 0.1$, $P(r) = 0.2$, $P(r) = 0.3$).

Different from other existing recommender systems for taxi drivers, which mainly focus on extracting energy-efficient transportation patterns based on traveling time/length and recommending a sequence of potential pick-up points for taxi drivers (Yuan et al., 2010, 2011; Ge, Xiong, Tuzhilin, et al., 2010), the MNP recommendation problem focuses on providing an entire driving route with maximum net profits for a taxi driver. Along this line, there are two major challenges for solving the MNP recommendation problem. First, how to calculate the parameters $g(r)$, $P(r)$ of each segment r from the historical pick-up data. Second, how to efficiently search an optimal route from the complex directed-cyclic route segmentation network. In the following section, we will introduce our solutions for the above two challenges, respectively.

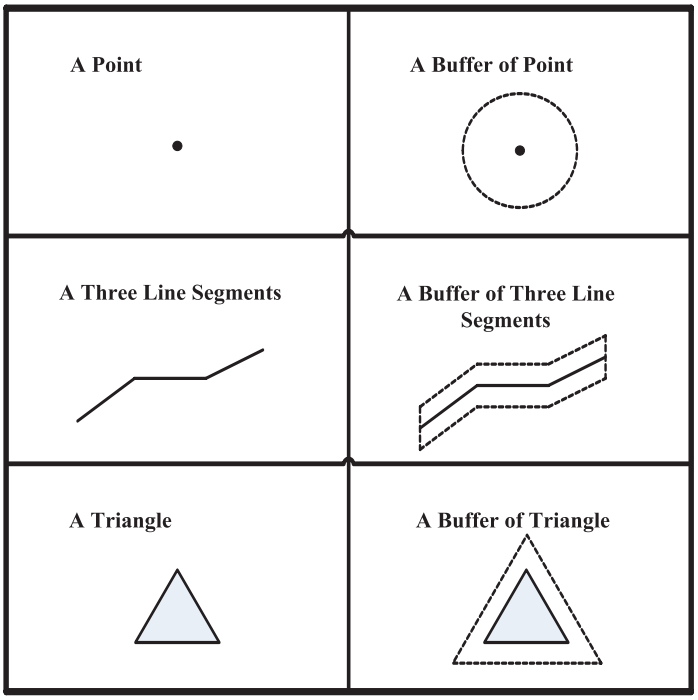
2.3 Maximum Net Profit (MNP) Recommendation

In this section, we introduce the technical details of our solutions for the MNP recommendation problem.

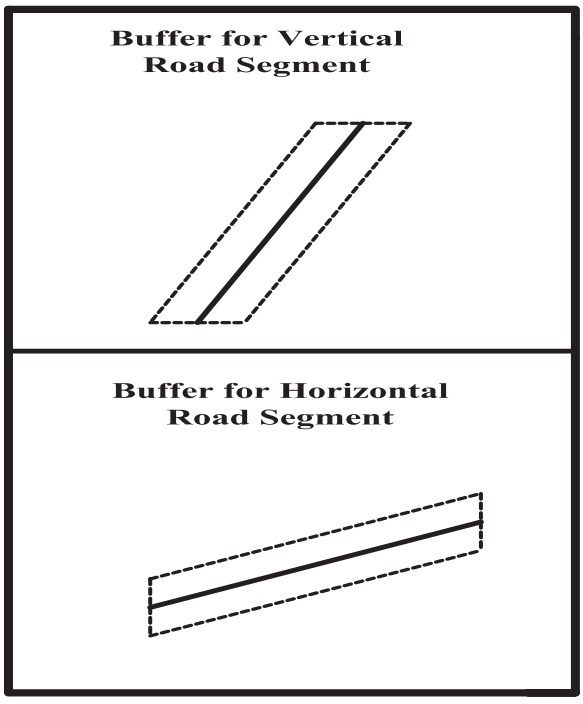
2.3.1 Parameter Estimation with Road Buffer

To accurately obtain the taxi driver's current location and the parameters for estimating net profit, i.e., $P(r)$ and $g(r)$, we exploit the *road buffer* estimation for each road segment. Specifically, in geographic information systems, a buffer is a zone of specified distance around the spatial object. The boundary of the buffer is the solid line of equal distance to the edge of the object. Figure 2.3(a) is an illustration of different buffer operations, such as buffers on a point, three line segments and a polygon (Xiong et al., 2004). Intuitively, people would like to wait for taxis at road side instead of in the middle of road, and the pick-up points of taxis are always around the road side. Therefore, when calculating for the number of historical pick-up events, we need to build a buffer around each road segment for obtaining the new boundaries of the road. This road buffer usually looks like a rectangle surrounding the road, which is similar with the buffer operation of three line segments. Particularly, the size of the buffer depends on the demands of different real-world problems.

To build the road buffer, we need to define the vertical road and the horizontal road first. To be specific, by using the longitude and the latitude of the starting and the ending points of each road segment, we can calculate the tangent value of this road segment. If the absolute value of the tangent is greater than 1, then we regard the corresponding road as vertical road, otherwise it is a horizontal road. For each



(a) Different buffer operations



(b) Buffer operations on road segments

Figure 2.3. Buffer Operations

vertical road, we keep the longitudes of its starting and ending points and extend the corresponding latitudes to west and east. For each horizontal road, we keep the latitudes of its starting and ending points and extend the corresponding longitudes to north and south. The above buffer operation results in new boundaries formed by four vertex coordinates. For example, Figure 2.3(b) shows the buffer operations on vertical and horizontal road segments.

Given the historical pick-up data and the road buffers, we are able to calculate the total number of pick-up events in each road segment r, which indicates how frequently a pick-up event can happen when cabs travel across each road segment. Let N_r^0 denote the number of times that taxis are vacant in the buffer of road segment r, and and N_r^* denote the number of times that taxis had pick-up events in the buffer of segment r. Thus, the probability of pick-up event for each road segment r, i.e., $P(r)$, can be estimated as

$$P(r) = \frac{N_r^*}{N_r^0 + N_r^*}. \tag{2.6}$$

From the i-th historical pick-up event in segment r, we can also obtain the earnings $Fee(i; r)$ in Equation 2.1. Furthermore, the road length $L(r)$ and real-time traveling time $T(r)$ can be estimated from the historical data or some external resources, such as Google Map. Therefore, the net profit of $g(r)$ can be calculated by Equation 3.3. Particularly, the value $T(r)$, $g(r)$ and $P(r)$ of each road segment r can be pre-stored in corresponding node of the road segment network (e.g., Figure 2.1).

2.3.2 MNP Route Recommendation

In this subsection, we introduce how to solve the problem of NMP recommendation by different strategies.

Brute-Force Recommendation Strategy

After obtaining the road segment network, we can leverage it for generating route candidates can MNP recommendation. To this end, we first propose a Brute-Force strategy for this task based on the Breadth-First search. Specifically, the recommendation algorithm is shown in Algorithm 1. In this algorithm, we keep a route queue Q for generating a set of route candidates C, and the function $MNP(C)$ in Step 5 is used for finding the optimal route with maximum net profit in candidate set C. However, such Brute-Force method for searching the MNP route is not efficient, since it has to check all possible routes with length M in G.

Lemma 1 *Given a fixed cruising length M and the road segment network $G = \{V, E\}$, where $|V| = N$, the computational complexity of searching an optimal MNP route by Brute-Force algorithm is $\mathcal{O}(MN^{M-1})$*

Proof Obviously, the total number of route candidates in road segment network G is $\leq N^{M-1}$, and computing the net profit for each route needs M operations. Thus, the complexity of searching optimal MNP route is $\mathcal{O}(MN^{M-1})$

Intuitively, the computational complexity of the Brute-Force algorithm is too high to satisfy the needs of real-world applications. There are some algorithm can save the reaching time of freeway travel in real world. To this end, we further propose

Algorithm 1 Brute-Force based MNP Recommendation

Input 1: road segment network $G = \{V, E\}$;

Input 2: the cruising length M;

Input 3: taxi driver's current segment r_1;

Output: the MNP route R^*;

Initialization: A route queue $Q = \{R_0\}$, where $R_0 = \{r_1\}$;

1: $C = \emptyset$;
2: //get route from queue Q;
3: $R = Q.del()$;
4: **if** $(R = \emptyset)$ **do**
5: **return** $R^* = MNP(C)$;
6: **else if** $(|R| == M)$ **do**
7: $C = C \cup R$;
8: **else if** $(|R| < M)$ **do**
9: //r_k is the last road segment in R;
10: **for each** $(r_i \in V, \exists e_{ki} \in E)$ **do**
11: //add route from queue Q;
12: $Q.add(R \cup \{r_i\})$;
13: **go to Step 3;**

another recommendation strategy based on the recursive characteristic of the net profit function.

Recursive Recommendation Strategy

By observing the form of the net profit of routes, we can re-write the Equation 2.4 as follows.

$$G(R, r_1, M) = g(r_1) + \big(1 - P(r_1)\big)G(R - r_1, r_2, M - 1), \tag{2.7}$$

where $R = (r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_M)$. Indeed, the special form of total net profit can be realized by a recursion algorithm. To this end, for each road segment r_1, we can denote all the route candidates starting from r_1 as a *recursion tree* structure. Specifically, the recursion tree of a road segment can be defined as follows.

Definition 5 *[Recursion Tree] The recursion tree Υ_{r_1} of a road segment r_1 is a tree, where each node represents a road segment and the root node is r_1. Moreover, for each node r_i in the recursion tree, it has a children node set that equals to $r_i.next[]$.*

For example, Figure 2.4 shows an example of the recursion tree of road segment A. In this dissertation, we propose a method $RTree(r, M)$ for building a M-depth recursion tree Υ_r for r, which is shown in Algorithm 2. Particularly, the tree Υ_r obtained by our algorithm will hold M node sets $\Upsilon_r.level[i]$ $(1 \leq i \leq M)$, which represents the nodes in the i-th level of the tree. With this structure, the MNP recommendation from segment r_1 can be separated into several simpler MNP recommendation tasks recursively. Take Figure 2.4 as an example, we can develop a bottom-up method to compute the MNP route with length 3, of which the net profit is denoted as $G(A, 3)$. Specifically, according to the definition of net profit, we can obtain

Algorithm 2 RTree(r,M)

Input 1: road segment r_1 as root node;

Input 2: the depth M of recursion tree;

Output: a M-depth recursion tree Υ_r;

Initialization: $Depth = 1$; $\Upsilon_r.level[i] = \emptyset$ $(1 \leq i \leq M)$;

1: $\Upsilon_r.root = r$; $\Upsilon_r.level[1] = \{r\}$;

2: **if** $(Depth \geq M)$ **do**

3: **return** Υ_r;

4: **else**

5: **for each** $(r_{cur} \in \Upsilon.r.level[Depth])$ **do**

6: $\Upsilon_r.level[Depth + 1] \cup = r_{cur}.next[]$;

7: $Depth+ = 1$;

8: **go to Step 2;**

$G(A,3) = g(A)+(1-P(A))\times max\{G(B;2), G(C;2), G(F;2); G(E;2)\}$, where the net profit of each MNP route with length 2 can also be computed by the profit of their sub-routes. For example, we have $G(B;2) = g(B) + (1 - P(B)) \times max\{G(D;1), G(I;1)\}$, and the profit of each individual segment (i.e., leaf nodes of the tree) can be directly computed by its profit, e.g., $g(D)$. Therefore, given a recursion tree of r, we can obtain the MNP route with length M by recursing $M - 1$ times. Specifically, in this research we develop a recursion algorithm $rNMP(r, K)$ for MNP recommendation, which is shown in Algorithm 3. By implementing our algorithm with parameters $r = r_1$ and $K = M$, the MNP route starting from road segment r_1 with length M and corresponding MNP value will be obtained.

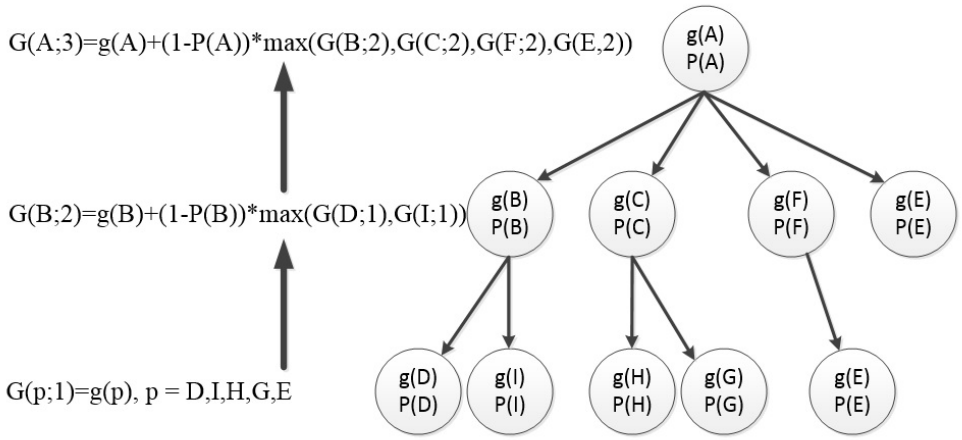


Figure 2.4. The recursion tree representation of route network. We can calculate the MNP $G(R, A, 3)$ from the leaf nodes of the tree.

Lemma 2 *Given a M-depth recursion tree Υ, where $\forall r \in \Upsilon$, $|r.next[]| \leq N$, the complexity of searching an optimal MNP route by the recursion method is $\mathcal{O}(N^{M-1})$*

Algorithm 3 rMNP(r,K)

Input 1: recursion tree of Υ_r;

Input 2: the depth M of recursion tree;

Output: the MNP value and route stating from r;

1: $Depth = M - K + 1$;
2: **if** $(Depth == M)$ **do**
3: $Profit = \emptyset$;
4: $Route = \emptyset$;
5: **for each** $(r_i \in \Upsilon_r.level[Depth])$ **do**
6: $Profit[i] = g(r)$;
7: $Route[i] = r_i$;
8: **return** $(Max(Profit),Max(Route))$;
9: **else**
10: $Profit = \emptyset$;
11: $Route = \emptyset$;
12: **for each** $(r_i \in \Upsilon_r.level[Depth])$ **do**
13: $(Profit^*, Route^*) = rMNP(r_i, K-1)$;
14: $New_route[i] = r_i \cup Route^*$
15: $Profit[i] = g(r_i) + (1 - p(r_i)) \cdot Profit^*$;
16: **return** $(Max(Profit),Max(Route))$;

Proof Assume that the computational cost of finding $G(R, r_1, M)$ is $T(M)$, obviously we have $T(M) \leq NT(M-1)+1$. Moreover $\forall r$ satisfies $|r.next[]| = N$, the computation can be separated into N sub-problems. Particularly, for route with only one segment, we have $T(1) = 1$. Meanwhile, after recursing $M-1$ times, we have $T(M) \leq N^{M-1}T(1) + \frac{N^{M-1}}{N-1}$. Therefore, the computational complexity of searching optimal MNP route by recursing tree is $\mathcal{O}(N^{M-1})$

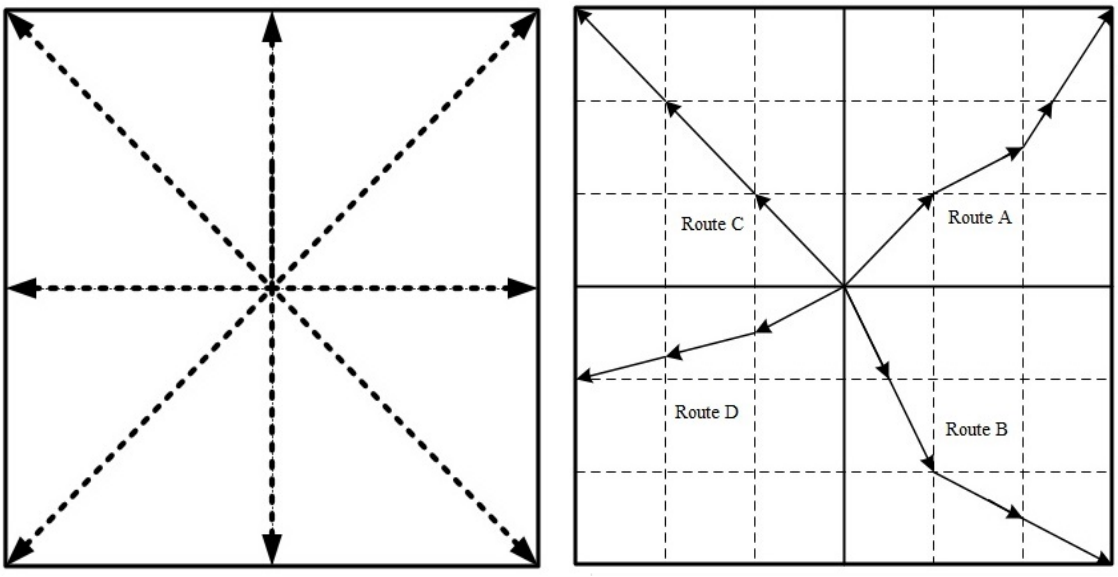
Although the recursion tree can achieve more efficient recommendation than the Brute-Force method, the computational cost increases significantly as M becomes larger. According to the discussion in Section 2.2, we can set an upper bond Λ for M, since the average increasing rate of the net profit is very low after $M > 5$. Therefore, we set $\Lambda = 5$ in our experiments.

