Contrast Data Mining of Multi-source Heterogeneous Trajectory Data

Li Li
ORCID: 0000-0003-3158-7468

Submitted in total fulfilment of the requirements of the degree of

Doctor of Philosophy

School of Computing and Information Systems
THE UNIVERSITY OF MELBOURNE

January 2020

Abstract

The rapid growth of location-acquisition and mobile computing techniques has led to an increasing availability of human trajectory data. This raises the challenge of detecting and understanding human mobility from these trajectory datasets to extract useful knowledge in a variety of domains, such as business management and urban computing. In this thesis, we focus on research into knowledge discovery from multi-source heterogeneous trajectory data. To be specific, five research questions in three scenarios are studied. The details are as follows.

The first research question is how to perform trajectory pattern identification and anomaly detection for pedestrian flows. We propose to adopt contour maps as the visualization method of the origin-destination flow matrix to describe the distribution of pedestrian movements in terms of entry/exit areas. By transforming the origin-destination flow matrix into a dissimilarity matrix, a visual clustering algorithm is applied to visually cluster the most popular and related areas. We also propose a clustering-based algorithm to detect normal/abnormal time periods with similar/anomalous pedestrian flow patterns. Our results on one synthetic and one real-life dataset validate the effectiveness of our proposed algorithms.

The second research question is how to perform contrast pattern mining from multi-source datasets in retail environments. Given the sales data and customers' trajectory data, in order to find patterns where there has been a big change in one dataset but little change in the other dataset, we define a new kind of contrast pattern, conditional contrast patterns, which are a subset of traditional contrast patterns in one kind of dataset conditioned on a property of these patterns in another kind of dataset. Accordingly, we propose an algorithm based on tree search for mining these patterns. Experiments on a

synthetic dataset as well as a real-life retail dataset show that our proposed patterns are more informative and actionable for decision makers than traditional contrast patterns, and our tree-based algorithm has good performance in terms of computational efficiency.

Three research questions are studied in the third scenario, i.e., human behavior analysis in heterogeneous mobile networks. First, we focus on identifying the underlying geographical corridors of trajectories generated in mobile networks. We propose a hierarchical multi-scale trajectory clustering algorithm for corridor identification by analyzing the non-homogeneity of the spatial distribution of cell towers and users' movements. Results on a three-week real-life dataset from China Mobile show that our method can achieve the best performance with more than 10% improvement in clustering quality compared with other state-of-the-art methods.

Identifying static corridors plays an important role in managing networks for the long term design of a network. However, there is also a great opportunity for dynamically reconfiguring a network in response to changes in traffic flows. Therefore, in our fourth work, we propose a framework based on contrast data mining to identify significantly different corridors during different time periods. Contrast corridors are defined and a distance measure based on Hausdorff distance and earth movers' distance is proposed to calculate the dissimilarity between the identified corridors. Experimental results on synthetic as well as real-life datasets show that our method can effectively and robustly detect contrast corridors from trajectories generated from different time periods in mobile networks by improving the F1 score by 20% on average.

Finally, we focus on how to design caching strategies at the edge of networks. Edge caching in mobile networks can improve users' experience, reduce latency and balance the network traffic load. Considering that cells located in different places have different levels of predictability due to the heterogeneity of mobile users' content preferences and mobility, we propose an adaptive edge caching algorithm based on content popularity as well as the individual's prediction results to provide an optimal caching strategy, aiming to maximize the cache hit rate with acceptable file replacement cost. Our results on a real-life dataset as well as simulation data show that our method is more appropriate for resource-limited and heterogeneous network than other methods.

In summary, we have proposed several trajectory data mining approaches to extract useful knowledge from heterogeneous trajectory data or multi-source datasets in three different scenarios. We have shown that our proposed methods can achieve better performance compared to existing state-of-art techniques on a variety of real-life datasets.

Declaration

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Li Li, January 2020

Acknowledgements

I would like to acknowledge all the people who have contributed to this thesis and helped me during my PhD study. Their support and encouragement make this journey colorful and fruitful.

First, I would like to express my gratitude and sincere appreciation to my excellent supervisors, Prof. Christopher Leckie and Dr. Sarah Erfani, for their strong support and warm help. I feel very lucky to be supervised by them during my PhD candidature in the past four years. Without Chris, I would not have the chance to study abroad and have this memorable journey. He provided many insightful ideas and constructive suggestions in this research, inspired me with confidence in research and teaching, and generously supported me to attend conferences. I would like to sincerely thank Sarah, who joined my supervision from my second year. She also provided good advice in our meetings, helped me review our papers and always encouraged me when I feel frustrated. A special acknowledgment goes to Dr. Chien Aun Chan who provided valuable data, professional knowledge in communication engineering and helpful advice to my research. I would also like to thank my advisory committee chair Prof. Frank Vetere for his guidance and support.

I would like to thank the China Scholarship Council for funding my study at the University of Melbourne, the School of Computing and Information Systems for providing the whole training program and the administrative staff who helped me. I would also like to express thanks to my friends and fellow students, Oscar, Huiping, Yixin, Shima, Tabinda, Liyan et al., for their help to me and some insightful discussions with them.

My deepest gratitude goes to my family. I would like to thank my parents and parents-in-law for their encouragement and support. My special gratitude goes to my

husband, Dr. Fang Xu, for his love, understanding and encouragement. Although we have different research areas, he is always there to help me and willing to discuss with me. I feel so grateful to have him with me in this journey even though we have been separated for almost four years in two far away cities, Melbourne and Manchester.

Preface

This thesis has been written at the School of Computing and Information Systems, The University of Melbourne. Each chapter is based on manuscripts published or under review for publication. The main contributions of the thesis are discussed in Chapters 3-7 and are based on the following publications:

Published Papers

P1. **Li Li**, Christopher Leckie. Trajectory Pattern Identification and Anomaly Detection of Pedestrian Flows Based on Visual Clustering, in *International Conference on Intelligent Information Processing (ICIIP)*, 2016 (Chapter 3 contains material from this publication).

P2. **Li Li**, Sarah Erfani, Christopher Leckie. A Pattern Tree based Method for Mining Conditional Contrast Pattern of Multi-Source Data, in *IEEE International Conference on Data Mining Workshops (ICDMW)*, 2017 (Chapter 4 contains material from this publication).

P3. **Li Li**, Sarah Erfani, Chien Aun Chan and Christopher Leckie. Multi-scale Trajectory Clustering to Identify Corridors in Mobile Networks, in *ACM International Conference on Information and Knowledge Management (CIKM)*, 2019 (Chapter 5 contains material from this publication).

P4. **Li Li**, Sarah Erfani, Chien Aun Chan and Christopher Leckie. Adaptive Edge Caching based on Popularity and Prediction for Mobile Networks, in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2019 (Chapter 7 contains material from this publication).

P5. **Li Li**, Sarah Erfani, Chien Aun Chan and Christopher Leckie. Discovery of Contrast Corridors from Trajectory Data in Heterogeneous Dynamic Cellular Networks Using Contrast Mining, in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2020 (Chapter 6 contains material from this publication).

Contents

List of Figures

List of Tables

Acronyms

ARIMA AutoRegressive Integrated Moving Average

CCP Conditional Contrast Pattern

CDR Call Detail Record

CP Contrast Pattern

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DFM Discrete Fréchet Metric

DFS Depth First Search

DTW Dynamic Time Warping

ED Euclidean Distance

EDR Edit Distance on Real sequence

EMD Earth Mover's Distance

EP Emerging Pattern

FEP Fuzzy Emerging Pattern

FM Fréchet Metric

GA Genetic Algorithms

GPS Global Positioning System

GMM Gaussian Mixture Models

GSM Global System for Mobile Communications

HD Hausdorff Distance

HMM Hidden Markov Models

IRD Interpolated Route Distance

iVAT improved Visual Assessment of cluster Tendency

JEP Jumping Emerging Pattern

LCSS Longest Common SubSequence

LDA Latent Dirichlet Allocation

LFU Least Frequently Used

LOF Local Outlier Factor

LRU Least Recently Used

LSS Location Sharing Services

LSTM Long Short Term Memory

MAC Media Access Control

MDL Minimum Description Length

MHD Modified Hausdorff Distance

O-D Origin-Destination

OPTICS Ordering Points To Identify the Clustering Structure

PC Popularity Caching

PCA Principal Component Analysis

QoE Quality of Experience

RFID Radio-Frequency IDentification

RSS Received Signal Strength

SJEP Strong Jumping Emerging Pattern

SSPD Symmetrized Segment-Path Distance

t2vec Trajectory to Vector

VAT Visual Assessment of cluster Tendency

WLAN Wireless Local Area Networks

ZBDD Zero-suppressed Binary Decision Diagram

Chapter 1
Introduction

RAPID improvements in location acquisition technologies have led to the increasing availability of trajectory data for different types of objects, such as pedestrians, customers, vehicles and animals. This has motivated research into data mining techniques that can extract insights into the behavior and movement patterns of such objects in domains such as business management, urban computing, social media and transportation. The objective of this thesis is to study human mobility patterns from multi-source heterogeneous datasets using unsupervised machine learning techniques. In the following sections, we begin with a brief introduction to the motivation for our work. Then we present the existing challenges and our research questions in this thesis. Finally, the organization of the thesis and a summary of our contributions are presented.

1.1 Motivation

Trajectory data can be obtained through a variety of passive or active sensing modalities, e.g., traditional travel survey data, Global Positioning System (GPS) data, Call Detail Record (CDR) data in mobile networks, video cameras, Wireless Local Area Networks (WLAN), Bluetooth, Radio-Frequency IDentification (RFID) and other non-conventional sources, such as mobile social media and check-in data [5]. As a consequence, it is becoming easier to generate large-scale trajectory data traces of moving objects.

Generally, a trajectory is represented by a sequence of locations sampled from the continuous movement of the moving object with corresponding timestamps, i.e., $T = \{(p_1, t_1, a_1), ..., (p_i, t_i, a_i), ..., (p_n, t_n, a_n)\}$, where p_i is the spatial location of the sampled

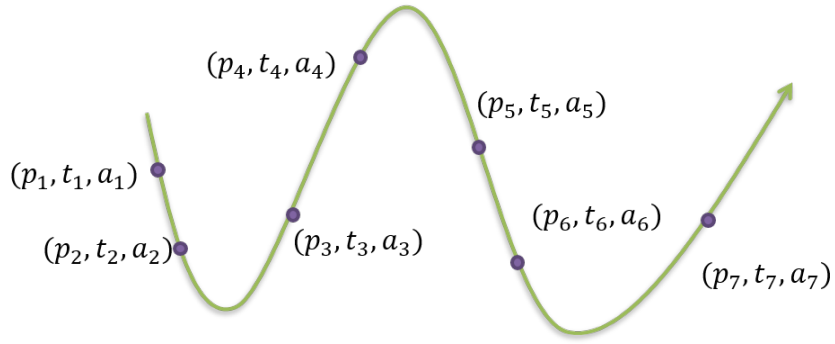


Figure 1.1: Illustration of a trajectory with seven sampled points.

point at timestamp t_i, and a_i is the additional associated attribute (e.g., purchased items of customers, requested files of mobile users, etc.), as shown in Figure 1.1. The encoding of the position may be represented in different ways depending on the recording technology or the trajectory representation method, such as Cartesian coordinates (longitude and latitude), grid cells or mobile cell id.

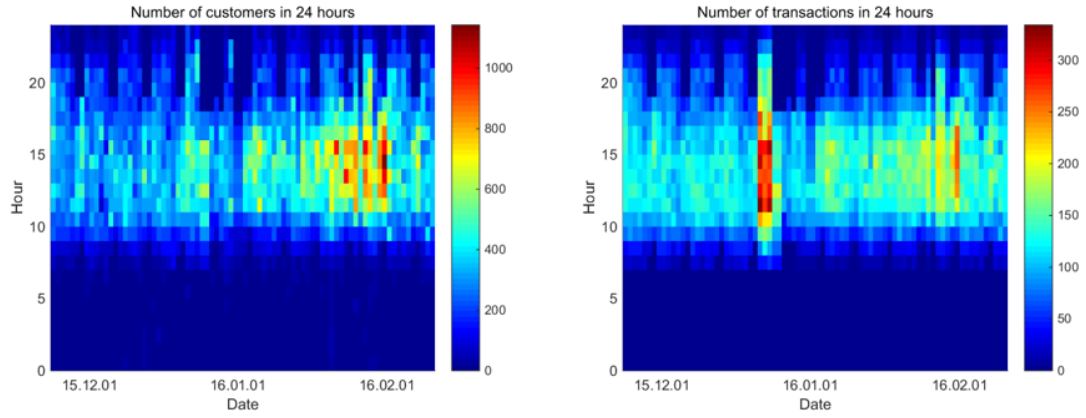
Similar to the general field of data mining, trajectory data mining aims to discover interesting knowledge from the trajectory data. A wide spectrum of applications can benefit from trajectory data mining, such as urban planning, traffic flow control, path discovery, travel recommendation, anomaly detection, location/destination prediction, as well as location-based services and applications. These applications can have significant benefits for individuals, commercial organizations or government agencies [38]. For example, by using anomaly detection on pedestrian trajectories, it is possible to detect the occurrence of major events, such as celebrations, parades, business promotions, accidents or disasters, which may be a risk to public security or safety. Another example is contrast mining of trajectory patterns of customers in retail stores, which can help assess how customer behavior is influenced by different factors such as time of day, day of week, store layout, supply shortage or shifts, advertising campaigns, in-store promotions or in-store signage. This knowledge can then be used to improve decision-making tasks for retail business owners.

1.2 Challenges

Trajectory data mining creates a range of challenges for traditional data mining techniques, due to several inherent properties of trajectory data that need to be taken into consideration. For example, normally in classical data mining problems, the instances are independent and identically distributed, whereas in trajectory data mining, instances are structurally related to each other in the context of space and time, and show varying properties over different spatial regions and time periods [5]. Consequently, trajectory mining techniques need to be able to capture these spatial and temporal structural relationships when characterizing and generalizing patterns of trajectories. In this thesis, we focus on solving three main challenges when dealing with trajectory data, namely anomaly detection, multi-source, and spatial and temporal heterogeneity.

C1: Anomaly detection The definition of trajectory anomaly is heavily dependent on the nature of the application. For example, an anomaly can be a trajectory or a segment of a trajectory that is significantly different from other trajectories in terms of some similarity measure on features, such as shape, direction or speed [134]. An anomaly can also be an event or a group of trajectories that do not conform to an expected pattern, which could be caused by traffic accidents, road blocks, disasters, celebrations, protests or other events. Therefore, it can be challenging to construct a general definition of anomalies for different application purposes, and provide appropriate methods for detecting these anomalies.

C2: Multi-source In many applications it is interesting and challenging to extract patterns form multi-source datasets. Many existing studies on trajectory data mining are primarily focused on the trajectory dataset alone, i.e., movement points with time stamps. However, in certain applications, additional information about human behavior can be obtained from other data sources. For example, in a retail environment, in addition to trajectory data on customers' visiting behavior (e.g., visits or browsing in different areas), we may also have access to information about their shopping behavior, which can be obtained from sales data (i.e., what purchases they made). As shown in Figure 1.2, based on the trajectory data of customers in a shop, we have summarized the number of customers visiting the shop per hour during 78 days from November 2015 to February 2016, which



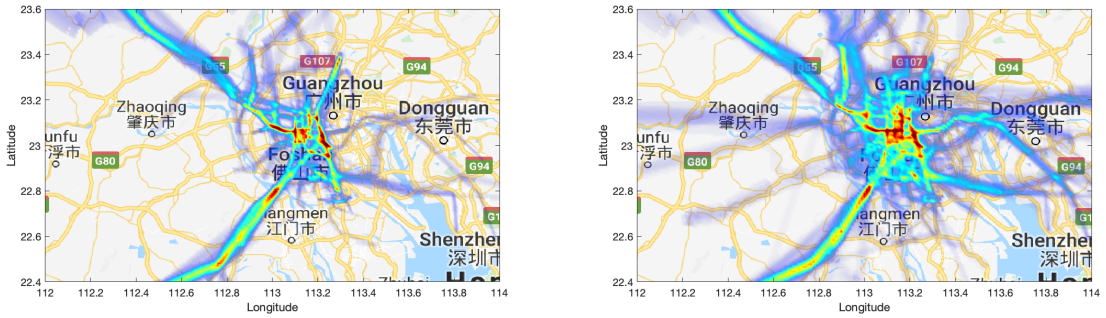
(a) number of customers (b) number of transactions

Figure 1.2: The number of customers and transactions per hour in different days.

is visualized in Figure 1.2a. Similarly, we can obtain the number of transactions per day from the sales data, as shown in Figure 1.2b. By comparing these two datasets, some interesting patterns can be discovered, e.g., identifying the sets of products where there has been an increase in sales but not an increase in the number of visiting customers. Knowledge of such a product or set of products is important as it highlights that the conversion rate of customers has improved. (Details of this dataset are given in Chapter 4.)

C3: Heterogeneity In classical trajectory data mining, trajectories are assumed to be homogeneous, which implies that each trajectory is generated by the same population and is thus identically distributed. However, trajectories can show spatial and temporal heterogeneity in varying ways in different application domains.

- *Temporal heterogeneity* For example, customers can show different visiting or purchasing behaviors at different times of a day or different days of a week, which can be observed in Figure 1.2. Thus, the patterns of customer behavior on weekdays can be different from the patterns on weekends. Figure 1.3 shows another example, which shows the heatmaps of the traffic density of mobile phone users on weekdays and weekends. It clearly shows that the traffic flows of mobile phone users are different in weekdays and weekends. (Details of this dataset are given in Chapter 5.)



(a) weekdays (b) weekends

Figure 1.3: The heat map of mobile phone users' traffic density on weekdays and weekends in a province of China.

- *Spatial heterogeneity* In mobile network management, the movement patterns of mobile users can vary over different spatial scales according to the density of different regions of the network, e.g., across the central business district, suburbs or rural areas of the network. Also due to the heterogeneity of users' movement patterns (e.g., different transportation methods), trajectories of different users or even the same user can have inconsistent granularity. Figure 1.4 shows the distribution of cell towers in one city in the southern part of China. Due to the non-homogeneity of the spatial distribution of cell towers, the spatial resolution of the trajectories generated from mobile networks can vary from several hundred meters to a few meters.

1.3 Research Questions and Objectives

Considering the challenges of trajectory data mining of multi-source heterogeneous datasets, in this thesis, five research questions about human mobility analysis are studied in three different scenarios: (1) To address **C1**, we studied the pedestrian movement patterns in public places; (2) Customers' shopping and visiting behaviors are studied in retail environments to address the second challenge **C2**; (3) Three research questions are studied

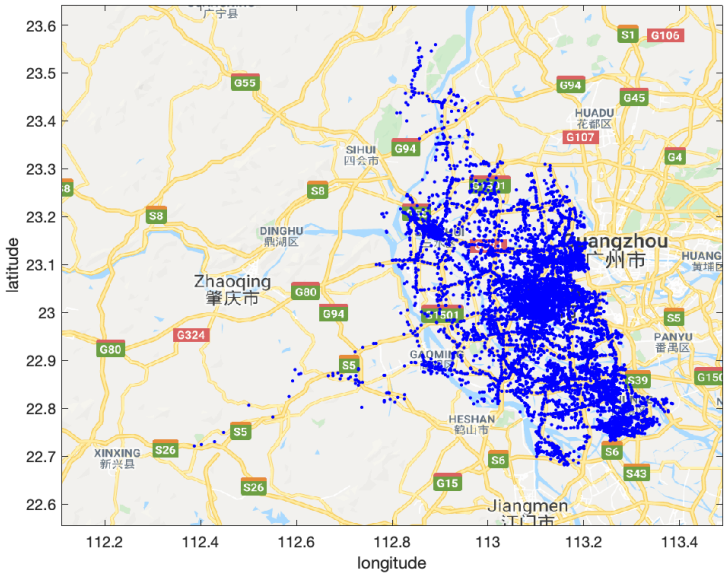


Figure 1.4: The cellular tower distribution in one city in China.

for analyzing phone users' mobility patterns in mobile networks to address the spatial and temporal heterogeneity challenges as mentioned in **C3**.

Q1: How to discover and describe the movement patterns hidden in trajectories, and identify any misbehavior or interesting events using visualization methods?

Understanding patterns of pedestrian movement is useful in applications such as pedestrian flow management, public security, and safety. One challenge is how to discover and describe the movement patterns hidden in trajectories, and identify any misbehavior or interesting events. Therefore, three related research questions are studied: (1) how to visually identify and summarize related sets of pedestrian flows over a given time period; (2) how to visually identify which time periods exhibit similar patterns of pedestrian flows; and (3) how to visually identify which time periods have experienced anomalous flow patterns.

Q2: How to extract useful rules from trajectory data as well as other associated datasets?

Trajectory data is often associated with other datasets, e.g., in a retail environment, the trajectory data can be obtained as well as the sales data. Identifying or extracting rules

from multi-source data can provide more insightful or helpful knowledge than using one single dataset in isolation. For example, shop managers may be interested in finding the relationships or differences between customer movement and sales data. Contrast patterns are itemsets that frequently occur in one dataset while not in another [30]. These patterns have been successfully applied to many data mining domains, such as prediction, classification, and clustering. However, none of the previous studies has considered extracting contrast patterns from different types of datasets. Therefore, our second question is how to extract useful rules from multi-source datasets using contrast mining techniques.

Q3: How to identify common corridors from the heterogeneous trajectory data of phone users in mobile networks?

Identifying the underlying geographical pathways/corridors shared by users in mobile networks can help the network providers better understand the movement patterns of the mobile users, provide them with insights on the deployment of networks and base stations and the utilization of network resources. However, compared with other types of trajectories, mobile trajectories are coarse, and their granularity varies due to the inconsistent density of cell towers. Our third question is how to identify common corridors from the heterogeneous trajectory data of phone users in mobile networks.

Q4: How to discover what are the significant changes in corridors in dynamic heterogeneous mobile networks?

Identifying static movement corridors in mobile networks plays an important role in managing resources for the long term design of network. However, with the introduction of a new generation of cellular network technology, such as 5G, there is an emerging opportunity for dynamically reconfiguring the network in response to changes in the traffic flows by time of day. Therefore, considering the temporal heterogeneity in trajectory data, our fourth research question is how to discover what are those significant changing corridors in mobile networks.

Q5: How to design an appropriate edge caching strategy in mobile networks?

One important decision-making context where trajectory patterns can in mobile networks is in caching content in the mobile network. Edge caching in mobile networks

can help improve users' experience, reduce latency and balance the network traffic load. However, edge caching requires suitable strategies for determining what files to pre-fetch at which cell and at what time. Due to the heterogeneity of users' content preferences and mobility, caching based only on popularity has limitations. Therefore, considering the spatial heterogeneity in trajectory data, our fifth question is how to design an appropriate edge caching strategy in mobile networks.

1.4 Organization and Contributions

The core chapters of this thesis are mostly derived from the refereed publications made during my PhD candidature. Figure 1.5 provides an overview of the structure of the thesis. In this section, we provide an overview of the focus and contributions of each chapter in the thesis.

Chapter 2 – Literature Review

In this chapter, we present a general review of three unsupervised learning tasks in trajectory mining, i.e., trajectory pattern mining, trajectory clustering and anomaly detection. We provide a literature review of the recent work in different application scenarios, indoor human mobility analysis and mobile phone user behavior analysis in mobile networks, in terms of data acquisition techniques and existing trajectory mining techniques. We also raise open questions which we believe are worthy of further research.

- We review existing unsupervised data mining techniques for trajectory data. We propose to utilize contrast pattern mining techniques to better understand the differences and changes in different datasets.
- Techniques for trajectory clustering are also reviewed in terms of different similarity measures and clustering algorithms.
- We give a survey of the trajectory data acquisition techniques and application questions in two scenarios, i.e., indoor human mobility analysis and phone users' behavior analysis in mobile networks.

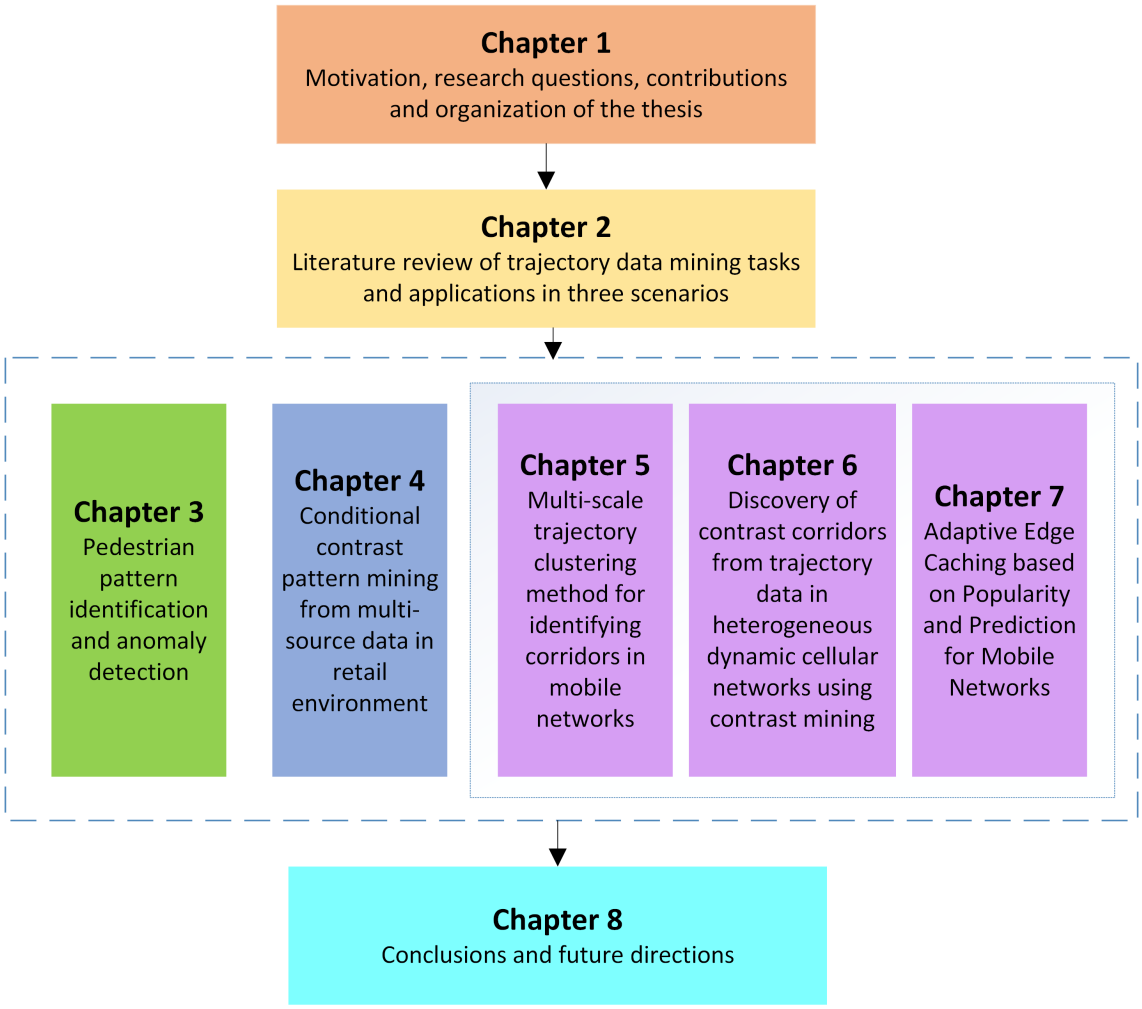


Figure 1.5: Thesis organization.

Chapter 3 – Related pedestrian flow identification and anomaly detection of pedestrian flows based on visual clustering

In this chapter, we study research question **Q1**, the problem of trajectory data analysis and anomaly detection, which falls into the category of trajectory data mining. Two common approaches used to address this problem are statistical methods combined with classification and clustering-based methods. Many existing approaches to address this problem have the limitations that they focus on the details of trajectories but do not consider the characteristics of the overall trajectory distribution.

In this chapter, we focus on three specific questions. The first question is how to summarize related pedestrian flows. We propose to transform the origin-destination flow matrix into a dissimilarity matrix first. Then a visual clustering algorithm is adopted to find related pedestrian flows. The second question is how to detect normal/abnormal time periods/pedestrian flows. We propose a clustering-based method to identify time periods with abnormal pedestrian flows. Finally, we utilize contour maps to visualize the distribution of identified anomalous pedestrian flows. The main contributions of this work are as follows:

- Considering the distribution of trajectories, we propose a method based on visual clustering to find related pedestrian flows.
- We propose a novel method based on the improved Visual Assessment of cluster Tendency (iVAT) algorithm [111] to detect normal/abnormal time periods with similar/anomalous pedestrian flow patterns.
- Synthetic and large, real-life datasets are used to validate the effectiveness of our proposed algorithms.

The publication arising from the work in this chapter is paper P1 listed in the Preface.

Chapter 4 – A pattern tree based method for mining conditional contrast patterns of multi-source data

There have been many contrast pattern mining methods proposed in the literature. However, none of them has considered mining contrast patterns for different kinds of data

sources. In this chapter, we propose a new kind of contrast pattern, conditional contrast patterns for multi-source data, which differ from other existing types of patterns - it is a subset of contrast patterns, which can be more *informative* and *instructive* for decision makers since it utilizes *information from additional related data sources*. The main contributions of this work are as follows:

- We introduce a new type of contrast pattern, Conditional Contrast Patterns (CCP). Three conditions for mining CCPs are also proposed.
- We propose a conditional contrast pattern tree based method for mining CCPs.
- We evaluate our method and compare its performance with two baseline methods on a synthetic dataset. The results demonstrate better efficiency of our method in comparison with the other two methods.
- We also apply our method to one real-life retail dataset, which includes the transaction data and customer behavior of each customer in a retail store in Melbourne. The results show the practicality of our proposed method.

The publication arising from the work in this chapter is paper P2 listed in the Preface.