2.3.3 Top-K Route Recommendation

Based on the above algorithms, our recommender system can recommend an optimal MNP route for a single taxi driver. However, in real life, an ideal recommender system must be capable of recommending multiple taxi drivers in the same area simultaneously. In this section, we address this problem and introduce a minimum redundant strategy for the recommendation process in the real world.

Intuitively, a straightforward recommendation strategy is to recommend the optimal driving route to all available drivers. However, if we recommend the same route to too many drivers at the same time, it will cause an overloaded problem and degrade the performance of the recommender system. The overloaded problem is a classic problem which has been widely studied. For example, the load balancing

mechanism distributes requests among web servers in order to minimize the execution time (Z. Xu & Huang, CS213 Univ. of California,Riverside; Grosu & Chronopoulos, 2004). In our problem, we can treat multiple empty cabs as jobs and multiple optimal drive routes as computers. Instead of solving this overloaded problem by exploiting existing load balancing algorithm, we want to focus on the direction characteristics in the mobile recommender system and exploit a direction-based clustering (DEN) method (Zhou et al., 2010) to distribute the empty cabs by following the top-K optimal drive routes (Ge, Xiong, Zhou, et al., 2010; Yuan, Sun, Tian, Chen, & Liu, 2009).



(a) (b)

Figure 2.5. (a) Direction-based clustering; (b) Top-K route Recommendation.

Before recommending driving routes to taxi drivers, we first rank all the route candidates according to their net profits and obtain the top-K driving routes. After recommending the top ranked route to the first taxi driver, we need to calculate

the correlation between this route and all other $K-1$ candidate routes, and then recommend the route with the lowest correlationship (Tang et al., 2011) to the second driver.

In order to calculate the correlation between those candidate routes, we first partition the space into grids and turn the movement statistics in each grid into a vector which represents the probabilities of moving directions within the grid. Then, we transform the direction information of the taxis' movement into the same data format, and further partition each small grid into 8 direction bins. For example, in Figure 2.5(a) the angle of each bin has a range of $\pi/4$. Next, we transform each grid into a direction vector $g = (p_1, p_2, p_3, ..., p_8)$, where each p_i is the probability of moving towards direction i within this grid and $p_i = f_i / \sum_{k=1}^{8} f_k$, where f_i is the frequency of moving objects that have passed this grid and has the direction along the direction i.

For instance, as shown in Figure 2.5(b), we first recommend route A to the first taxi driver, route B, C and D are other candidate routes at the same time and same location. Then we divide the space into small grids and get the direction vectors for each grid. A driving route candidate which has lowest correlation with the previous recommending route is usually the one with a different driving direction in the beginning. Therefore, we only need to analyze the first n grids to decide the driving directions. We combine the direction vectors in n grids together and get a vector with $8n$ elements for each candidate route. For example, the vector for route A is $g(A) = (p_{11}, p_{12}, ...p_{n7}, p_{n8})$. Then, we calculate the correlation of those vectors for each pair of candidate routes. Thus, the correlation between route A and B can

be computed by $\rho(A, B) = Cov(g(A), g(B))/\sigma_g(A)\sigma_(g(B)$. If route B has the lowest correlation with route A, we will recommend route B for next coming empty cab.

2.4 Experimental Results

To validate the efficiency and effectiveness of the proposed recommender system, extensive experiments are performed on real world data sets collected in the San Francisco Bay Area in 30 days.

2.4.1 Experimental Data

Taxi GPS Traces. In the experiments, we use the real-world taxi GPS traces collected by the Exploratorium-the museum of science, art and human perception through the cabspotting project. The mobility traces are the records of the cabs' driving states in consecutive time, with each be represented as a tuple, (*latitude, longitude, fare identifier, time stamp*). By cleaning the dataset, we obtained 89,897 pick-up and drop-off activities in total. Generally, we assume that most drivers would follow the suggested driving route provided by the Google Map, thus we can get the fare related to the specific trip and the fare information can also be used to calculate the profits concerning the trip. The following Figure 2.6 is an example of one hundred taxi drivers' pick-up points in 30 days in the San Francisco Bay Area, with each red point representing one pick-up activity. Figure 2.7 is an heat map illustration of pick-up probabilities. Here, different color and area of circles represent different pick-up probabilities. This map shows there are lots of pick-up activities around the Market Street of San Francisco, which is a very busy street with lots of shopping places and museums. Other pick-up hot spots including Fisherman's Wharf, Divisadero St,

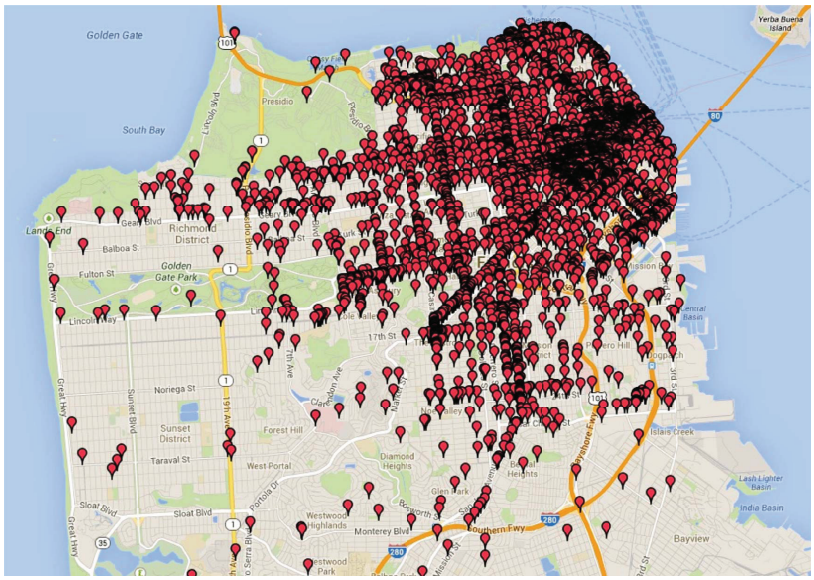


Figure 2.6. A demonstration of pick-up points in the dataset.

Cathedral Hill and Western Addition.

Road Network Data. Because the quality of existing road networks in San Francisco is not sufficient. We build the road network dataset of San Francisco by using google API. First, we searched for all the street names in San Francisco. Second, we run the google API to find out if there is an intersection between two streets. We keep a record of each intersection point. Figure 2.8(a) is an illustration of our intersection points. Then, we use each intersection point to search the nearest points in four different directions and connect those 5 points together. Therefore, we can obtain four different connected road segments with starting points and ending points. However, as the yellow line in Figure 2.8(b), we may accidentally connect two intersections with no road between them. To solve this problem, we calculate the distance of those two intersections by using coordinates and compare it with the driving distance measured by the Google map. If there is a road between those two points, those two distances should be very close to each other. If the distances are not close to

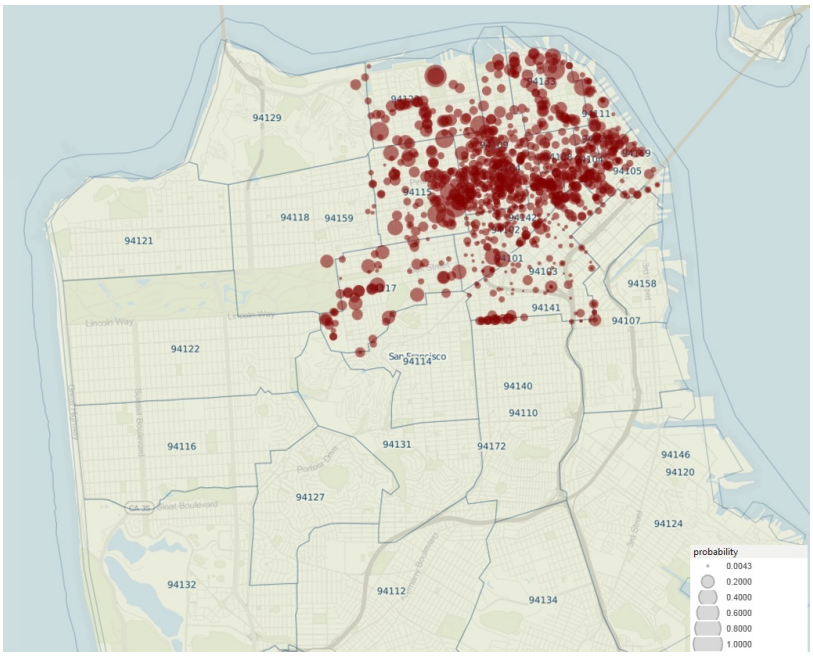


Figure 2.7. The heat map of pick-up probabilities in San Francisco bay area.

each other, it means there is no road between those two intersections and we delete this road segment from the road network dataset.

The road network dataset contains 5391 roads in the San Francisco Bay Area, with each consisting of the ID, starting points, ending points and we also calculate the historical pick-up probability and net profit associate with each road segment. For each road, several coordinates of the intermediate points may be recorded and there are also some noise points. After removing the noise points, we selected 2,149 roads with high pick-up probability for our experiment. Then, we can build road buffer with the starting points and the ending points in those road segments.

By matching the pick-up coordinates of the Road Network Dataset with the Taxi Dataset, we are able to get 87,688 valid pick-up activities which can be located in the road segments, therefore the two data sets are combined together with each pick-up

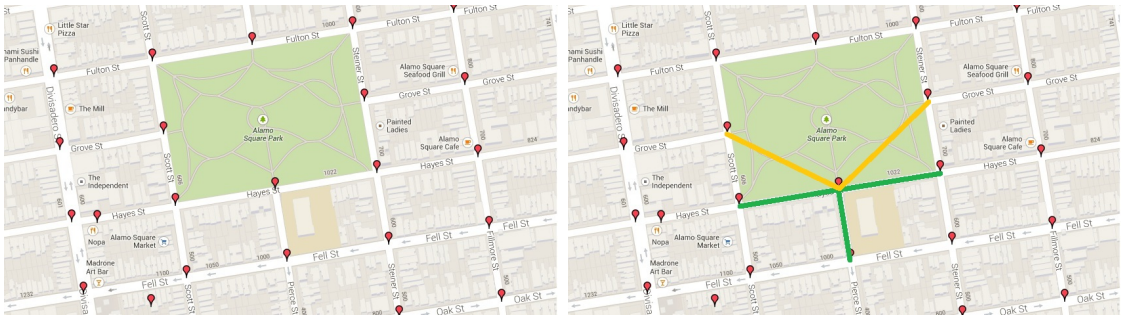


Figure 2.8. (a) Intersections; (b) Connected Road Segments.

point mapped to the constructed road buffer. To implement the proposed algorithm, we also need to calculate the pick-up probability and the net profit for each road segment in those road segments. This has already been presented in Section 2.3.

Finally, we get the coordinates of the starting and the ending points for each road segment, along with the pick-up probability, the net profit and the average driving time in this road segment. Note that the average driving time is estimated as the distance of each road segment divided by the average driving speed in the San Francisco Bay Area.

2.4.2 Empirical Studies on Recommendations

Here, we provide two case studies. One case study is on cost effective route recommendations. Another case study is on top-K recommendations.

A Case Study on Cost-Effective Route Recommendations

Here, we show two examples of MNP route recommended by our approach and compare it with the suggested route by the Google map. Specifically, in Figure 2.9 and Figure 2.10, we plot the optimal driving route suggested by our recommender system at a randomly selected initial location of the target cab. We also assumed that the

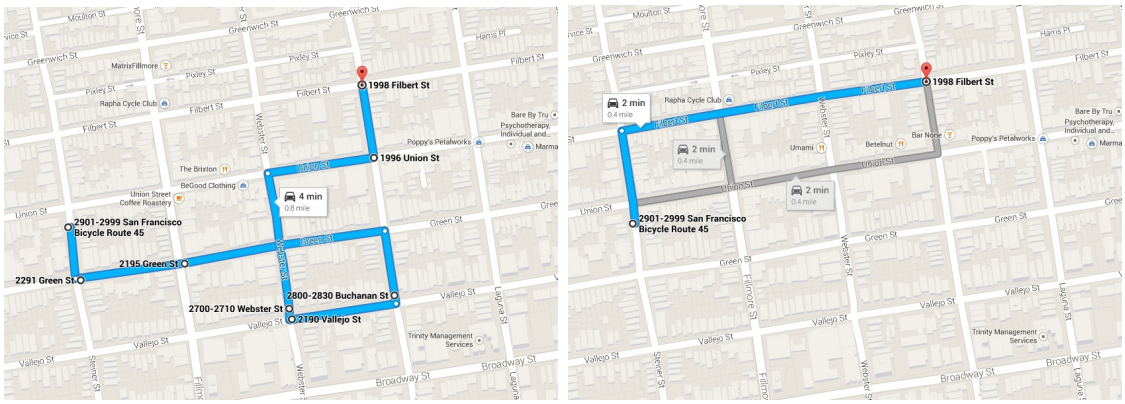


Figure 2.9. Cost-Effective Route Recommendation Case Study (a)

driver's expected cruising length is 5, and after every 5 road segments, the system will use the current location as the new starting point for search and restart the recommendation process. In order to do the comparison, we calculate the real driving time of each trip of taxi drivers and restart our recommendation system until the total driving time in those MNP routes is equal to the real driving time of each trip. Then, we connect those MNP routes together and this is the entire driving route that should be recommended to the drivers. In those figures, the left figures are the driving route recommended by the MNP recommender system and the right figures are the route suggested by the Google Map based on the shortest driving distance. However, this driving route suggested by the Google map cannot maximize taxi drivers' net profit.

Recently, most recommender system can only suggest a sequence of hot spots to taxi drivers. There is no such recommendation system that can suggest an entire driving route. If taxi drivers do not know how to drive to the nearest hot spot, he or she has to follow the driving route provided by the Google map. However, both the pick-up probability and the potential net profit may be very low along those routes.

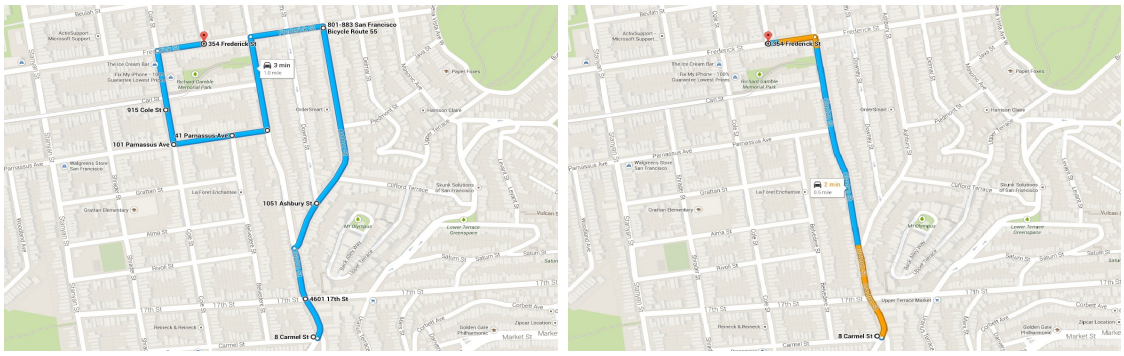


Figure 2.10. Cost-Effective Route Recommendation Case Study (b)

The drivers have a high probability of losing money until they reach the next hot spot. Our recommender system can improve potential net profits for taxi drivers compared to the routes suggested by the Google map.

A Case Study on Top-K Recommendations

In Section 2.3, we introduced a minimum redundant strategy to recommend the Top-K driving routes and solved the overloaded problem. In figure 2.4.2, we demonstrate the top K driving routes starting from the same location, where K equals to 4 in this case. The figure shows that each route has different driving directions and the correlations between those driving distances are very small. Therefore, the minimum redundant strategy can improve the performance of our recommender system.

2.4.3 Route Recommendation for Inexperienced Taxi Drivers

Given one specific location, our proposed algorithm can recommend several routes with high expected utility for drivers. The algorithm is especially applicable for inexperienced drivers, since they lack of knowledge about the roadmap and the local driving routes that can make profits. To validate the effectiveness of the proposed

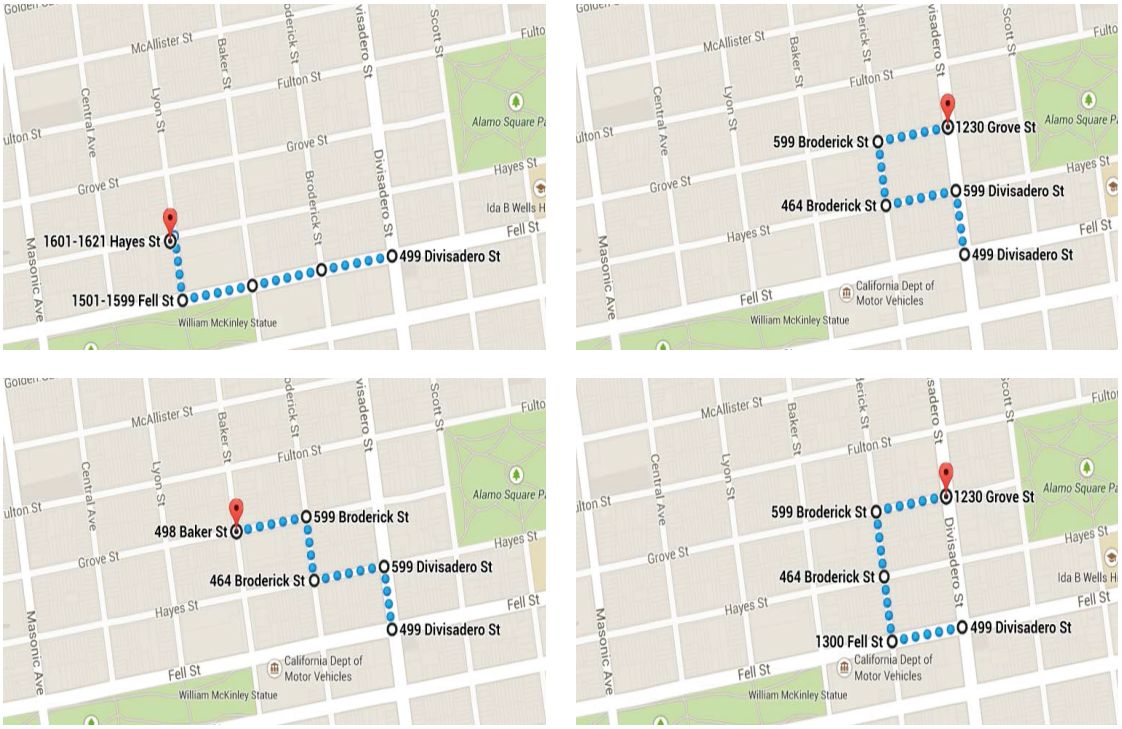


Figure 2.11. The top 4 driving routes starting from the same location (longitude: -122.4376221 and latitude: 37.77407074)

algorithm, we firstly divide all the drivers into two categories based on their average net profits. The top 10% drivers in the dataset are treated as 'experienced' drivers, while the others are 'inexperienced'. Therefore, the driving routes of the experienced drivers are used as training set and we recommend driving routes for the inexperienced drivers.

We define driver's event $\mathbf{e}$ as a consecutive sequence of 'roam $\rightarrow$ pick up $\rightarrow$ drop off', by extracting the pick-up and drop-off activities of each user, we can reconstruct each event. For each driver, we define the location where the driver starts to search for potential pick-ups as l_0, and after roaming in Δt time, the driver picks up passengers at location l_1 and drive for $\Delta t'$ and drop off at l_2. Let $r_{i,j}$ denotes the road segment between location l_i and l_j, then event e can be represented with $(r_{0,1}, \Delta t, r_{1,2}, \Delta t')$, and the unit time profit of the event can be calculated as $p_e = \frac{pr_{12}}{\Delta t + \Delta t'}$. Thus, the proposed algorithm starts with the location l' which is neareset to l_0, and return a sequence of recommended potential pick-up points and road segments.

The performance of the recommended driving route is measured by the average net profit per unit time p_r, and it is compared with the average unit net profit of the inexperienced drivers, i.e., $p_d = \frac{\sum p_e}{|\mathbf{e}|}$.

The statistical experiment results for recommended driving routes for inexperienced drivers are shown in Table 2.1, the average net profits per unit time outperforms the real profit of the inexperienced drivers.

We first plot the distribution for the net profit per unit time, i.e., the number of events for specific profit values, as shown in Figure 2.12. The net profit per unit time of our recommended route is compared with the inexperienced taxi drivers'

Table 2.1. Net Profits per Unit Time

	Recommender System	Inexperienced Drivers
Mean	0.038148	0.024162
SD	0.017815	0.018455

performance based on statistical histogram. The blue bar of the histogram shows the net profits from our recommendation results and the red bar shows the profits from the inexperienced taxi drivers. As we can see from the figure, the recommendation events mostly positioned on bigger values. This indicates that our recommender systems provide higher profit routes than the real routes by inexperienced drivers.

To further investigate the performance of the recommender system, we also study the difference of net profit per unit time between the recommended routes and the drivers' real routes for each event, i.e. $p_r - p_e$. As shown in Figure 2.13, the X axis is the difference between the profits of the recommended results and the inexperienced taxi drivers' profits. We can see that most of dot points are positioned to the right of $X = 0$, meaning that the profits of our recommended routes outperform the profits of the routes by the inexperienced drivers.

Then, we evaluated the performance of the Brute-Force recommendation strategy and the performance of the recursive recommendation strategy. This experiment was conducted across 1000 randomly picked starting points. We only compared the running time for five road segments, because the increasing rate of pick-up probability in Equation 4 is less than 10% after increasing more than 5 road segments. As shown in Figure 2.14, the red line is the running time for the Brute-Force recommendation

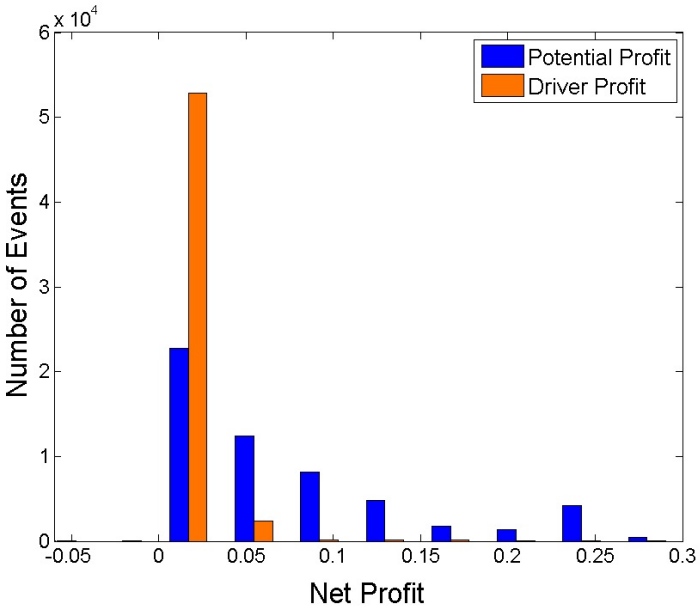


Figure 2.12. Net Profit Statistics. The blue bar represents the potential profits of our optimized routes and the orange bar represents the profits of taxi drivers' traditional routes ranked below top 10%

strategy and the black line is the running time of the recursive Strategy. We can see that the recursive strategy can lead to better efficiency compared to the Brute-Force strategy. Note that all the experiments were conducted on a Windows 7 with Intel(R) Core(TM)i5-3210 CPU and 6.0 GB RAM.

To sum up, the experiments showed that the cost-effective recommender system could help inexperienced taxi drivers find better routes so as to maximize their potential profits. Also, the recursive strategy can help to efficiently identify the recommended optimal routes.

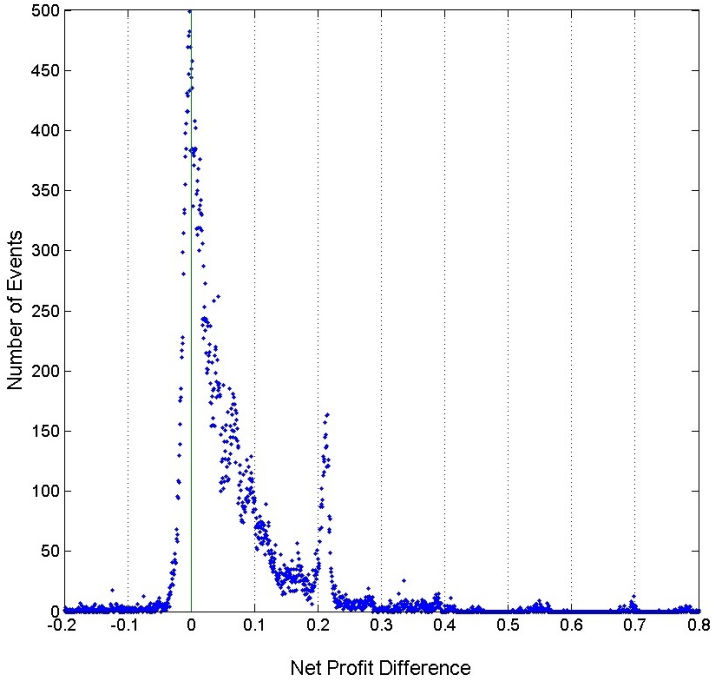


Figure 2.13. Profit Difference. X axis is Net Profit Difference between our optimized routes and taxi drivers' traditional routes ranked below top 10%. Y axis is the number of events

2.5 Concluding Remarks

In this chapter, we proposed a cost-effective recommender system for taxi drivers to maximize their profits by providing profitable driving routes. To be specific, we first provided a net profit objective function for evaluating the driving routes before finding a customer. Then, we proposed a graph based approach to efficiently generate candidate driving routes for finding passengers. As a result, we can use the net profit objective function to rank each candidate route and make recommendations to taxi drivers in a cost-effective way. An unique perspective of our recommender system is that it can recommend an entire driving route instead of only recommending a

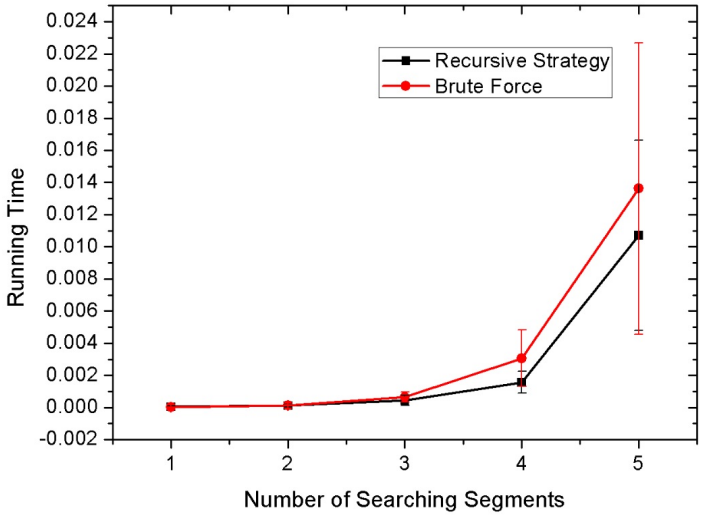


Figure 2.14. A Comparison of the Running Time. The red line represents the running time of the Brute-Force strategy and the black line represents the running time of the Recursive strategy

sequence of discrete pick-up points. Also, the drivers are able to maximize their profits within the fixed time period by following the recommended driving routes. Finally, the extensive experiments on a real-world data set collected from the San Francisco Bay area clearly validated the effectiveness of the proposed recommender system.

CHAPTER 3

ENHANCING RECOMMENDER SYSTEM FOR TAXI DRIVERS WITH BUSINESS EFFECTIVE STRATEGIES

Nowadays, the development of GPS technology and the collection of abundant taxi transaction records bring the key to a more effective mobile recommender system for taxi drivers. Indeed, recent efforts have been made on recommending a sequence of pick-up points for the purpose of minimizing the distance for searching next customer. However, in the real world, the profits of taxi drivers are strongly related to multiple factors such as the effective driving hours, the driving distances, and the potential earnings from passengers. Moreover, these recommender systems suffer from taxi overload problem which dramatically reduce the potential earnings of the taxi drivers who follow the same recommended strategy but come lately. To address these challenges, in this chapter, we propose an enhancing recommender system for taxi drivers with business effective strategies. The goal is to maximize their profits by following the recommended next-customer searching strategies. Specifically, for each taxi driver, we first dynamically estimate the pick-up possibility of each road segment according to its surrounding taxi activities during previous time periods. With the considerations of pick-up possibility, driving distance and potential earnings from the next passenger, a net profit objective function for evaluating the potential profits of route candidates is constructed and optimized by solving a maximum net profit route

searching problem. Then, this maximum net profit from our route searching strategy is compared with a virtual station waiting strategy, which recommends taxi drivers to wait in a specific area (usually near hot spots like station exits, shopping malls, etc.) instead of searching driving routes. In the end, we recommend the strategy with the maximum net profit to taxi drivers. We also carry out extensive experiments on real-world data collected from Beijing City and San Francisco Bay area. The experimental results clearly validate the efficiency and the effectiveness our propose recommender system.

3.1 Introduction

In our earlier work (Qu et al., 2014) in Chapter 2, we presented a cost-effective recommender system for taxi drivers. The goal of this recommender system is to maximize their profits when following the recommended routes for finding passengers. Particularly, instead of recommending a sequence of pick-up points and letting the driver decide how to get to those points, this recommender system is capable of providing an entire driving route.