Chapter 5 – A Corridor identification algorithm in mobile networks

Identifying the underlying geographical corridors of trajectories generated in mobile networks helps mobile operators to better manage and orchestrate the allocation of network resources. However, compared with other types of trajectories, mobile trajectories are coarse, and their granularity varies due to the inconsistent density of cell towers. Previous trajectory clustering approaches are not appropriate for trajectory data in mobile networks since they fail to consider the heterogeneity of mobile networks. In this chapter, we propose a hierarchical multi-scale trajectory clustering algorithm for corridor identification by analyzing the non-homogeneity of the spatial distribution of cell towers and users' movements. The main contributions of this work are as follows:

- To measure trajectory similarity on different scales we propose a distance measure based on Hausdorff distance [50] that considers the cell density distribution.

- A two-level clustering method based on the developed distance measure is proposed. Corridors are identified and represented as weighted graphs based on the clustering result.

- Experiments are conducted over the real-life data of mobile users in a southern province in China to evaluate our approach. Results show that by considering the heterogeneity of mobile networks, our method can achieve the best performance with more than 10% improvement in clustering quality compared with state-of-the-art methods.

The publication arising from the work in this chapter is paper P3 listed in the Preface.

Chapter 6 – Discovery of contrast corridors from trajectory data in heterogeneous dynamic cellular networks using contrast mining

Identifying static corridors plays an important role in managing networks for the long term design of the network. However, with the introduction of new generations of cellular network technology, such as 5G, there is a great opportunity for dynamically reconfiguring the network in response to changes in traffic flows by time of day. However, most studies treated the network as temporally homogeneous in their analyses. In this chapter, we focus on the problem of identifying what are those significant changing corridors based on contrast data mining. The main contributions of this work are as follows:

- We propose a distance measure to calculate the dissimilarity between corridors in heterogeneous mobile networks.

- We propose a contrast corridor mining algorithm based on Earth Movers' Distance [94] to detect the differences/changes in movement patterns during different time periods.

- Experiments over the real-life data of mobile users as well as a synthetic dataset are conducted to evaluate our approach. These experiments show that our method can effectively and robustly detect contrast corridors from trajectories generated from

different time periods in mobile networks by improving the F1 score by 20% on average.

The submission arising from the work in this chapter is paper P5 listed in the Preface.

Chapter 7 – An adaptive edge caching strategy for mobile networks

Edge caching in mobile networks can improve users' experience, reduce latency and balance the network traffic load. Due to the heterogeneity of users' content preferences and mobility, caching based only on popularity has limitations. The challenge is how to design a suitable strategy for determining what files to pre-fetch at which cell and at what time. Considering that cells located in different places have different predictabilities, in this chapter, we propose an adaptive edge caching strategy for mobile networks based on content popularity as well as the individual's prediction results to provide an optimal caching strategy, which aims to maximize the cache hit rate with acceptable file replacement cost. The main contributions of this work are as follows:

- Considering that cells located in different places have different degrees of predictability, we propose an adaptive edge caching algorithm based on content popularity as well as the individual's prediction results to provide an optimal caching strategy, which aims to maximize the cache hit rate with acceptable file replacement cost.
- A heuristic optimization strategy based on genetic algorithms is presented, along with a prediction model based on an improved Markov model for each user according to the historical data. In the model, similar users are clustered based on their behavior patterns.
- We evaluate our algorithm on a simulation dataset as well as a 3-week real-life dataset from China Mobile. The results indicate that our optimal caching strategy can improve the cache hit rate compared with other methods, especially when the storage capacity is small and the similarity in content requests of users is low.

The publication arising from the work in this chapter is paper P4 listed in the Preface.

Chapter 8 – Conclusions and future directions

In this chapter, we conclude the thesis with a summary of the key research outcomes and a discussion of future directions.

Chapter 2

Background on Unsupervised Learning for Trajectory Data

The prevalence of tracking methods and technologies has generated a large amount of spatial trajectory data. Such trajectory data provides us with an unprecedented volume of information to better understand moving objects, which plays an important role in a broad range of applications, such as urban computing, location-based services, and environment science. Many trajectory data mining techniques have been proposed for knowledge discovery and pattern mining in the literature, which can be generally divided into two categories, i.e., supervised learning and unsupervised learning. The focus of this thesis is on unsupervised learning, the goal of which is to learn the inherent structure or distribution in the data without using explicitly provided labels. In this chapter, we first review the related work of three unsupervised learning tasks in trajectory mining, i.e., trajectory pattern mining, trajectory clustering and anomaly detection. The relevant theories and concepts are also provided. Then we give an overview of trajectory data acquisition techniques and existing trajectory mining techniques in the context of two application areas that are most related to our research, namely, indoor human mobility analysis and mobile phone user behavior analysis in mobile networks.

2.1 Unsupervised Learning Tasks for Trajectory Data

Similar to data mining, trajectory data mining can involve supervised learning tasks (e.g., trajectory classification and location prediction [117]) that learn from labeled training data, and unsupervised learning tasks that focus on finding underlying patterns or models based on unlabeled data. Given the focus of this thesis, in this section, we review three important unsupervised learning tasks for trajectory data, i.e., trajectory pattern mining, trajectory clustering and trajectory anomaly detection.

2.1.1 Trajectory Pattern Mining

Trajectory pattern mining concentrates on identifying rules that describe the movement patterns hidden in the trajectory data, such as frequent patterns, contrast patterns, group patterns, and periodic patterns, which we describe as follows.

Frequent Sequential Patterns

Frequent sequential patterns are subsequences (a sequence of visited locations) that occur in a trajectory dataset with a frequency no less than a user-specified threshold, which can be used for travel recommendation, next place prediction, and trajectory compression. Considering the spatio-temporal characteristics of trajectory data, frequent sequential patterns are also defined as a frequent sequence of locations with similar transition times between the locations. In [135], this problem was categorized into two subproblems, which are sequential pattern mining in free space [119,123], and sequential pattern mining in road networks [115]. In free space, normally the original trajectories need to be simplified or re-formed in terms of higher-level regions (e.g., change a sequence of points to a sequence of clusters), using techniques such as clustering [90] or stay point detection [19]. An alternative approach is to divide the area into uniform grids [136], then trajectories can be represented as a sequence of grid cell identifiers. Similarly, in a road network, by using map-matching, trajectories can be represented as a sequence of road segment identifiers [114].

Group Patterns

Group pattern mining aims at discovering a group of objects that move together for a certain period of time. Based on the spatio-temporal closeness of a group or the internal structure of a group, various group patterns have been studied in the literature, such as flocks [46], convoys [55], and swarms [68]. A flock is defined as a group of objects traveling together within a disk of some user-specified size for at least m consecutive timestamps, a convoy is a group of densely connected objects traveling together for at least m consecutive timestamps, and a swarm is an extension of the convoy pattern, which is

a group of objects traveling together for at least m timestamps (not necessarily consecutive). Normally clustering-based algorithms are adopted for group pattern mining.

Periodic Patterns

Periodic patterns aim to extract the regular movement patterns of an object [18]. For example, people commute between their work place and home every weekday, and some animals migrate repeatedly every year. The discovery of periodic patterns can help compress trajectory data and predict the behavior of a moving object. A common approach for mining periodic patterns is to apply frequent sequential pattern mining, e.g., an algorithm for retrieving maximal periodic patterns from trajectories was proposed in [18].

Contrast Patterns

Contrast pattern mining aims to quantify and describe the differences between two given multivariate datasets, which was first proposed by Bay and Pazzani [9]. Contrast patterns are often defined as patterns whose supports differ significantly among the datasets that are under contrast [30]. If the supports of all the frequent itemsets in two datasets are very similar, then we can say there is no significant change between these two datasets and vice versa. For example, in transactional data, a transaction t_i is said to contain a pattern X if X is a subset of t_i. Then the support count, $\sigma(X, D)$, of one pattern X in a dataset D can be stated as:

$$\sigma(X, D) = |\{t_i | X \subseteq t_i, t_i \in D\}|,$$

where $|\cdot|$ denotes the number of elements in a set. Then the support for a pattern X in a dataset D is calculated as:

$$supp(X, D) = \frac{\sigma(X, D)}{N}, \tag{2.1}$$

where N is the total number of transactions in D.

Then given two datasets D_i and D_j under contrast, the growth rate of a pattern X is

defined as:

$$gr(X, D_j) = \begin{cases} 0 & \text{if } supp(X, D_j) = supp(X, D_i) = 0; \\ \infty & \text{if } supp(X, D_j) > 0, supp(X, D_i) = 0; \\ supp(X, D_j)/supp(X, D_i) & \text{otherwise.} \end{cases} \quad (2.2)$$

and the support difference ($supp_\delta(X, D_j)$) of a pattern X with respect to datasets D_i and D_j is defined as:

$$supp_\delta(X, D_j) = supp(X, D_j) - supp(X, D_i). \quad (2.3)$$

The definition of a contrast pattern can then be given as follows.

Definition 2.1. ***Contrast pattern*** *Given a growth rate threshold (or a support delta threshold), a pattern X is called a contrast pattern for dataset D_j if $gr(X, D_j)$ (or $supp_\delta(X, D_j)$) is greater than the threshold. Dataset D_j is called the positive dataset, and the other dataset D_i is called the negative dataset. Specifically, if the support of a pattern in the negative dataset is 0, whereas in the positive dataset it is nonzero, the pattern is called a jumping emerging pattern.*

Here in the problem of contrast mining on trajectory data, our aim is to find discriminative patterns (e.g., sequences, graphs, matrices or tensors) that occur frequently in one dataset and infrequently in another. For example, in the work of Wang et al. [114], a framework for discovering the impact of road closures on traffic flows was proposed. By computing the growth rate of traffic flows on n-Edgesets, the emerging n-Edgesets were selected by using the Local Outlier Factor (LOF), and by comparing the frequency of the neighboring edge of emerging n-Edgesets, frequent emerging networks were also detected. The results showed that the proposed LOF-based method was more effective than a method whose thresholds were chosen arbitrarily. That is because in the former method the distribution of the growth rate is taken into account, which can provide useful information for the threshold setting. Note that here we treat contrast pattern mining as an unsupervised learning task, because we do not treat the data from the positive or negative datasets as labeled data.

2.1.2 Trajectory Clustering

The objective of trajectory clustering is to find clusters of similar trajectories based on their movement characteristics. Although there have been some studies on developing trajectory-specific approaches [2, 40], most state-of-the-art clustering algorithms are extensions of traditional clustering algorithms with an appropriate definition of distance measure (or similarity function). Therefore, in this section, we first review existing work related to trajectory clustering from these two perspectives, i.e., distance measures and clustering algorithms. Then a brief review of clustering-based anomaly detection is provided.

Distance Measure

Many distance measurements have been proposed for measuring the dissimilarity between trajectories.

- *Euclidean Distance (ED)* Euclidean distance is the point-by-point distance between two trajectories in Euclidean space [36]. It requires the number of points in the two trajectories to be consistent. However, in real applications, the number of sampling points in each trajectory may vary and the lengths of two trajectories can be different.

- *Dynamic Time Warping (DTW)* DTW [11] was proposed to find the distance between two temporal sequences with the best alignment, which uses a recursive approach to search all possible point combinations between two trajectories to find the alignment with the minimal distance.

- *Longest Common SubSequence (LCSS)* LCSS [109] was proposed to address the challenges of noisy data, by finding the longest common subsequence existing in two trajectory sequences, which is generally solved recursively. In this approach, there are two user-defined thresholds in this distance measure that need to set.

- *Edit Distance on Real sequence (EDR)* The idea of EDR [23] is to count the number of edit operations (insert, delete, replace) that are necessary to transform one sequence

into the other.

These measures are all based on the spatial proximity between sampled locations (point-to-point), and hence are easily affected by the sampling strategies adopted. Some distance measures were proposed to measure the distance between two polygonal curves based on the shape of the trajectories.

- *Hausdorff Distance (HD)* Hausdorff Distance [50] is used to measure how far two trajectories are from each other, which is defined as the maximum of all the distances from each point in one trajectory to the closet point in the other trajectory.

- *Fréchet Metric (FM) or Discrete Fréchet Metric (DFM)* FM [3] or DFM [34] measures the similarity between two curves by taking into account location and time ordering, which is defined as the maximum distance a point on the first curve has to travel as this curve is being continuously deformed into the second curve.

- *Symmetrized Segment-Path Distance (SSPD)* In [12], the authors proposed a new metric for vehicle trajectories called SSPD based on point-to-segment distance. The method returns the mean distance instead of the maximum distance, which is less sensitive to noise.

- *Trajectory to Vector (t2vec)* Recently, an approach has been proposed to use deep learning for a distance similarity measure. t2vec was proposed by Li et al. in [67], which is a seq2seq-based algorithm to learn representations of trajectories. It is robust to sampling rate variations and noisy sample points. Trajectories are represented as vectors using their proposed model and then Euclidean distance is adopted to measure the similarity.

Table 2.1 compares the distance measures we mentioned above from six perspectives: *parameters*, which indicates whether the algorithm has parameters to be tuned; *anti-noise*, which means whether the measure is sensitive to noise or not; *metric*, which shows whether the distance measure is a metric or not; *unifying length*, which indicates whether the two trajectories under comparison need to have equal numbers of sampling points; *temporal*, which shows whether temporal information is considered when calculating the

Table 2.1: A comparison of different trajectory similarity measures.

measure	parameters	anti-noise	metric	unifying length	temporal	complexity
ED	✗	✗	✓	✓	✗	$O(n)$
DTW	✗	✗	✗	✗	✗	$O(n^2)$
LCSS	✓	✓	✗	✗	✗	$O(n^2)$
EDR	✓	✓	✗	✗	✗	$O(n^2)$
HD	✗	✗	✓	✗	✗	$O(n^2)$
FM	✗	✗	✗	✗	✗	$O(n^2)$
SSPD	✗	✓	✗	✗	✗	$O(n^2)$
t2vec	✓	✓	✗	✓	✗	$O(n)$

distance; *complexity*, which shows the time complexity of the algorithm, where the two trajectories under comparison have n sampling points on average. Here the complexity is the time complexity of the basic distance measure algorithms without any particular optimization methods.

There are also some other distance measures proposed for specific problems. For example, in [60], a distance measure was defined as the weighted sum of three components, i.e., perpendicular distance, parallel distance, and angle distance. In another example [105], the authors proposed a distance function called Interpolated Route Distance (IRD), which computes the distance between each pair of aligned points (real or interpolated) to cope with asynchronous sampling rates.

Clustering Algorithms

Clustering is the task of dividing a set of objects into groups (clusters) that are meaningful, useful or both. The goal is that the objects within a group should be similar (or related) to one another and different from (or unrelated to) the objects in other groups [103]. There are different ways to categorize clustering techniques. For example, in [48], the major fundamental clustering methods are classified into four categories, i.e., partitioning methods, hierarchical methods, density-based methods, and grid-based methods. In the rest of this section, we divide the clustering techniques used for trajectory data into four categories and discuss them separately.

- *Prototype-based methods* Prototype-based methods aim to find a prototype that defines a cluster so that the objects in one cluster are closer to the prototype than to the prototype of any other cluster. The prototype is often a centroid, i.e., the average of all the points in the cluster. It can also be a medoid when a centroid is not meaningful. For example, in [59], Larson et al. utilized k-medoids clustering to find feasible centroid paths for different groups of people. The prototype can also be a statistical distribution. Each distribution corresponds to a cluster and the parameters of each distribution provide a description of the corresponding cluster. For example, in [40], a trajectory clustering algorithm was proposed based on a regression mixture model and the expectation maximization algorithm.

- *Hierarchical methods* Hierarchical methods create a hierarchical decomposition of the given set of data objects. These methods can be classified into two categories, agglomerative and divisive. The agglomerative approach starts with individual points as clusters, and then the two closest clusters are merged into a new cluster successively until only one cluster remains or a termination condition holds. This method has been adopted in papers such as [136–138]. The divisive approach starts with all the objects in one cluster. Then a cluster is split into smaller clusters until eventually each object is in one cluster, or a termination condition holds [103].

- *Density-based methods* Density-based methods locate regions of high density that are separated from one another by regions of low density. Several density-based clustering algorithms have been proposed, such as Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [35] or Ordering Points To Identify the Clustering Structure (OPTICS) [4]. For example, in [60], a line segment clustering method was proposed based on DBSCAN. In [82], T-OPTICS, an extension of OPTICS was proposed to compare and cluster trajectories.

- *Graph-based methods* Graph-based methods took a graph-based view of the data, i.e., the data is represented as a graph. Data objects are represented as vertices and the proximity between two data objects is represented by the weight of the edge between the corresponding nodes. For example, in [66], Li et al. utilized a graph-

theoretic clustering method for trajectory clustering, which takes on an iterative bi-partition manner and splits a "dominant set" from a weighted graph in each round.

2.1.3 Trajectory Anomaly Detection

In contrast to trajectory patterns that frequently occur in trajectory data, anomaly detection focuses on rare patterns. A trajectory anomaly (also called an outlier) is defined as a trajectory (or subtrajectory) which is significantly different from others in terms of some distance metric [73], such as shape, direction, location or travel time. The anomalies can also be events or observations (represented by a collection of trajectories) that do not conform to an expected pattern (e.g., traffic congestion caused by accidents) [135]. There have been many methods proposed for anomaly detection for different kinds of data. Han et al. [48] categorized anomaly detection methods into three types: statistical methods, proximity-based methods, and clustering-based methods. In the following, we review the related papers to these three types.

- *Clustering-based methods* Clustering-based methods assume that the normal data objects belong to large and dense clusters, whereas outliers belong to small or sparse clusters, or do not belong to any clusters. Clustering-based methods can be adopted for detecting anomalous trajectories. For example, in [61], an anomalous subtrajectory detection method was proposed based on the subtrajectory clustering algorithm proposed in [60].

- *Proximity-based methods* Proximity-based methods (such as distance-based methods and density-based methods [48]) assume that an object is an outlier if the nearest neighbors of the object are far away in feature space, that is, the proximity of the object to its neighbors significantly deviates from the proximity of most of the other objects to their neighbors in the same data set.

- *Statistical-based methods* Statistical methods (also known as model-based methods) make assumptions of data normality. They assume that normal data objects are generated by a process with a known statistical distribution, and that the data not

following the model are outliers. For example, Pang et al. [84] proposed a statistical model, which adapts a likelihood ratio test to find anomalous regions for monitoring the emergence of unexpected behavior based on GPS data from taxis.

2.2 Applications

Different application areas often require the design of specific trajectory mining algorithms and the discovery of new types of patterns for two reasons. First, the trajectory data generated by different recording techniques have different characteristics. For example, GPS trajectory data can be generated from individuals and vehicles with a relatively high sampling rate and accuracy, but it is only available in the open air and the accuracy may be affected by the urban canyon effect [27]. Mobile phone trajectory data do not have such problems, but the sampling rate depends on phone users' network usage patterns. Therefore, different data preprocessing techniques and mining algorithms need to be designed for different data acquisition methods even for the same mining task. Second, the mining tasks may vary in different application areas, and different additional data is required to help with mining tasks. For example, in a retail environment, the main motivation of customers' trajectory data mining is to help improve the profit of the business, while the mining of pedestrian data in public areas is mainly for public security or urban planning. Therefore, more specific algorithms need to be designed to help with decision-making.

To date, many trajectory data mining techniques have been proposed for solving problems in different application areas. In this section, we first review the related work of indoor human mobility analysis in terms of the corresponding trajectory acquisition techniques and two application scenarios, i.e., pedestrian analysis and customer behavior analysis. Then we review the related work of phone users' behavior analysis in mobile networks, in terms of the data acquisition techniques, data characteristics, application problems, and the corresponding solving techniques.

2.2.1 Indoor Human Mobility Analysis

Understanding human movement patterns in indoor environments, such as shopping malls, stores, and stations, plays an important role in public security, crowd management, or other specific applications. In this section, we review several trajectory data acquisition techniques in indoor environments and provide a brief review about two scenarios, pedestrian analysis and customer behavior analysis.

Trajectory data acquisition techniques

Several types of sensing technologies have been developed for tracking human movements in indoor environments, such as RFID tags, Bluetooth beacons, WLAN Received Signal Strength (RSS), and video cameras.

- *Video camera* Human tracking within the field of view of a camera generates the moving trajectories of human objects over time by locating their positions in each frame of a given video sequence. It has been applied in many applications such as public security, transportation control, and urban planning. For instance, video surveillance systems can be installed in public places, such as airports, subways, railway stations and shopping malls, which can help protect the security of public facilities and citizens and provide insights about the use of that infrastructure [124]. Another example is in a retail environment, it can be used to track customer behavior, which includes the movement, product handling, verbal contact with the store staff and display viewing [99]. It can not only provide path information, but other features of customers like demographic groups (gender, age, ethnicity) based on visual appearance, and the facial expressions of shoppers, which can capture their emotional reactions to the in-store environment. However, it may be limited to locations within the direct line of sight of the cameras.

- *WLAN RSS* WLAN based positioning techniques use the Media Access Control (MAC) addresses of mobile phones to track the movement path of customers, and have been commonly used in indoor localization systems [107]. This technique can be cost-effective as it uses existing infrastructure. The problems with this approach

are that the accuracy is not high and not everyone would use the Wi-Fi or would like to be tracked.

- *Bluetooth beacon* Beacons provide data about when a person or object is within range of a certain point (the beacon location). Beacons can be attached to the objects/walls in a store. Devices within the range of a single beacon can pick up the radio signal, and the distance between the device and beacon can be measured by the signal strength. Companies, such as Senion[1] and Crosscan[2], have adopted this techniques. Similar to WLAN, it has a low set up cost. It has been used in applications for environments such as museums and shops, for tracking users' movement behavior. For example, Pierdicca et al. [86] used Bluetooth beacon sensors to detect and locate mobile phones in retail environments, which can be used to generate a heat map of detected positions over time.

- *Radio-Frequency IDentification (RFID)* RFID systems consist of three components: a RFID tag or smart label, a RFID reader, and an antenna. RFID tags contain an integrated circuit and an antenna, which are used to transmit data to the RFID reader (also called an interrogator). The reader then converts the radio waves to a more usable form of data. RFID has been widely used for customer behavior analysis in retail environments [59]. The shopping paths of customers can be tracked by attaching tags to trolleys/baskets through the store. It has relatively high accuracy, but the setting up of the RFID system can be expensive, because both scanning and scanned devices need to be installed [21]. The sample of tracked customers may be biased, since this method may exclude customers without trolleys/baskets, and customers may also sometimes leave their carts behind to shop in the store aisles. Another drawback is that trolleys/baskets may not always be available in some kinds of shops, such as clothing shops and small-sized stores.

[1] https://senion.com/
[2] https://crosscan.com/

Scenario 1: Pedestrian Analysis in Public Area

Understanding the movement patterns of pedestrians in a public area can provide valuable guidance to city planning, public security and safety. Trajectory data can be easily obtained by video cameras in public areas, such as stations, sport stadiums or shopping malls. In the literature, many research questions have been studied, such as anomaly detection in public places, pedestrian movement prediction and layout design. We briefly introduce some of them as follows.

- *Anomaly detection* Anomaly or outlier detection is the search for individuals or groups who do not conform to an expected moving pattern. For example, Chong et al. [25] proposed a framework of hierarchical crowd analysis and anomaly detection. From historical trajectory data, the regions of interest are first learned based on a Hierarchical Dirichlet Processes grouping. Then the statistical template of the pedestrian distribution is learned for identifying observations that differ from this normal profile. Afiq et al. [1] provided a review of the major techniques applicable for classifying abnormal behavior in a crowded scene, including the use of Gaussian Mixture Models (GMM), Hidden Markov Models (HMM), optical flow methods and spatio-temporal techniques.

- *Pedestrian movement prediction* Pedestrian movement prediction aims to estimate pedestrians' future locations in terms of trajectories based on their previously observed movements. Yi et al. [124] proposed an algorithm for pedestrian behavior modeling by including stationary crowd groups as a key component. Given a source and a destination, the optimal walking path can be predicted by optimizing an objective function. A method for destination prediction was also proposed in the paper. In [14], a novel hierarchical Long Short Term Memory (LSTM) based network is proposed to consider both the influence of the social neighborhood and scene layouts.

- *Group pattern discovery* Group pattern discovery aims to efficiently discover moving objects that are found in close proximity to each other for a period of time. This can help with the understanding of pedestrian behavior, such as group gathering

and group dispersion, in public places. In [95], a time window based method was proposed to discover groups of pedestrians of varying group pattern semantics. At first, pairs of moving objects over time are discovered and then the evolving groups are identified by expanding pairs to larger groups. In [14], a sequential method is proposed to detect and track small groups (convoys). A two-phase algorithm was implemented which consists of a density clustering phase and an intersection phase.

- *Layout design* Layout design focuses on how to design facilities for pedestrians, such as the arrangement of walkways, exits, staircases and the shape of the rooms or corridors [98, 101]. For example, Sun et al. [101] conducted a series of pedestrian experiments to investigate the effectiveness of adding a funnel shape buffer zone in front of the bottleneck entrance of a subway station. The results show that the funnel shape can improve traffic efficiency at the bottleneck, especially under large pedestrian traffic volumes.

In Chapter 3, we propose to use an origin-destination matrix representation to describe pedestrian flow patterns, and a visual clustering based method is proposed to address the problem of anomaly detection of pedestrian flows.

Scenario 2: Customer Behavior Analysis in Retail Environments

The rapid growth of e-commerce has provided retailers with diverse sources of information about their online customers. This is has driven a substantial body of research into data analytics for modeling the behavior of on-line customers [52]. However, the growth of e-commerce has not entirely displaced traditional brick-and-mortar retailing, and physical stores are still a major channel of customers for many retail businesses [121]. The cost of staff and maintaining these physical retail outlets means that retailers need to constantly improve their stores to meet the needs of customers and increase revenue and profitability. Indoor location technologies can provide information about the customers' location, which can be used in advanced analytics and visualizations. In the following, we review some applications based on location data of customers in retail environments.

- *Customer segmentation* Customer segmentation addresses the discovery of customer behavioral segments, the analysis of segment characteristics and the utilization of segment data [121]. For example, some studies have focused on identifying typical in-store travel paths in supermarkets. A shopping path is the route a customer follows from the point of entering the store to the point of leaving the store. Trajectory clustering is a common technique adopted to solve this problem. Larson et al. [59] proposed an algorithm based on k-medoids clustering to report feasible centroid paths, which considered the spatial constraints in the store. Fourteen "canonical path types" are identified to represent typical grocery store travel paths. The study in [83] created a hierarchical clustering based on the stores in a shopping mall that customers visited.

- *Store layout optimization* By analyzing customers' movement patterns in a store, shop owners can design a better shop layout that can increase shopper traffic, conversion rates and profitability. In [54], a field experiment for 11 months was conducted to develop algorithms for store layout optimization. They compared the positioning data and transaction data before and after critical store layout optimization decisions to identify which customer movement patterns generated the highest sales.

- *Change detection of customer behavior* In [99], customers' behaviors during three time periods, mornings (8:00 - 12:00), afternoons (12:00 - 16:00) and evenings (16:00 - 20:00), are compared by using two types of visualization methods. The first method is called pixel models, which visualize all places in the store that have been visited at least once. The second method is called square models. The shop was divided into several equal-sized grids, and the cumulative number of visits in each grid is visualized using different grayscale tones.

- *Prediction and recommendation* Some customer behavior prediction/recommendation models have been proposed based on the movement path and transaction data. For example, Tsai et al. [106] proposed a shopping behavior prediction system based on rule matching by considering movement patterns and purchase transactions, where

store similarity and item similarity are considered in the prediction module.

- *Relationship between shopping path and purchase* By combining the shopping path data with sales data, shop managers can better understand the relationship between customers' visiting behavior and shopping behaviors, and help with decision making. In [57], to examine grocery shoppers' movement direction within the store and its influence on their shopping behavior (purchase value and walking distance), the authors introduced and tested two hypotheses. The results show that the shoppers who move in a clockwise direction spend more money and walk a longer distance than those who walk in an anti-clockwise direction. Other works have focused on finding the relationship between the shopping distance/stay time and the purchase amount, such as [53, 102].

However, many of the current works only focus on the trajectory data of customers to extract the movement patterns of customers. Some studies aim to find out the relationship between shopping paths and customers' purchase behavior. However, none of them focus on finding changes in their relationships by considering the trajectory data and sales data at the same time. In Chapter 4, we propose a new kind of pattern, conditional contrast patterns, based on contrast data mining techniques to find patterns that change significantly in one dataset but not in the other.

2.2.2 Phone Users' Behavior Analysis in Mobile Networks

The fast development of mobile networks and the popularity of mobile phones enable the availability of large volumes of mobile phone users' trajectory data, such as Call Detail Records (CDRs), GPS records and Global System for Mobile Communications (GSM) cell-tower data. Such trajectory data provide us with an opportunity to understand phones users' movement behavior, which can help with the optimization of network resources, urban planning and the improvement of a user's experience. In this section, we first review the data acquisition techniques in mobile networks, and then present a review on a variety of applications from two perspectives, i.e., network oriented and user oriented analysis.

Trajectory data acquisition techniques

- *GPS* GPS records the locations (latitude and longitude) of a user along with the timestamps when those location measurements were made. Although GPS data can normally provide data with a high frequency and accuracy, there are several limitations [69]: (1) it may become unavailable when the user is indoors or underground, (2) the GPS signal may be poor when using it in street canyons with tall buildings, (3) collecting GPS data continuously may consume devices' energy quickly, and (4) privacy is also an issue of using GPS data.

- *CDRs* CDRs provide information about calls/messages made over a mobile phone service, including the time when the calls/messages took place and the identity of the cell tower with which the phone was associated. However, CDRs are only generated when a call is made or a message is sent/received, which is a limitation for human mobility analysis. A report from Ofcom[3] shows that call volumes stayed almost stable from 2012 to 2018 and the average number of monthly messages per subscriber continued to decline every year, which may limit the utility of CDRs.