While the cost-effective recommender system in chapter 2 can efficiently recommend an entire driving route to taxi drivers. In this chapter, we further developed an enhanced recommender system with business effective strategies for taxi drivers. This approach is motivated by the following observation: taxi drivers do not always want to drive around to find their next passenger. Indeed, they may pick up a passenger faster just by waiting in line at a hot pick-up spot, i.e., virtual station, such as a hotel, a restaurant, a movie theater, a train station, or any other popular places that

have abundant passengers waiting for taxis. Therefore, it is necessary to take those virtual stations into consideration in developing recommendation strategies for maximizing taxi drivers' profits. As we stated in the previous chapter, the incomes of taxi drivers are strongly correlated with their effective driving hours. In this chapter, we developed the previous cost-effective recommender system and provide two different recommendation strategies. First of all, we remain the previous cost effective recommender strategy and rename it as route searching strategy. Secondly, based on our real life observation, we provided another strategy which can recommend a hot pick-up spot to taxi drivers and let them wait in line. We call this type of recommendation strategy as virtual station waiting strategy.

The design goal is to maximize taxi drivers' profits in unit time when following the recommended strategies for finding passengers. Specifically, we design a joint learning algorithm to evaluate the potential profits of different strategies. Those strategies could be routing around a neighborhood by following a route with the maximum profit or waiting at a virtual station with the highest ratio of the potential profit over the waiting time depending on the given time period and the current location of the driver. Take the route searching strategy, and the virtual station waiting strategy into consideration, the enhanced recommender will effectively compute and recommend the most profitable strategy.

To achieve the design goal and recommend the most profitable strategy which allows the taxi drivers to maximize their profits by following the recommended strategy, there are several factors to be considered. First, it is necessary to know the pick-up probabilities along the route and the virtual station. Second, it should be able to

compute the potential profit of picking a customer at a given location. Third, the potential driving time/waiting time on the route or at a virtual station should be estimated. Indeed, all these issues can be solved by mining the historical Taxi GPS traces. However, a key challenge is to combine the impacts of all these factors. In this chapter, we develop a joint learning strategy to collectively integrate the impacts of the above factors. This joint learning strategy can be used for evaluating the potential net profits of the driving strategies. Then, we develop a graph representation of road networks and provide a Brute-Force strategy to generate all optimal driving route for finding passengers. In addition, the search for candidate route searching strategy is essential a combinatorial search problem. The computational cost is prohibited. Therefore, we further develop a pruning strategy to reduce the search space and improve the computational performances. As mentioned in Chapter 2, we first change the graph representation of road networks to a new structure, namely a recursion tree, based on the special form of the net profit function. Then, we design a novel recursion strategy based on the recursion tree for searching optimal candidate routes in an efficient way.

When recommending driving routes to taxi drivers, we also provide two strategies for making a better load balance for the recommendations happening at the same location. Specifically, we provide Top-K route recommendation and a novel dynamic MNP strategy to solve this load unbalance problem.

Finally, to validate the efficiency and the effectiveness of the proposed recommender system, extensive experiments are performed on real world data sets collected from Beijing and San Francisco. These data sets include the historical records of the

cab's pick up and drop off events (with latitude, longitude, vendor id, rate code, and time stamp for each event). We also mutually labeled 50 famous virtual stations in Beijing. Then we use those virtual stations as the training set and define other virtual stations from our data sets. In the end, the experimental results clearly validate both the effectiveness and the efficiency of the proposed recommender system.

Overview. The remainder of this chapter is organized as follows. In Section 3.2, we formulate the problem of enhancing recommender system with business effective strategies for taxi drivers and introduce some preliminaries. Section 3.3 provides a detailed description of our recommender system. In Section 3.4, we report the experimental results. Finally, Section 3.5 concludes this work.

3.2 Problem Formulation

In this section, we first introduce some preliminaries that provide the platform for our enhancing recommender system and then formally define the problem of Maximum Net Profit (MNP) recommendation for taxi drivers.

3.2.1 Preliminaries

In order to provide an exact searching route, we first decompose the original trajectories into a sequence of road segments, which are the basic units of taxi searching route and road network. Moreover, we pay special attention to some road segments with special properties, i.e., high frequent region or taxi pick-up stops, and regard them as "virtual station candidates".

Road Network Structure

A road segment is the specific representation of a portion of a road with uniform properties. The start and end points are decided by an intersection, a roundabout or a dead end. A route is a sequence of adjacent road segments following which a taxi driver can search for new customers. A road segment network is a graph structure that contains all possible searching routes and is centered by a taxi driver's start searching location. Formally, the road network structure is defined as follows.

Definition 6 (Road Segment) *A road segment r is the representation of a road portion associated with a start point $r.s$ and an end point $r.e$. A road segment has several static properties including one-way indicator and segment length. If the one way attribute $r.one = 1$, r can only be traveled from $r.s$ to $r.e$. Each segment r has several adjacent segments that are reachable directly from r. The adjacent segments form a segment set $r.next$, which satisfies $\forall r_i \in r.next$ iff. $r.e = r_i.s$.*

Definition 7 (Route) *A route R of length n is a sequence of n connected road segments that a taxi driver can follow to search for next customer, i.e., $R = (r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_n)$, where $r_{k+1}.s = r_k.e$ $(1 \leq k < n)$. The start point and the end point of a route R can be represented as $R.s = r_1.s$ and $R.e = r_n.e$.*

Definition 8 (Road Segment Network) *A road segment network G can be represented by a directed graph $G =< V, E >$, where $V = \{r_i\}$ is the node set that consists of all road segments, and E is the edge set, which satisfies $\exists e_{ij} = (r_i, r_j) \in E$ iff. $r_j \in r_i.next$.*

Virtual Station

Virtual stations are hot spots in certain road segments associated with lots of pick-up events. In real life, we can discover two types of potential virtual stations. The first type is real taxi stations regulated by city governments. Most of those official taxi stations are in busy traffic areas. For example, there are several taxi stations near JFK airport and also Penn Station in New York City. Because they are regulated by the city government, there can be a lineup area for taxis and a waiting space for passengers. However, even waiting at some official taxi stations may most likely outperform the route searching strategy, drivers may still have a chance to gain a lower profit in certain time slots. For instance, waiting at a taxi station in JFK airport at 6 PM maybe not a good idea on a work day, because it is very easy to find a passenger in Manhattan and drivers do not need to waste gas and time for waiting. Therefore, we need to prune out those low profit official virtual stations and treat the rest as the candidates of virtual stations when developing recommendation algorithms.

The second type of potential virtual stations are located in those areas with no official taxi station. It may be near a restaurant, an attraction or just a road segment near some residential building. Even though, the city government does not regulate those area as an official taxi station, taxi drivers can still find some places in the nearby road segments and wait for passengers in line. In this case, we can leverage the taxi drivers' historical driving trajectories to identify those virtual stations.

In general, we define virtual station V as a subset of road segments S with taxi

stands functions. Taxi drivers searching for customers in these areas will have to stop and wait in a line to pickup next customer. Typically, virtual stations are located within some high frequency regions, for example, business districts, station exits, hotels entrances, etc. Different from other segments where taxi drivers can pickup a passenger without waiting, virtual stations have some distinct properties including high demand, high profit, stop & wait and temporality that need to be considered independently.

- **High Customer Demand.** Similar to a hot taxi stand, virtual stations located in an area of high customer densities, such that customers usually stay in a line waiting for taxis. As a result, the customer pick-up in a virtual station is guaranteed.

- **High Pickup Earnings.** Long distance travelers are more willing to wait in line for a taxi. Thus, the earning associated with the pick-up events at a virtual station should be relatively high.

- **Stop & Waiting.** Even though there may be no taxi waiting in line at a virtual station for a certain time, it is just a special case and rarely happen. Most likely, taxi drivers need to stop at a virtual station and spend some time waiting in line. Therefore, the average waiting time associated with trips start from virtual station should be longer than regular trips.

- **Dynamic Properties.** The traffic patterns of modern cities keep changing during the day. Typically, the traffic flow at a virtual station varies over the

daytime. For instance, some road segments are identified as virtual stations during rush hours while being treated as a passing search segment at other times. Therefore, virtual stations are not static.

3.2.2 Route Searching vs Virtual Station Waiting Strategies

According to the distinct properties of these two kinds of road segments (searching segment and virtual station), taxi drivers usually take two different strategies for next-customer searching: route searching strategy and virtual station waiting strategy.

Route searching strategy

Route searching strategy recommends taxi drivers to look for next-customer by searching along a route. Since pick-up is not guaranteed by following a recommended route, the net profits of taxi drivers with route searching strategy will be weighted by the pick-up possibilities.

For each taxi driver driving through segment r, the net profit $g(r)$ consists of two components, namely *potential earning* $e(r)$ and *potential cost* $c(r)$. Specifically, we define the potential earnings $e(r)$ as the expected earnings of searching for a passenger in segment r, which is computed by

$$e_s(r) = \frac{\sum_{i=1}^{N_r} Fee(i;r)}{N_r} P_s(r), \tag{3.1}$$

where N_r is the number of pick-up events in segment r during a given time period, $Fee(i;r)$ is the earning from the i-th pick-up event and $P_s(r)$ is the pick-up possibility in segment r. Meanwhile, the potential cost $c_s(r)$ is defined as the cost of passing

through segment r without any pickups, i.e.,

$$c_s(r) = \big(1 - P_s(r)\big)\big(L(r) \cdot Gas + T(r) \cdot CF\big), \tag{3.2}$$

where $L(r)$ is the length of segment r, Gas is the price of gas per unit distance (e.g., per mile), $T(r)$ is the traveling time through segment r and CF is short for *Company Fee* that is the opportunity cost per unit time (e.g., per minute). Indeed, $T(r)$ is sensitive to the real-time traffic conditions. For example, a traffic jam will result in a high $T(r)$, and thus bring a high cost of $T(r) \cdot CF$. In this case, the segment will not be recommended by our model. In particular, because route searching taxi drivers will drive through the virtual stations without any pickups, the pickup possibility $P_s(r)$ equals to 0. Therefore, the potential earnings of a virtual station located in a searching route equals 0, and the potential cost of Equation 3.2 is then modified to $c_s(r) = L(r) \cdot Gas + T(r) \cdot CF$.

Furthermore, the net profit of segment r of our route searching strategy is computed by

$$g_s(r) = e_s(r) - c_s(r). \tag{3.3}$$

Based on the definitions above, we can further define the net profit for each searching route R. Specifically, given a route $R = (r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_M)$ starting from r_1, its total net profit can be computed by

$$G_s(R, r_1, M) = g_s(r_1) + \sum_{i=2}^{M} g_s(r_i) \prod_{j=1}^{i-1} \big(1 - P_s(r_j)\big). \tag{3.4}$$

Intuitively, the net profit of route R is the sum of the net profit of road segments $\{r_i\}$ contained in R, which is weighted by the possibility of not picking up any passenger in previous segments (i.e., r_1 to r_{i-1}).

As we already proved and showed in Chapter 2 Figure 2.2 with the possibility weights in net profit, the route searching taxi drivers will not consider the segments which are far away from her current location because the expected profit there is very low. To be more specific, we can define the average increasing rate of net profit as $\tau = \frac{<G(R,r_i,M+1)>-<G(R,r_i,M)>}{<G(R,r_i,M+1)>}$ to indicate the profit increase when increasing one more road segment in the route. We can observe that the increasing rate is less than 10% after increasing more than 5 road segments. Indeed, the average pick-up probability of each road segment in our experiments is always less than 0.1, therefore it is possible for us to set an upper bound Λ for route length M in Equation (3.4).

3.2.3 Parameter Estimation with Road Buffer

We also exploit the road buffer estimation for each road segment by using the same method as we already described in Chapter 2. Then we can calculated the total number of pick-up events in each road segment r and the probability of pick-up event in this road segment can be calculated by

$$P(r) = \frac{N_r^*}{N_r^0 + N_r^*}. \tag{3.5}$$

We obtain the earnings $Fee(i;r)$ in Equation 3.1, the road length $L(r)$ and real-time traveling time $T(r)$ as described in chapter 2. Therefore, the net profit of $g(r)$ can be calculated by Equation 3.3. The value $T(r)$, $g(r)$ and $P(r)$ of each road segment r can also be pre-stored in corresponding node of the road segment network.

Virtual Station Waiting Strategy

The virtual station waiting strategy recommends taxi drivers to drive directly to a virtual station and wait for next customer. Compared to the route searching strategy, taxi drivers will pay extra waiting time as penalty for a guaranteed pick-up. Therefore, given a route R starting from r_1 to a virtual station r_M, the potential net profit is calculated as

$$G_w(R, r_1, M) = e_w - c_w, \tag{3.6}$$

where e_w is the average earning of a pickup event in virtual station r_M, which can be computed in the similar way of Equation 3.1. In particular, the cost of searching VS r_M, i.e., c_w, should include the gas and the time penalty of driving to the specific virtual station, which can be computed by

$$c_w(R, r_1, M) = \sum_{i=1}^{M} \big(L(r_i) \cdot Gas + T_s(r_i) \cdot CF\big) + T_w(r_M) \cdot CF, \tag{3.7}$$

where the definition of Gas and CF are the same with Equation 3.2. $T_w(r_M)$ is the average waiting time in virtual station r_M.

3.2.4 Problem Definition

After introducing the difference between the route searching strategy and the virtual station waiting strategy, here we propose a Maximum Net Profit (MNP) recommendation problem to help taxi drivers decide the strategy (i.e., route searching or VS waiting) and the optimal route, which can help to maximize their net profit per unit time. Specifically, the MNP problem is formally defined as follows.

Definition 9 (Problem Statement) *Given the current location $LCab \in r_1$ of a*

taxi driver, a set the nearby virtual stations $V = \{vs_1, vs_2, ...vs_n\}$*, a fixed cruising length* M*, and a set of route candidates* $\mathcal{R}$*, the MNP recommendation problem is to recommend a strategy* i *and a route* $R^* \in \mathcal{R}$*, which has the maximum net profit.*

$$R^* = \underset{i \in \{s,w\}, R \in R}{\operatorname{argmax}} G_i(R, r, M). \tag{3.8}$$

Different from other existing recommender systems for taxi drivers, which mainly focus on extracting energy-efficient transportation patterns based on traveling time/length and recommending a sequence of potential pick-up points for taxi drivers (Yuan et al., 2010, 2011; Ge, Xiong, Tuzhilin, et al., 2010), the MNP recommendation focuses on providing an entire driving route with maximum net profits for a taxi driver or discovering a nearby virtual station with high earnings and low opportunity cost. Along this line, there are two major challenges for solving the MNP recommendation problem. First, how to calculate the net profit of each segment r from the historical taxi transaction data. Second, how to efficiently search an optimal route from the complex directed-cyclic route segmentation network. In the following section, we will introduce our solutions for the above two challenges, respectively.

3.3 MNP Recommendation with Business Effective Strategies

In this section, we formally define the problem of enhancing recommender system for taxi drivers with business effective strategies and introduce our joint learning algorithm.

3.3.1 Maximum Net Profit Recommendation Strategy for Joint Learning Algorithm

In this subsection, we introduce how to implement NMP recommendation with respect to different strategies. Generally, there are three steps to recommend the best business effective strategies to taxi drivers. First, we need to get the net profit from route searching strategy. Second, we calculate the net profit from virtual station waiting strategy. Third, we compare those two profits and choose the strategy that can maximize our net profit. In the end, we can integrate the above two strategies into a joint learning framework.

Recursive Algorithm for Route Searching Strategy

As we mentioned in Chapter 2, by observing the form of the net profit of routes, we can re-write the Equation (3.4) as follows.

$$G(R, r_1, M) = g(r_1) + \big(1 - P(r_1)\big)G(R - r_1, r_2, M - 1), \tag{3.9}$$

where $R = (r_1 \to r_2 \to \cdots \to r_M)$. Indeed, the special form of total net profit can be realized by a recursion algorithm. To this end, for each road segment r_1, we can denote all the route candidates starting from r_1 as a *recursion tree* structure. Specifically, the recursion tree of a road segment can be defined as same as Definition 5 in chapter 2.

However, the computational cost for recursion increases significantly as M becomes larger. According to the discussion in the end of Section 3.2.2, we can set an upper bond Λ for M, since the average increasing rate of the net profit is very low after $M > 5$. Therefore, we set $\Lambda = 5$ in our experiments.