- *GSM cell-tower data* In recent years, Internet-connected mobile devices have penetrated every aspect of individuals' life, work and entertainment. According to a report from CISCO[4], by 2022 wired devices will account for 29 percent of IP traffic, and Wi-Fi and mobile devices will account for 71 percent of IP traffic [26]. The report from Ofcom also shows that the average monthly data volumes per mobile user increased by about 25% to 3GB per month in 2018. In a cellular network, each cell tower (base station) covers a small geographical area. If phone users access the Internet, their positions can be passively detected by the cell towers that provide Internet data to them. Thus, a mobile phone user's trajectory can be represented as a sequence of cell tower IDs with corresponding timestamps. The locations of cell towers can be obtained from some open databases of cell towers, which normally include the latitude and longitude coordinates for each cell towers. In addition, other information may be also available, such as what types of service the users

[3]https://www.ofcom.org.uk/research-and-data/multi-sector-research/cmr/interactive-data
[4]https://www.cisco.com/

Table 2.2: A comparison of different data types in mobile network for mobility analysis.

Data Type	Spatial Resolution	Temporal Resolution	# of Users	Accuracy	Availability
CDR	cell-level	low	large	low	easy
GPS	m	high	limited	high	hard
GSM	cell-level	medium	large	medium	easy
LSS	m	low	limited	low	hard

Table 2.3: Description of mobile network data records.

Field Name	Description
user ID	identity number of mobile phone users
cell ID	identity number of cell towers
latitude	latitude of the corresponding cell tower
longitude	longitude of the corresponding cell tower
application	application the user is using
service type	type of the service, such as email, instant message
timestamp	recording time

are using, the download/upload bytes, and the download/upload speed. Table 2.3 shows a data set description of this type of network data. Although GSM cell-tower data is similar to CDRs as they can both be collected by telecommunications companies, GSM cell-tower data are collected periodically, e.g., every 5 minutes, which usually has a much higher temporal resolution than CDRs. However, the spatial resolution can vary from several hundred meters to a few meters determined by the cell tower density of different areas.

- *Location Sharing Services (LSS)* Some researchers also use location information from LSS, such as Facebook, WeChat and Foursquare, to analyze human's movement behavior.

Table 2.2 summarizes a comparison of the properties of the aforementioned four data types in terms of resolution, the number of users under consideration, accuracy and availability for network operators.

Data preprocessing is necessary to make the mobile network data usable for mobility

analysis. One common problem is the oscillation phenomenon. An oscillation happens when the device intermittently switches between cell towers instead of connecting to the nearest cell tower [118]. One solution is to aggregate the cell stations, such as utilize the semantic tags of the cell towers or group cell towers that are close together into clusters [16, 96]. In order to avoid relying on other data sources and using cell tower location directly (not cell clusters), several techniques have been proposed to resolve oscillation for mobile phone traffic data [88, 118]. In [118], an algorithm called DECRE is proposed, which consists of four steps (Detect, Expand, Check, and REmove) to detect and remove oscillations. Qi et al. [88] analyzed five conditions that may cause oscillations and proposed a heuristic algorithm called SOL, based on three different time periods, i.e., Stable Period, Oscillation Period and Leap Period, to reduce oscillations.

Applications

This type of data can be applied in a variety of urban-related applications, such as mobility pattern mining [133], urban planning, traffic prediction [32,110,113,120,127] and social network structure. In [133], Zhang et al. found that cellular network data can provide a finer granularity of location and movement information, and investigated the difference of mobility properties between network data and CDR data. Qiao et al. [89] proposed a mobility analytical framework for big mobile data. There have been some reviews about mobile phone network data analysis. For example, Calabrese et al. [17] surveyed techniques related to the use of telecommunication data for urban sensing. Techniques proposed in the literature are categorized into two levels, individual level and aggregation level. Zhang et al. [126] have presented a comprehensive survey of the crossovers between deep learning and mobile wireless networking research.

In the following, we summarize the techniques for processing mobile phone network data in the literature and divide them into two categories, i.e., user-oriented analysis and network-oriented analysis.

Network oriented Network oriented analysis means that the focus is on analyzing the dynamics and properties of the whole network in order to understand and optimize the

network.

- *Network traffic prediction of each cell* One popular research direction is traffic load prediction for each cell. Normally, a dataset provides the throughput of some cells in one area, then prediction algorithms such as AutoRegressive Integrated Moving Average (ARIMA) [32, 120] or neural network [110, 113, 127] can be applied for the prediction of the traffic load in different cells or cell clusters.
- *Land use inference* Another application is land use inference. Understanding land usage is crucial for practitioners and researchers to perform urban analysis and planning. For example, Pei et al. in [85] proposed a semi-supervised fuzzy c-means clustering approach to infer land-use types using CDRs data. In [79], two land use patterns, commercial/business/industrial and residential, are discovered by applying non-negative matrix factorization.

User oriented Finding the mobility patterns of phone users in mobile networks, such as Origin-Destination (O-D) matrix inference [7, 16], home-work place detection [56], transport mode detection [6, 63], user segmentation [56, 132], and behavior (location and/or service type) prediction [58, 75].

- *Transport mode detection* Li et al. [63] referred to the road network to identify transportation modes of mobile users. Approaches for subway and train trip detection and bus trip detection are proposed separately. For example, based on a K-nearest reference system and a D meters coverage tower reference system, cell towers are defined as reference towers of the closest station. Then the sequence of cell tower ids is transformed into a sequence of bus/subway stations. Then the transport mode can be identified by matching it with the travel time schedule of the subway and trains. Bachir et al. [6] proposed to combine Bayesian inference and clustering for transport mode detection using CDR data. Features are extracted from both mobile and transport networks, and sectors are clustered to find the transport probabilities of each sector. Then Bayesian inference was adopted to compute trajectories' transport mode probabilities. In their subsequent work [7], the identified transport modes are utilized for the inference of O-D flows.

- *User segmentation* Ben-Gal et al. [10] analyzed trajectories of semantic locations to find users who have similar movement behavior or mobility lifestyle, even when they live in different areas. A grouping scheme, Lifestyle-Based Clustering (LBC) was proposed, where the mobility and movement of each user is represented by a Markov model, and the Jensen-Shannon distances were calculated among pairs of users to measure their dissimilarity.

- *Behavior prediction* The behavior prediction of mobile users includes the prediction of the service type and next place of a user [58]. For example, in [58], an online algorithm was proposed to learn routes between important locations and predict the next location of users. Lu et al. [75] proposed a sequential pattern mining method called Cluster-based Temporal Mobile Sequential Pattern Mine (CTMPSP-Mine). By using the discovered patterns, a prediction method based on similar users and temporal properties was proposed.

- *Underlying geographical corridors identification* A corridor can be treated as a pathway that is frequently traversed by a considerable number of mobile users. A three-phase approach based on grids was proposed in [136] to discover trajectory corridors using the Discrete Fréchet distance. Recently, the authors in [137] proposed a method for detecting a set of corridors from GPS trajectories using the Minimum Description Length (MDL) principle. In their subsequent work [138], a grid-based method was proposed to transfer trajectories into grid cell sequences and then mine frequent corridors from the grid cell sequences.

Most work has focused on GPS data or CDR data but not on the mobile network GSM data, and the methods they proposed cannot deal well with the heterogeneity of GSM data. In Chapters 5, 6 and 7, we propose several algorithms to deal with GSM trajectory data generated in mobile networks to help with network resource management and deployment. To be specific, in Chapter 5, we address the problem of corridor identification from GSM mobile network data by proposing a multi-scale trajectory clustering algorithm. Then, in Chapter 6, we propose a method to discover corridors that are significantly different during two different time periods based on contrast data mining to

help understand changes in mobile network traffic flows. In Chapter 7, we focus on how to utilize the data mining techniques to help improve the network performance. An adaptive optimization algorithm is proposed to improve the edge caching hit rate by combining the popularity of files and results from our prediction model.

2.3 Summary

In this chapter, we first provided a review of unsupervised learning techniques for trajectory data and then discussed three application scenarios, including pedestrians in public places, customers in retail stores and phone users in mobile networks, in terms of the data acquisition techniques and research questions. Although there have been several studies on different applications, there has been little research on the heterogeneous characteristics in trajectory data. In the following chapters, we focus on five research questions in three different scenarios. Meanwhile, the detailed literature review for existing data mining techniques in each specific research question will be included in each of the following five chapters.

Chapter 3

Trajectory Pattern Identification and Anomaly Detection of Pedestrian Flows Based on Visual Clustering

Extracting pedestrian movement patterns and determining anomalous regions/time periods is a major challenge in data mining of massive trajectory datasets. In this chapter, we apply contour map and visual clustering algorithms to visually identify and analyze areas/time periods with anomalous distributions of pedestrian flows. Contour maps are adopted as the visualization method of the origin-destination flow matrix to describe the distribution of pedestrian movement in terms of entry/exit areas. By transforming the origin-destination flow matrix into a dissimilarity matrix, the iVAT visual clustering algorithm is applied to visually cluster the most popular and related areas. A novel method based on the iVAT algorithm is proposed to detect normal/abnormal time periods with similar/anomalous pedestrian flow patterns. Synthetic and large, real-life datasets are used to validate the effectiveness of our proposed algorithms.

The publication arising from the work in this chapter is paper P1.

3.1 Introduction

There is growing interest in the problem of extracting useful information from massive trajectory datasets derived by various sensing methods. Understanding patterns of pedestrian movement is useful in applications such as pedestrian flow management, public security and safety. A major challenge in pattern analysis of pedestrian movement is how to discover and describe the movement patterns hidden in trajectories, and identify any misbehavior or interesting events.

The main approaches to trajectory data analysis and anomaly detection fall into the

category of trajectory data mining. To detect and recognize special events, two common approaches used to address this problem are statistical methods combined with classification and clustering-based methods. Many existing approaches to address this problem have the limitations that they focus on the details of individual trajectories, but do not consider the characteristics of the trajectory distribution.

Therefore, we address this limitation of existing approaches by proposing the use of contour maps and visual clustering. Contour maps are a very useful visualization tool for three-dimensional data, which we adopt to visually describe the connection between different subareas and describe the distribution of trajectories. Visual clustering methods such as Visual Assessment of cluster Tendency (VAT) and iVAT [13, 111] are proposed to visually assess the clustering tendency of a set of objects. By using the VAT/iVAT approach, we are able to visualize and determine the possible number of clusters of locations or the periods with similar activity, and then determine abnormal areas/days with significantly different trajectory distributions.

The main contributions of this chapter are as follows. First, we use a visualization method to describe pedestrian movement distributions in terms of their origin and destination points. By transforming the origin-destination flow matrix into a dissimilarity matrix, we visually cluster the most popular flows using the VAT/iVAT algorithms. The popular flow patterns are important for monitoring the safety and security of public areas. Second, we propose a novel method of detecting abnormal time periods with anomalous pedestrian trajectory distributions based on the results of the VAT/iVAT algorithms. By doing so, it is possible to detect the occurrence and impact of special events. Finally, we evaluate our methods and make relevant comparisons on a large, real-life dataset, the Edinburgh informatics forum database [78], and demonstrate the effectiveness of our proposed algorithms.

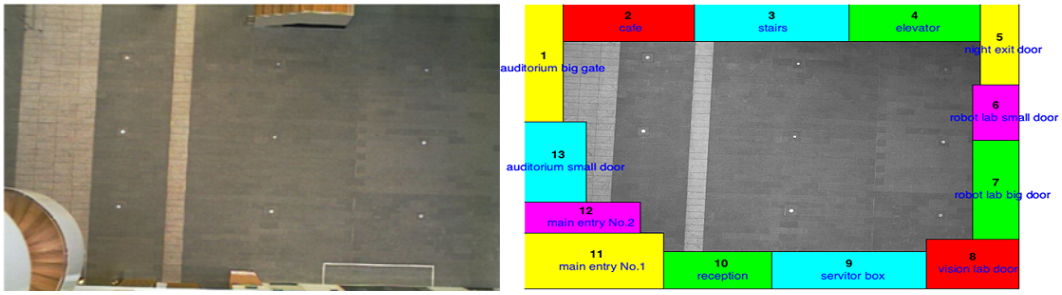
3.2 Related Work

An important aspect of a monitoring system is to detect significant events or unusual behavior in that environment. By mining pedestrian trajectories, it is possible to detect

the occurrence of major events, such as celebrations, parades, business promotions, accidents, disasters and others, which may be threats to public security or safety. With the increasing availability of big trajectory data, there have been various research methods on the detection and recognition of anomalous social events from trajectory data. In our work, we focus on the challenges of how to detect normal patterns of flows, and how to detect abnormal pedestrian flow patterns.

The first problem we address is to find and visualize the most visited subareas in a given region. In [44], an algorithm for extracting popular regions was proposed, with the popular regions defined as regions with trajectory densities greater than a given threshold, which is manually prescribed instead of adaptively adjusted. Therefore, we aim to find a parameter-free algorithm for this problem. Liu et al. [72] designed real-time analytical method for spatial-temporal data of daily travel patterns in metropolitan urban environments. The data for analysis of Liu et al. [72] are taxi traces and smart card records, whereas in contrast we mainly focus on pedestrian trajectories. While they provided an analysis on travel patterns, they did not consider the problem of detecting anomalies. Liu et al. [73] and Lu et al. [76] shared the same idea of extracting entry/exit points and using length analysis to derive indoor scene structure and identiy abnormal motion behaviors. However, the methods in [73, 76] only detect particular instances of trajectories. In our work, using the same dataset [73, 76], we focus on using the origin-destination matrix as a whole to describe the pedestrian flow patterns and cluster the most visited areas.

The second problem we address is how to detect a set of time periods with abnormal trajectory motion patterns. In the literature, there have been many research studies on anomaly detection methods for traffic or pedestrian data. Pang et al. [84] proposed a statistical model, which adapts likelihood ratio tests to find anomalous regions for monitoring the emergence of unexpected behavior based on GPS data from taxis. Chawla et al. [22] adopted Principal Component Analysis (PCA) to detect traffic anomalies from GPS data. In [22], moving activities of a crowd were simulated as the movements of a group of points, and the distribution of point groups is described with fractal dimensions. PCA was used to remove the disturbed factors from a feature vector and maintain only relevant information. Witayangkurn et al. [116] proposed a framework based on a



(a) Video image (b) Functional areas

Figure 3.1: Video image and functional areas of the Edinburgh informatics forum.

hidden Markov model to construct a pattern of spatial-temporal movement of people in each area in a grid during each time period. However, they focused on using changes in the population to detect anomalies. However, we intend to detect anomalous time periods of trajectory distributions based on the use of visual clustering methods, which can partition the data into clusters in a visual manner. In this way, we can provide a robust, unsupervised approach for clustering periods of normal pedestrian activity and visually highlighting periods of anomalous pedestrian activity.

3.3 Problem Statement

Our aim is to find the most popular routes of pedestrians in a given region and check which areas are the most related by visualizing the motion patterns, and to detect and visualize abnormal time periods by comparing the distributions of motion patterns. An outlier or anomaly in a dataset is considered to be an inconsistent observation (or subset of observations) compared with the remainder of that set of data, such as a substantial change in the popularity of pedestrian routes.

Suppose that the structure of the monitored area is known and can be divided into k subareas $A = \{A_1, A_2, ..., A_k\}$ according to the functions of different areas or some other predefined labeling. For example, as shown in Figure 3.1 (from the Edinburgh informatics forum database), there are 13 functional areas in this forum.

The dataset consists of a set of detected people walking through these subareas. Suppose we are given the trajectory data $T = \{T_{day_1}, T_{day_2}, ..., T_{day_m}\}$ covering a set of m days

$TP = \{day_1, day_2, .., day_m\}$, where on day_i a set of trajectories $T_{day_i} = \{T_{i1}, T_{i2}, ..., T_{in_i}\}$ has been collected, where n_i is the number of trajectories on day_i, and it is assumed that the trajectory data is the outcome of a robust tracking system. Each trajectory is composed of triplets (x, y, t) containing X/Y coordinates and sampling time $T_{ij} = \{(x_1, y_1, t_1), (x_2, y_2, t_2), ..., (x_l, y_l, t_l)\}_{ij}$, where l is the number of points in trajectory T_{ij}. Given a trajectory dataset T, our aim is to find and visualize the most visited subareas $A' \subset A$ and identify a set of time periods $TP' \subset TP$ with abnormal trajectory motion patterns.

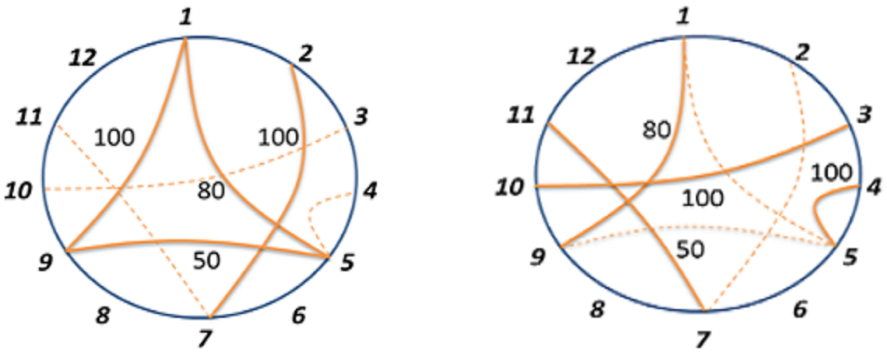
To address this aim, we examine three related research questions: (1) how to identify and summarize related sets of pedestrian flows over a given time period; (2) how to identify which time periods exhibit similar patterns of pedestrian flows; and (3) how to identify which time periods have experienced anomalous flow patterns? We present our approach to these research questions in the following three sections.

3.4 Case Study - Edinburgh Pedestrian Flow

The trajectory dataset from the University of Edinburgh is used as a case study. The dataset provides the scene under surveillance and its configuration, which covers most of the main hall, as shown in Figure 3.1. The most significant features of the hall are that there are many entry and exit points, i.e., the main entrance to the building, lifts, access to the Atrium, access to the second part of the hall, staircase, reception desk, and the four other exits, which means that there are a variety of possible pedestrian flows in this area.

The dataset consists of a set of detected targets of people walking through the Informatics Forum, the main building of the School of Informatics at the University of Edinburgh. The valid data covers 118 days of observation, resulting in about 90,000 observed trajectories in total. Substantial differences are observed between weekdays (Mon-Fri) and weekends. The average number of trajectories on weekdays is 932, which is significantly larger than the number of 140 on weekends.

Some papers [73, 76, 78] provided clustering methods to exact the pedestrian flows from trajectory data in terms of these entry and exit points. Based on the pedestrian trajectories detected from video images in Majecka et al. [78], which are represented by



(a) case A (b) case B

Figure 3.2: Two synthetic cases. Italic numbers from 1 to 12 represent 12 different areas. Lines and dashed lines between different areas represent the large pedestrian flows and small pedestrian flows respectively, and numbers next to lines indicate the size of pedestrian flows.

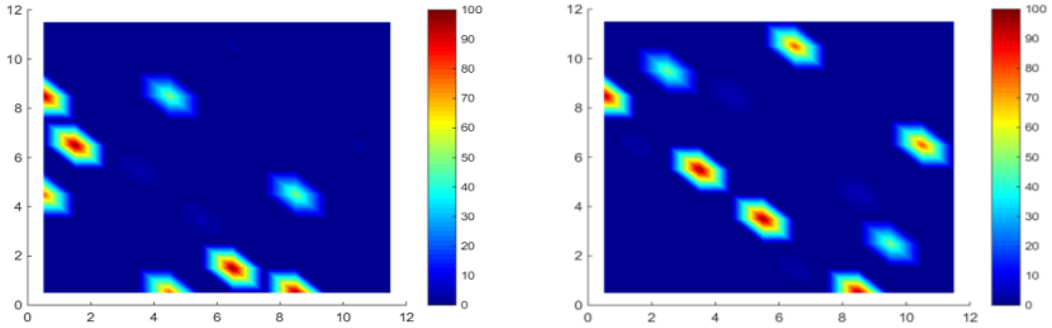
a sequence of centroid positions, we provide a visual clustering using contour maps and iVAT.

3.5 Summarizing Related Flows

The first question we address is how to identify and summarize related sets of pedestrian flows over a given time period. The challenges in addressing this question are how to summarize a given pedestrian flow matrix so that a user can identify the dominant flows, and then how to identify related subsets or clusters of flows. In this way, we can summarize the pedestrian activity over a given period of time. In this section, we illustrate methods to summarize a flow matrix and detect normal/abnormal days by identifying related flows.

3.5.1 Synthetic Cases

The rows and columns of a contour map reflect the order in which entry/exit pairs are labeled, but it does not reflect the clustering relationships between flows, so we would like to visually group related flows. Consider the following synthetic example.



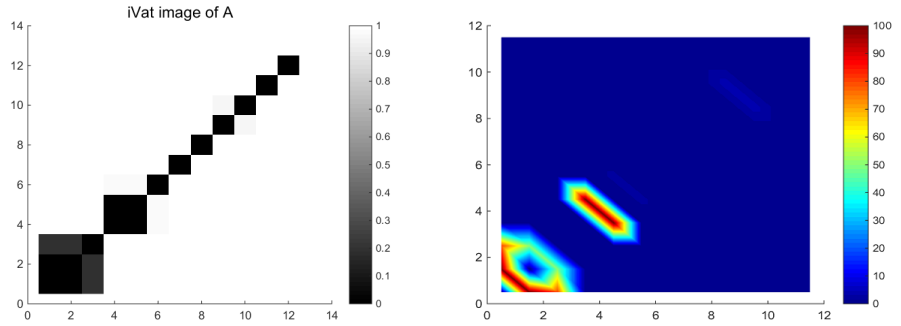
(a) case A (b) case B

Figure 3.3: Contour maps of the two synthetic cases in Figure 3.2.

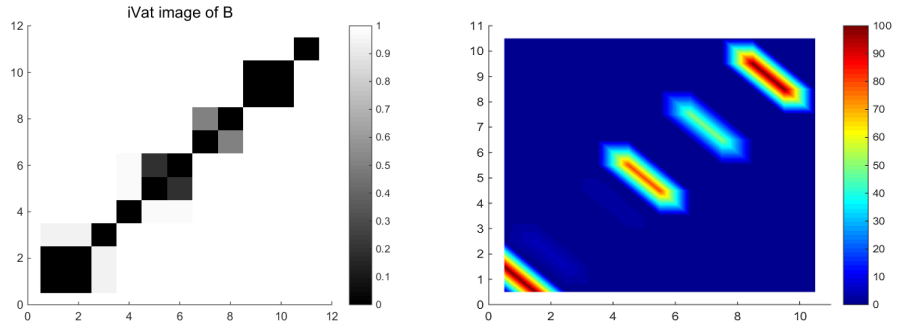
As shown in Figure 3.2, these two synthetic cases both have four relatively high pedestrian flows 50, 80, 100 and 100, but the relationships between flows are different. To visually display the origin-destination pair distribution characteristics, we introduce contours to represent the flow matrix. Contours indicate equal valued regions with the same color. This is similar to a heat map, and the characteristics of the matrix distribution can be visually analyzed. The application of contours helps us better visualize and compare the distribution characteristics of the origin-destination flow matrix, which would otherwise be hard to analyze only by the values of the matrix. The contour maps of the two cases are shown in Figure 3.3 respectively.

Although we can easily find the distribution of trajectories flows and detect origin/destination pairs that have high pedestrian flows in Figure 3.3, we cannot easily identify related flows, since the light areas are scattered on the contour map. The main challenge is how to reorder the rows/columns of a contour map to group related flows. In our work, we propose to treat this as a visual clustering problem.

VAT and iVAT are useful tools for visual assessment of clustering tendency, as is shown by Bezdek et al. [13] and Wang et al. [111]. The VAT algorithm displays a reordered dissimilarity matrix D as a gray-scale image with a modified version of Prim's minimal spanning tree algorithm. The iVAT algorithm augments VAT by applying a path-based distance transform to the input dissimilarity data before VAT images are made. It reorders the dissimilarity matrix of the given set of objects so that it can display any clusters as dark blocks along the diagonal of the image, and a diagonal dark block appears in



(a) iVAT-A (b) contour map-A



(c) iVAT-B (d) contour map-B

Figure 3.4: Results on synthetic examples.

the iVAT image only when a tight group exists in the data. We only provide the results based on the iVAT algorithm. The main steps of iVAT are:

Step 1: Transform input dissimilarity matrix $D \rightarrow D'$ using a path-based distance;

Step 2: VAT is applied to reorder $D' \rightarrow D'^{*}$, resulting in an iVAT image $I(D'^{*})$ whose $(i,j)^{th}$ element is a scaled dissimilarity value between objects o_i and o_j.

Since a dissimilarity matrix D is the input data to the iVAT algorithm, a method to transform the origin-destination matrix F to a dissimilarity matrix D is proposed in our approach. Considering that the origin-destination flow matrix F is non-symmetric (F_{ij} and F_{ji} may be different), the first step is to transform the flow matrix to be symmetric. There are three methods to derive a symmetric matrix S: (1) $S_{ij} = S_{ji} = max(F_{ij}, F_{ji})$; (2)$S_{ij} = S_{ji} = min(F_{ij}, F_{ji})$; (3) $S_{ij} = S_{ji} = (F_{ij} + F_{ji})/2$.

This symmetric flow matrix S can be normalized by using $S'_{ij} = S_{ij}/S_{max}$, where S_{max} is the value of the largest element in S. Then we can compute the dissimilarity matrix

D. If $i \neq j$, $D_{ij} = 1 - S'_{ij}$; otherwise, $D_{ij} = S'_{ij}$, where S' is the normalized symmetric transferring matrix. When $i = j$, the dissimilarity between the same area is 0; when $i \neq j$, the dissimilarity between two areas decreases as the normalized symmetric flow matrix S' increases, which means that high pedestrian flows result in low dissimilarity values, and vice versa.

To verify the effectiveness of iVAT, we apply it to the two synthetic examples. The iVAT image results and reordered contour maps are in Figure 3.4. The reordering of these two cases are both Areas {1 9 5 2 7 11 3 10 4 6 8 12}. However, the clustering results are different. For case 1, the clustering is {(1 9 5) (2 7) 11 3 10 4 6 8 12}, i.e., Areas 1, 9 and 5 are strongly related and also 2 and 7. For case 2, the clustering is {(1 9) 5 2 (7 11) (3 10) (4 6) 8 12}. The results indicate that iVAT can cluster the related areas correctly.

3.5.2 Case of Real Trajectory Data

Next, we test our method of visual clustering on the real trajectories from one Sunday (20-Jun-2010). We assume that the structure of the scene is known, so we can classify trajectories based on the location of their first (start) and last (end) regions. For example, given $T = \{(x_{start}, y_{start}, t_{start}), ..., (x_{end}, y_{end}, t_{end})\}$ with $(x_{start}, y_{start}, t_{start}) \in A_{11}$ and $(x_{end}, y_{end}, t_{end}) \in A_6$, then T belongs to (11,6).

By counting all the trajectories, we obtain an origin-destination flow matrix (i.e., the frequency-adjacency matrix) F. In this matrix, the value of $F(i, j)$ represents the number of trajectories which start at A_i and end at A_j. Note that the origin-destination matrix can be asymmetric, e.g., Table 3.1 on Sunday (20-Jun-2010). Applying the iVAT algorithm, there are 12 clusters for all 13 areas, as is shown in Figure 3.5. The clustering result is {1 (11 2) 8 3 5 12 4 10 13 6 7 9}.

Using the iVAT results, we obtain the reordered origin-destination flow matrix, which is shown in Table 3.2. For most origin-destination pairs, there are few trajectories between them (S_{ij} is small compared with S_{max}), leading to $S'_{ij} \sim 0$ and $D_{ij} \sim 1$. Thus, most clusters/area groups contain only one area, except the cluster containing A_{11} and A_2, which means Areas A_2 and A_{11} have high pedestrian flows and they are most related to each other.

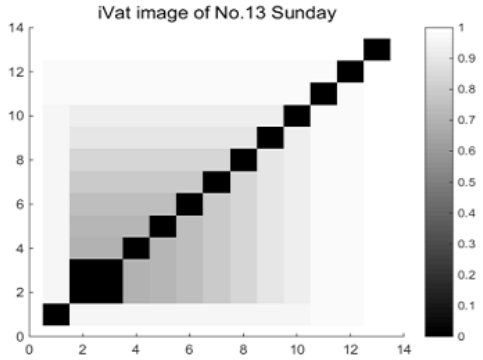


Figure 3.5: iVAT image on the No.13 Sun.

Table 3.1: Origin-destination flow matrix.

Area	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
13	0	8	0	0	0	0	0	2	0	0	0	0	0
12	1	17	0	0	0	0	0	0	0	0	3	0	0
11	4	102	5	2	0	0	0	11	0	11	0	4	1
10	0	3	1	0	3	0	0	0	0	0	11	1	1
9	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	16	29	0	6	0	2	0	0	0	14	0	3
7	0	0	0	0	1	0	0	1	0	0	0	0	0
6	0	0	0	0	1	0	0	0	0	0	2	0	0
5	0	7	26	17	0	0	0	4	0	1	0	0	0
4	0	3	8	0	13	0	0	0	0	0	1	0	0
3	0	1	0	8	25	0	0	16	0	2	20	0	0
2	0	0	2	5	5	0	0	32	0	4	103	21	2
1	0	2	0	0	0	0	0	0	0	0	0	0	0

Table 3.2: Reordered origin-destination matrix.