MNP Route Recommendation with Virtual Station Waiting Strategy

After obtaining the maximum net profit from routing strategy, we need to calculate the maximum net profit from virtual station waiting strategy. We have a set of all N virtual stations' longitude and latitude, the average profit and the waiting time for each virtual station from historical data and the taxi drivers' starting road segment. Then we calculate the cost of gas and company fee by using the shortest driving route from the taxi driver's current location to this virtual station. We also take the waiting time penalty in this virtual station into consideration. In the end, we use the profit minus the cost for those virtual stations and get the final net profit from each virtual station of different starting points. The largest value is the the maximum net profit from our virtual station waiting strategy and this virtual station is the best station that we need to recommend to taxi drivers with the waiting strategy.

3.3.2 Load Unbalance Problem

Based on the above strategy, our recommender system can recommend an optimal MNP route for a single taxi driver. However, in real life, an ideal recommender system must be capable of recommending multiple taxi drivers in the same area simultaneously. To this end, in this subsection, instead of Top-K route recommendation strategies in chapter 2, we further address this problem by introducing a novel dynamic MNP recommendation load lalance strategy.

Dynamic MNP Strategy

Although the top-K route recommendation can alleviate the load unbalance problem, it still has limited capability in heavy traffic area. Therefore, we further introduce

another method, i.e., dynamic MNP recommendation strategy, for solving this problem.

In real life, as long as the road segment is not a virtual station, the pick up probability keeps changing after each pick-up event in this road segment and in its nearby road segments. Let us recall that in MNP route searching strategy, we use the average pick-up probability $P_s(r)$ to estimate the potential net profit of road segment r. Since $P_s(r)$ is a static value learned from historical pick-up events, the calculation for this strategy is very efficient. However, if we keep doing recommendation by using this static probability, we may accidentally send all the taxi drivers to the same road segment for searching next customer. To address this problem, we propose to leverage another dynamic pick-up probability $P^0_{sd}(r,t)$ to replace $P_s(r)$. Specifically, we separate a time period to several small time slots, each time slot may be a couple minutes. Then we represent the current dynamic pick-up probability $P^0_{sd}(r,t)$ as an integration of the pick-up probability in this road segment and also in its nearby road segments in time slot $t-1$. Based on the historical data, we can run regression functions and analyze the relationship between the target road segment pick-up probability $P^0_{sd}(r,t)$ at time t and the previous pick-up probability $P^k(r,t-1)$ ($k \geqslant 0$) for the k-th directly conjuncted road segments of target road segment.

By using the above regression functions, we can rewrite Equation (3.4) as following:

$$G_s(R, r_1, t, M) = g_s(r_1, t) + \sum_{i=2}^{M} g_s(r_i, t) \prod_{j=1}^{i-1} \big(1 - P^0_{sd}(r_j, t)\big). \tag{3.10}$$

Intuitively, by using the above objective function, our model is more realistic and

it can solve the load unbalance problem with a context-aware manner. Particularly, in real applications, the DEN based top-K route recommendation and the dynamic MNP recommendation can be implemented as complementary strategies. For example, the system can first use dynamic strategies to generate context-aware route candidates, and then use the DEN method to conduct top-K route recommendation.

3.4 Experiments

In this section, we evaluate the performances of enhancing recommender systems for taxi drivers with effective strategies by using real-word taxi trajectory data in Beijing and San Francisco Bay area.

3.4.1 The Experimental Data

Taxi GPS Traces. In the experiments, we used two sets of real-world taxi GPS traces collected in San Francisco Bay area and Beijing. The mobility traces are the records of cabs' driving states in consecutive time, with each represented as a tuple, (*latitude, longitude, fare identifier, time stamp*). We use 536 taxi drivers data in San Francisco and obtained 89,897 pick-up and drop-off activities in a 30-day period. Moreover, we calculated the taxi fare related to each pick-up and drop-off pair by using a taxi fare calculator online (*San Francisco Taxi Fare Calculator*, n.d.). For the data set in Beijing, we chose 3,258 most active taxi drivers and selected their driving trajectories in a 7-day peroid. In the end, we obtained 1,345,560 pick-up and drop-off activities in total for Beijing.

Figure 3.1 shows an example of one hundred taxi drivers' pick-up points in a 30-day peroid in San Francisco Bay Area, with each red point representing one pick-up

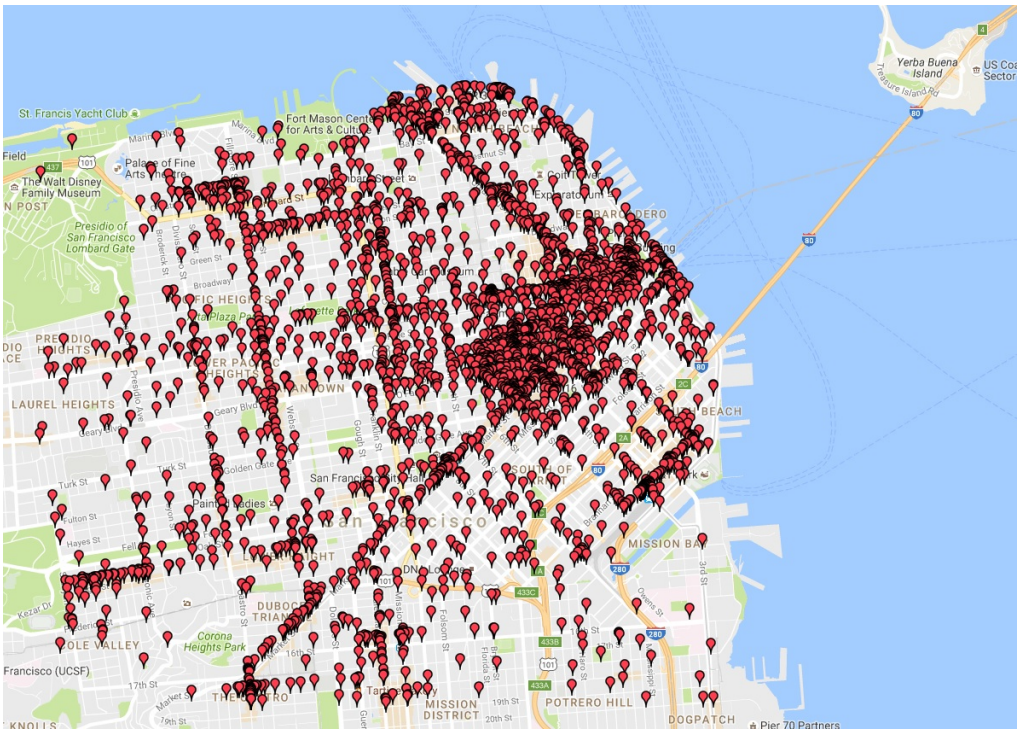


Figure 3.1. Pick-up Events in San Francisco Bay Area

activity. Figure 3.2 is the heat map illustration of corresponding pick-up probabilities. Here, different color and area of circles represent different pick-up probabilities. As we mentioned in Chapter 2, this map shows there are lots of pick-up activities around Market Street of San Francisco, which is a very busy street with lots of shopping places and museums. In addition, other pick-up hot spots include Fisherman's Wharf, Divisadero St, Cathedral Hill and Western Addition.

Beijing Road Network Data. The high quality road network data of Beijing is provided by (*Beijing Road Network Data*, n.d.), which includes two parts. The first part includes vertices data, including the ID numbers, the starting points and the ending points' geographic data for each road segment. The second document includes the edges data, where if two road segments are connected to each other,

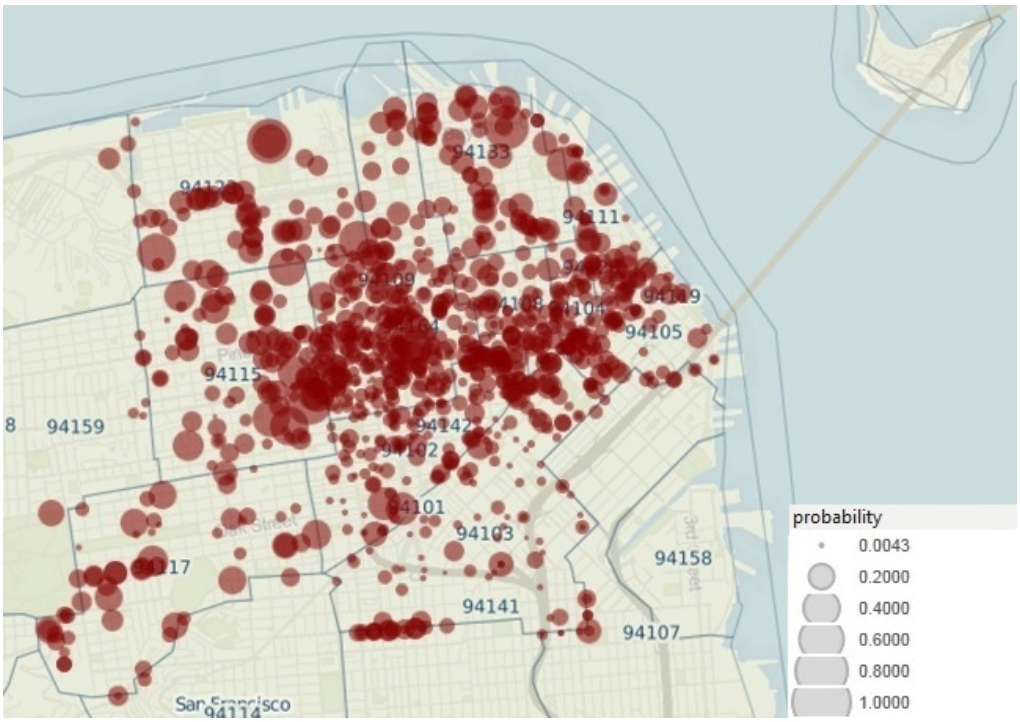


Figure 3.2. Pick-up Heat Map in San Francisco Bay Area

we kept their road segment IDs and gave an edge ID to this connection. In our experiments, we mainly focused on the road segments inside the third ring road of Beijing due to the urban area distribution. In the end, we obtained 18,301 road segments in Beijing.

San Francisco Road Network Data. Because the quality of existing road networks in San Francisco is not sufficient. We built the road network data set of San Francisco by using Google API as described in Chapter 2.

3.4.2 Data Preprocessing

Beijing Data Set. The traffic pattern in Beijing has temporal differences during the day. Those differences will highly affect the associated pick-up probability, the net profit and the average driving time for each road segment. Therefore, we investigated

taxi drivers' behaviors in three different time slots, 8-11 am, 13-15 pm and 17-20 pm, respectively. The above three time slots include morning, evening traffic hours and also a non traffic time interval. Moreover, the taxi's driving speed in the same road segment for different time slots are also different. Therefore, we used the average historical driving data of each road segment in different time slot as the average driving speed of corresponding road segment. Then, we matched the taxi trajectories in Beijing taxi data set into the Beijing road network, and calculated the corresponding pick-up probabilities and the net profits. In the end, we obtained the coordinates of the starting and the ending points for each road segment in Beijing, along with the pick-up probability, the net profit and the average driving time in this road segment for different time slots.

San Francisco Data Set. By matching the pick-up coordinates of road network data with the trajectories, we finally obtained 87,688 valid pick-up activities which can be located in the road segments, therefore the two data sets in San Francisco are combined together with each pick-up point mapped to the corresponding road segment. To implement the proposed algorithm, we also need to calculate the pick-up probability and the net profit for each road segment. In the end, we got the coordinates of the starting and the ending points for each road segment, along with the pick-up probability, the net profit and the average driving time in this road segment. Note that the average driving time is estimated as the distance of each road segment divided by the average driving speed in San Francisco. Different from Beijing data set, we did not split the data into different time slots due to the quality of data collection.

Figure 3.3. Three types of status identifier sequence

Virtual Station. The main challenge for our research is how to find those virtual stations defined in section 3.2. Indeed, virtual stations should have four properties, which are high customer demand, high pick up earnings, long waiting time and dynamic properties. Therefore, it is necessary for us to look at the taxis' trajectory data and prune out those locations, which can satisfy the properties of virtual stations. Indeed, the taxi trajectories data includes 4 elements for each record, i.e., (*latitude, longitude, status identifier, time stamp*). Note that, the status identifier are the current status of occupation. If there are customers in the cab by that time, the identifier equals to 1. Otherwise, this value should equals to 0. Specifically, there are three types of status identifier sequence in one small road segment. The first type is all 0s, which means a passing by event of a vacant taxi. Because there should be high

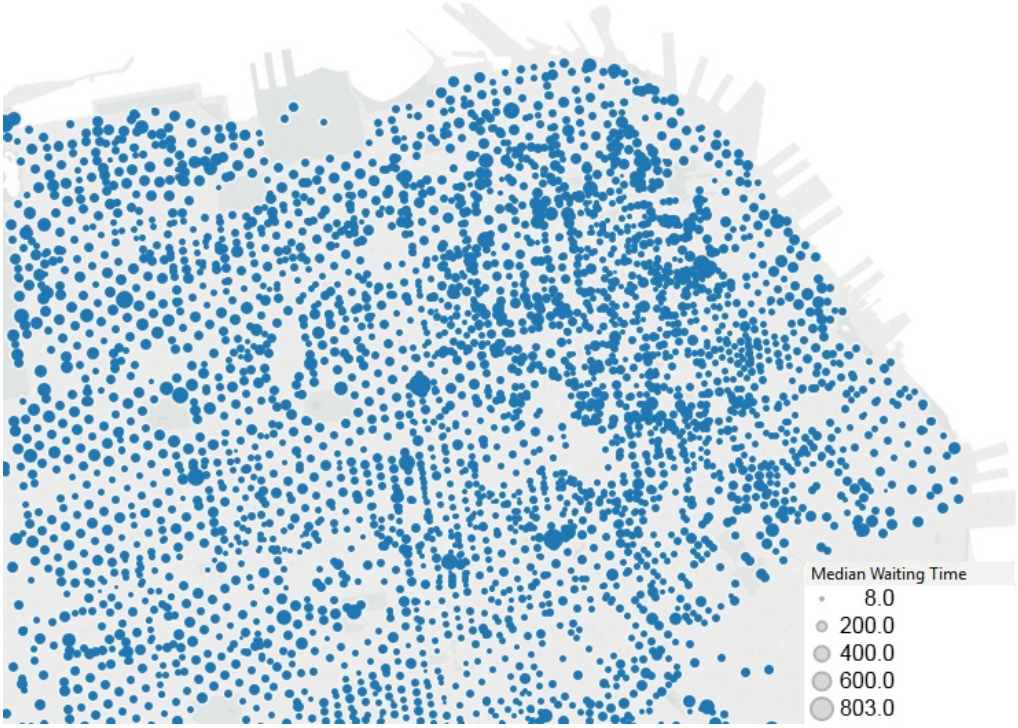


Figure 3.4. Median waiting time in road segments in San Francisco Bay Area

customer demands near a virtual station, those passing by events can not contribute to a virtual station. The second type is couple 0s followed by a 1. This is a pick up event because the status identifier changed from vacant to occupied. However, because the number of 0s are not large, it is more likely that the taxi did not wait in a line before they picking up a customer. Those type of events also cannot contribute to a virtual station. Instead of that, those patterns are more like a pick up event by using routing strategy. The third type of status identifier is lots of 0s followed by a 1. Lots of 0s means the cab wait in line for a relatively long time and the 1 means a pick up event. This accords with the property of virtual station. Therefore, if there are lots of type III status identifier sequence around a point in a time slot, this point is a virtual station. Some examples of status sequence are showed in Figure 3.3.

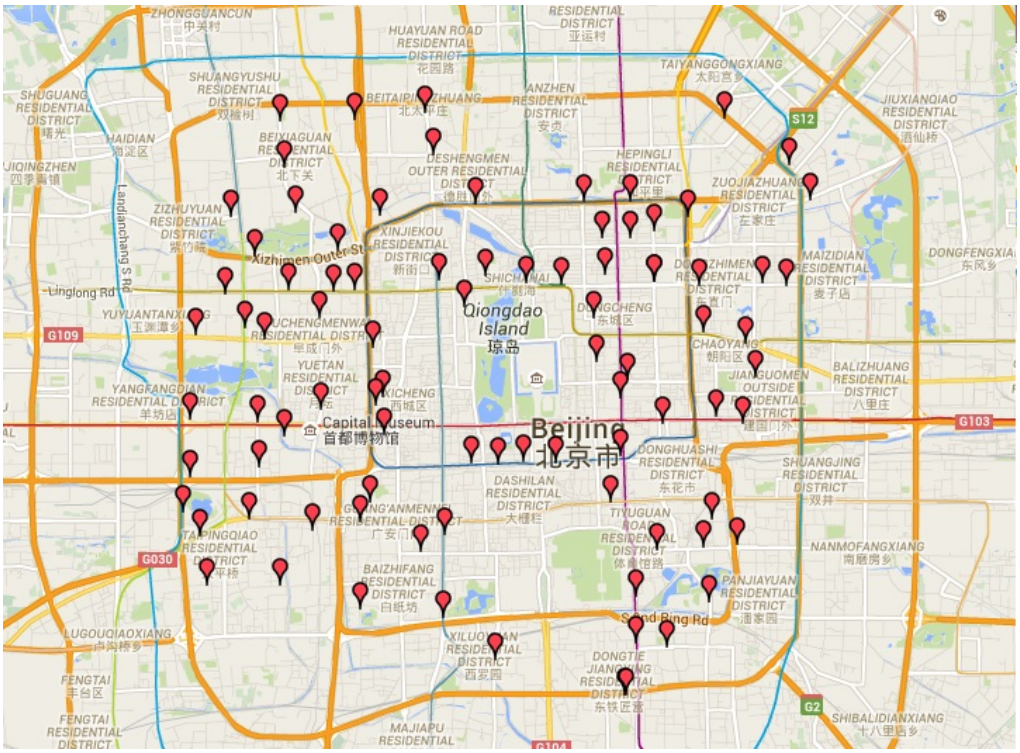


Figure 3.5. Virtual Station

In order to find out those virtual stations in different time slots. We should look at taxi driver's behaviors first. Here, we choose 2 minutes as the threshold. If a taxi stayed in one location for more than 2 minutes with no traffic jam in that road segment, we identify it as a waiting event. We can recognize that there are more passing events than waiting events, however, some drivers will choose waiting strategy if it is necessary. We further calculated the median waiting time for each road segment in San Francisco Bay Area and plotted it in Figure 3.4. The bigger the blue dots indicate longer median waiting time in this road segment. After further looked into those road segments, we found out that the roads with large median waiting time usually have big super markets, hotels or universities. However, because of the terrain of San Francisco, lots of road segments are up and down. It is not very convenience

for taxi drivers to park and wait in a line. Therefore, the waiting behavior is not very obvious. We used the same method to find the waiting events in Beijing and the road segments with potential virtual stations. Since the terrain of Beijing is flat, it is much more easier for taxi drivers to choose waiting strategy. After investigating the properties of virtual station and the corresponding taxi drivers' trajectories, we find around 90 virtual stations inside the third ring road of Beijing as showed in Figure 3.5. Most of those virtual stations are near hospitals, museums, hotels and sports centers. Those places are consistence with our prediction on the potential virtual stations.

3.4.3 The Active Region of Visual Station

Waiting in the line of a virtual station can bring more profit in unit time than cruising in some case, especially when the current location of a taxi is already near a virtual station. However, if the current location is far away from any virtual station, the taxi driver will waste lots of gas and time to drive to a virtual station. In those cases, the waiting strategy cannot always win the route searching strategy. In the real world, we should suggest taxi drivers to employ waiting strategy only if their current locations are within in a certain distance of a virtual station. We call these locations the active region of virtual station. Moreover, depending on different travel patterns in different time slots in a day, those active region of each virtual station should be different.

In this research, we calculated the profit $G_w(R, r_1, M)$ from waiting strategy associated to each road segment near a virtual station and compare this profit to the profit $G_s(R, r_1, M)$ of route searching strategy. If the profit from waiting is higher than the profit from route searching strategy, then we say this road segment is in the

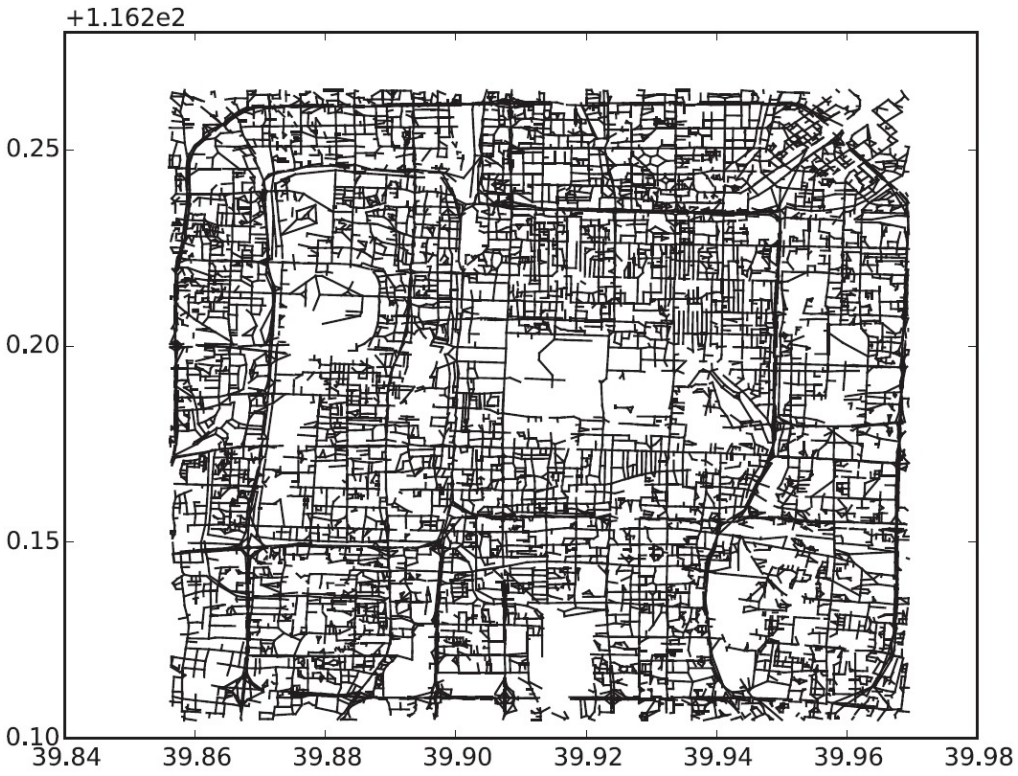


Figure 3.6. Base Map

active region of the virtual station. Figure 3.6 is a base map of Beijing road network in third ring road and Figure 3.7, 3.8, 3.9 shows the different active ranges of virtual stations in three different time slots in Beijing, where active regions of different virtual stations are showed in different color. From the results, we can find that both virtual stations and their active region changed from time to time.

3.4.4 Recommendation for Inexperienced Taxi Drivers

Given one specific location, our proposed joint learning algorithm can recommend several routes with high expected profits for drivers. This algorithm is especially applicable for inexperienced drivers, since they are usually lack of knowledge about the road map and the local driving routes/vitual stations that can make high profits.

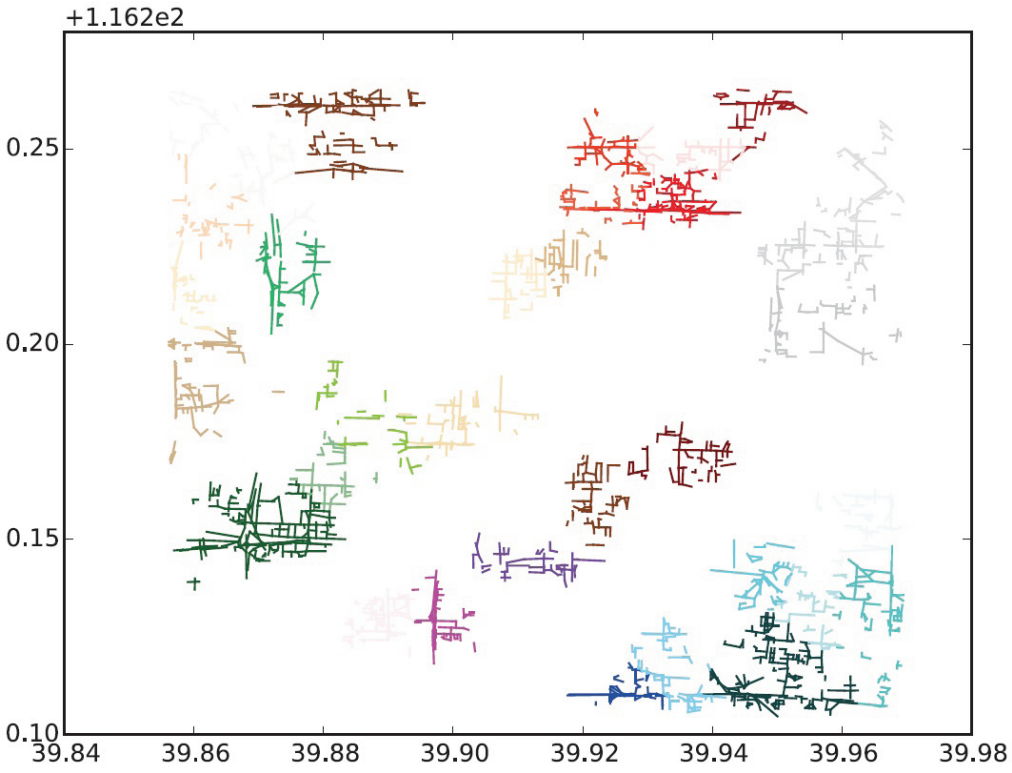


Figure 3.7. Virtual Stations Active Regions from 8 to 11

To validate the effectiveness of the proposed algorithm, we firstly divide all the drivers into two categories based on their average net profits. The top 10% drivers in the data set are treated as "experienced" drivers, while the others are "inexperienced". Therefore, the driving routes of experienced drivers are used as training set and we recommend driving routes for the inexperienced drivers.

In route searching strategy, we define driver's event $\mathbf{e}$ as a consecutive sequence of "roam $\rightarrow$ pick up $\rightarrow$ drop off". By extracting the pick-up and drop-off activities of each driver, we can reconstruct each event. For each driver, we define the location where the driver starts to search for potential pick-ups as l_0, and after roaming in Δt time, the driver picks up a passenger at location l_1 and drive for $\Delta t'$ and drop

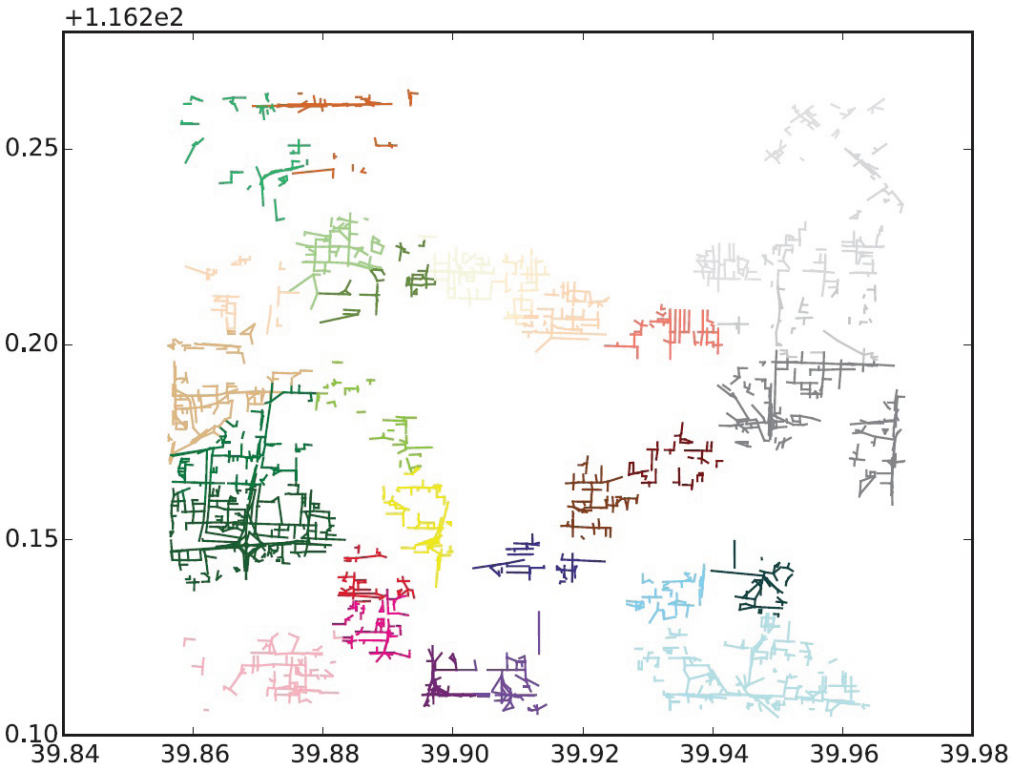


Figure 3.8. Virtual Stations Active Regions from 13 to 15

off at l_2. Let $r_{i,j}$ denotes the road segment between location l_i and l_j, then event e can be represented with $(r_{0,1}, \Delta t, r_{1,2}, \Delta t')$, and the unit time profit of the event can be calculated as $pr_e = \frac{pr_{12}}{\Delta t + \Delta t'}$, where pr_{12} is the total profit during the trip. Thus, the proposed algorithm starts with location l' which is neareset to l_0, and return a sequence of recommended potential pick-up points and road segments. The performance of the recommended driving route is measured by the average net profit per unit time $pr_s = \frac{\sum p_e}{|\mathbf{e}|}$.