Area	**1**	**11**	**2**	**8**	**3**	**5**	**12**	**4**	**10**	**13**	**6**	**7**	**9**
9	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	1	0	1	0	0	0	0	0	0	0
6	0	2	0	0	0	1	0	0	0	0	0	0	0
13	0	0	8	2	0	0	0	0	0	0	0	0	0
10	0	11	3	0	1	3	1	0	0	1	0	0	0
4	0	1	3	0	8	13	0	0	0	0	0	0	0
12	1	3	17	0	0	0	0	0	0	0	0	0	0
5	0	0	7	4	26	0	0	17	1	0	0	0	0
3	0	20	1	16	0	25	0	8	2	0	0	0	0
8	0	14	16	0	29	6	0	0	0	3	0	2	0
2	0	103	0	32	2	5	21	5	4	2	0	0	0
11	4	0	102	11	5	0	4	2	11	1	0	0	0
1	0	2	0	0	0	0	0	0	0	0	0	0	0

3.6 Identifying Time Periods with Similar Flows

The second question we address is how to identify which time periods exhibit similar patterns of pedestrian flows. The challenges in addressing this question are how to compare the flow patterns of different time periods, and how to identify which time periods have similar flow patterns. This enables users to profile normal activity.

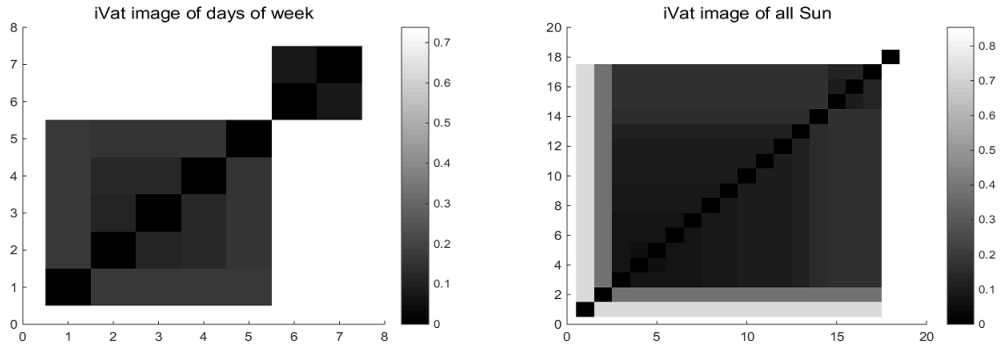
3.6.1 Comparing Flow Patterns

Given flow matrices F_i from time period T_i and F_j from time period T_j, we require a measure of how similar are the flows between these two time periods, i.e., we require a distance measure $d(F_i, F_j)$. We use the Frobenius norm, $\|F_i - F_j\|_F = \sqrt{Tr\left[(F_i - F_j)(F_i - F_j)^T\right]}$, which reflects the pairwise difference of individual flows between the same pairs of location, where $(F_i - F_j)^T$ is the transpose of $F_i - F_j$, and Tr means the trace of the matrix. For example, given $F_i = \begin{pmatrix} 0 & 5 \\ 3 & 2 \end{pmatrix}$ and $F_j = \begin{pmatrix} 2 & 1 \\ 2 & 4 \end{pmatrix}$, then the Frobenius norm $\|F_i - F_j\|_F = \sqrt{Tr\left[\begin{pmatrix} 20 & -10 \\ -10 & 5 \end{pmatrix}\right]}$.

3.6.2 Identifying Similar Time Periods

Given a set of flow matrices $F = \{F_1, F_2, ..., F_m\}$ corresponding to m different time periods $T_1, T_2, ..., T_m$, we would like to group or cluster these flow matrices so that we can identify which time periods have similar flow patterns. For example, if F contains seven flow matrices, each corresponding to the average flows on each day of the week, then we would like to detect F in order to identify which days have similar pedestrian traffic.

To achieve this goal, we again make use of the iVAT algorithm. First, we create a $m \times m$ distance matrix D_F, where the $(i, j)^{th}$ entry in D_F is $d(F_i, F_j) = \|F_i - F_j\|_F$. The distance matrix can be normalized by using $norm(D_F) = (D_F - min(D_F))/(max(D_F) - min(D_F))$, where $min(D_F)$ and $max(D_F)$ are the minimum and maximum values in D_F respectively. We then reorder the normalized D_F using iVAT to produce D'_F, which should visually



(a) 7 days (b) all Sun

Figure 3.6: iVAT image.

reorder the clusters of time periods with similar flow patterns.

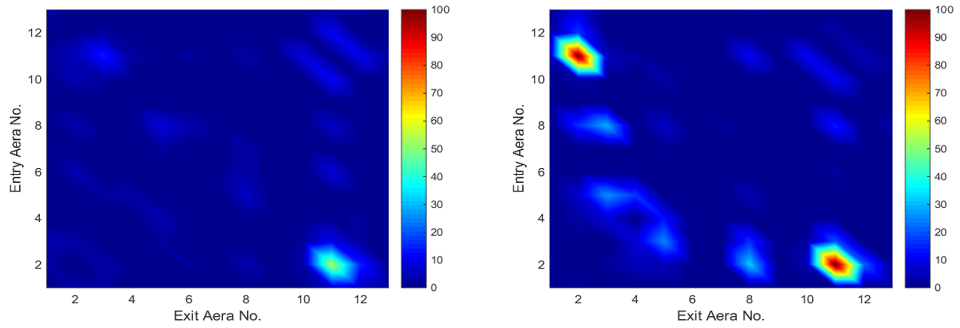
We evaluated our proposed method on the data set of 118 days, which has been classified into seven groups, corresponding to 7 days of the week. Then the averaged flow matrices of the 7 days of the week (7 samples, i.e., Sun, Mon, Tue, Wed, Thu, Fri, Sat) are compared. The iVAT results are shown in Figure 3.6a.

The ordering of the iVAT image is {(4 6 3 5 2) (1 7)}, and it shows two clusters, corresponding to clusters of weekdays (Wed, Fri, Tue, Thu, Mon) and the weekend (Sun, Sat).

3.7 Identifying Anomalous Flow Patterns

Once we have a profile of normal flow patterns over different periods of time, our final question is how to identify which time periods have experienced unusual or anomalous flow patterns. The challenge in addressing this question is how to identify individual time periods in which the pedestrian flows significantly differ from what is expected. This enables users to detect when an anomaly has occurred, and to analyze how the pedestrian flows during that time period differ from what is expected.

Given a set of flow matrices $F = \{F_1, F_2, ..., F_m\}$, the aim of visual anomaly detection is to detect a subset of these flow matrices that are anomalous or outliers compared to the rest. To be specific, if the number of objects in a cluster is fewer than a user-defined



(a) No.4 Sun (b) No.13 Sun

(c) No.14 Sun (d) average Sun

Figure 3.7: Contour of abnormal and normal Sundays.

threshold δ_a, then these objects are defined as outliers. As before, we use the Frobenius norm to compare flow patterns from different time periods, and construct D_F. We then reorder the distance matrix D_F using iVAT to generate D'_F. Here we set $\delta_a = 1$, which means that if a time period does not strongly belong to any cluster, then it is anomalous. When we visualize D_F, any anomalous time periods should appear as singleton dark blocks, which are significantly different from the larger clusters in F. For example, consider the set of flow matrices for all Sundays, the iVAT result is shown in Figure 3.6b.

The iVAT result shows that Sunday has four clusters {(13) (4) (8 15 16 5 9 17 18 3 12 11 7 10 1 6 2) (14)}, which means that the anomalous time periods are $TP' = \{Sun_{13}, Sun_4, Sun_{14}\}$. The average of normal Sundays contour map, and No.4, No.13 and No.14 Sunday contour maps are shown in Figure 3.7.

The contour maps indicate that each of the three anomalous Sundays has different high value regions, and all these three time periods are significantly different from the

distribution of the average of normal Sundays. For example, in Figure 3.7(b), the top left area and lower right area are very bright, which indicates that there are lots of people moving between A_2 and A_{11} as the most visited areas, corresponding to the two largest values $F(11,2) = 102$ and $F(2,11) = 103$ in the origin-destination flow matrix respectively. Some other areas are rather dark, indicating few people moving between these area pairs. There are some regions with relatively bright color, indicating relatively high value of trajectory numbers between corresponding area pairs, e.g., $A_2 \rightarrow A_8$, $A_5 \rightarrow A_3$, $A_8 \rightarrow A_3$. Similar analyses can be applied to other days of week.

3.8 Conclusion and Future Work

In this chapter, we use the origin-destination matrix to discover and characterize the connectivity between places or regions. In order to find and visualize related areas, we introduce a contour map to represent the origin-destination flow matrix, and propose a visual and parameter-free area clustering method based on the VAT/iVAT algorithms. To detect and visualize abnormal days with significantly different flow patterns, an iVAT based method is also developed. The results on synthetic data and the Edinburgh informatics forum database show that our methods can effectively cluster related areas and identify normal/abnormal pedestrian flow patterns. Possible future research directions are to discuss on the scalability of the method on large data and to modify the proposed method for data stream analysis.

In the next chapter, we focus on how to extract interesting and actionable patterns using not only a trajectory dataset but also other associated datasets.

Chapter 4

A Pattern Tree Based Method for Mining Conditional Contrast Patterns of Multi-Source Data

In the previous chapter, we studied how to detect anomalies from pedestrian trajectory data. In this chapter, we focus on finding significant changes from multi-source data based on contrast mining techniques. Contrast patterns are itemsets that frequently occur in one dataset while not in another. These patterns have been successfully applied to many data mining domains, such as prediction, classification and clustering. However, none of the previous studies has considered extracting contrast patterns from different types of datasets. Therefore, we introduce a new type of contrast pattern, CCPs (Conditional Contrast Patterns), which are a subset of traditional Contrast Patterns in one kind of dataset conditioned on a property of these patterns in another kind of dataset. Accordingly, we propose an algorithm based on tree search for mining Conditional Contrast Patterns (CCPs), which can compress the datasets into a tree representation. We evaluate our proposed method in comparison with two other methods (Brute force and Apriori-based methods) on a synthetic dataset as well as a real-life retail dataset. The results show that CCPs are more informative and actionable for decision makers than normal Contrast Patterns (CPs), and our tree-based algorithm has the best performance in terms of efficiency.

The publication arising from the work in this chapter is paper P2.

4.1 Introduction

Contrast data mining aims to quantify and describe the differences between two given multivariate datasets, where a contrast pattern is defined as a pattern that occurs frequently in one dataset and infrequently in the second dataset [9]. Finding such changes or contrast patterns in datasets is useful in many applications, such as urban traffic man-

agement [62, 114], medical diagnosis [41], customer analysis [97, 121], and anomaly detection [65]. However, current research in the literature cannot answer questions like *If we have data from multiple types of data sources, how can we extract interesting patterns by considering different datasets at the same time?* or *Can we find patterns where there has been a big change in one type of dataset but little or no change in the other?*

There have been many contrast pattern mining methods proposed in the literature, such as border based [31], tree structure based [37], and Zero-Suppressed Binary Decision Diagrams [74]. The border based method can represent many Emerging Patterns (EPs) (a kind of contrast pattern) by using borders. The tree structure based method is inspired by the Frequent Pattern tree method [49]. In [74], the Zero-suppressed Binary Decision Diagram (ZBDD) method was proposed to deal with high dimensional data. However, none of them has considered mining contrast patterns by utilizing different kinds of data sources.

Alternatively, to produce a high-quality representation and improve the generalization performance [128], one can use datasets from multiple sources. Most existing multi-source (or multi-view) learning methods mainly focus on how to incorporate knowledge extracted from multiple databases [92]. In that approach, different types of data from multiple sensors are integrated to obtain more detailed information using fusion techniques, instead of extracting contrast patterns. Therefore, to benefit from multiple types of data sources in contrast pattern mining we introduce a new type of pattern, called CCPs, which aims to find more interesting contrast patterns by utilizing supplementary data sources (i.e., different types of data) and conditioning contrast factors. Note that we focus on extracting patterns from data generated from multiple data sources, but not data with heterogeneous data types.

One typical application of our proposed CCP is customer analysis. For example, in a retail environment, shop managers may be interested in finding the relationships or differences between sales data and customers' behavior. If we use traditional contrast data mining techniques, we can only use one type of data, e.g., sales data, and find combinations of products that are frequently purchased together on one day but not on another day. However, by using conditional contrast pattern mining, we can make use of both

sales data and customers' behavior data to find sets of products where there has been an increase in sales but little or no increase in the number of customers, which represents a stronger buying intention of customers. These patterns can be more interesting and helpful for decision makers.

Therefore, we propose a Conditional Contrast Pattern tree (CCP-tree) search method, which has two steps: (*1*) CCP-tree construction, and (2) CCP mining. In the first step, we build a CCP-tree by scanning every instance in the datasets. A CCP-tree, similar to the frequent pattern tree, builds a compressed representation of the input data. Then each instance in the datasets can be mapped onto a path in the tree, and the node in the CCP-tree has fields to record the counts in different datasets. In the second step, by using depth-first search and applying three predefined subjective conditions of a CCP, we are able to extract all the CCPs. These conditions ensure that the patterns we find are not only the CP in one kind of data, but also exhibit the required properties in another kind of data.

Contributions. The main contributions of this chapter are as follows:

- We introduce and define a novel class of pattern, called a Conditional Contrast Pattern (CCP). Three conditions for mining CCPs are also proposed. This class of pattern differs from other existing types of patterns - it is a subset of contrast patterns, and it can be more *informative* and *instructive* for decision makers since it utilizes *information from additional related data sources*.

- We propose a conditional contrast pattern tree based method for mining CCPs. The counts in all datasets are recorded by the CCP-tree. This method is complete, meaning that all itemsets satisfying our conditions can be found using our method.

- The proposed method is efficient. The complexity of mining CCPs does not depend on the number of instances.

- We evaluate our method and compare its performance with two baseline methods on a synthetic dataset. The results demonstrate better efficiency of our method in comparison with the other two methods.

- We also apply our method to one real-life retail dataset, which includes the transaction data and customer behavior of each customer in a retail store in Melbourne. The results show the practicality of our proposed method.

4.2 Related Work

Contrast patterns are often defined as patterns whose support differ significantly among the datasets that are under study, which were first proposed by Bay and Pazzani [9]. If the supports of all the frequent itemsets in two datasets are very similar, then we can say there is no significant change between these two datasets, and vice versa. These patterns have been successfully applied to many areas. For example, in the work of [114], frequent emerging networks are detected to discover the impact of road closures on traffic flows.

In the literature, various kinds of contrast patterns have been proposed, such as Jumping Emerging Pattern (JEP) [64], Strong Jumping Emerging Pattern (SJEP) [37] and Fuzzy Emerging Pattern (FEP) [42]. Jumping emerging patterns [64] are defined as itemsets whose support changes suddenly from zero in one dataset to nonzero in another dataset, i.e., the growth rate is infinity. Strong jumping emerging patterns [37] are subsets of emerging patterns, such that the support of the second dataset should be above a threshold and the subsets of a SJEP are not a SJEP. The authors in [37] also proposed NEPs (Noise-tolerant EPs) and GNEPs (Generalized NEPs). In [42], FEPs are proposed as a combination of the concepts of fuzzy logic and emerging patterns [31][30]. In [24], shared emerging patterns are proposed for mining the similarity of two datasets.

A key challenge for mining contrast patterns is how to reduce the computational complexity. A brute force approach for mining contrast patterns is to enumerate all combinations of items and calculate their respective supports in each dataset, and then find contrast patterns according to the change in these supports. However, this method is not efficient because of the number of candidate itemsets. Therefore, in the literature, many different kinds of contrast mining methods have been proposed in terms of the data structures used in the mining algorithms. These contrast pattern algorithms can be categorized into: (i) border based [31], (ii) tree based [37], and (iii) ZBDD based [74]. In

[31], a suite of emerging patterns mining algorithms is proposed, which discover a class of EPs by manipulating only the borders of two collections. This algorithm can finish quickly even when the number of EPs is large since the discovered EPs can be represented using the borders. Tree-based contrast data mining is based on the use of a tree structure [30]. This method takes advantage of a tree representation to compress the input datasets by sharing patterns with common prefixes. Therefore, it only searches itemsets that are known to occur in the dataset. In several studies [37], this method has been found to work well in practice. A ZBDD based method is proposed in [74]. This method can store input data or output patterns in a highly compressed form and is advantageous for mining high dimensional datasets.

Although many different kinds of contrast patterns have been proposed in previous work, none of the methods has considered utilizing multiple types of data sources. In our study, we mainly focus on how to find interesting contrast patterns from multiple data sources. In the information domain, most information fusion techniques only focus on how to fuse data from multiple sources [135]. However, little work has been done on how to find contrast patterns based on other types of data sources.

4.3 Problem Statement

Finding interesting patterns is important to decision makers, especially from multiple types of data sources. Our aim is to find interesting contrast patterns by utilizing additional related data sources. Contrast patterns are itemsets whose support change significantly from one dataset to another. In our work, we are interested in finding contrast patterns that change significantly in one data source, while the change in another data source is not notable. We introduce a new concept, a Conditional Contrast Pattern (CCP), which consists of a baseline and a set of conditional contrast factors, where small changes make big differences.

Suppose we have two datasets from different sources, $D = \{D_1, D_2\}$ and $S = \{S_1, S_2\}$, where subscripts $1/2$ indicate negative/positive datasets respectively. For instance, in retail environments, the first dataset D is retail transaction data, which includes datasets

Table 4.1: Transaction data in D_1 (Morning of 11-10-15).

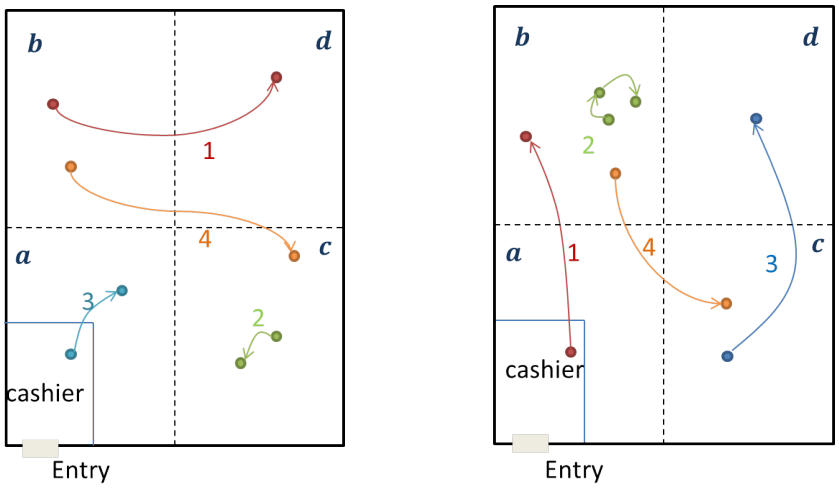
time-date	Item code	Zone code
10:20:20 11-10-15	B1	b
10:20:20 11-10-15	C2	c
10:30:35 11-10-15	B2	b
10:55:35 11-10-15	A2	a
10:55:35 11-10-15	B3	b
11:30:01 11-10-15	D2	d

Table 4.2: Transaction data in D_2 (Afternoon of 11-10-15).

time-date	Item code	Zone code
15:25:07 11-10-15	A2	a
15:25:07 11-10-15	B2	b
15:25:07 11-10-15	C2	c
15:25:07 11-10-15	D1	d
16:05:01 11-10-15	A1	a
16:05:01 11-10-15	B1	b
16:05:01 11-10-15	D1	d
16:35:03 11-10-15	A1	a
17:02:01 11-10-15	A1	a
17:02:01 11-10-15	B3	b

D_1 and D_2. Table 4.1 and Table 4.2 provide examples for these two datasets respectively. Each record contains the time, item code and zone code. As shown in Table 4.1, there are four transactions generated in the morning of 11-10-2015. The second dataset S contains spatial data, which records the shopping path of each customer. Datasets S_1 and S_2 are shown in Figure 4.1a and 4.1b respectively. In the morning, four customers are detected, e.g., customer 1 visited zones b and d, while customer 4 visited zones b and c. Here we only considered the zones where the customer stayed longer than a certain dwell-time threshold.

Let $I = \{i_1, i_2, ..., i_n\}$ be the set of all items in each type of dataset. Each instance in the datasets contains a subset of items chosen from I, and a collection of items is defined as an itemset X. For example, in the example shown in Table 4.3, there are 4 items in the itemset, $I = \{a, b, c, d\}$. Instance $\{b, c\}$ in dataset D_1 means that one customer bought products of categories b and c in time period 1, while another instance $\{b, c\}$ in dataset S_1 means in time period 1 a customer visited zones where products of category b and c are



(a) customers trajectories in S_1 (Morning of 11-10-15)

(b) customers trajectories in S_2 (Afternoon of 11-10-15)

Figure 4.1: An example of the trajectories of customers.

Table 4.3: Example of datasets of two time periods from two different data sources.

D_1	D_2	S_1	S_2
{b, c}	{a, b, c, d}	{b, d}	{a, b}
{b}	{a, b, d}	{c}	{b}
{a, b}	{a}	{a}	{c, d}
{d}	{a, b}	{b, c}	{b, c}

sold. Here, we should notice that $\{b, c\}$ in D_1 and $\{b, c\}$ in S_1 probably track two different customers. There are 15 possible candidate itemsets. They are $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$, $\{a, b\}$, $\{a, c\}$, $\{a, d\}$, $\{b, c\}$, $\{b, d\}$, $\{c, d\}$, $\{a, b, c\}$, $\{a, b, d\}$, $\{a, c, d\}$, $\{b, c, d\}$, and $\{a, b, c, d\}$. Here notice that each line in Table 4.3 shows just one record in each dataset, and the records in the same line are not necessarily from the same customer.

The count of an itemset is the number of instances containing it in the dataset. Here the support of an itemset $supp(X)$ is defined as the absolute occurrence frequency (i.e., the count of an itemset) instead of the definition of relative proportion used in some papers. The counts of all the itemsets are shown later in Table 4.4.

Table 4.4: Counts of all itemsets of four datasets.

	a	b	c	d	ab	ac	ad	bc	bd	cd	abc	abd	acd	bcd	abcd
D_1	1	3	1	1	1	0	0	1	0	0	0	0	0	0	0
D_2	4	3	1	2	3	1	2	1	2	1	1	2	1	1	1
S_1	1	2	2	1	0	0	0	1	1	0	0	0	0	0	0
S_2	1	3	2	1	1	0	0	1	0	1	0	0	0	0	0

Definition 4.1. *The growth rate of an itemset in D is:*

$$gr(X,D) = \begin{cases} \frac{supp(X,D_2)}{supp(X,D_1)} & supp(X,D_1) \neq 0 \\ \infty & supp(X,D_1) = 0 \;\&\; supp(X,D_2) \neq 0 \\ 1 & supp(X,D_1) = 0 \;\&\; supp(X,D_2) = 0 \end{cases} \tag{4.1}$$

Definition 4.2. *The ratio of an itemset between D and S is:*

$$Ratio_{D|S}(X) = \begin{cases} \frac{gr(X,D)}{gr(X,S)} & gr(X,S) \neq 0 \\ \infty & gr(X,S) = 0 \;\&\; gr(X,D) \neq 0 \\ 1 & gr(X,S) = 0 \;\&\; gr(X,D) = 0 \\ 1 & gr(X,S) = \infty \;\&\; gr(X,D) = \infty \end{cases} \tag{4.2}$$

Definition 4.3. *Given two different kinds of datasets $D = \{D_1, D_2\}$ and $S = \{S_1, S_2\}$, where D and S both contain one negative and one positive dataset, a conditional contrast pattern from D_1 to D_2 is an itemset X that satisfies the following conditions:*

C1. $Supp(X, D_2) \geq min_{supp}$;
C2. $Growth(X, D) \geq min_{growth}$;
C3. $Ratio_{D|S}(X) \geq min_{ratio}$

The first condition C1 means the support of CCP X is greater than a user-defined threshold min_{supp}. This constraint ensures that the pattern should be contained in a minimum number of transactions in dataset D. Condition C2 guarantees that the support of the pattern increases significantly in dataset D. The first two conditions make sure that

the patterns we find are CPs in dataset D. In the last condition, we constrain the ratio between the growth rates of patterns in datasets D and S. This ensures the growth rate of X in dataset D is significantly greater than its growth rate in dataset S.

Our aim is to find all patterns whose support increased significantly in the transaction dataset, while not changing much in the spatial dataset. For example, for itemset $\{a\}$, its growth rate in dataset D is $gr(\{a\}, D) = 4/1 = 4$, while in dataset S, its growth rate is only 1, which means there is no significant increase in the number of visits to the area selling product a. Therefore, $\{a\}$ is regarded as a CCP. Knowledge of such a product is important as it represents a product when the conversion rate of customers has improved. For itemset $\{c, d\}$, the growth rate of it in dataset D is $gr(\{c, d\}, D) = 1/0 = \infty$, and the growth rate of it in dataset S is still ∞. It is not a CCP, since it does not satisfy condition $C3$.

4.4 CCP Tree-based Method

In this section, we propose a novel method for mining Conditional Contrast Patterns (CCPs) based on a CCP tree, which is similar to a Pattern tree [37]. The main difference is that in a CCP tree we need to record values for two different kinds of datasets. There are two steps in our algorithm: constructing the CCP tree and finding CCPs. First, we provide the definition of a CCP tree.

4.4.1 Definition of a CCP tree

A CCP tree is a compressed representation of the input data. It can be constructed by reading the data sets one transaction at a time and mapping each transaction onto a path in the tree. Since different transactions may have common items, their paths may overlap. Figure 4.2 shows a CCP tree constructed by using the datasets in Table 4.3. Each node N in the CCP tree contains an ordered set of items. Suppose we have m items in node N, then each node can be denoted as $N.items[i], i = 1, ..., m$. Each item has 6 fields: $items.name$, $item.countD1$, $item.countD2$, $item.countS1$, $item.countS2$ and $item.child$. In the field of an item, $item.name$ records the item name; $item.countD1$, $item.countD2$, $item.countS1$ and

*item.countS*2 are the count of the item in datasets D_1, D_2, S_1 and S_2 respectively; pointer *item.child* stores all the children nodes of the current item. The root of the CCP tree is also a node that contains items, which is different from the null root for an FP tree.

4.4.2 Construction of a CCP tree

To illustrate the process of constructing a CCP tree, we use the example provided in Table 4.3.

(i) Datasets D_1 and D_2 are scanned once to determine the support count of each singleton item. Infrequent items can be discarded in this process. The remaining items are sorted in descending order of support counts, and the item names and support information are stored in a header table. If we set the minimum support to 1.0, we can obtain the ordered item list: $b:6 \rightarrow a:5 \rightarrow d:3 \rightarrow c:2$.

(ii) The second pass is made to construct the CCP tree. For example, for the case in Table 4.3, we have the first transaction in D_1, $\{b,c\}$. We first sort each transaction by the ordered item list. In this case, it is still $\{b,c\}$. Then a node with one item $\{b,1,0,0,0,p.child\}$ is created as the root node, and a node with item $\{c,1,0,0,0,\{\}\}$ is stored as the child node of item b. For the second transaction in D_1, we scan the current tree to check if there is an existing path. Because the root node already contains item b, we only increase its field *countD*1, i.e., the node is updated to $\{b,2,0,0,0,p.child\}$. For transaction $\{a,b\}$, it can be sorted as $\{b,a\}$. Similarly, item b in the root node is updated to $\{b,3,0,0,0,p.child\}$, and this time, item a is added to the children of item b, which means that item b has a new child item $\{a,1,0,0,0,\{\}\}$. For the last transaction $\{d\}$, since there is no d in the root node, the root node has a new item $\{d,1,0,0,0,\{\}\}$. This scanning process continues until every transaction in datasets D_1, D_2, S_1 and S_2 has been mapped onto one of the paths in the CCP tree. The final CCP tree is shown in Figure 4.2, and the pseudo-code is shown in Algorithm 1 and Algorithm 2.

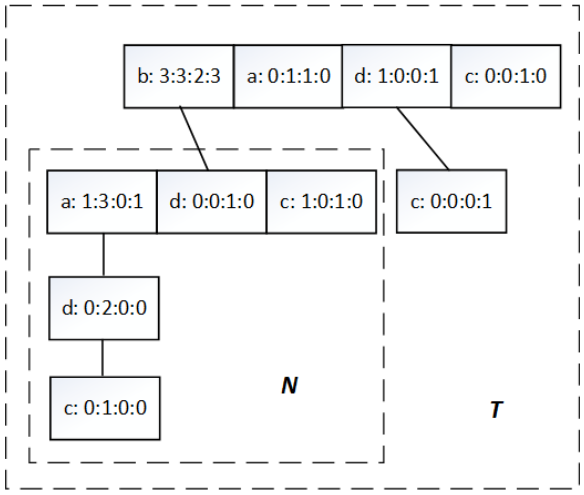


Figure 4.2: CCP tree of the example data set.

Algorithm 1 *create_tree*(D_1, D_2, S_1, S_2, min_{supp})

Input: All the datasets and minimum support threshold of D_2
Output: The final CCP tree

$T = \{\}$;
for trans in D_2 **do**
    count the frequency of each singleton item;
delete item in D_2 with frequency lower than min_{sup};
sort remaining items;
for *trans* in D_1, D_2, S_1 and S_2 **do**
    sort trans;
    insert_tree(*trans*, T);
return T

Algorithm 2 *insert_tree*(P, T)

Input: One sorted transaction P; T is a CCP tree
Output: The updated pattern tree T

search T for $T.item[i] = P[0]$;
if $T.item[i]$ is not found **then**
    insert p at the appropriate place in T obeying the order, denoted as $T.item[i]$;
    $T.item[i].countD1 = 0$;
    $T.item[i].countD2 = 0$;
    $T.item[i].countS1 = 0$;
    $T.item[i].countS2 = 0$;
increase $T.item[i].countD1, countD2, countS1, countS2$ by 1 according to the class label of the instance;
if $P[1:]$ is not empty **then**
    if $T.item[i]$'s subtree is empty **then**
        create a new node N as $T.item[i]$'s subtree;
let $N \leftarrow T.item[i]$'s subtree;
call *insert_tree*($P[1:]$, N);

4.4.3 Mining CCPs

After building the CCP tree, we mine CCPs from the tree. To obtain the count of each itemset, we propose to use the merging subtree algorithm, which is inspired by [37]. The main idea is that the subtree of the current item should be merged with the tree where the current item is located before we decide whether an itemset is a CCP. For example, before we check whether $\{b\}$ is a CCP or not, we need to merge b's subtree N to the original tree T. We check all the itemsets using a Depth First Search (DFS) strategy. This means that after checking $\{b\}$, we need to check itemset $\{b, a\}$ (list order is $b \rightarrow a \rightarrow d \rightarrow c$). Because item $\{a\}$'s subtree is not empty, its subtree should be merged with tree N. Thus, for all the itemsets including item b, the order of checking should be $\{b\}$, $\{b, a\}$, $\{b, a, d\}$, $\{b, a, d, c\}$, $\{b, d\}$, $\{b, d, c\}$ and $\{b, c\}$ in our case, as shown in Figure 4.3. The pseudo-code of *merge_tree* is shown in Algorithm 4.

For every item in the header table, if we find it in the current root node of the tree and it has a subtree, merge its subtree into the current tree. Then we add the item to an accumulation itemset $\beta = \alpha \cup T.item[i]$, which records the itemset we are checking. If the item satisfies our CCP conditions, the itemset β is a CCP. Then we keep searching its subtree if the item satisfies the first condition C1 and its subtree is not empty. Here, we

use condition $C1$ to decide whether to check the superset of itemsets, because condition $C1$ is anti-monotone. If β does not satisfy the minimum support threshold, neither will its superset. The algorithm stops when all items in the header table have been checked.

The final merged tree is shown in Figure 4.3, from which we can see that the counts of all the itemsets are consistent with the statistics of the counts of all possible itemsets shown in Table 4.4. Here we set $min_{supp} = 1$, $min_{growth} = 2$, $min_{ratio} = 2$. This consistency demonstrates that by using our method, we can obtain the correct counts of each itemset. After applying our conditions, we derive 10 CCPs from all the itemsets. They are $\{a\}$, $\{a,b,d\}$, $\{a,b,d,c\}$, $\{a,b,c\}$, $\{a,d\}$, $\{a,d,c\}$, $\{a,c\}$, $\{b,d\}$, $\{b,d,c\}$ and $\{d\}$. We can see that pattern $\{c,d\}$ is a jumping emerging pattern in the transaction dataset, while it is not a conditional contrast pattern, since it does not satisfy condition $C3$.

Algorithm 3 *mine_tree* (T, α, min_{supp}, min_{growth}, min_{ratio}, *headerTable*)

Input: T is a pattern tree, α is an accumulating itemset, min_{supp}, min_{growth} and min_{ratio} are the minimum support, minimum growth rate, and minimum ratio, which are designed by the user, *headerTable* stores the sorted singleton item list

Output: A set of CCPs

```
23 for item in headerTable do
24     if item is found in T's items T.item[i] then
25         if T.item[i]'s subtree M is not empty then
26             merge_tree(M, T);
27         β = α ∪ T.item[i];
28         if T.item[i] satisfies CCP conditions then
29             generate an CCP β of D2;
30         if T.item[i]'s subtree N is not empty and T.item[i].countD2 ≥ min_supp then
31             mine_tree(N, β, min_supp, min_growth, min_ratio, headerTable);
```

4.4.4 Analytical Results

Space complexity During runtime, we not only need to store the counts of each node, but also the pointers between nodes. This may result in increased storage. To solve this problem, we can adopt the method proposed in [37] and [49], which partitions the data set into a set of projected data sets and mine patterns for each projected dataset.

Algorithm 4 *merge_tree*($T1$, $T2$)

Input: $T1$ and $T2$ are two pattern trees
Output: An updated tree $T2$ after being merged with $T1$

```
for each item of T1, T1.item[i] do
    search T2 for T2.item[j] = T1.item[i];
    if T2.item[j] is found then
        update T2.item[j] by adding the count of T1.item[i] correspondingly;
    else
        copy and insert T1.item[i] with its counts and child node at the right place in T2,
          and denoted as T2.item[j];
        continue;
    if T1.item[i]'s subtree M is not empty then
        if T2.item[j]'s subtree is empty then
            create a new node N as T2.item[j]'s subtree;
        N ← T2.item[j]'s subtree;  call merge_tree(M,N);
```

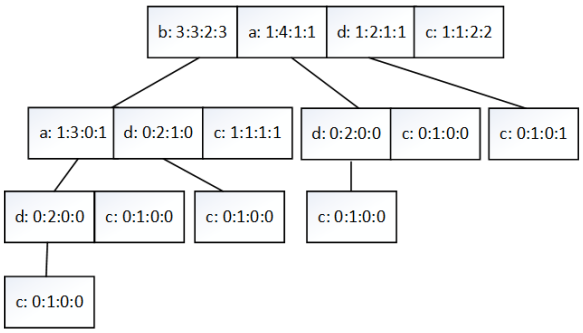


Figure 4.3: Final merged tree of the illustrative dataset.

Computational complexity A detailed analysis of the time complexity for the CCP tree based algorithm is shown below.

Construction of CCP tree: First, we need to scan all the datasets once to obtain the header table. In the second scan, for each transaction, we need to find the right position for each item in the transaction to insert in the current CCP tree. Assuming that N_t is the total number of transactions, w is the average transaction width, and the algorithm complexity of sorting items is $O(w \cdot log(w))$, then the computational complexity of constructing the CCP tree is $O(w \cdot log(w)N_t)$.

Mining CCPs: The best case happens when all the transactions have the same set of items, which means the CCP tree contains only a single branch of nodes. If all the

candidates satisfy the CCP conditions, the algorithm complexity will be $O(2^w)$. In the worst case, all the possible itemsets appear in the transactions and their counts are all greater than the minimum support threshold. In this case, we build the biggest CCP tree. Assuming that d is the number of singleton items, the computational complexity of mining CCPs is $O(d \cdot 2^{d-1})$ in the worst case. In practice, since not all the patterns will exist in transactions and not all the patterns satisfy condition $C1$, the computational complexity will be lower than the worst case $O(d \cdot 2^{d-1})$.

Completeness proof The construction of the CCP tree is exhaustive, and all the counts in different datasets are recorded in the tree node. This preserves complete information for CCP mining. By merging the tree, we can obtain the exact counts for each itemset. In our mining algorithm, all possible itemsets are checked except some itemsets that may be pruned by using condition $C1$, since this condition is anti-monotone. We also test the accuracy of our method using a brute force method, which checks all combinations of items. The experiments show that the results of these two methods are the same.

4.5 Performance Evaluation

In this section, we evaluate the performance of our CCP tree based method. We describe the synthetic and the real-life dataset in Section 5.1, and baseline methods in Section 5.2. Then we compare our method with other methods in terms of efficiency and quality for synthetic data in Section 5.3. In the last section, we show the results derived from the real dataset.

All experiments are performed on a laptop with 2.6 GHz Intel® Core™ $i7-5600U$ CPU, 16 GB memory, and Windows 7 64bit Enterprise operating system. The programs are coded in Python (version 3.6.0).

4.5.1 Synthetic Dataset

For synthetic data, we simulate a store with N_p zones in terms of the product categories. Suppose that we have five categories of customers, who have different probabilities in

buying different products. Note that costumers may visit a set of zones and purchase several categories. The probability of a customer belonging to category j buying products in zone i is $P_b(i,j)$, which is chosen arbitrarily, and the value chosen has no effect on the evaluation of our algorithm.

The chance that a customer of category j entered zone i, $P_v(i,j)$ is decided by a sale ratio, $Ratio$, namely:

$$P_v(i,j) = P_b(i,j) \cdot Ratio,$$

wherein $Ratio(j) \geq 1$. Then we can get the probabilities of customers visiting different areas according to the formula above. We also suppose that the probabilities of different categories of customers $P(Cate(j))$ are equal. For example, if there are 5 categories of customers, $P(Cate(j)) = 0.2$ for $j = 1, ..., 5$.

By changing the parameter $Ratio$, we can generate dataset D and S of two different time periods. For example, setting $Ratio = 1.1$, we can generate dataset D_1 and S_1. Similarly, setting $Ratio = 1.2$, we can generate another two datasets D_2 and S_2.

4.5.2 Real Dataset – Melbourne Retail Shop

In this dataset, 78 days of transaction data and spatial data in one store located in Melbourne were collected from November 2015 to February 2016. In the store, a set of sensors are installed, which monitor the position of individual customers over time by detecting the WiFi MAC address of a customer's mobile phone. Note that some customers may not have a mobile phone, or WiFi is not activated on their phones. We assume that the proportion of this kind of customer is constant among different days. Since in our problem, we only pay attention to the change of the number of customers, these customers in constant proportion will have no effect on our results. Each trajectory represents one customer visiting the store, and each transaction represents one customer buying product(s) in the store. Based on the category of items on the shelf, the store is divided into eight polygon zones, with the cashier zone excluded. The trajectory of one customer is composed of a sequence of $2D$ points with a unique WiFi MAC address. Based on which zone each $2D$ point occurs in, each trajectory is translated to a set of zones. Then, a threshold

of total dwell time (30 seconds in the current case) is applied to discard customers with short dwell time. Also, trajectories that occurred out of business hours are considered as those generated by staff, which are removed. Transaction data with a negative amount indicates a refund, which should not be taken into account in our experiment. In all, there are 215,748 valid trajectories and 109,752 valid transactions. The name and detailed layout of the store cannot be revealed for confidentiality reasons. The names of the 8 zones in the store are also anonymized.

4.5.3 Evaluation

Baseline Mining Methods

We compare our proposed method with two other methods: brute force method, and Apriori-based method. The brute force method considers all the combinations of items. It scans the datasets every time it calculates the counts of all the itemsets. Since the first proposed condition satisfies the anti-monotone property, we can apply the Apriori principle to help us reduce the number of possible candidate itemsets.

Results for Synthetic Data

To evaluate the efficiency of our method, we compared the running time of the three approaches. Figure 4.4 shows the running time of the three methods as the number of transactions in each dataset varied from 100 to 20,000. Other parameters setting are: $N_p = 15$, $min_{supp} = 2$, $min_{growth} = 2$, $min_{ratio} = 2$.

We can see that the running time of the brute force method increased dramatically as the number of transactions increases. Here, we only provide the results of the brute force with less than 3000 transactions. When the number of transactions is below 3000 transactions, the Apriori-based method has comparable run time to the CCP-tree based method. However, as the number of transactions further increases, the Apriori-based method becomes slower, while the running time of our proposed method stays almost stable.

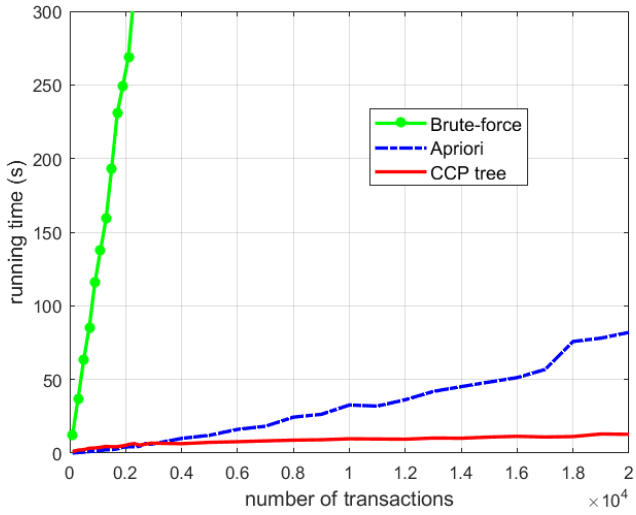


Figure 4.4: CCP mining time v.s. number of transactions of synthetic data.

We also compared the number of patterns we can find using the CCP-tree based method and CP-tree based method [37]. The numbers of CPs and CCPs are also shown in Figure 4.4 as the number of transactions in each dataset increased from 100 to 20,000. We can see that the number of CCPs is less than the number of CPs, since we filter CCPs from CPs by utilizing information from other datasets. The growth rate of the number of CCPs is also slower than CPs. This is because as the number of transactions increased, there would be more patterns that satisfy the conditions of CPs, while for CCPs, the number of transactions has little influence on the number of CCPs we can find.

Table 4.5 shows the running time of the three methods as the number of items varies. Here the number of transactions in each dataset is 3000. The brute force method is still the slowest. With only 15 items, its running time can be 61 times longer than the other two methods. When the number of items is no more than 20, the Apriori-based method is a little faster than the CCP-tree based method. While when the number of items reached 25, the CCP-tree based method is about 1.5 times faster than the Apriori-based method. In practice, the running time of our proposed method may be reduced by using compiled languages and algorithm parallelization.

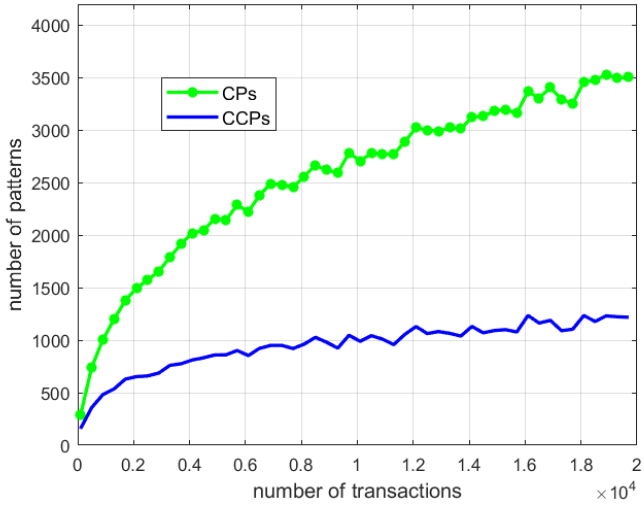


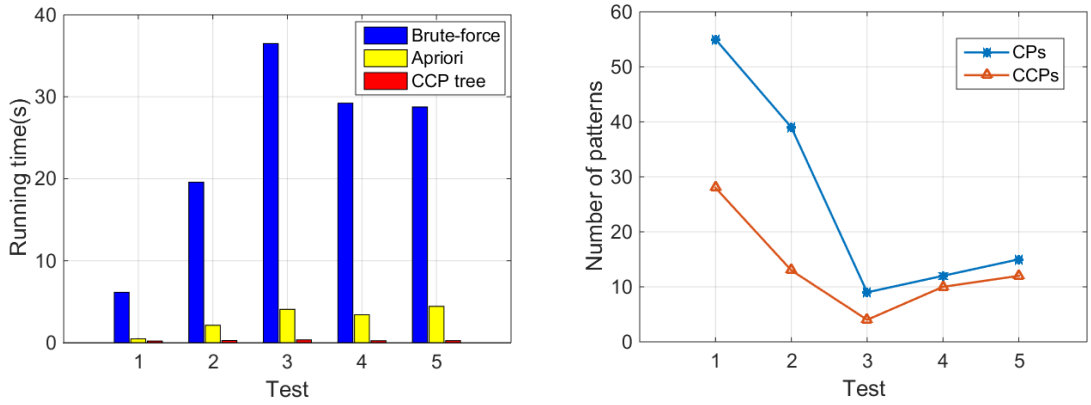
Figure 4.5: Number of CCP and CP v.s. number of transactions of synthetic data.

Table 4.5: Running time depending on the number of items on synthetic data.

	Running time(s)		
	15 items	20 items	25 items
Brute force	371.98	-	-
Apriori	5.95	94.67	1384.49
CCP tree	6.08	95.24	986.83

Table 4.6: Description of the five tests on real data: Transaction data and spatial data of customers.

Test	**Data set**		**Number of Transactions**			
	Negative	Positive	D1	D2	S1	S2
1	26/12/15-28/12/15	22/12/15-24/12/15	2,397	7,366	4,706	8,340
2	25/01/16-31/01/16	01/02/16-07/02/16	9,910	12,195	19,754	29,372
3	Sundays	Mondays	18,129	19,156	34,566	36,965
4	Sundays	Saturdays	18,129	18,852	34,566	35,300
5	Tuesdays	Mondays	17,147	19,156	33,281	36,965



(a) Running time comparison

(b) Number of CCPs and CPs

Figure 4.6: Results on real data - transaction data and spatial data of customers.

Results for Melbourne Retail Shop

In a retail environment, emerging patterns of purchased items are often analyzed to provide sales strategies. Intuitively, sale increases relate to more customer visits. It is the patterns whose sales change significantly but with little variation in customer visits that are of special interest. These patterns fit well with our proposed conditional contrast patterns (CCPs), and can be obtained using our CCP tree based mining algorithm.

We select 5 representative time periods from the data to test the performance of our algorithm. The first time periods in the comparison are $26/12/2015 - 28/12/2015$ and $22/12/2015 - 24/12/2015$, which are 3 days after the Christmas Day and 3 days before it. The second comparison is between $01/02/2016 - 07/02/2016$ and $25/01/2016 - 31/01/2016$, which are one week before the start of school and one week after it. The last three comparisons are Sundays and Mondays, Sundays and Saturdays, and Mondays and Tuesdays. The description of the time period is shown in Table 4.6.

Then we obtain five sets of datasets $\{D = D_1 \cup D_2, S = S_1 \cup S_2\}$. The detailed numbers of transactions in D_1, D_2, S_1 and S_2 are shown in Table 4.6. In these experiments, the parameter threshold settings are $min_{supp} = 5$, $min_{growth} = 2$, $min_{ratio} = 2$.

The running time of the three methods is shown in Figure 4.6a. As shown in Figure 4.6a, in all the five tests, the running time of the brute force is the highest, and it strongly depends on the number of transactions. The Apriori-based method is much bet-

ter than brute force, but it also depends on the number of transactions. Our method is the fastest, with all the running times less than 0.5 second, and the number of transactions has little influence on its performance.

The number of CCPs and CPs are shown in Figure 4.6b. We can see that the number of CCPs is less than CPs, which means that we extract fewer contrast patterns by utilizing two kinds of datasets and these patterns are more informative and actionable for decision makers. We also notice that in *test 1* and *test 2*, although the number of transactions is less than the number of transactions in *test 3*, *test 4* and *test 5*, the number of CCPs in the first two cases is larger than the number in the last three cases. This means that the time periods under contrast are more different than other time periods.

4.6 Conclusion and Future Work

In this chapter, we have introduced a new kind of contrast pattern, conditional contrast patterns. In contrast to normal contrast patterns, CCPs are able to use information from other data sources. We also have proposed a conditional contrast pattern tree based algorithm for mining CCPs. By comparing our method with two other baseline methods (brute force method and Apriori-based method) on synthetic data, we have evaluated the efficiency of our method. The results show that our CCP tree based method is the most efficient one among these methods. A real-world case study relating to customer behaviors has been applied to further explain our proposed method.

Overall, we believe that we have proposed a useful and interesting contrast pattern for multi-source mining, and our proposed CCP method is practical in analyzing data from retail environments. Also, we believe CCPs can be applied to other areas, such as classification. In the future, we would like to develop more time-efficient methods, such as approximate methods, for mining CCPs.

In the next chapter, we will focus on knowledge discovery and contrast mining from the trajectories generated from heterogeneous mobile networks.

Chapter 5

Multi-scale Trajectory Clustering to Identify Corridors in Mobile Networks

In the previous two chapters, the trajectory data we studied are generated from pedestrian in indoor environments, such as a station or a shop. In the following three chapters, we mainly focus on the study of trajectory data generated from large-scale heterogeneous mobile networks.

Deployment and management of large-scale mobile edge computing infrastructure in 5G networks has created a major challenge for mobile operators. The ability to extract common users' trajectories (i.e., corridors) in mobile networks helps mobile operators to better manage and orchestrate the allocation of network resources. However, compared with other types of trajectories, mobile trajectories are coarse, and their granularity varies due to the inconsistent density of cell towers. To identify the underlying geographical corridors of users in mobile networks, we propose a hierarchical multi-scale trajectory clustering algorithm for corridor identification by analyzing the non-homogeneity of the spatial distribution of cell towers and users' movements. To measure trajectory similarity on different scales we propose a distance measure based on Hausdorff distance that considers the cell density distribution. Common corridors are represented as weighted graphs as the final results, which can not only highlight users' frequent paths but also users' movement pattern between cell towers. The proposed methods are validated using real-life datasets provided by China Mobile. Results show that by considering the heterogeneity of mobile networks, our method can achieve the best performance with more than 10% improvement in clustering quality compared with state-of-the-art methods.

The publication arising from the work in this chapter is paper P3.

5.1 Introduction

The development of smart devices and data networks (from 2G/3G/4G to the new generation 5G) has provided mobile users with faster and easier access to the mobile In-

ternet. The movements of mobile users can be detected and collected by their mobile phones, which results in large volumes of human mobility data, such as GPS tracks and CDR traces available to mobile operators. Recently, studies showed that this data can be helpful in human dynamics studies [133]. When mobile users access the Internet, their trajectories can be passively detected by identifying the sequence of locations of the cell towers that provide Internet data to them. This kind of data acquisition method has several advantages, such as low energy consumption, wide range coverage and fine time granularity (applications may send or receive packets periodically when running in the background and people may access the Internet while moving).

In our work, we focus on the problem of identifying the underlying geographical corridors of trajectories generated in mobile networks. A corridor can be treated as a pathway that is frequently traversed by a considerable number of mobile users. In Figure 5.1, there are 4 distinct continuous trajectories sharing a corridor (marked as a gray area). Identifying corridors provides us an opportunity to understand the movement patterns of users in mobile networks, which will then help service providers in the deployment of networks and base stations, and the management of network resources. For example, the costs of deploying a large number of mobile edge computing [70] servers at the edge of mobile networks could be minimized by focusing on deployment at corridors. Furthermore, the ability to identify the temporal dynamic patterns of corridors could help the orchestration of 5G network resources through network function virtualization [81] and network slicing [39].

However, there are challenges in identifying corridors generated in mobile networks. First, the trajectories represent discrete approximations of original continuous paths derived by users using different travel modes with varying speeds. Thus, in mobile networks, sometimes trajectories of the same path may result in totally distinct cell sequences. Second, the cell towers are distributed heterogeneously, i.e., the density of cell towers varies at different places. For example, in rural areas or suburbs, the cell towers are distributed coarsely, whereas in urban places, the density of cells is higher. Therefore, using the absolute distance measures between trajectories may not be suitable for finding the real corridors. Figure 5.2 illustrates the cell sequences generated from the continuous

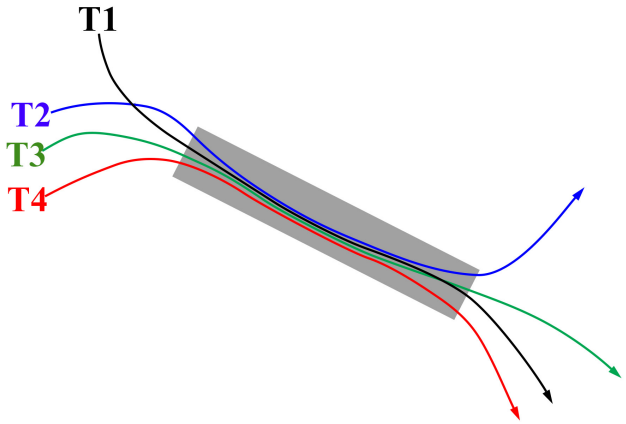


Figure 5.1: Example of four trajectories sharing one corridor - continuous trajectories.

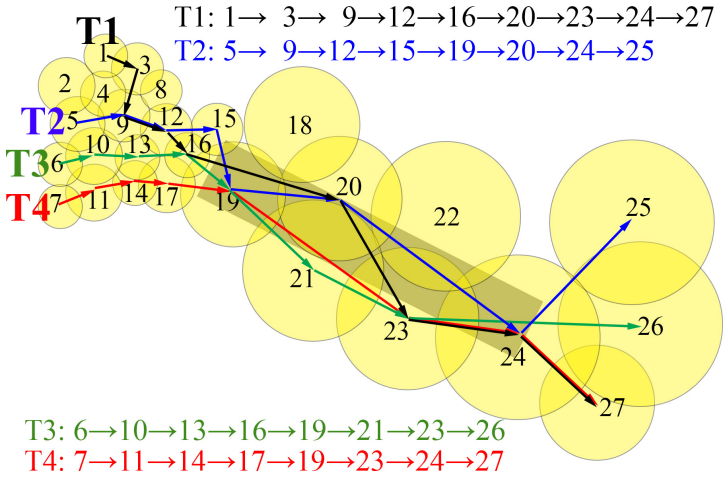


Figure 5.2: Real-life mobile trajectories of example in Figure 5.1. (The yellow circle shows the coverage of a cell tower.)

trajectories in Figure 5.1. Although $T1$, $T2$, $T3$ and $T4$ all share one corridor, their cell sequences are different. We also notice that trajectories which are close to each other can have large distances between them in areas with sparse cells, while distinct trajectories may be closer to each other in dense areas. To measure the similarity between trajectories, the density of cells needs to be considered in the distance measure instead of directly using the absolute distance.

Therefore, we propose a multi-scale trajectory clustering algorithm for identifying users' corridors in mobile networks. To the best of our knowledge, it is the first study that focuses on the corridor identification of human trajectories generated from heterogeneous mobile data networks. Our main contributions are: (1) We propose a new distance measure based on Hausdorff distance to calculate the similarity between sub-trajectories by considering the distribution of cells. (2) We propose a two-level clustering method based on the developed distance measure. Corridors are identified and represented as weighted graphs based on the clustering result. (3) We conduct experiments over the real-life data of mobile users in a southern province in China to evaluate our approach, which show improvements in both interpretability and clustering quality.

5.2 Related Work

In the literature, generally there are two strategies for solving the trajectory clustering problem. The first one is clustering the whole trajectory. The second one is clustering the segments of trajectories to find the common sub-trajectory, which is more conservative compared to complete trajectories and was first proposed in [60].

The authors in [60] proposed a partition-and-group framework for clustering trajectories, TRACLUS, which enables the discovery of common sub-trajectories, based on a trajectory partitioning algorithm that uses the Minimum Description Length (MDL) principle. They also defined the distance measure between two single line segments as the weighted sum of three measurements, i.e., perpendicular distance, parallel distance and angle distance. A clustering method based on DBSCAN for line segments is also proposed. Finally, the notion of the representative trajectory of a cluster is also

provided based on sweeping. However, the proposed algorithm for clustering is sensitive to its input parameters, and the representative trajectory of a cluster primarily supports straight movement patterns but cannot identify complex (e.g., circular, polygonal) motions, which are common in real-world applications. To solve this problem, a three-phrase approach was proposed in [136] to discover trajectory corridors (i.e., frequently followed paths) using the Discrete Fréchet distance. In the first step, trajectories are segmented into sub-trajectories (short polygonal curves not line segments) using meshing-grids, and then the sub-trajectories in each grid cell are clustered separately using a hierarchical clustering method to avoid accumulation. In the third phrase, local clusters in each cell are concatenated together to construct the final corridors. In [45], a trajectory clustering method based on motifs (frequently occurring substrings) was proposed. Trajectories are simplified first and partitioned according to some predefined motion patterns, such as wide left turn and short left turn. Then the algorithm computes motifs and maps subtrajectories corresponding to motifs into some feature space. Finally, DBSCAN clustering is applied and representative trajectories are obtained using the method mentioned in [60]. The authors claimed that their method is better than [60], because they can detect more patterns and not just relatively straight curves. Recently, the authors in [137] proposed a method for detecting a set of corridors from GPS trajectories using the MDL principle. The DTW distance measure is adopted, and for inferring the most likely path of a sparse uncertain movement a graph-based approach was proposed. In their consecutive work [138], they proposed to use a grid-based method to transfer trajectories into grid cell sequences and then Latent Dirichlet Allocation (LDA) was applied to discover frequent sets. Then a hierarchical clustering algorithm was applied for grouping trajectories. Finally, the MDL principle was adopted for corridor selection. A summary of this prior work is given in Table 5.1.

Although there has been extensive literature on mining the trajectory data or finding common human mobility patterns, few of them focus on mobile network data. In [89], the authors proposed a method based on the Apriori algorithm to find the frequent hotspot sequences, which is different from our work. Here we choose to use the location of cellular towers directly to find the common pathways. To the best of our knowledge, this is

Table 5.1: A summary of papers related to corridor/pathway identification.

Reference	Lee et al. [60]	Zhu et al. [136]	Zygouras et al. [137, 138]
Data Type	GPS	GPS	GPS
Segmentation	MDL principle	grid meshing	grid meshing
Similarity	weighted sum of three measurements	Discrete Fréchet distance	DTW
Clustering	DBSCAN	hierarchical clustering	hierarchical clustering
Identification	sweeping	inter-grid concatenation	MDL principle
Spatial	✓	✓	✓
Temporal	✗	✗	✗
homogeneous	✓	✓	✓
heterogeneous	✗	✗	✗

the first study that focuses on the pathway identification of human trajectories generated from heterogeneous mobile data networks.

5.3 Problem statement

Suppose that in a mobile network there are N cellular towers, which is represented as $C = \{c_1, c_2, ..., c_N\}$, and each cell tower has a unique identifier and is associated with its coordinates.