We then compare the average net profit per unit time pr_s from routing strategy to the average net profit per unit time pr_w from the waiting strategy. In order to calculate pr_w, we need to construct a waiting event. For each driver's event $\mathbf{e}$, there

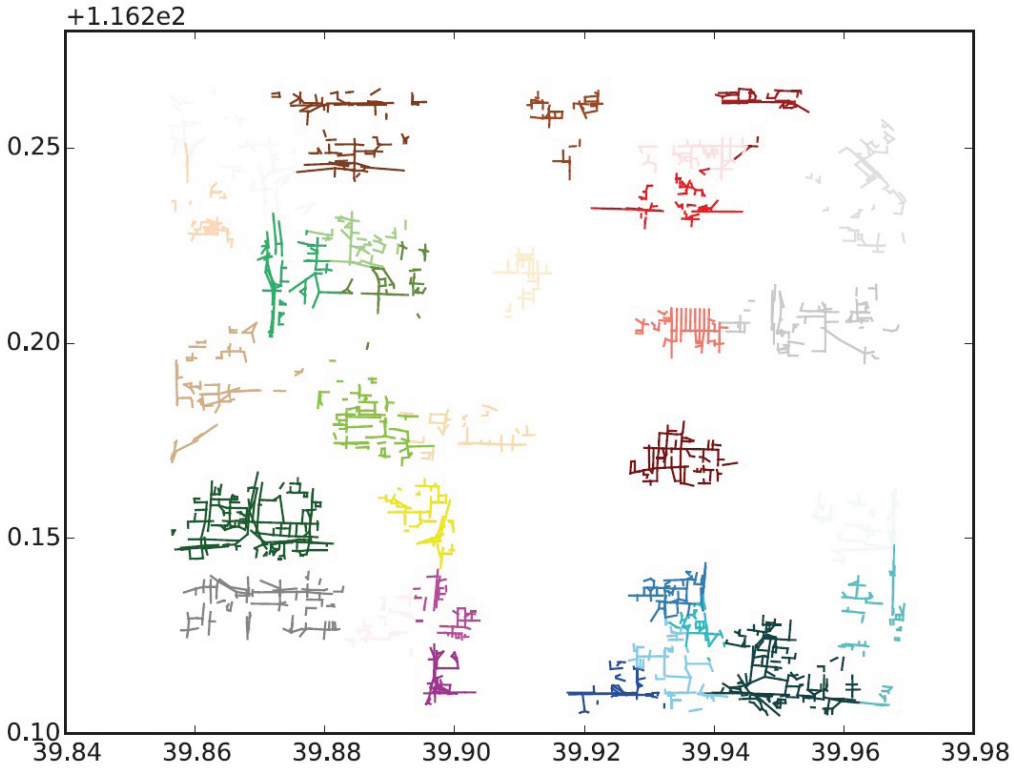


Figure 3.9. Virtual Stations Active Regions from 17 to 20

is a consecutive sequence of "driving to virtual station → waiting in virtual station → pick up → drop off". For each driver, we define the location where the driver starts driving to a virtual station as l_0. Then the drive spends Δt_d time to arrive a virtual station and spend Δt_w to wait for a customer in line. Then the driver picks up passengers at this virtual station **VS** and drive for $\Delta t'$ and drop off at l_2. Event **e** can be represented with $(r_{0,VS}, \Delta t_d, \Delta t_w, r_{VS,2}, \Delta t')$, and the unit time profit of the event can be calculated as $p_e = \frac{pr_{VS,2}}{\Delta t_d + \Delta t_w + \Delta t'}$. Similarly, we have $pr_w = \frac{\sum p_e}{|\mathbf{e}|}$.

If $pr_s > pr_w$, our recommender system should recommend route searching strategy with an entire driving route to the inexperienced taxi drivers. Otherwise, the recommender system should recommend a virtual station that the driver can wait for

Table 3.1. Net Profits per Unit Time

		Recommender System	Inexperienced Drivers
8-11am	Mean	0.66179	0.36957
13-15pm	Mean	0.65899	0.37571
17-20pm	Mean	0.63606	0.36778

next customer in line. Table 3.1 shows the statistical results of net profit between recommended routes/virtual stations and the choices of inexperienced drivers. Note that, due to the data quality, here we only conducted experiments based on Beijing data set. From the results, we can observe that the average net profits per unit time of recommendations clearly outperform the real profit of the inexperienced drivers, which validates the effectiveness of our recommendation strategies.

To further investigate the performance of the recommender system, we also study the difference of net profit per unit time between the recommended strategies and the drivers' real net profit for each event, i.e. $p_r - p_e$. As shown in Figure 3.10, the X axis is the difference between the profits of the recommended results and the inexperienced taxi drivers' profits. We can see that most of dot points are positioned to the right of $X = 0$, meaning that the profits of our recommended strategies outperform the profits of the strategies chosen by the inexperienced drivers.

3.4.5 Empirical Studies on Recommendations

Here, we provide some empirically studies to validate the effectiveness of our recommendation system.

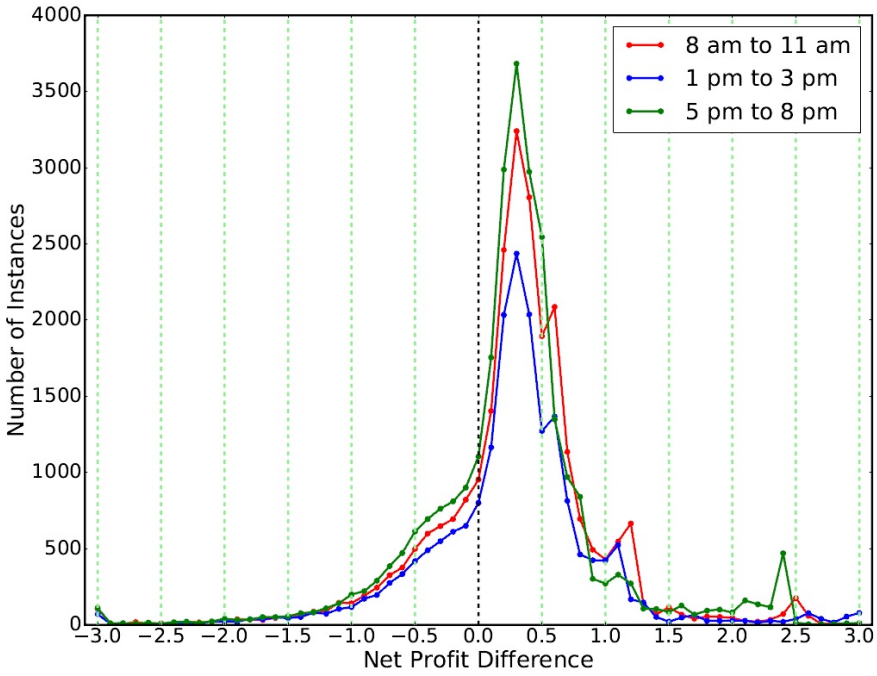


Figure 3.10. Profit Difference. X axis is the Net Profit Difference between our strategy and taxi drivers' traditional strategies ranked below top 10%, Y axis is the number of events

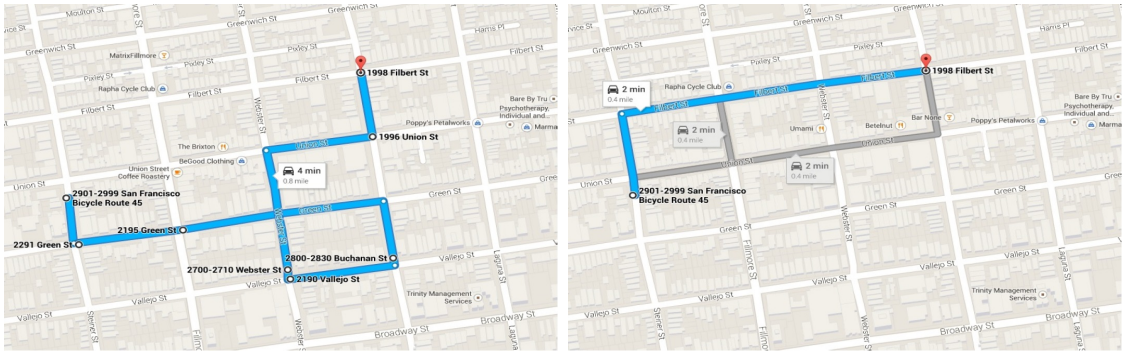


Figure 3.11. Enhancing Recommender System Case Study (a)

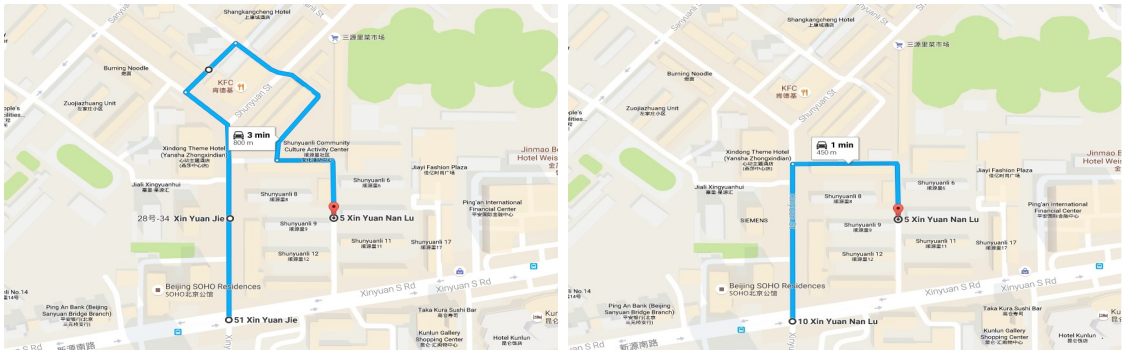


Figure 3.12. Enhancing Recommender System Case Study (b)

Case Study on Driving Route Recommendation

Indeed, different from previous studies that focus on recommending a sequence of pick-up points and letting the driver decide how to get to those points, our recommender system is capable of providing an entire driving route for taxi drivers. Therefore, here we first provide a case study on route searching strategy. Specifically, we show two examples of route searching strategy by our approach and compare it with the suggested route by the Google map. In Figure 3.11 and Figure 3.12, we plot the optimal driving route suggested by our recommender system at a randomly selected initial location of the target cab in San Francisco and Beijing separatively.

We also assumed that the driver's expected cruising length is 5, and after every 5 road segments, the system will use the current location as the new starting point for search and restart the recommendation process in this case study. The total searching time of our recommender system equals to the real searching time of the taxi drivers. In those Figures, the left figures are the driving routes recommended by the MNP recommender system and the right figures are the routes suggested by the Google Map based on the shortest driving distance. In both cases, our recommender system attends to suggest the drive to cruising around a neighborhood. Meanwhile, google map suggests to choose the shortest driving distance. However, the driving routes suggested by the Google map cannot maximize taxi drivers' net profit.

Recently, most recommender system can only suggest a sequence of hot spots to taxi drivers. There is no such recommendation system that can suggest an entire driving route. If taxi drivers do not know how to drive to the nearest hot spot, he or she has to follow the driving route provided by the Google map. However, both the pick-up probability and the potential net profit may be very low along those routes. The drivers have a high probability of losing money until they reach the next hot spot. Our recommender system can improve the potential net profits for taxi drivers compared to the routes suggested by the Google map.

Case Study on Top-K recommendation

In Section 3.2, we introduced a top-K driving strategy for addressing the overloaded problem. In figure 2.4.2 in Chapter 2, we demonstrate the top-K driving routes starting from the same location in San Francisco Bay Area, where K equals to 4 in

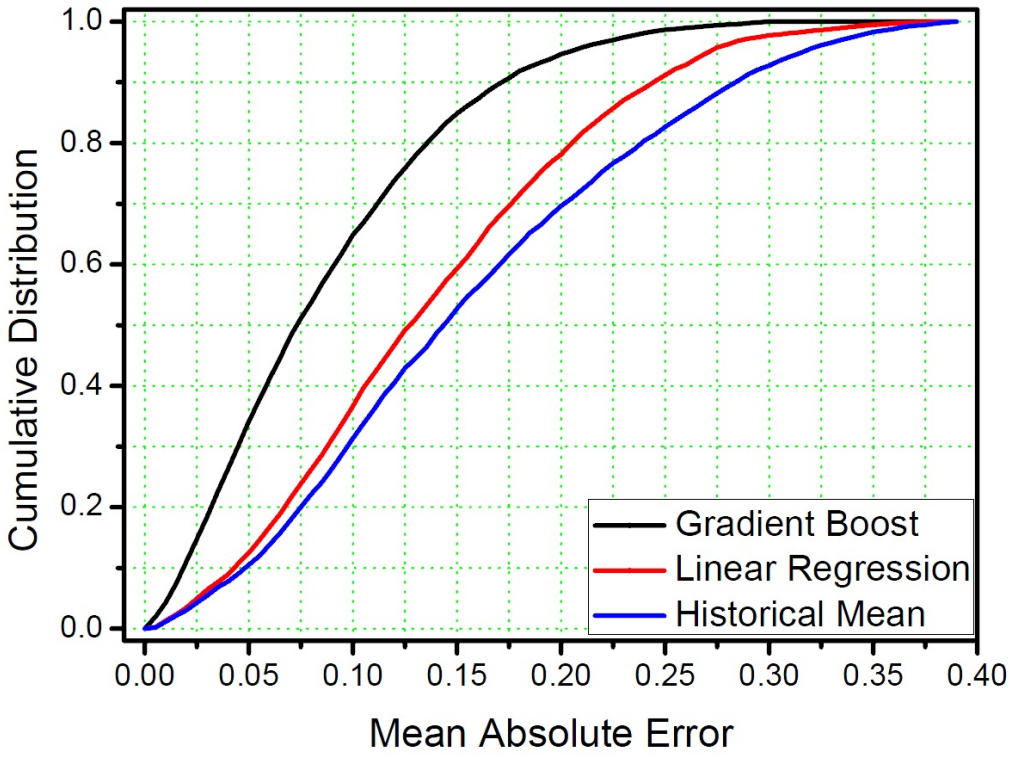


Figure 3.13. The performance of regression models

this case. The figure shows that each route has different driving directions and the correlations between those driving distances are very small. Therefore, the top-K strategy can improve the performance of our recommender system.

Case Study on Dynamic MNP strategy

The experiments showed that the joint learning algorithm recommender system could help inexperienced taxi drivers find better business effective strategies so as to maximize their potential profits. However, as we already discussed in Section 3.3, using the average pick-up probability for a road segment in a certain time slot may cause overload problem. In order to find the relationship between the pick-up probability of target road segment and the pick-up probabilities of its first, second, third con-

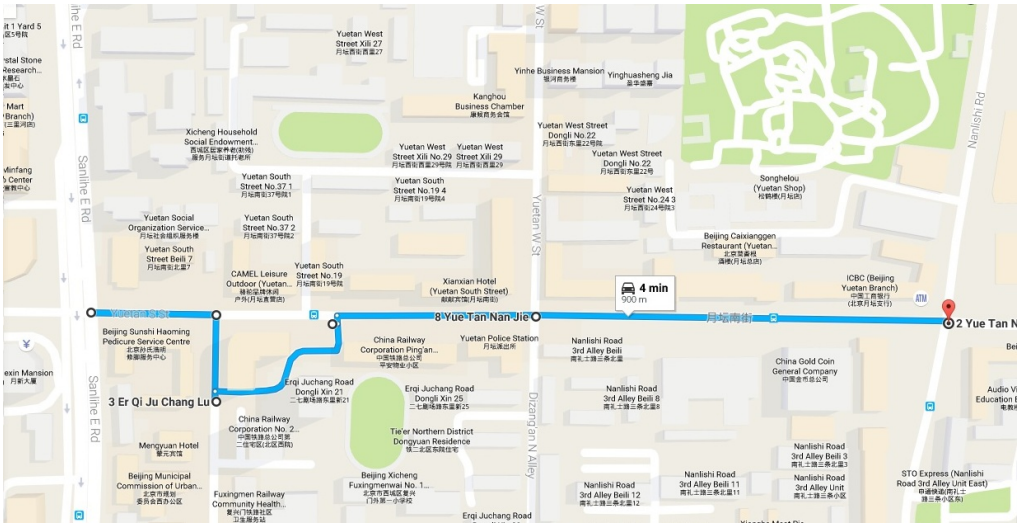


Figure 3.14. Driving route suggested with static pick-up probability

Figure 3.15. Driving route suggested with dynamic pick-up probability in the 4th time slot

Figure 3.16. Driving route suggested with dynamic pick up probability in the 14th time slot

juncted road segments and also the pick-up probability of this target road segment in the previous time slots, we conduct experiments to compare several regression models, including **Linear Regression**, **Gradient Boosted Tree**, and **Historical Mean** (i.e., using the average pick-up probability in each time period as prediction value without considering the pick-up event in nearby neighborhood). Specifically, we use the data in Beijing from 7:30am to 11:30am in one day as our experimental data and separated this 4-hour time period to 24 small time slots, i.e., each time slot contains 10 minutes. Then we calculated the pick-up probability in each target road segment in time slot t, $t-1$ and the average pick-up probabilities of its first, second and third directly conjuncted road segments in time slot $t-1$. Note that, we pruned out those road segments with all pick-up probabilities equals to 0. In the experiments, we randomly select 80 percent data as the training set and test our models with another 20 percent data. Figure 3.13 shows the performance of above

three models, where we can see that the gradient boosted tree regression outperforms other baselines.