Definition 5.1. ***Trajectory*** *A trajectory can be represented as a sequence of states in a given period of time: $Traj = \{s_1, s_2, ..., s_n\}$, where n is the number of cells visited by the user. A state is defined as $s = (c, t, stay)$, where c is the cell ID, t is the time when the user entered the current cell, $stay$ is the stay time in the current cell.*

Definition 5.2. ***Tracklet*** *A tracklet, T, is a directed fragment of a trajectory $Traj$, i.e., $T = \{s_a, ..., s_b\}$, where $1 \leq a < b \leq n$.*

Definition 5.3. ***Corridor*** *A corridor, cor, is a cluster of similar tracklets, which represents the movement of a certain number of mobile users, denoted as $cor = \{T_1, T_2, \ldots, T_K\}$. It satisfies the following conditions:*

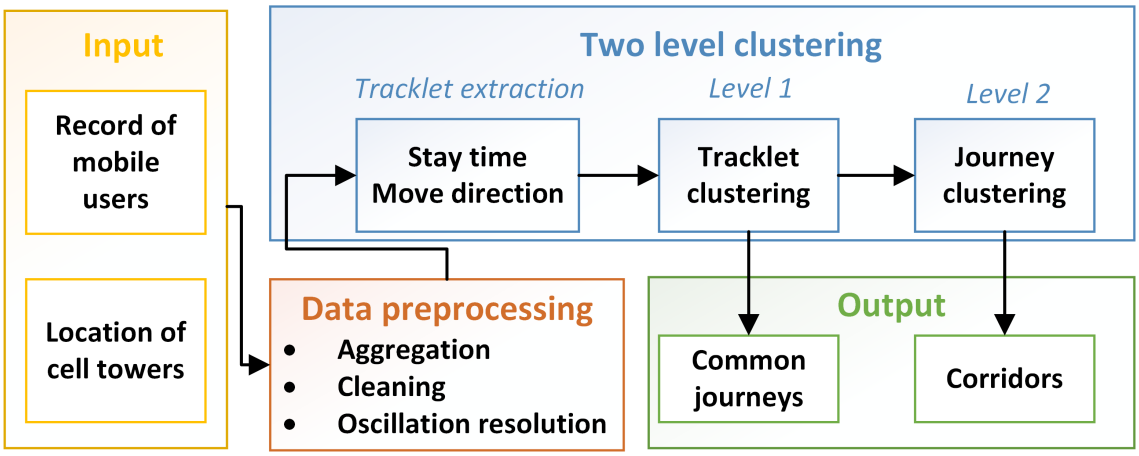


Figure 5.3: Flowchart of corridor identification.

1. $K \geq \delta$*, which means the number of tracklets in cor should be no less than a threshold* δ*;*

2. The tracklets in one corridor are similar, where the similarity measurement of tracklets is described in Section 5.4.2.

Problem: Given the historical trajectories of a set of M mobile users $Traj = \{Traj_1, Traj_2, \ldots, Traj_M\}$, our task is to identify a set of corridors $COR = \{cor_1, cor_2, \ldots\}$ of users in the network.

5.4 Multi-scale Corridor Identification

There are four steps in the corridor identification algorithm, as shown in Figure 5.3. First, data are preprocessed by data aggregation, data cleaning and oscillation resolution. In the second step, trajectories are partitioned into tracklets according to our defined constraints on stay time and movement direction. Then tracklets are clustered together to extract common journeys shared by users, and clustered again in the second level to identify corridors.

5.4.1 Data Preprocessing

There are three steps in data preprocessing.

Aggregation The original data include logs of mobile phone users' activities. Each log is generated every 5 minutes, which contains the timestamp and the ID of the cell tower that the phone is connected to. Mobile phone users' logs are aggregated if the consecutive logs have the same cell ID, then each state of a user includes the cell ID, the time when the user entered the cell, and the stay time in the cell, as defined in Section 5.3.

Cleaning Users who have only a few records are discarded since they have insufficient data for analysis.

Oscillation resolution Oscillation is a common phenomenon and major problem in the location data collected from mobile cellular networks. This occurs when the mobile phone intermittently switches between cells quickly or when the mobile users are located near the boundary of different cells. There has been some research on how to resolve this problem to enable further mobility modeling (e.g., [88, 118]). Here, the DECRE (Detect, Expand, Check, REmove) algorithm [118] is adopted, since this method does not rely on other data sources and uses the cell locations (rather than cell clusters) to represent locations of mobile users.

5.4.2 Two-level Clustering

Tracklet Extraction from Trajectories

Tracklet extraction is similar to trajectory segmentation, wherein the main task is to identify when a trip starts or ends. We propose to extract the tracklets based on both temporal and spatial heuristics. We suppose that in a trip, the movement direction should not change a lot, and the stay time at the start and end cells should be reasonably long. According to these two basic assumptions, a tracklet $T = \{s_a, ..., s_b\}$ $(1 \leq a < b \leq n)$ extracted from a trajectory $Traj = \{s_1, s_2, ..., s_n\}$, needs to satisfy the following two constraints:

1. $\Delta(dir(s_{i-1}, s_i), dir(s_i, s_{i+1})) \leq dir_{thres}, \forall i \in [a+1, b-1]$;
2. $stay_i \leq stay_{thres}, \forall i \in [a+1, b-1]$,

where $dir(\cdot,\cdot)$ is the movement direction of two consecutive states, dir_{thres} is the threshold of direction difference, which makes sure that the heading direction of the mobile user does not change dramatically, and $stay_i$ is the stay time in cell c_i, which should be no greater than a time threshold $stay_{thres}$.

The pseudo-code for tracklet extraction from trajectories is shown in Algorithm 5.

Algorithm 5 *Trajectory2tracklet*

Input: A trajectory $Traj = \{s_1, s_2, \ldots, s_n\}$; two thresholds dir_{thres}, $stay_{thres}$
Output: A set of tracklets in TL

```
Initialize TL = {} ;
Add the first line segment T_curr = {s_1, s_2} ;
for each state in trajectory s_i ∈ Traj, (2 ≤ i ≤ n − 1) do
    if Δ(dir(s_{i−1}, s_i), dir(s_i, s_{i+1})) ≤ dir_thres and stay_{s_i} ≤ stay_thres then
        append s_{i+1} to T_curr ;
    else
        add T_curr to TL ;
        T_curr = {s_i, s_{i+1}} ;
```

First Level Clustering

A two-level clustering scheme is proposed to group similar trajectories. On the first level, similar tracklets are grouped in one cluster and dissimilar tracklets into different clusters. In our problem, it is difficult to determine the number of clusters and all the tracklets are not necessarily assigned to one specific cluster. Therefore, we propose to use the modified sequential algorithm [104], which is not sensitive to noise and does not need to set the number of clusters.

In order to measure the dissimilarity between two tracklets, we propose a distance measure based on Hausdorff distance. Hausdorff distance represents the maximum mismatch level between two point sets [50]. Instead of using the distance between points, here we use the distance between a point and line segment. Moreover, to overcome the problem caused by cell heterogeneity, we propose to normalize the distance by introducing a normalization factor, which is calculated based on the cell density. We suppose that the density contributed by a cell follows a Gaussian distribution. The mean vector is the center of the cell tower, and that three times the standard deviation is equal to the cover-

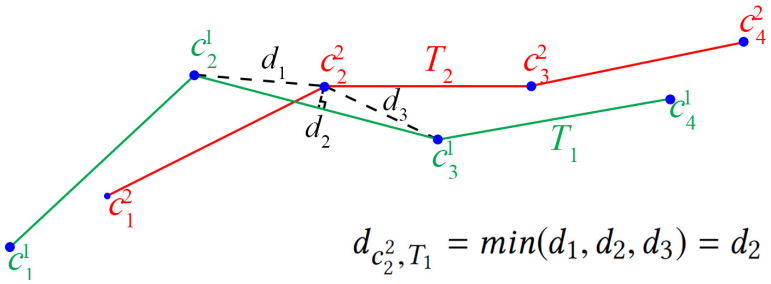


Figure 5.4: Illustration of distance measure between T_1 and c_2^2.

age radius of the cell. Therefore, the accumulated density at each cell in the network is considered as the density of the cell in a network, which is:

$$\rho\left(c_i\right) = \sum_{j=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_j} exp(-\frac{d_{c_i,c_j}^2}{2\sigma_j^2}), \tag{5.1}$$

where d_{c_i,c_j} is the spherical distance between centers of two cells c_i and c_j, and $\sigma_j = r_j/3$, r_j is the coverage radius of c_j. Then, the distance from c_i to T_j is:

$$d_{c_i,T_j}^{\rho} = \alpha \cdot d_{c_i,T_j} = \frac{N \cdot \rho\left(c_i\right)}{\sum_{k=1}^{N} \rho(c_k)} \cdot d_{c_i,T_j}, \tag{5.2}$$

where α is the normalization factor and $d_{c,T}$ is the distance between cell c and tracklet T, i.e., the shortest distance from the cell center to all line segments in tracklet T. For example, in Figure 5.4, the distance between cell c_2^2 and tracklet T_1 is the shortest distance from the cell c_2^2 to the three line segments (c_1^1, c_2^1), (c_2^1, c_3^1) and (c_3^1, c_4^1) of T_1, i.e., $d_{c_2^2,T_1} = min(d_1, d_2, d_3) = d_2$. If c_2^2 is in a dense area, then α will be greater than 1, and d_2 will turn larger; otherwise, d_2 will be normalized to a lower value. Specifically, when the network is homogeneous, $d_{c_i,T_j}^{\rho} = d_{c_i,T_j}$ since $\rho\left(c_i\right) = \rho\left(c_j\right), \forall c_i, c_j \in C$. Then we can define the Modified Hausdorff Distance.

Definition 5.4. ***Modified Hausdorff Distance*** *Given two tracklets* $T_1 = \{s_1^1, s_2^1, ..., s_m^1\}$ *and* $T_2 = \{s_1^2, s_2^2, ..., s_n^2\}$, *the Modified Hausdorff distance* $dist(T_1, T_2)$ *is defined as:*

$$dist(T_1, T_2) = max(\Delta(T_1, T_2), \Delta(T_2, T_1)), \tag{5.3}$$

$$\Delta(T_i, T_j) = max(d^{\rho}_{c^i_1,T_j}, d^{\rho}_{c^i_2,T_j}, \ldots, d^{\rho}_{c^i_m,T_j}). \tag{5.4}$$

We refer to the clusters from the first level as *journeys*, since the tracklets in one cluster are similar to each other in terms of movement path, start point and end point. Common journeys are selected if the number of tracklets in the journey is greater than a user-specified threshold.

Second Level Clustering

On the second level, the similarity of journeys is evaluated on a grid-based strategy, and these similar journeys are grouped using hierarchical clustering. First, the whole region is divided in both horizontal and vertical directions to form small rectangular grids. A high density in a certain grid indicates a large number of tracklets in this journey pass through this small area. Then, the similarity of two journeys is calculated as the similarity between their corresponding density matrices. Here, the similarity s between two m by n density matrices $A = (a_{ij})$ and $B = (b_{ij})$ is evaluated as the cosine similarity:

$$s(A, B) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} \cdot b_{ij}}{\sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2} \cdot \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} b_{ij}^2}}. \tag{5.5}$$

Since each element in the density matrix is non-negative, s ranges between 0 and 1. A higher value indicates a higher similarity between two journeys.

Given the clustering result of a set of journeys, the density matrix of all the journeys, and a frequency threshold δ, we take two steps to find the corridors in each cluster. In the first step, for each cluster, areas that have frequencies higher than the threshold δ are identified. A binary matrix can be obtained, with each element indicating whether or not the frequency of the corresponding area is above the threshold. In the second step, each tracklet in one cluster is checked by the binary matrix, and the parts that pass through high-frequency areas are extracted as a new tracklet in a corridor. Then the tracklets in each corridor are represented and visualized as a weighted graph. For example, as shown in Figure 5.5, journey 1 and 2 are identified from the first level clustering. Since these

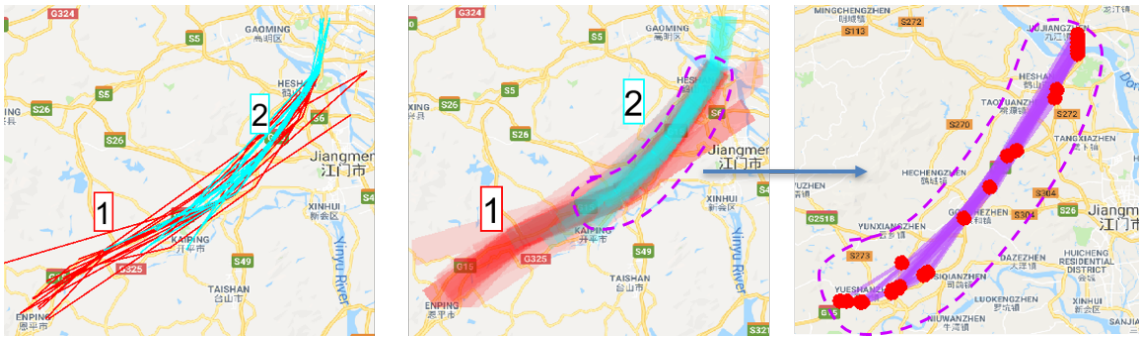


Figure 5.5: Illustrative example of corridor identification.

two journeys share some common parts they are clustered together in the second level clustering, and the common parts are identified as a weighted graph in the rightmost subfigure, where the small red circles represent cell towers and the line width reveals the traffic volume.

Algorithm 6 $identify\_corridor(clusters, M, \delta)$

Input: $clusters = \{c_1, ..., c_k\}$ generated from the second level clustering, each cluster contains several common journeys; the density matrix of all the common journeys M; one threshold δ

Output: Common pathways $P = \{p_1, ..., p_k\}$

for each cluster c_i in $clusters$ **do**
  Initialize $p_i = \{\}$;
  density matrix $M_{c_i} \leftarrow$ sum of density matrices of the common journeys in c_i ;
  binary matrix $B_{c_i} \leftarrow$ element of M_{c_i} greater than δ labels 1, otherwise labels 0 ;
  for each commom journey in cluster c_i **do**
    for each tracklet T_j in the common journey **do**
      $T_{new} \leftarrow$ extract part of T_j that located in the high frequency area B_{c_i} ;
      append s_{i+1} to T_{curr} ;

5.5 Experiments and Results

The real-life dataset was originally collected by China Mobile which contains 5,000 mobile users from a province in South China (nearly 80,000 cell stations). The cell locations (longitude and latitude) of each user is recorded every 5 minutes in a time period of three weeks (from 23:55 14/11/2015 to 23:50 05/12/2015). In the following experiments, we

Table 5.2: Distance measure comparison for the tracklets.

Tracklets	**Ours**	HD	DFM	DTW	LCSS	EDR	SSPD
T_1, T_2	**3.50**	0.75	0.81	2.98	0.18	0.18	1.10
T_3, T_4	**1.00**	1.48	24.01	7.66	0.67	0.45	1.12
T_2, T_3	**30.08**	24.97	25.46	69.07	0.89	0.74	13.38

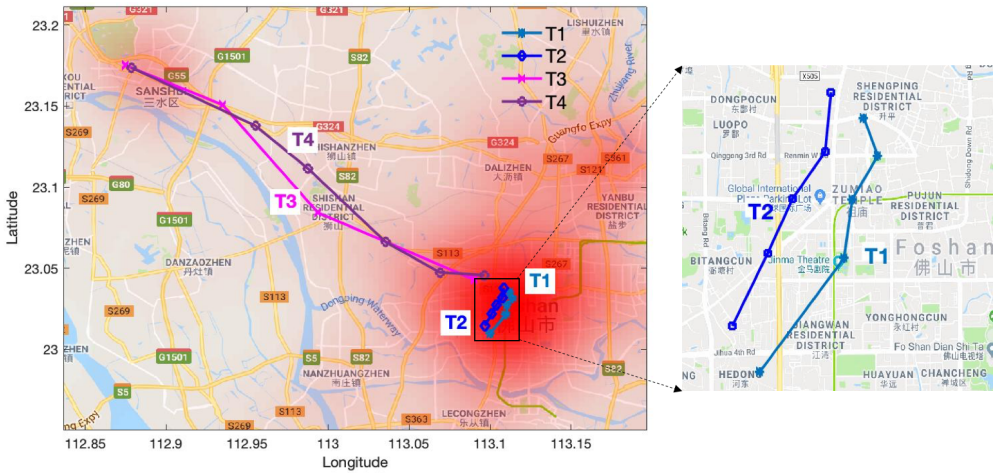


Figure 5.6: Illustrative example of tracklet distance measure.

studied three cities, Foshan, Shenzhen and Guangzhou.

5.5.1 Evaluation of the Distance Measure

Here we compared our distance measuring method, Modified Hausdorff Distance (MHD), with Hausdorff Distance (HD) [50], Discrete Fréchet Metric (DFM) [34], Dynamic Time Warping (DTW) [11], Longest Common SubSequence (LCSS) [109], Edit Distance on Real sequence (EDR) [23] and Symmetrized Segment-Path Distance (SSPD) [12]. As shown in Figure 5.6, four tracklets (T_1, T_2, T_3 and T_4) are extracted from our data of Foshan city. T_1 and T_2 are two tracklets within the CBD area, whereas T_3 and T_4 are tracklets commuting between the CBD of Sanshui District and the CBD of Foshan City. Then we calculate the distance between T_1 and T_2, and the distance between T_3 and T_4 using the method in Section 5.4.2. The results in Table 5.2 show that only our proposed MHD method can identify that T_3 and T_4 are closer than T_1 and T_2.

Then we compare the first level clustering results of these distance measures on three

cities. The quality of a clustering algorithm can be evaluated using several validity indices proposed in the literature [103, 125], such as Davies-Bouldin Index and Dunn's Index [33]. Here, we propose to adopt three evaluation metrics, silhouette coefficient (SC), cohesion (Co) and separation (Se) to verify the advantages of our proposed method. For the i^{th} tracklet, the computation method is given as:

1. Calculate its average distance to all other tracklets in its cluster, which is denoted as a_i;
2. For other clusters that do not contain the tracklet, calculate the tracklet's average distance to all the tracklets in the given cluster. Denote the minimum value as b_i;
3. The Silhouette Coefficient of the i^{th} tracklet can be obtained by $s_i = (b_i - a_i)/max(a_i, b_i)$.

Then, the Silhouette Coefficient of a cluster is the averaged Silhouette Coefficients of all tracklets belonging to the cluster. Similarly, the overall measure of the clustering results is the average of all the tracklets' Silhouette Coefficients. The value of the Silhouette Coefficient is in a range of $[-1, 1]$. A positive and high value of the Silhouette Coefficient indicates a good clustering.

In addition, we also use the averaged distance within clusters and the averaged distance between clusters to measure the cohesion Co and separation Se of the clustering result. The formulas are given as:

$$Co(c_i) = \frac{1}{|c_i|} \sum_{x \in c_i, y \in c_i} dist(x, y), \tag{5.6}$$

$$Se(c_i) = min_{j \neq i} \frac{1}{|c_i||c_j|} \sum_{x \in c_i, y \in c_j} dist(x, y), \tag{5.7}$$

where $|\cdot|$ represents the number of tracklets in the cluster. The overall Co and Se are the averaged cohesion and separation of all the clusters, respectively.

As shown in Table 5.3, our proposed method considering density can achieve the best results with a high SC, a relatively low Co and a high Se. The SSPD [12] uses a similar idea of calculating the distances between points and line segments instead of points-to-points. The difference is that SSPD uses the mean value of all the distances whereas our

Table 5.3: Clustering comparison of different methods.

Method	Foshan			Shenzhen			Guangzhou		
	SC	*Co*	*Se*	*SC*	*Co*	*Se*	*SC*	*Co*	*Se*
HD	0.33	6.72	19.40	0.47	6.48	16.27	0.49	9.73	25.38
DFM	0.31	4.94	8.50	0.31	16.25	25.10	0.45	24.90	48.56
DTW	0.26	21.32	31.40	0.42	24.31	54.54	0.37	15.63	33.68
LCSS	0.14	0.67	0.81	0.29	0.49	0.74	0.35	0.45	0.71
EDR	0.06	0.62	0.68	0.23	0.45	0.59	-	-	-
SSPD	0.32	3.29	5.80	0.43	3.30	7.01	0.46	2.60	7.98
Ours	**0.47**	9.40	26.98	**0.58**	4.76	26.93	**0.69**	2.85	15.76

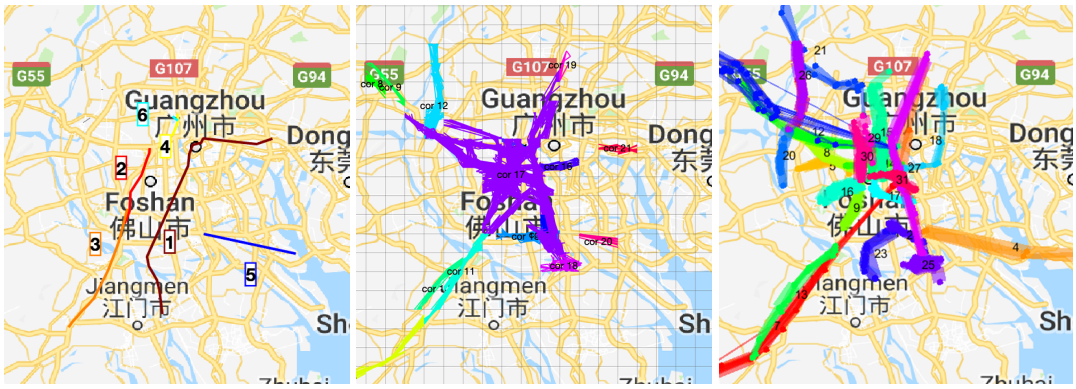
(a) Heat map generated without considering the cell distribution.

(b) Heat map generated by our proposed method.

Figure 5.7: Heat maps generated on users from Foshan City.

proposed method uses the maximum value. This also explains why SSPD can achieve better results than other methods, since they use the averaged distance which reduces the effect of noise to some extent.

Figure 5.7 shows the heat maps of the clustering results by using our proposed method and the method without considering the cell distribution. In Figure 5.7a, since the heterogeneous distribution of cells is not considered, in the city center area, pathways are not easy to be detected. Moreover, some frequent pathways traversing rural areas can not be identified. However, the result of our proposed method as shown in Figure 5.7b can clearly show users' movement flows in dense areas, as well as in sparse areas.



(a) TRACLUS [60] (b) FDMG [136] (c) our proposed

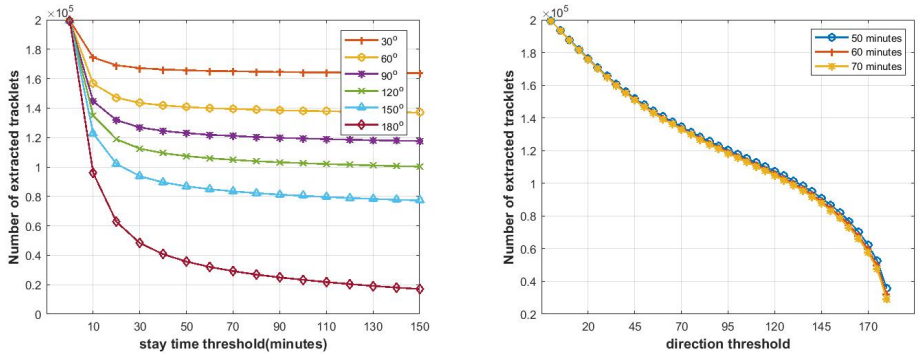
Figure 5.8: Corridors identified using different methods.

5.5.2 Corridor Evaluation

Figure 5.8 compares the corridors identified by our method and two other methods [60,136]. In the urban areas, our proposed method can clearly identify multiple corridors, which are either missing in Figure 5.8a or indistinguishable in Figure 5.8b. Meanwhile, by considering the heterogeneous distribution of cell towers (refer to Section 5.4.2), our proposed method can identify more corridors that follow the road directions in the suburbs.

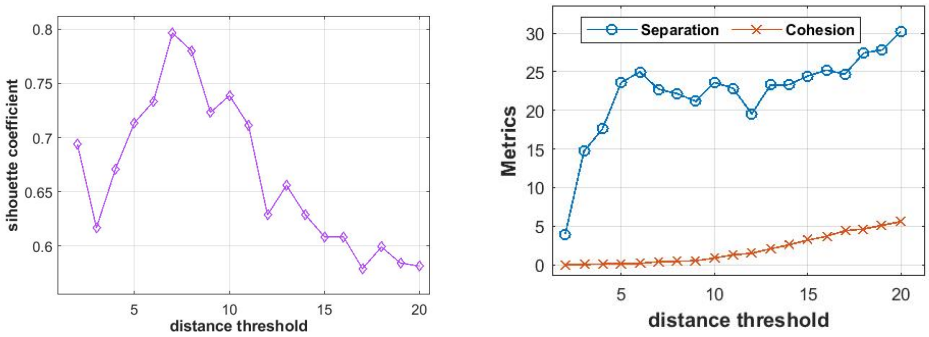
The stay time threshold $stay_{thres}$ and dir_{thres} in Section 5.4.2 are selected when the change rate of the number of extracted tracklets is the lowest. Figure 5.9 studies the variation of the number of extracted tracklets as a function of threshold parameters $stay_{thres}$ and dir_{thres}. In Figure 5.9a, the number of extracted tracklets decreases as $stay_{thres}$ becomes larger, and the most rapid decline occurs at low values of $stay_{thres}$. When $stay_{thres}$ exceeds $60min$, the number of extracted tracklets remains essentially unchanged with only a slight decrease for each dir_{thres}. Thus, here $stay_{thres} = 60min$ is applied to the experiments. In Figure 5.9b, the number of extracted tracklets consistently decreases as dir_{thres} turns larger, with the most rapid decrease when dir_{thres} approaches 180°. We select dir_{thres} to be 90°, which corresponds to the position with the minimum rate of change. A variation of $stay_{thres}$ from $50min$ to $70min$ is observed to have negligible effect on the shape of the curve.

The distance threshold dis_{thres} in the first-level clustering is chosen when SC reaches



(a) $stay_{thres}$ (b) dir_{thres}

Figure 5.9: Parameter analysis on $stay_{thres}$ and dir_{thres}.

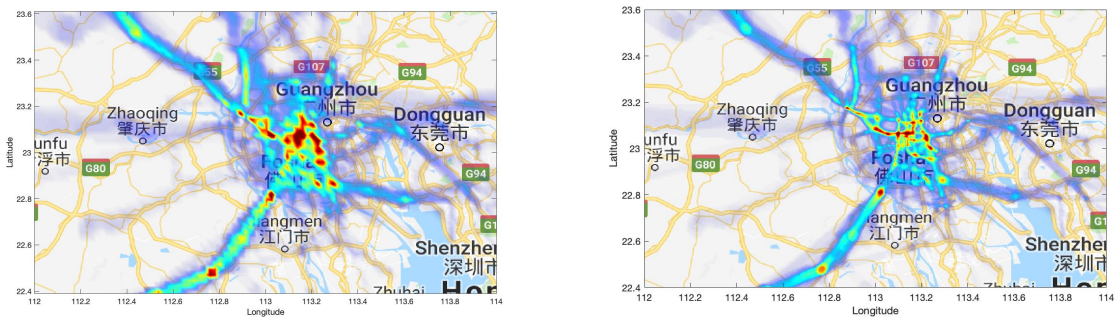


(a) Silhouette coefficient (b) Cohesion and separation

Figure 5.10: Parameter analysis on dis_{thres}.

the maximum whilst the number of identified corridors are relatively high. We calculate the *SC*, *Co* and *Se* with several different dis_{thres} values, ranging from 2 to 20, as shown in Figure 5.10. The silhouette coefficient, as shown in Figure 5.10a, measures how similar an object is to its own cluster as compared to other clusters. Figure 5.10b shows the variations of *cohesion* and *separation*. The result with a high value of separation and a low value of cohesion is preferred. The results are based on the data in Foshan city, which indicates that when the distance threshold dis_{thres} is between 5 and 10, we can make sure the silhouette coefficient and separation value are high and the cohesion value is relatively low. Therefore, in our experiment, the value of 7 is chosen as the distance threshold dir_{thres}.

An appropriate grid size is chosen to balance the quality of clustering and computing



(a) 160 × 150 meshing grids

(b) 480 × 450 meshing grids

Figure 5.11: Heat maps in different grid sizes.

time. We investigate the effect of spatial resolution, i.e., the granularity of the mesh grid. Besides the currently applied resolution of 320 × 300 (see Figure 5.7b), two cases with coarser (160 × 150) and finer (480 × 450) mesh grids are compared in Figure 5.11. For the coarser mesh grid, the size of each grid is $2.26km \times 2.56km$. Figure 5.11a indicates some artificially inter-connected regions with high density in the urban area, which is an artifact compared with the result in Figure 5.7b. Thus, a mesh size of 160 × 150 is believed to be inadequate. On the other hand, when the mesh size is increased by 50% to 480 × 450, the size of each grid is $1.13km \times 1.28km$. Figure 5.11b indicates a similar result as compared with that in Figure 5.7b. No additional features can be extracted with increased spatial resolution. However, an increase in the spatial resolution by 50% leads to a rapid increase in the computational cost, with the computation time increasing from $424s$ to $3112s$. Thus, a mesh size of 320 × 300 as applied in the present study is sufficient and appropriate.

5.6 Conclusion

In this chapter, a multi-scale trajectory clustering algorithm for corridor identification in mobile networks is proposed. We consider the non-homogeneous distribution of cell towers in our proposed distance measure, which gives better clustering results of trajectories compared to other state-of-the-art distance measures. For example, compared with

using the Hausdorff distance measure, our proposed method improved cluster quality by more than 10%. The identified corridors are shown as a weighted graph, which can better show users' movement patterns within a frequent path. Our findings could help mobile operators to identify the key focus areas (i.e., corridors) in large-scale deployments of 5G networks for cost minimization.

In the next chapter, we focus on finding the significantly different corridors during different time periods, which can help network managers better understand the dynamics and changes in mobile phone users' movement patterns.

Chapter 6

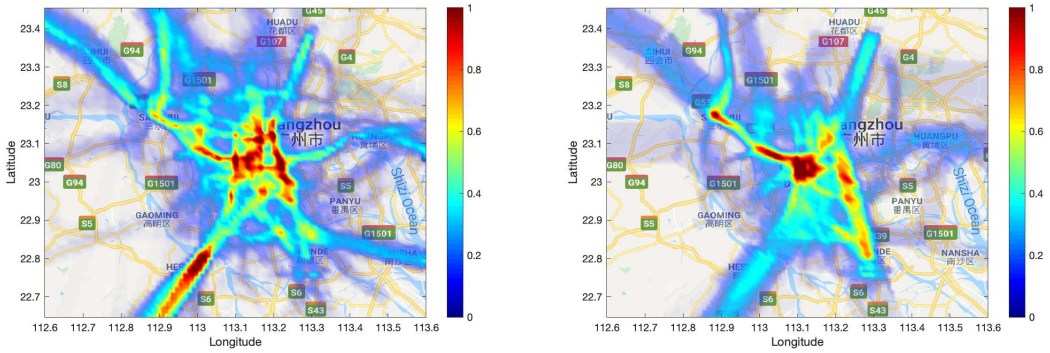
Discovery of Contrast Corridors from Trajectory Data in Heterogeneous Dynamic Cellular Networks Using Contrast Mining

In our previous chapter, we studied how to extract static movement patterns of mobile users from the trajectories generated during a specific time period to help with the management and orchestration of network resources. However, movement patterns of mobile users are not static over time. Understanding significant differences in mobile users' movement during different time periods can provide insights for mobile operators to dynamically reconfigure the network in response to the changes in traffic flows by time of day. Therefore, in this chapter, we propose a framework based on contrast data mining to identify significantly different movement patterns, which we model as corridors, during different time periods. To measure the difference, an improved distance measure based on a modified Hausdorff distance and earth movers' distance is proposed to calculate the dissimilarity between the identified corridors, which considers the heterogeneity of mobile networks. To further extract the significantly different corridors, we formulate the definition of contrast corridors of mobile users' movement. Experimental results on synthetic datasets as well as real-life datasets collected by China Mobile show that our method can effectively and robustly detect contrast corridors from trajectories generated from different time periods in mobile networks by improving the F1 score by 20% on average.

The paper arising from the work in this chapter is paper P5.

6.1 Introduction

A major challenge in the management of mobile networks is how to provide high bandwidth coverage to large numbers of mobile users. In particular, understanding and dis-



(a) morning (b) afternoon

Figure 6.1: Heat maps of mornings and afternoons. (The results are obtained based on the methods in Chapter 5.)

covering significant changes in the movement patterns of users in mobile networks can help service providers in the deployment of networks and base stations, and the management of network resources. Some studies have focused on identifying the underlying geographical corridors of users, which can be treated as pathways that are frequently traversed by a considerable number of mobile users [136, 137]. However, most studies tried to find the pathways based on the data during one specific time period and treated the network as temporally homogeneous in their analyses.

Therefore, in our work, we focus on the problem of identifying what are those significant changing corridors, which we model as contrast patterns that can be used to identify targets for network configuration. We consider two challenges from real-life data. The first challenge is that mobile trajectories are coarse and their granularity varies due to non-uniform spatial distribution of cell towers. Thus, it is necessary to propose a distance measure that can deal with the heterogeneous scales when measuring the dissimilarity between trajectories. The second challenge is that identifying static corridors plays an important role in managing networks for the long term design of network, but with the introduction of new generations of cellular network technology, such as 5G, there is a great opportunity for dynamically reconfiguring the network in response to changes in traffic flows by time of day. For example, users' movement patterns might be different in the morning to the patterns in the afternoon, as shown in Figure 6.1.

In this chapter, we propose to use contrast data mining on trajectories generated in mobile networks to analyze the changes in phone users' movement patterns during different time periods. To the best of our knowledge, this is the first study that focuses on the temporal changes of human trajectories generated from heterogeneous mobile data networks. Our main contributions are: (1) We propose a modified Hausdorff distance to measure the dissimilarity between corridors in heterogeneous mobile networks. (2) We propose a contrast corridor mining algorithm based on Earth Movers' Distance to detect the differences/changes in movement patterns during different time periods. (3) We conduct experiments over the real-life data of mobile users in a southern province in China as well as a synthetic dataset to evaluate our approach, which shows improvements in both interpretability and detection accuracy.

6.2 Related Work

Contrast patterns are often defined as patterns whose supports differ significantly among the datasets that are under contrast [30], which can describe discriminative behavior between classes or emerging trends between datasets with respect to a property of interest by means of an understandable representation [43]. In the problem of contrast mining on trajectory data, the aim is to find discriminative patterns (e.g., sequences, graphs, matrices, tensors) that occur frequently in one dataset and infrequently in another. For example, in the work of Wang et al. [114], a framework for discovering the impact of road closures on traffic flows was proposed. By computing the growth rate of traffic flows on n-Edgesets, the emerging n-Edgesets were selected by using the LOF [15]. In our work, we focus on finding the discriminative corridors, which are represented as directed weighted graphs, between two trajectory data sets generated in different time periods.

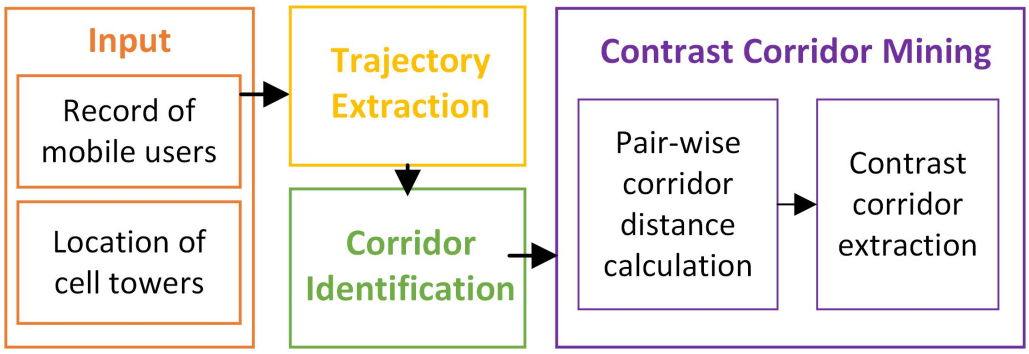


Figure 6.2: Framework of our proposed method.

6.3 Overview of Problem

In this section, we introduce some definitions and formulate our problem. Suppose that in a mobile network, there are N cellular towers, which are represented as $C = \{c_1, c_2, ..., c_N\}$, and each cell tower has a unique identifier and is associated with its coordinates. According to the definition in Chapter 5, a corridor can be treated as a pathway that is frequently traversed by a considerable number of mobile users. It can be represented as a graph, denoted as $cor = \langle V, E \rangle$, where V is the set of cells in the corridor, E is the set of all the edges in the graph and the weight of each edge represents the traffic load between the corresponding two nodes.

Here we focus on characterizing the major differences between these corridors in different time periods. Specifically, given the historical trajectories of M mobile users during two different time periods, i.e., the positive trajectory data set $TRAJ^+ = \{Traj_1^+, Traj_2^+, \ldots, Traj_M^+\}$ and the negative trajectory data set $TRAJ^- = \{Traj_1^-, Traj_2^-, \ldots, Traj_M^-\}$, and the identified corridor sets $COR^+ = \{cor_1^+, cor_2^+, \ldots\}$ and $COR^- = \{cor_1^-, cor_2^-, \ldots\}$, our research question is how to detect the significantly different corridors, *contrast corridors*, between COR^+ and COR^-. In order to answer this question, we study the following two sub-problems:

(1) How to measure the dissimilarity between two corridors, i.e., how can we calculate the distance between corridors?

(2) How to define and mine contrast corridors?

The framework of our proposed method is illustrated in Figure 6.2. As introduced

in Chapter 5, the inputs are the record of mobile users and the location of cell towers (longitude and latitude). Each record includes user ID, cell ID and the corresponding timestamp. There are three main steps to identify the contrast corridors in different time periods. The first step is trajectory extraction. In the first step, data are preprocessed by data aggregation, data cleaning and oscillation resolution [118]. Then the trajectory of each user is extracted according to our definition in Section 5.3. In the second step, corridors are identified by using the method proposed in Section 5.4. The final step is to identify significantly different corridors by using contrast mining. In particular, the pair-wise distances between corridors are calculated and contrast corridors are formally defined and extracted.

6.4 Methodology for Contrast Corridor Mining

In the following subsections, we describe the details about our methodology for contrast corridor mining.

6.4.1 Distance Measure for Corridors

In order to measure the distance between two corridors, we propose an algorithm that is based on Earth Mover's Distance (EMD) [94]. The EMD is a method that is applied to evaluate the dissimilarity between two multi-dimensional distributions in some feature space where a distance measure between single features, which we call the ground distance, is given. It is proportional to the minimum amount of *work* required to change one distribution into the other, where a unit of work is defined as the amount of work needed to move a unit of weight by a unit of *ground distance*. Here some other distance measurements for two distributions, such as K-L divergence (Kullback-Leibler divergence), cannot be adopted to measure the distance between corridors. This is because K-L divergence cannot take the geographical information of corridors into consideration, whereas the geographical information can be modeled as ground distance in EMD.

Suppose there are two corridors $cor_p = G_p \langle V_p, E_p \rangle$ and $cor_q = G_q \langle V_q, E_q \rangle$, where V denotes the vertices and E represents the edges in the corridors. For each non-zero edge

in a corridor, the weight of it, w, is defined as the weight of the edge, which indicates the traffic volume between two cells. Each corridor can be treated as a distribution, which has a set of edges with their corresponding weights, i.e., a *cor* can be represented as $\{(e_1, w_{e_1}), (e_2, w_{e_2}), ...\}$, where e is an edge in the graph and w_e is the weight of edge e. $\{(e_1, w_{e_1}), (e_2, w_{e_2}), ...\}$ is also called the *signature* of the distribution.

Then the ground distance matrix between two corridors is defined as $D = [d_{kl}]$ ($k = 1, ..., m, l = 1, ..., n$), where m and n is the number of edges in corridor p and q respectively, and d_{kl} is the ground distance between edge $e_k \in E_p$ and edge $e_l \in E_q$. Considering the non-uniform distribution of cell towers, we propose to calculate the ground distance between two edges based on a modified version of the distance measure proposed in Chapter 5, which will be described in Section 6.4.2. Then we can formulate the problem as a linear programming problem.

Given the corresponding signatures of two corridors and the ground distance matrix, our aim is to find a flow $F = [f_{kl}]$, where f_{kl} is the flow between edges e_k and e_l, that minimizes the overall cost defined as:

$$WORK(p, q, \mathbf{F}) = \sum_{k=1}^{m} \sum_{l=1}^{n} f_{kl} d_{kl}, \tag{6.1}$$

subject to the following constraints:

1. $f_{kl} \geq 0$;
2. $\sum_{l=1}^{n} f_{kl} \leq w_{e_k}, 1 \leq k \leq m$;
3. $\sum_{k=1}^{m} f_{kl} \leq w_{e_l}, 1 \leq l \leq n$;
4. $\sum_{k=1}^{m} \sum_{l=1}^{n} f_{kl} = min(\sum_{k=1}^{m} w_{e_k}, \sum_{l=1}^{n} w_{e_l})$;

The first constraint means that only the weight from cor_p can be moved to cor_q. The second constraint ensures that the total weight moved from an edge in cor_p to cor_q should be no more than its own weight, and the third constraint ensures that the total weight moved to any edges in cor_q should be no more than their own weight. The last constraint

requires that the total weight moved should be equal to the total weight of the lighter corridor.

Then the EMD is defined as the work normalized by the total flow:

$$EMD(cor_p, cor_q) = \frac{\sum_{k=1}^{m}\sum_{l=1}^{n} f_{kl} d_{kl}}{\sum_{k=1}^{m}\sum_{l=1}^{n} f_{kl}}. \tag{6.2}$$

6.4.2 Multi-scale Hausdorff Distance

In order to calculated the ground distance between two edges, here we adopt the distance measure proposed in Chapter 5 with additional modifications by considering the direction. Here we briefly describe it and then introduce our modifications on it. Hausdorff distance is the maximum distance of a set to the nearest point in the other set, which represents the maximum mismatch level between two point sets [50].

Algorithm 7 $\Delta(T_1, T_2)$

Input: Two tracklets T_1 and T_2
Output: $\Delta(T_1, T_2)$

$\Delta(T_1, T_2) = 0$; **for** each c_i^1 in T_1 **do**
  $d_{c_i^1 T_2} = \infty$; **for** each line segment $c_j^2 c_{j+1}^2$ in T_2 **do**
    calculate the distance $d_{c_i^1 (c_j^2 c_{j+1}^2)}$ between a point c_i^1 and a line segment $c_j^2 c_{j+1}^2$;
    $d_{c_i^1 T_2} = min(d_{c_i^1 T_2}, d_{c_i^1 (c_j^2 c_{j+1}^2)})$;
  calculate $\alpha_{c_i^1}$ using Equation (5.1) and (5.2) ; $\Delta(T_1, T_2) = max(\Delta(T_1, T_2), \alpha_{c_i^1} \cdot d_{c_i^1 T_2})$;
calculate β using Equation (6.3), (6.4), (6.5); $\Delta(T_1, T_2) = \beta \Delta(T_1, T_2)$.

This measurement can be used to find the closeness of two set of points. However, it ignores the direction information of sequences. Therefore, we propose to add another factor that considers the direction of movement. Here we use the accumulated direction of all the segments in one tracklet as the direction of movement of the tracklet. Given a tracklet $T_i = \{c_1, c_2, ..., c_m\}$, the direction of movement of it can be calculated as:

$$dir_{T_i} = \sum_{i=1}^{m-1} dir(s_i, s_{i+1}). \tag{6.3}$$

Then the direction difference between two tracklets T_i and T_j is:

$$dir(T_i, T_j) = \frac{dir_{T_i} \cdot dir_{T_j}}{|dir_{T_i}| \cdot |dir_{T_j}|}. \tag{6.4}$$

The direction distance will be used to obtain the second normalization factor β, which can be calculated as:

$$\beta = \begin{cases} 1/dir(T_i, T_j), & dir(T_i, T_j) > 0 \\ \infty. & dir(T_i, T_j) \leq 0 \end{cases} \tag{6.5}$$

Then the distance between two tracklets after normalized is:

$$\Delta(T_1, T_2) = \beta \cdot \Delta(T_1, T_2). \tag{6.6}$$

The pseudo-code is provided in Algorithm 7.

6.4.3 Contrast Corridor Mining

Algorithm 8 Contrast Corridor Mining

Input: a positive corridor set P and a negative corridor set Q
Output: a set of contrast corridors Cor_{con}

$Cor_{con} = \varnothing$;
for each cor_{p_i} in P **do**
  $common = 0$; **for** each cor_{q_j} in Q **do**
    calculate the EMD distance $EMD(cor_{p_i}, cor_{q_j})$ and the flow matrix F between cor_{p_i} and cor_{q_j} ; $supp = \frac{\sum_{k=1}^{m}\sum_{l=1}^{n} f_{kl}}{\sum w_{p_i}}$; **if** $EMD(cor_{p_i}, cor_{q_j}) < dis_{thres}$ and $supp > supp_{thres}$ **then**
      $common = 1$;
  if $common \neq 1$ **then**
    $Cor_{con} = Cor_{con} \cup \{cor_{p_i}\}$;

Given two different datasets, two sets of corridors $P = \{cor_{p_1}, ..., cor_{p_i}, ..., cor_{p_{N_1}}\}$ and $Q = \{cor_{q_1}, ..., cor_{q_j}, ..., cor_{q_{N_2}}\}$ can be found respectively using our proposed corridor identification method, where N_1 and N_2 are the numbers of corridors identified in these

two datasets, and cor_{p_i} $(1 \leq i \leq N_1)$ and cor_{q_j} $(1 \leq j \leq N_2)$ represent each individual corridor in these two datasets.

Then we define an emerging corridor as follows:

Definition 6.1. ***Emerging Corridor*** *A corridor cor_{p_i} in P is defined as an emerging corridor if it satisfies any of these two following conditions:*

1. ***Distance*** *If the distance between corridor cor_{p_i} and any other corridors in Q is greater than a threshold dis_{thres}, then the corridor is treated as an emerging corridor in P, which indicates that cor_{p_i} is sufficiently different from any corridors in Q.*

$$EMD(cor_{p_i}, cor_{q_j}) \geq dis_{thres}, \forall cor_{q_j} \in Q; \tag{6.7}$$

2. ***Support*** *If two corridors are similar to each other but a certain amount of earth of one corridor cannot move to the other corridor, then it will be treated as an emerging corridor. Satisfaction of this condition means that the corridors under contrast are similar but their supports are significantly different.*

$$\frac{\sum_{k=1}^{m}\sum_{l=1}^{n} f_{kl}}{\sum w_{p_i}} \leq supp_{thres}, \forall cor_{q_j} \in \left\{cor_q | EMD(cor_{p_i}, cor_q) \leq dis_{thres}, cor_q \in Q\right\}, \tag{6.8}$$

where $\sum w_{p_i}$ is the sum of the weights of all the edges in corridor cor_{p_i}.

The pseudo-code is provided in Algorithm 8.

6.5 Experiments and Results

In this section, we first evaluate our proposed algorithm on a synthetic dataset in terms of accuracy, F1 score, precision and recall. Then two real-life case studies are conducted to further evaluate the effectiveness of our method.

6.5.1 Contrast Mining on Synthetic Data

To validate the effectiveness of our proposed method, we compare it with two other baseline methods:

1. **Non-density** This method is generally similar to our proposed method except for the distance metric between two edges, i.e., the original Hausdorff distance is adopted. (The non-density method can be any other state-of-the-art method that does not take the heterogeneity into account, such as DFM, DTW, LCSS, EDR and SSPD.)

2. **Node-based** The ground distance is defined as the spherical distance between two nodes, and for each node in a corridor, its weight is defined as the degree of the node, which is the number of edges that are incident to the node.

Synthetic data are generated by utilizing the real corridors identified by the method proposed in Chapter 5. We generate positive and negative datasets based on the data of Foshan City. The details are as follows.

Given a set of corridors $COR = \{cor_1, cor_2, \ldots, cor_n\}$, we first divide the corridor set into two equal-sized subsets (here we assume n to be even.), i.e., $COR_1 = \{cor_1, \ldots, cor_{n/2}\}$ and $COR_2 = \{cor_{n/2+1}, \ldots, cor_n\}$. The corridors in COR_2 are considered as the negative dataset, i.e., $COR^- = COR_2$. Then the corridors in the positive dataset COR^+can be generated by considering the following four scenarios:

1. $n/2$ corridors have same distribution as the corridors in COR_2;

2. $n/2$ corridors have same distribution as the corridors in COR_1;

3. $n/2$ corridors have same cell set as the cell set of corridors in COR_2 but with distinct distributions;

4. $n/2$ corridors have same distribution as the corridors in COR_2 but with significantly different amount of traffic volume.

Corridors generated by scenario 1 are treated as common corridors for both positive and negative datasets, whereas corridors generated by scenarios 2, 3 and 4 are emerging corridors according to our definition.

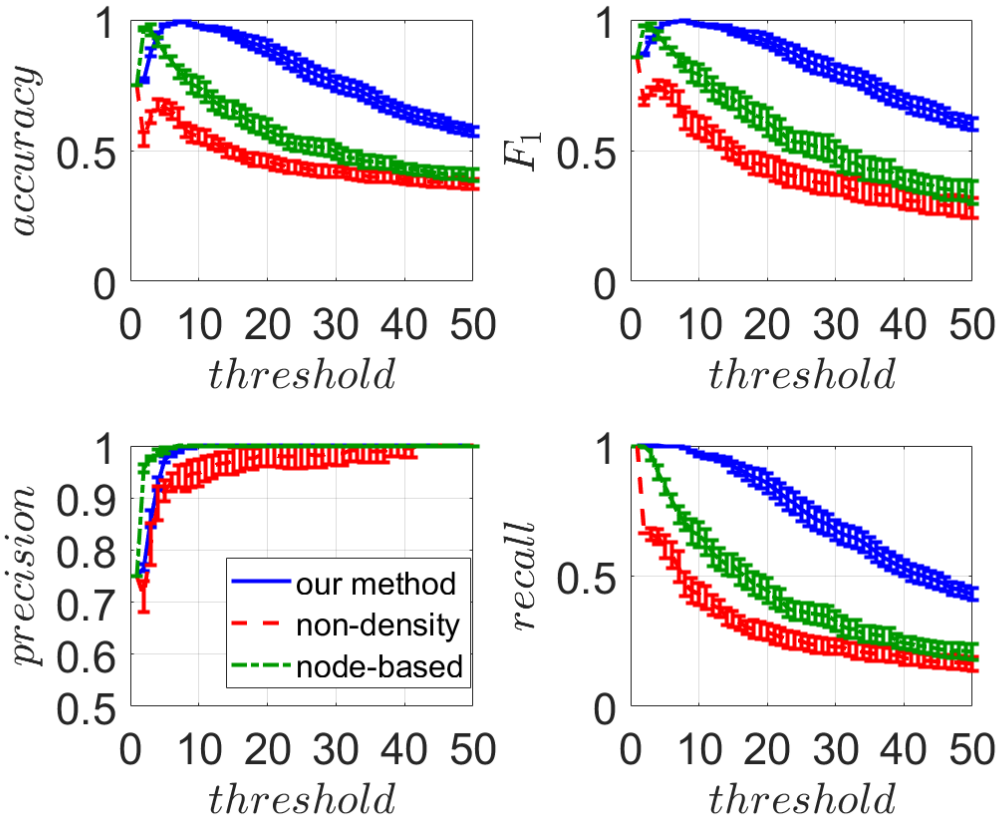
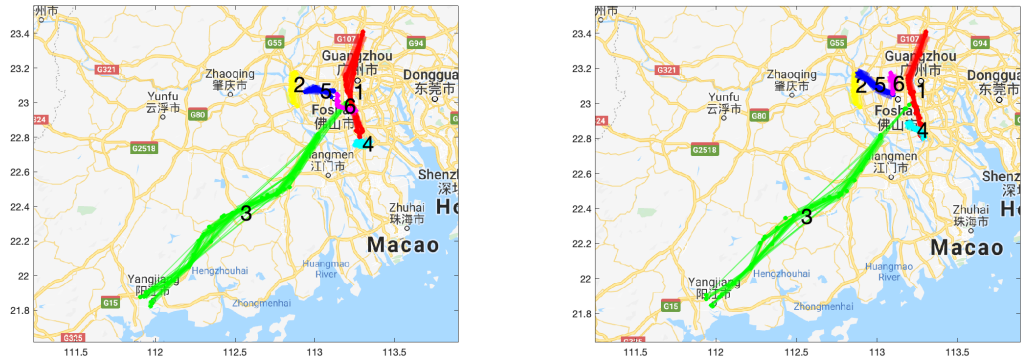


Figure 6.3: Performance comparison in emerging pattern finding between our proposed method and two reference methods.

Figure 6.3 shows the comparison between our proposed method and the other two baselines in terms of accuracy, F_1 score, precision and recall. The results indicate that our proposed method performs better than the other two methods with varying distance thresholds (which has been normalized for comparison). In the node-based method, the direction of the movement pattern is ignored. Therefore, some close-located corridors may be treated as very similar even though they have different directions/connections between nodes.



(a) positive (b) negative

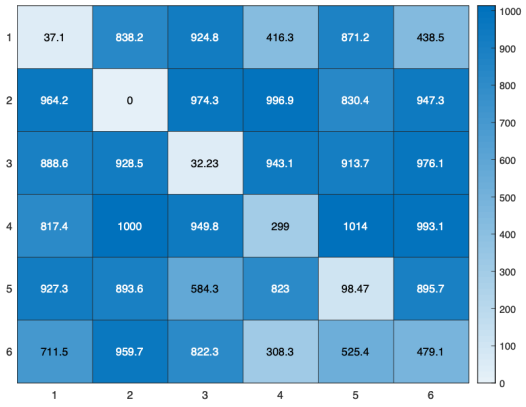
Figure 6.4: Two sets of corridors under contrast.

6.5.2 Contrast Mining on Real Data

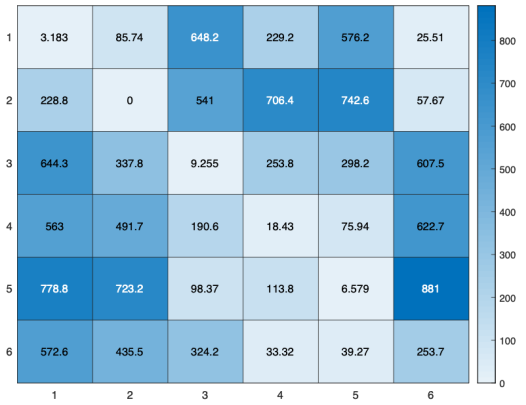
The real-life dataset was originally collected by China Mobile, which contains 5,000 mobile users from a province in South China. The cell locations (longitude and latitude) of each user were recorded every 5 minutes in a time period of three weeks (from 23:55 14/11/2015 to 23:50 05/12/2015).

First, we select two corridor sets to illustrate the effectiveness of our proposed method compared with two other baselines. As shown in Figure 6.4, there are six corridors in each of the two corridor sets under contrast. While three of the corridors are the same in both the positive and negative datasets, the other corridors are different from each other. Figure 6.5 shows the distance matrices we obtained by using different methods. By using our proposed method, clearly similar corridors have smaller distances between each other. Although the non-density and node-based method can also identify similar corridors, incorrect conclusions may be obtained when calculating the distance between corridor 5 in the positive dataset and corridor 5 in the negative dataset. This is because these two corridors have similar directions and they are located close to each other in a dense area. Our method can deal with this well since it normalizes the distance by considering the heterogeneous distribution of cell towers.

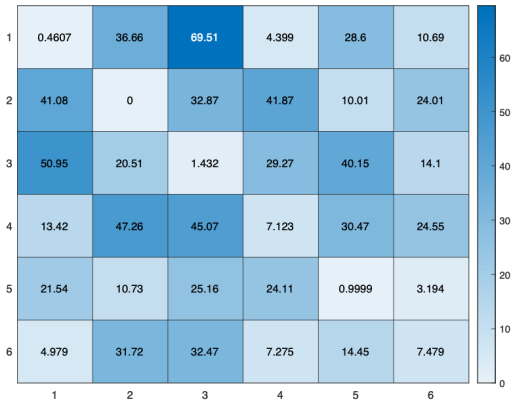
Then two sets of experiments are conducted using the dataset in Foshan City, i.e., weekday (Mon-Fri) vs weekend (Sat-Sun) and morning (0:00-12:00) vs afternoon (12:00-



(a) our method



(b) non-density



(c) node-based

Figure 6.5: The distance matrix between two sets of corridors obtained by three different methods.

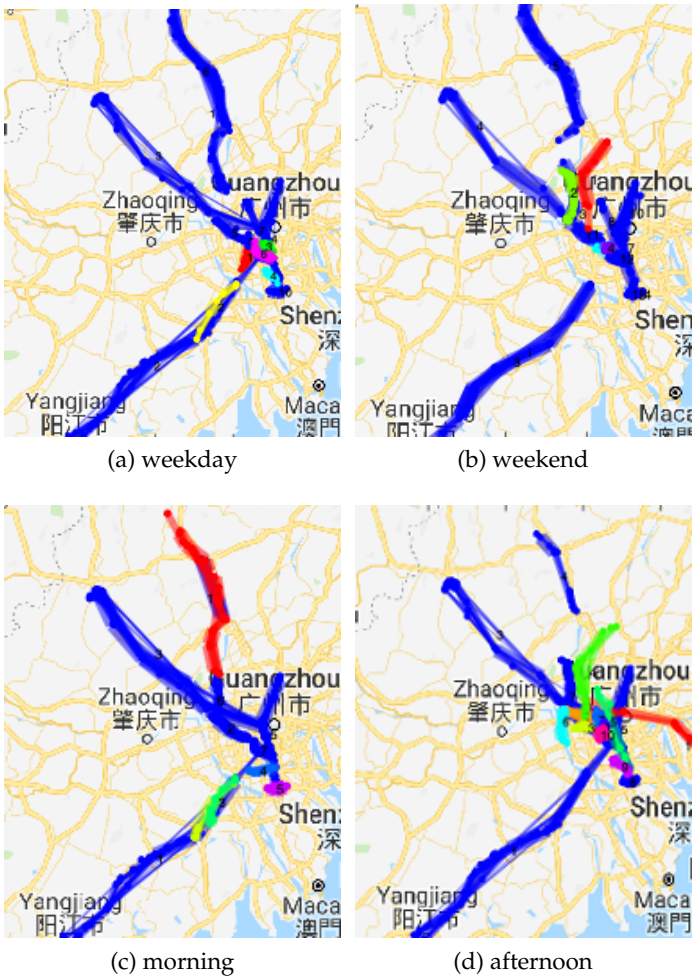
0:00). The EMD between weekdays and weekends in Figure 6.6a is 0.7632. There are more emerging corridors identified on weekdays in Figure 6.6a, and most of them are located in the central area, which indicates that people would move more frequently on weekdays, especially in the central area. The emerging patterns on the weekdays may be attributed to work commutes. By contrast, some emerging patterns surrounding the urban area (e.g., beltway) are identified on weekends, as shown in Figure 6.6b, and these patterns may be explained by leisure activities out of the city center on weekends.

The results of morning vs afternoon (Figure 6.6c and Figure 6.6d) indicate that the city is more active in the afternoon compared with in the morning, since more emerging corridors are identified in the afternoon. Most emerging corridors in the morning are located outside the city area, while in the afternoon some emerging corridors are identified in the central area. Note that the afternoon time period starts from 12:00 to 0:00, which means the corridors are generated not only from the afternoon activity but most of the evening movement of mobile users.

6.6 Conclusion

In this chapter, a framework for mining the changes in the movement patterns of mobile users is proposed. We consider the non-homogeneous distribution of cell towers in the distance measure, which is more appropriate for trajectories generated in mobile networks compared to other state-of-the-art distance measures. A contrast corridor mining algorithm is also proposed to find significant changes between the corridors generated in different time periods. Both synthetic and real-life datasets are applied to validate the effectiveness of our proposed method. Contrast corridors can be effectively detected from trajectories in mobile networks, and our method outperforms others by an average 20% improvement in the F1 score. Our findings could help mobile operators to identify the key focus areas (i.e., corridors) in large-scale deployments of 5G networks for cost minimization, and the ability to identify the temporal dynamics of corridor patterns can help the management and orchestration of 5G network resources.

In the next chapter, we will focus on the edge caching problem in mobile networks



(a) weekday (b) weekend

(c) morning (d) afternoon

Figure 6.6: Identified common and emerging corridors in different time periods. Corridors with blue color indicate they are common corridors in two time periods, while other colors are emerging corridors.

considering the heterogeneity of users' content preferences and mobility to further improve the overall performance in mobile networks.