Moreover, we demonstrate a case study between 7:30am and 11:30am in Beijing by using gradient boosted tree regression with a random start searching point. Figure 3.14 3.15 3.16 show the driving routes for static pick-up probability and dynamic pick-up probabilities in the 4th and 14th time slots. We can see that the pick-up probability in a long time period are not always the same and the changing of this probability may also change the suggested driving routes.

3.5 Concluding Remarks

In this chapter, we proposed an enhancing recommender system with business effective strategies for taxi drivers to maximize their profits by providing profitable driving strategies. Specifically, we first designed a joint learning framework to evaluate the potential profits of different strategies. Those strategies could be routing around the neighborhood by following a route with the maximum net profit or waiting at a virtual station with the highest ratio of the potential profit over the waiting time depending on the given time period and the location of the driver. Then, by mining the historical taxi GPS traces, we developed a recursive algorithm for efficiently generating optimal driving route for route searching recommendation strategy. As a result, we can use the net profit objective function to rank each candidate route and make recommendations to taxi drivers in a cost-effective way. Furthermore, we also provided two strategies to make a better load balance for the recommendations happening at the same location. An unique perspective of our recommender system is that it can recommend an entire

driving route instead of only recommending a sequence of discrete pick-up points. Also, the drivers are able to maximize their profits within the fixed time period by following the recommended driving strategies. Finally, we conducted extensive experiments on real-world data sets collected from Beijing and the San Francisco Bay area, and the experimental results clearly validated the effectiveness of the proposed recommender system.

CHAPTER 4

CONCLUSIONS AND FUTURE WORK

In this chapter, we conclude the dissertation with an overall review and a general discussion about future work.

4.1 Review of Disseration

In this dissertation, we addressed the differences between mobile recommender systems and traditional recommender systems, and developed mobile recommender systems with business effective strategies. In the following, we briefly summarize the contributions we made on the mobile recommendation domains:

- In CHAPTER 2, we proposed a Cost-Effective Recommender System for Taxi Drivers based on the analysis of taxi GPS traces. This recommender system can provide an entire driving route that maximizes taxi drivers' net profits instead of recommend a sequence of pick-up points. We calculated the weight for each small road segment and use a novel recursive tree algorithm to generate candidate driving route efficiently. Then we developed a MNP objective function and evaluate the profit for each candidate route. Finally, we exploit top-K recommendation strategy to solve the load unbalance problem.

- In CHAPTER 3, we further developed an Enhancing Recommender System for Taxi Drivers with Business Effective. In this work, we enhanced the previous

Cost Effective Recommender System with business effective strategies, such as route searching strategy and virtual station waiting strategy. We firstly defined several virtual stations based on their special properties in different time slots. Then we calculated the potential net profit in unit time for each virtual station based on historical data. We also designed a joint learning framework with a special net profit objective function to evaluate the potential profits for both strategies and recommend the one can provide maximum potential net profit. Instead of Top K recommendation strategy, we provided a dynamic MNP strategy to make better load balance for recommendations happening at the same location.

4.2 Future Research Directions

With the success of developing mobile recommender systems for taxi drivers with business effective strategies, it is worthwhile to extend its usage in other domains. Indeed, there are lots of existing works in mobile recommender system (J. Liu et al., 2015; J. Liu, Sun, Chen, & Xiong, 2016), POIs (Sun et al., 2015; Yao, Fu, Liu, Liu, & Xiong, 2016; Y. Liu, Liu, Liu, Qu, & Xiong, 2016) and urban computing (Niu, Liu, Fu, Liu, & Lang, 2016; Y. Liu et al., 2014; J. Liu et al., 2017). In general, my long-term research objective is to make efforts to connect those areas such as data mining, mobile recommender system, and urban computing in a coherent way.

- There has been a huge amount of taxi trajectory, public transportation and road network data accumulated. My current research is to establish a recommender system for taxi drivers by using taxi trajectory and road network data. It

will be more interesting to extend this idea to combine public transportation data and urban computing for smart transportation management. Indeed, by observing daily taxi and public transportation data, I can identify the locations with higher chance of severe traffic congestion. Then, it is possible to improve the road networks based on the identified time and location sensitive patterns and provide suggestions to local governments by developing effective big data solutions for urban planning and intelligent transportation management.

- In addition to taxi trajectory data, there are also a rich supply of points of interest and real estate data nowadays. It would be interesting to combine the use of such data together with human mobility data and public transportation data to discover the function areas of cities, such as working zones, living zones and recreation zones. It is also possible to build a real estate recommender system to meet the commuting needs of residents.

- Finally, the taxi route optimization problem is a dynamic routing problem with stochastic outcomes. A driver can abort a recommended route early if a customer is picked up at any of the road segments. This type of sequential decision making problems with stochastic outcomes can be formulated as finite-time Markov Decision Processes (MDPs) and can be solved by using the backward induction algorithm. Equivalently, a linear programming model can also be constructed to solve the aforementioned MDP model. I believe that by implying the new model we are capable to design an efficient and scalable algorithm and solve the recommendation problem within the shortest possible time.

BIBLIOGRAPHY

Abowd, G., Atkeson, C., & al et. (1997). Cyber-guide: A mobile context-aware tour guide. *Wireless Networks*, *3(5)*, 421-433.

Adomavicius, G., & Tuzhilin, A. (2005). Towards the next generation of recommender systems: A survey of the state-of-the art and possible extensions. *TKDE*.

Averjanova, O., Ricci, F., & Nguyen, Q. N. (2008). Map-based interaction with a conversational mobile recommender system. In *The 2nd int'l conf on mobile ubiquitous computing, systems, services and technologies.*

Beijing road network data. (n.d.). `http://www.datatang.com/data/45422`.

Bell, R. M., & Koren, Y. (2007). Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Data mining, 2007. icdm 2007. seventh ieee international conference on* (pp. 43–52).

Cena, F., Console, L., & al et. (2006). Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. *AI Communications*, *19(4)*, 369-384.

Chakrabarti, S., Ester, M., Fayyad, U., Gehrke, J., Han, J., Morishita, S., et al. (2006). Data mining curriculum: A proposal (version 1.0). *Intensive Working Group of ACM SIGKDD Curriculum Committee*, 140.

Cheverst, K., Davies, N., & al et. (2000). Developing a context-aware electronic tourist guide: some issues and experiences. In *the sigchi conference on human factors in computing systems* (p. 17-24).

Deshpande, M., & Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, *22*(1), 143–177.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, *17*(3), 37.

Ge, Y., Liu, C., Xiong, H., & Chen, J. (2011). A taxi business intelligence system. In *Proceedings of the 17th acm sigkdd international conference on knowledge discovery and data mining* (pp. 735–738).

Ge, Y., Xiong, H., Liu, C., & Zhou, Z.-H. (2011). A taxi driving fraud detection system. In *Data mining (icdm), 2011 ieee 11th international conference on* (pp. 181–190).

Ge, Y., Xiong, H., Tuzhilin, A., Xiao, K., Gruteser, M., & Pazzani, M. (2010). An energy-efficient mobile recommender system. In *Proceedings of the 16th acm sigkdd international conference on knowledge discovery and data mining* (pp. 899–908).

Ge, Y., Xiong, H., Zhou, Z. hua, Ozdemir, H., Yu, J., & Lee, K. C. (2010). Top-eye: top-k evolving trajectory outlier detection. In *Proceedings of the 19th acm international conference on information and knowledge management* (pp. 1733–1736).

Grosu, D., & Chronopoulos, A. T. (2004). Algorithmic mechanism design for load balancing in distributed systems. *IEEE TSMC-B*, *34(1)*, 77-84.

Heijden, H. van der, Kotsis, G., & Kronsteiner, R. (2005). Mobile recommendation systems for decision making 'on the go'. In *Icmb*.

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Proceedings of the fifteenth conference on uncertainty in artificial intelligence* (p. 289-296). Stockholm, Sweden.

Huang, J., Huangfu, X., Sun, H., Li, H., Zhao, P., Cheng, H., et al. (2015). Backward path growth for efficient mobile sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, *27*(1), 46–60.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th acm sigkdd international conference on knowledge discovery and data mining* (pp. 426–434).

Liu, J., Li, Q., Qu, M., Chen, W., Yang, J., Xiong, H., et al. (2015). Station site optimization in bike sharing systems. In *Data mining (icdm), 2015 ieee international conference on* (pp. 883–888).

Liu, J., Sun, L., Chen, W., & Xiong, H. (2016). Rebalancing bike sharing systems: A multi-source data smart optimization. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1005–1014).

Liu, J., Sun, L., Li, Q., Ming, J., Liu, Y., & Xiong, H. (2017). Functional zone based hierarchical demand prediction for bike system expansion. In *Kdd'17*.

Liu, S., Wang, S., Liu, C., & Krishnan, R. (2015). Understanding taxi drivers routing choices from spatial and social traces. *Frontiers of Computer Science*, *9*(2), 200–209.

Liu, Y., Liu, C., Liu, B., Qu, M., & Xiong, H. (2016). Unified point-of-interest recommendation with temporal interval assessment. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1015–1024).

Liu, Y., Liu, C., Yuan, N. J., Duan, L., Fu, Y., Xiong, H., et al. (2014). Exploiting heterogeneous human mobility patterns for intelligent bus routing. In *Data mining (icdm), 2014 ieee international conference on* (pp. 360–369).

Miao, F., Han, S., Lin, S., Stankovic, J. A., Zhang, D., Munir, S., et al. (2016). Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach. *IEEE Transactions on Automation Science and Engineering*, *13*(2), 463–478.

Miller, B. N., Albert, I., & al et. (2003). Movielens unplugged: Experiences with a recommender system on four mobile devices. In *international conference on intelligent user interfaces*.

Mooney, R. J., & Roy, L. (1999). Content-based book recommendation using learning for text categorization. In *Workshop recommender systems: Algorithms and evaluation*.

Nagy, G., & Salhi, S. (2005). Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, *162*(1), 126–141.

Niu, H., Liu, J., Fu, Y., Liu, Y., & Lang, B. (2016). Exploiting human mobility patterns for gas station site selection. In *International conference on database systems for advanced applications* (pp. 242–257).

Pazzani, M. (1999). A framework for collaborative, content-based, and demographic filtering. *Artificial Intelligence Review*.

Powell, J. W., Huang, Y., Bastani, F., & Ji, M. (2011). Towards reducing taxicab cruising time using spatio-temporal profitability maps. In *International symposium on spatial and temporal databases* (pp. 242–260).

Qian, S., Cao, J., Mouël, F. L., Sahel, I., & Li, M. (2015). Scram: a sharing considered route assignment mechanism for fair taxi route recommendations. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining* (pp. 955–964).

Qin, G., Li, T., Yu, B., Wang, Y., Huang, Z., & Sun, J. (2017). Mining factors affecting taxi drivers incomes using gps trajectories. *Transportation Research Part C: Emerging Technologies*, *79*, 103–118.

Qu, M., Zhu, H., Liu, J., Liu, G., & Xiong, H. (2014). A cost-effective recommender system for taxi drivers. In *Kdd'14* (pp. 45–54).

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 acm conference on computer supported cooperative work* (pp. 175–186). ACM Press.

Rong, H., Zhou, X., Yang, C., Shafiq, Z., & Liu, A. (2016). The rich and the poor: A markov decision process approach to optimizing taxi driver revenue efficiency. In *Proceedings of the 25th acm international on conference on information and knowledge management* (pp. 2329–2334).

San francisco taxi fare calculator. (n.d.). `http://www.taxifare.us/san_francisco_taxi_fare_estimator.html`.

Shao, D., Wu, W., Xiang, S., & Lu, Y. (2015). Estimating taxi demand-supply level using taxi trajectory data stream. In *Data mining workshop (icdmw), 2015 ieee international conference on* (pp. 407–413).

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, *2009*, 4.

Sun, J., Xiong, Y., Zhu, Y., Liu, J., Guan, C., & Xiong, H. (2015). Multi-source information fusion for personalized restaurant recommendation. In *Proceedings of the 38th international acm sigir conference on research and development in information retrieval* (pp. 983–986).

Tang, L.-A., Zheng, Y., Xie, X., Yuan, J., Yu, X., & Han, J. (2011). Retrieving k-nearest neighboring trajectories by a set of point locations. In *Advances in spatial and temporal databases* (pp. 223–241). Springer.

Tveit, A. (2001). Peer-to-peer based recommendations for mobile commerce. In *the 1st international workshop on mobile commerce.*

Xiong, H., Shekhar, S., Huang, Y., Kumar, V., Ma, X., & Yoo, J. S. (2004). A framework for discovering co-location patterns in data sets with extended spatial objects. In *Sdm.*

Xu, T., Zhu, H., Zhao, X., Liu, Q., Zhong, H., Chen, E., et al. (2016). Taxi driving behavior analysis in latent vehicle-to-vehicle networks: A social influence perspective. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1285–1294).

Xu, Z., & Huang, R. (CS213 Univ. of California,Riverside). Performance study of load balancing algorithms in distributed web server systems. In *Tr.*

Yao, Z., Fu, Y., Liu, B., Liu, Y., & Xiong, H. (2016). Poi recommendation: A temporal matching between poi popularity and user regularity. In *Data mining (icdm), 2016 ieee 16th international conference on* (pp. 549–558).

Ye, Z., Xiao, K., & Deng, Y. (2015). Investigation of simulated annealing cooling schedule for mobile recommendations. In *Data mining workshop (icdmw), 2015 ieee international conference on* (pp. 1078–1084).

Yuan, J., Sun, G.-Z., Tian, Y., Chen, G., & Liu, Z. (2009). Selective-nra algorithms for top-k queries. In *Advances in data and web management* (pp. 15–26). Springer.

Yuan, J., Zheng, Y., Xie, X., & Sun, G. (2013). T-drive: Enhancing driving directions with taxi drivers' intelligence. *Knowledge and Data Engineering, IEEE Transactions on*, *25*(1), 220–232.

Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G., et al. (2010). T-drive: driving directions based on taxi trajectories. In *Proceedings of the 18th sigspatial international conference on advances in geographic information systems* (pp. 99–108).

Yuan, J., Zheng, Y., Zhang, L., Xie, X., & Sun, G. (2011). Where to find my next passenger. In *Proceedings of the 13th international conference on ubiquitous computing* (pp. 109–118).

Zhang, C., Liang, H., & Wang, K. (2016). Trip recommendation meets real-world constraints: Poi availability, diversity, and traveling time uncertainty. *ACM Transactions on Information Systems (TOIS)*, *35*(1), 5.

Zhang, C., Liang, H., Wang, K., & Sun, J. (2015). Personalized trip recommendation with poi availability and uncertain traveling time. In *Proceedings of the 24th acm international on conference on information and knowledge management* (pp. 911–920).

Zhang, D., Sun, L., Li, B., Chen, C., Pan, G., Li, S., et al. (2015). Understanding taxi service strategies from taxi gps traces. *IEEE Transactions on Intelligent Transportation Systems*, *16*(1), 123–135.

Zheng, Y., Liu, Y., Yuan, J., & Xie, X. (2011). Urban computing with taxicabs. In *Proceedings of the 13th international conference on ubiquitous computing* (pp. 89–98).

Zheng, Y., Yuan, J., Xie, W., Xie, X., & Sun, G. (2010). Drive smartly as a taxi driver. In *Ubiquitous intelligence & computing and 7th international conference on autonomic & trusted computing (uic/atc), 2010 7th international conference on* (pp. 484–486).

Zhou, W., Xiong, H., Ge, Y., Yu, J., Ozdemir, H., & Lee, K. C. (2010). Direction clustering for characterizing movement patterns. In *Information reuse and integration (iri), 2010 ieee international conference on* (pp. 165–170).

Zhu, H., Chen, E., Yu, K., Cao, H., Xiong, H., & Tian, J. (2012). Mining personal context-aware preferences for mobile users. In *Proceedings of the ieee 12th international conference on data mining* (p. 1212-1217).