Chapter 7

Adaptive Edge Caching based on Popularity and Prediction for Mobile Networks

In the previous two chapters, we studied how to identify common movement patterns of mobile users and significant changes in traffic flows between different time periods, which can provide insights to help with the management of network resources and the deployment of base stations. In this chapter, we focus on identifying optimal strategies for caching at the edge of networks, i.e, cell stations, to further contribute to the orchestration of 5G network resources. Edge caching in mobile networks can improve users' experience, reduce latency and balance the network traffic load. However, edge caching requires suitable strategies for determining what files to pre-fetch at which cell and at what time. Due to the heterogeneity of users' content preferences and mobility, caching based only on popularity has limitations. Considering that cells located in different places have different predictability, we propose an adaptive edge caching algorithm based on content popularity as well as the individual's prediction results to provide an optimal caching strategy, aiming to maximize the cache hit rate with acceptable file replacement cost. A heuristic optimization strategy based on genetic algorithms is presented, along with a prediction model based on an improved Markov model for each user according to the historical data. In the model, similar users are clustered based on their behavior patterns. We evaluate our algorithm on a simulation dataset as well as a 3-week real-life dataset from China Mobile. The results show that our optimal caching strategy can improve the cache hit rate compared with other methods, especially when the storage capacity is small and the similarity in content requests of users is low.

The publication arising from the work in this chapter is paper P4.

7.1 Introduction

Content caching at the edge of the mobile network has attracted increasing research attention recently. Caching at the wireless network edge (close to users) can help reduce latency, balance network traffic load, and improve users' experience [87]. However, since the storage capacity for each cell is limited, it is important to design appropriate caching strategies to balance the trade-off between the quality of users' experience, and the limited storage capacity. In this chapter, we focus on the problem of finding a strategy for caching at the edge of the network.

To design an appropriate caching strategy, we need to answer the question: what contents should be pre-fetched, at which cells and when? This is a challenging task, due to the following factors: (i) *Limited storage size*: Since the storage space in the edge node is limited, it is not possible to cache all the contents. Therefore, only those content items that can make the most profit (maximum hit rate/minimum cost) should be selected to be cached. (ii) *Users' mobility*: Users in the network may frequently be handed off from one cell to another. (iii) *Users' heterogeneous preferences*: The probability that each content item is requested by a specific user during a certain period can differ among individuals.

A simple caching strategy is to choose the most popular contents at the network edge. The limitation of this method is that the real popularity of the contents in the network is unknown and users' behavior (movement and content requests) in the network can be dramatically different, in which case popularity-aware caching may not be effective. A recent study [77] shows that the behavior of users in the network, including content, location, and mobility, affects the performance of edge content caching. In [70], the authors also noted that statistical patterns of content requests both in aggregated form and on a per-user basis should be considered in the content deployment problem.

Although there has already been some research on mobile users' behavior prediction [90,100,131], the prediction of users' behavior may not be easy. For example, in a city center, people move in various directions in a nondeterministic manner. Their behavior may be not predictable, leading to low prediction accuracy. This will reduce the performance gain of edge caching and introduce extra costs, such as the energy consumed in the backhaul and edge cells due to ineffective content placement. An important property of cells

in this content is their predictability in terms of users' movements. The predictability of a cell is the observed likelihood of being correct when we predict that a user will move into a given cell next. For example, if the prediction model predicts that 100 users will next move into a given cell, but only 60 actually do so, then we consider the predictability of that cell is 0.6. In Figure 7.1, using the prediction model we propose, the average prediction accuracy of the top 2,000 most visited cells within a city in southern China is shown. We can see that different cells may have different levels of predictability. For those cells in area L, the prediction accuracy for users' behavior is low. Whereas for cells located along a main road (area H), the prediction accuracy is relatively high. This indicates that we should not totally rely on the prediction results when developing the caching strategy.

Therefore, we propose to consider not only the prediction results from the prediction model, but also the predictability of the cells. The basic idea is that, for cells with low predictability, we would prefer to cache the most popular files; whereas in other cells with high prediction accuracy, we would prefer to cache files according to the results from our prediction model. According to this, we propose to construct a proactive caching strategy considering two aspects: the popularity and the prediction results of users' location and content requests. In particular, we propose a formal model to optimize these two aspects of the proactive model.

To summarize, our main contributions are:

- We formalize the edge caching strategy problem in cellular networks, to maximize the hit rate over the whole network under a replacement constraint. An adaptive caching strategy based on general popularity and personalized predictions is also proposed, and a heuristic solution based on genetic algorithms is provided for solving the optimization problem.

- We propose a Markov based model for the prediction of users' behavior in terms of movement and requests, which is inspired by [90]. To overcome the "cold-start" limitation of Markov models when a new cell is visited, a clustering method is also proposed to find users who share similar behavior patterns.

- Simulations are provided to evaluate the performance of our proposed algorithm.

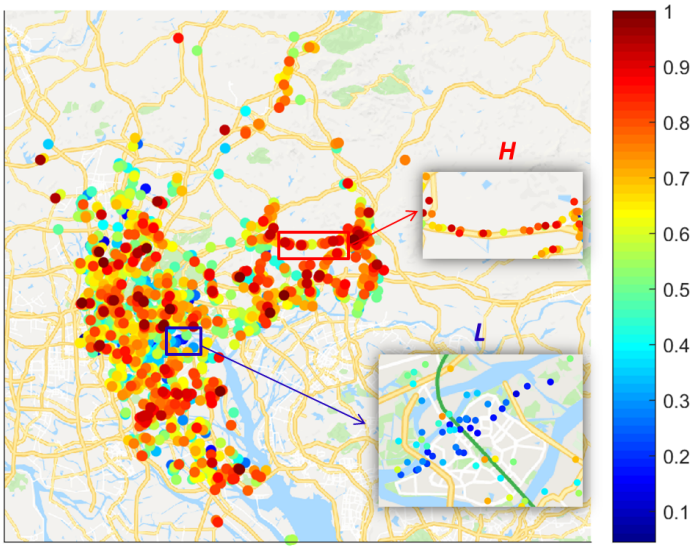


Figure 7.1: Averaged prediction accuracy of the top 2,000 most visited cells within one city in southern China. Area *H* denotes an area with high prediction accuracy, and area *L* indicates an area with low prediction accuracy.

We also test our algorithm on a real-life dataset from China Mobile. The results show that our algorithm outperforms all the existing schemes, especially when the cache capacity is low and the distribution of content popularity is skewed.

The remainder of this chapter is organized as follows. Section 7.2 provides a review of the related work. Section 7.3 introduces the model of the system and our adaptive edge caching strategy. In Section 7.4, a heuristic method based on genetic algorithms is proposed to solve our proposed optimization model. The prediction model is presented in Section 7.5. The details of our experiments on simulation data as well as real life data and a discussion of the results of our methods are shown in Sections 7.6 and 7.7. Section 7.8 concludes our work and proposes some future directions.

7.2 Related Work

The edge caching problem has attracted extensive attention recently due to its advantages for reducing latency, as well as relieving the heavy overhead burden on the network backhaul [80]. The current research mainly focuses on issues like the caching architecture

design, content deployment and delivery [130]. This paper falls within the problem of content deployment and delivery.

Two common caching schemes are Least Frequently Used (LFU) and Least Recently Used (LRU). They are simple but not robust methods, since their performance can be reduced by the heterogeneity of users in the network. To fully exploit the edge resources, as mentioned in several studies, e.g., [65, 91, 122], popularity-based cache placement schemes have been widely deployed to maximize the hit rate. For example, in [122], the authors found that content popularity varies at different locations. A linear prediction model is built to estimate the future hit rate of contents at different locations, and according to the results, contents that can have the highest hit rate will be cached. The authors in [65] proposed a model called PopCaching, which can predict the popularity and make the content caching decision on-line to maximize the hit rate. Naifu et al. [129] proposed a linear prediction model to estimate the future content requests based on historical data, and then an on-line cache replacement optimization model was built based on the future popularity prediction. Although these methods considered the change of popularity, none of them considered the characteristics of the mobile users' behaviors in the network.

Recently, some studies [70,77,112] have started to consider the movement and request characteristics of the mobile users in the wireless network. The basic idea is that if the users' trajectories and requests can be predicted based on historical data, then the cell can proactively cache appropriate files and the user can download the pre-fetched files along their trajectory. To achieve this, as mentioned in [70], we should not only consider the content popularity, but also the user's preference, which can be predicted and play a key role in the design of caching. In [112], the authors also pointed out that taking user mobility into consideration is critical for caching design in content centric wireless networks. Ge et al. [77] provided a thorough study on the behavior of mobile video users. A geo-collaborative caching strategy was also provided. They divided the cache storage into two parts according to the fractions of two types of users: single-location users and multi-location users. Dong et al. [71] argued that the common assumption that the preferences are identical among all users is not true in practice, and showed that

optimizing caching policy with individual users' preferences is beneficial. However, in [71], the user preferences were assumed known a priori, which actually cannot be known perfectly in advance in real-life applications.

In our work, we not only propose to integrate a personalized prediction model into the edge caching problem, but also provide an optimization model that can adaptively decide the preferences for popularity and prediction results. To the best of our knowledge, this is the first algorithm that considers the predictability of different cells when developing the cache deployment strategy.

7.3 System Model and Problem Formulation

In this section, we provide the description of the system model, and formulate the optimization problem.

7.3.1 Assumptions

Cells in the network: There are K cells in the network which can deploy caches, denoted as $C = \{c_1, c_2, ..., c_K\}$. However, the storage of a cell for caching is limited.

Proactive caching: We assume that the network is refreshed at fixed time slots. At the beginning of each time slot, the selected content files are pre-fetched at each cell so that user's content request can be processed with reduced latency. A time period with L time slots is represented as $T = \{t_1, t_2, ..., t_L\}$.

Files in the network: Suppose that there are N content files that may be requested in the network, denoted as $F = \{f_1, f_2, ..., f_N\}$. These files have been sorted by popularity, which is $p_F = \{p_1, p_2, ..., p_N\}$, where $p_i \geq p_j \geq 0$, $\forall i \leq j$, and $\sum_{f=1}^{N} p_f = 1$. We also assume that all files have the same unit size and the content popularity distribution does not change during the time period we considered.

Users in the network: We assume that the maximum number of mobile users in the network is M, and the mobile users in the network we are considering can leave or enter the network at any time. The user set is represented as $U = \{u_1, u_2, ..., u_M\}$. We also assume that at each time slot a user is served by its closest cell.

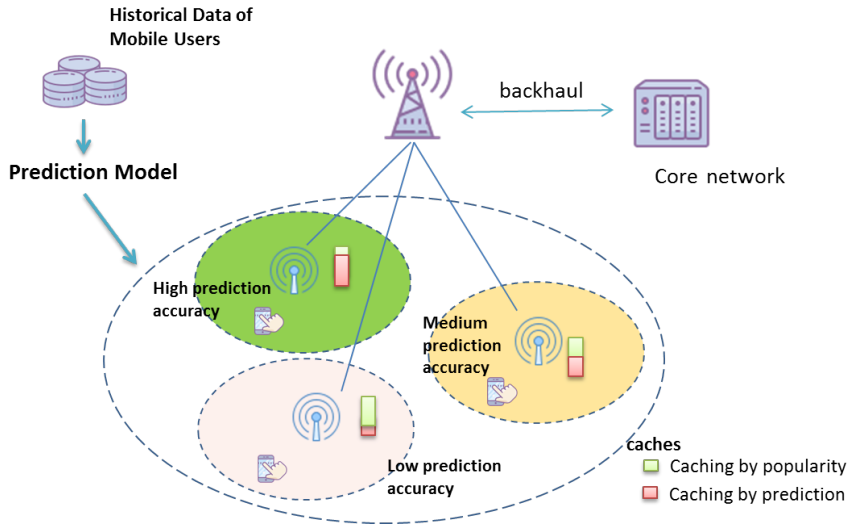


Figure 7.2: An example of the adaptive caching framework.

7.3.2 Caching Strategy

To improve the user experience at an acceptable cost, we plan to deploy different caching strategies for cells with different predictability levels. For the cells with a higher prediction accuracy, we prefer to pre-cache documents according to the users' prediction results, whereas in those cells where mobile users' behaviors are unpredictable, we consider pre-caching files according to their popularity, as shown in Figure 7.2. Based on historical data of the mobile users, a prediction model is built (refer to Section 7.5), and then an optimization model is proposed to obtain the preferences for popularity and personalized prediction results.

Therefore, we can represent the caching strategy as $\Phi = [\phi_1, ..., \phi_c, ..., \phi_K]$, where $c = 1, ..., K$, is the percentage of space that is used for popularity caching for cell c. For each cell, if we represent the files selected by popularity as F^1, and the files selected by prediction results as F^2, they should satisfy:

$$F^1, F^2 \subset F \; and \; F^1 \cap F^2 = \emptyset. \tag{7.1}$$

Suppose the storage of each cell is limited and fixed, denoted as s. Then the space limits for popularity caching and personalized caching of cell c would be $s \cdot \phi_c$ and $s \cdot$

$(1-\phi_c)$, respectively. This means F^1 and F^2 should satisfy:

$$|F^1| = s \cdot \phi_c, \;\; |F^2| = s \cdot (1-\phi_c), \tag{7.2}$$

where $|\cdot|$ denotes the number of files. If $\phi_c = 1$, it means that the files in cell c are cached solely according to the general popularity. If $\phi_c = 0$, it means that the files in cell c are cached by the personalized prediction. Equation (7.2) indicates that there are $s+1$ discrete values for selection in ϕ_c, i.e., $\phi_c \in \{0, 1/s, 2/s, \ldots, 1\}$.

The caching strategy for each cell can be described as the following two steps:

S1: First, the file f_i is cached according to the top popularity,

$$p_{f_i} \geq p_{f_j}, \;\; \forall f_i \in F^1, f_j \in F - F^1. \tag{7.3}$$

The cached file set F^1 contains the files of the highest popularity.

S2: Then we select files from the rest of the file set $F - F^1$ according to our prediction results. Files with the highest prediction confidences are selected. If we use $\hat{a}^t_{ufc} \in \{0,1\}$ to represent the prediction result of whether or not user u will request file f in cell c at time t, then the predicted request frequency for each cell can be calculated as:

$$\hat{q}^t_{fc} = \sum_{u=1}^{M} \hat{a}^t_{ufc}. \tag{7.4}$$

We select the file f_i based on the top prediction frequency:

$$\hat{q}^t_{f_ic} \geq \hat{q}^t_{f_jc}, \;\; \forall f_i \in F^2, f_j \in F - F^1 - F^2. \tag{7.5}$$

The cached files in F^2 have the highest prediction confidence. All the selected files in $F^1 \cup F^2$ will be cached in cell c at time t.

Based on these caching strategies, we can obtain the caching deployment of a file in a cell at a given time. We use a binary variable b^t_{fc} to represent whether or not the file f is

cached in cell c at time t,

$$b_{fc}^{t} = \begin{cases} 1 & file\ is\ cached \\ 0 & file\ is\ not\ cached \end{cases}. \tag{7.6}$$

The value of $\hat{a}_{ufc}^{t}$ is known a priori based on the prediction model, and the value of $\hat{q}_{fc}^{t}$ can be obtained accordingly. Given a value of ϕ_c, then the caching strategy b_{fc}^{t} of cell c can be achieved based on the popularity and the prediction.

7.3.3 Problem Formulation

Caching replacement: The replacement cost for refreshing caching files at time t is defined as the number of files that are cached at time slot t but not at the previous time slot $t-1$,

$$R_{c}^{t} = \begin{cases} \sum_{f=1}^{N} b_{fc}^{t} \cdot (1 - b_{fc}^{t-1}), & t \geq 2 \\ 0, & t = 1 \end{cases}. \tag{7.7}$$

We suppose that the replacement cost for all the cells at time t should be less than an upper threshold $Cost_r$,

$$\sum_{c=1}^{K} R_{c}^{t} \leq Cost_r. \tag{7.8}$$

Hit rate: A cache hit occurs when the requested data can be found in its cache. We use $a_{ufc}^{t} \in \{0, 1\}$ to represent whether the user u requests file f in cell c at time t or not. Then the average hit rate of cached data in cell c is:

$$H_{c}^{t} = \frac{\sum_{f=1}^{N} \sum_{u=1}^{M} a_{ufc}^{t} \cdot b_{fc}^{t}}{\sum_{f=1}^{N} \sum_{u=1}^{M} a_{ufc}^{t}}, \quad c \in C, t \in T. \tag{7.9}$$

The overall average hit rate of the network during the entire time period is:

$$\mathbf{H} = \frac{\sum_{t=1}^{L} \sum_{c=1}^{K} \sum_{f=1}^{N} \sum_{u=1}^{M} a_{ufc}^{t} \cdot b_{fc}^{t}}{\sum_{t=1}^{L} \sum_{c=1}^{K} \sum_{f=1}^{N} \sum_{u=1}^{M} a_{ufc}^{t}}. \tag{7.10}$$

Optimization model: In order to provide consistent Quality of Experience (QoE) to users, in our model, the objective is to maximize the hit rate of the whole network during a certain time period. Therefore, the optimization problem is:

$$max: \ \mathbf{H}(\Phi)$$
$$s.t. \ \textit{Equation} \ (7.2), \textit{Equation} \ (7.8).$$

The value of the objective function (7.10) can be calculated given a specific caching strategy, i.e., the value of Φ. In Algorithm 9, we provide the pseudo-code for calculating the objective function $\mathbf{H}(\Phi)$ in Equation (7.10) given $\Phi = [\phi_1, ..., \phi_c, ..., \phi_K]$. Apart from Φ, the other three inputs are the popularities of files which are assumed to be known a priori, users' real requests and user's predicted requests obtained by our prediction model, which will be discussed in detail in Section 7.5.

The optimal values are sought under two constraints, which are the division of the storage capacity constraint in Equation (7.2) and the total replacement cost constraint in Equation (7.8). The storage capacity constraint, which determines the proportion of cell storage used for content selected by popularity versus content selected by prediction, is directly related to the value of ϕ_c. As for the total replacement cost constraint, the optimal state of each cell, which is calculated independently of other cells, may not satisfy the total replacement cost constraint in Equation (7.8). Therefore, in calculating the optimal caching strategy, the states of all cells are essentially interconnected and need to be considered simultaneously.

For each cell c, there are $s+1$ possible discrete states for ϕ_c, i.e., $\phi_c \in \{0, 1/s, 2/s, \ldots, 1\}$. The non-linearity of the total replacement cost constraint in Equation (7.8) makes a fast algorithm with reduced complexity infeasible. The complexity of the problem, which includes K cells of $s+1$ discrete states, is $(s+1)^K$. The total replacement cost constraint is non-linear due to the non-linearity in Equation (7.7), and thus the optimization problem is non-linear and cannot be solved in polynomial time. This combinatorial problem makes it intractable to find the global optimal solution because of its exponentially large search space and the inclusion of a highly non-linear constraint. To solve the non-linear optimization problem, we propose to use stochastic optimization, such as Genetic Algorithms (GA), to find a probably local optimum. GA are considered as an appropriate choice for solving the current problem since the optimization model is highly non-linear and discontinuous [93]. High-quality solutions for this optimization can be obtained based on bio-inspired operators including mutation, crossover and selection. The details

are shown in the next section.

Algorithm 9 Calculate the Value of the Objective Function

Input: caching strategy $\Phi = [\phi_1, ..., \phi_c, ..., \phi_K]$, popularity of all the files, users' real request a^t_{ufc} and predicted request $\hat{a}^t_{ufc}$
Output: objective function $\mathbf{H}(\Phi)$

for each time $t \rightarrow 1, L$ **do**
  for each cell $c \rightarrow 1, K$ **do**
    $F^1 = \varnothing, F^2 = \varnothing$;
    // cache by popularity
    extract top $s \cdot \phi_c$ files to F^1 from F based on the popularity;
    // cache by prediction
    calculate the predicted request frequency $\hat{q}^t_{fc}$ using Equation (7.4) based on the predicted user request $\hat{a}^t_{ufc}$;
    extract top $s \cdot (1 - \phi_c)$ files to F^2 from $F - F^1$ based on the predicted request frequency;
    for each file $c \rightarrow 1, N$ **do**
      $b^t_{fc} \leftarrow 0$
      if $c \in F^1 \cup F^2$ **then**
        $b^t_{fc} \leftarrow 1$
      end if
    end for
  end for
end for
calculate **H** using Equation (7.10)
return hit rate **H**

7.4 Heuristic solution

Genetic Algorithms (GA), which were first proposed in [51], are a heuristic search and optimization technique inspired by natural selection, the process that drives biological evolution. In this section, a genetic algorithm is applied as the optimization search method, which is described in Algorithm 10. Basically, there are four steps.

S1: Generate a set of initial solutions (represented by chromosomes) as the first population. Each chromosome Φ, is a possible solution to the optimization problem. It stores an array of values, and each value (also called gene) represents the $\phi \in \{0, 1/s, 2/s, \ldots, 1\}$ value of each mobile cell.

S2: Calculate the fitness score $f(\Phi)$ of each chromosome Φ. The fitness function f determines how fit a chromosome is, which considers the objective function $\mathbf{H}(\Phi)$ in Equation (7.10) and the penalty of the constraint in Equation (7.8).

$$f = \mathbf{H}(\Phi) + g\left(max\left\{\sum_{c=1}^{K} R_c^t - Cost_r, 0\right\}\right), \tag{7.11}$$

where $g(\cdot)$ is the penalty function as proposed in [28].

S3: Generate n_P chromosomes as the successor population to replace the source population using the following steps.

Selection In the selection phase, we select two chromosomes Φ_1 and Φ_2 from the source population. The idea of the selection phase is to select the fittest individuals and let them pass their genes to the next generation. Individuals with high fitness have more chances to be selected for reproduction. Here Roulette Wheel Selection [8] is adopted.

Crossover We apply a crossover operator to Φ_1 and Φ_2 to generate a child chromosome Φ_{child} by combining the genetic information of two parent chromosomes. We adopt a k-point crossover operator [47], which incorporates k randomly picked crossover points from the parent chromosomes. The crossover rate r_c refers to the fraction of the next generation that are produced by crossover.

Mutation We apply an extended Power mutation [29], which is suitable for solving integer optimization problems, to produce Φ'_{child} by changing r_m of the gene values in Φ_{child}. Mutation helps introduce diversity within the population and prevent premature convergence.

Add the new child chromosome Φ'_{child} to the successor population.

S4: Evaluate the termination criterion. Calculate the fitness score $f(\Phi)$ in Equation (7.11) for n_P chromosomes. The algorithm terminates (converges) if it cannot produce new offspring that are significantly improved from those of the previous generation, and the chromosome with the highest fitness score is returned as the solution. If the criterion is not met, return to *S3*.

Algorithm 10 Genetic Algorithm

Input: population size n_P, crossover points k, crossover rate r_c, mutation rate r_m
Output: solution Φ

randomly generate n_P chromosomes as the first population;
evaluate the fitness score $f(\Phi)$ of each chromosome Φ in the first population;
while the termination condition is not satisfied **do**
  for $i \rightarrow 1, n_P$ **do**
    selection select two chromosomes Φ_1 and Φ_2 in the parent population according to the fitness scores;
    crossover apply k-point crossover to Φ_1 and Φ_2 to obtain the child chromosome Φ_{child} with a crossover rate r_c;
    mutation mutate a portion of r_m genes in Φ_{child} to generate Φ'_{child};
    append Φ'_{child} to the successor population;
  end for
end while
return Φ with the highest fitness score

7.5 Markov-based Prediction Model with User Clustering

A user's movements can be characterized by a Markov stochastic process, which assumes that the next state is only influenced by the current state and a fixed number of previous states. In this section, we develop our Markov-based prediction model for user's mobility and preferences. The framework is shown in Figure 7.3. Basically, there are two steps: (1) In the first step, users' movements and content request behavior (sequences) are extracted from the database; (2) In the second step, to address the cold start problem a prediction model with an improvement by user clustering is proposed for more robust prediction. Here we discuss the details about the second step, which includes a second-order Markov model improved by user clustering. The direct benefit of the user clustering algorithm is an improved prediction accuracy of the user's next position/file request.

7.5.1 Markov Model

Here a second-order Markov model is adopted. In the training process, the transition probability of one user traveling from cells (c_{i_1}, c_{i_2}) to cell c_j is denoted as $p_{(c_{i_1},c_{i_2}),c_j}$, which can be obtained by:

$$p_{(c_{i_1},c_{i_2}),c_j} = p(c_j|(c_{i_1}, c_{i_2})) = N_{ij}/N_i, \tag{7.12}$$

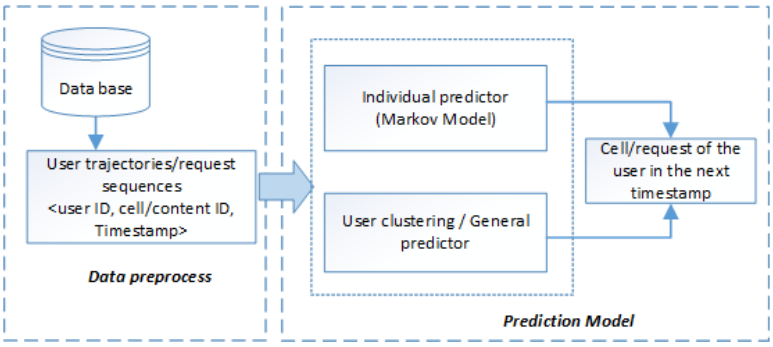


Figure 7.3: The framework of our Markov-based prediction model.

where N_{ij} is the occurrence of the user moving from cells (c_{i_1}, c_{i_2}) to cell c_j, and N_i is the frequency of cells (c_{i_1}, c_{i_2}) being visited by the user, (c_{i_1}, c_{i_2}) are the previous two cells visited by the user consecutively. The values of N_{ij} and N_i are calculated by the statistics of users in the data preprocessing.

Then in the prediction part, at time slot $t-1$, given the user's current cell c_{t-1} and previous cell c_{t-2}, we can predict the cell that the user will visit in the next time slot t to be:

$$\hat{c}_t = \arg\max_c \; p_{(c_{t-2}, c_{t-1}), c}. \tag{7.13}$$

Similarly, we can make the prediction for users' file request behavior in the next time slot $\hat{f}_t$. According to the predicted location $\hat{c}_t$ and file request $\hat{f}_t$, we can obtain whether or not the user will request file f in cell c at time t, i.e., the value of $\hat{a}^t_{ufc}$ as used in Equation (7.4).

7.5.2 User Clustering

To overcome the problem that Markov prediction may not work if the current cell has never occurred in the user's historical mobility behavior, we propose to utilize the historical mobility transition matrix from other similar users [90]. The basic idea is that, when the prediction cannot be made by using the user's own historical mobility pattern, we use the mobility pattern of users who have similar patterns to predict the next cell of this user.

Here we choose to use iVAT [13,111] for user clustering. iVAT is a useful tool for visual

assessment of clustering tendency, which displays a reordered dissimilarity matrix as a gray-scale image with a modified version of Prim's minimal spanning tree algorithm. There are three steps included as follows:

S1: *Feature selection*.

Each mobile user can be represented as a vector, $B_1 = [t_1, ..., t_j, ..., t_K]$, where K is the number of cells, and t_j is the number of time stamps the user has spent in cell c_j.

The visiting sequence is treated as a string, and the bigrams can be extracted for each user. Then a vector of $K(K-1)$ bigrams can be obtained, which is denoted as B_2.

After normalizing the two vectors B_1 and B_2 separately, we concatenate these two vectors together as the feature vector for each user.

S2: *Dissimilarity measurement*. Cosine distance is selected here as the dissimilarity measurement between two users.

S3: The dissimilarity matrix obtained from *S2* is fed to iVAT for clustering.

Figure 7.4 shows an example of the iVAT clustering result for 114 users in a small district. We can clearly see that seven clusters are detected for these users. Here notice that not all the users can be clustered together, since some users may behave quite differently from others. For those distinct users, the proposed improvement of the user clustering does not work.

The procedure of the prediction model is summarized as follows: we first extract the previous two cells of the user; if the user never visited these two cells consecutively before, the prediction model obtained by similar users will be applied, otherwise the individual prediction model will be applied for prediction.

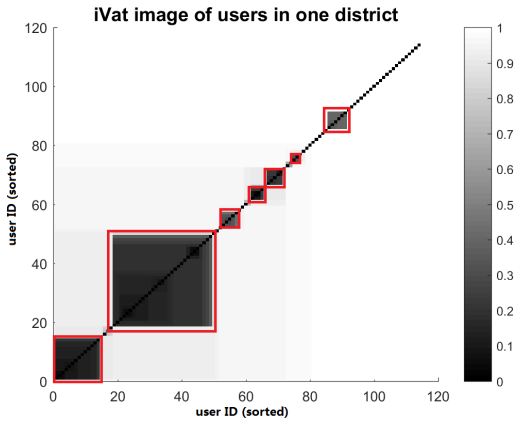


Figure 7.4: An example of user clusters of one district based on iVAT clustering.

7.6 Simulation

In this section, we describe our simulator to evaluate the effectiveness of our algorithm in terms of the cache hit rate compared with three benchmarks.

7.6.1 Simulation Setup

We simulate the mobility patterns of 2,500 users on a simulation area with 64 (8×8) cells using the smoothly truncated Levy walk algorithm [20], which simulates users' mobility pattern using the preset probability for travel distance, pause length, and change in direction. Users' content requests are modeled based on Poisson arrivals. The number of all the content files in the network is assumed to be 500,000. The popularity distribution of the files follows a Zipf distribution, which has been used in many existing works [91].

$$p_F \sim Zipf(\alpha, N),$$

where α is the distribution skewness parameter, and N is the total number of files in the network. The larger α is, the higher the skewness of the distribution. The total runtime of the simulation is around 1.5 hours.

7.6.2 Benchmarks

We compare the performance of our method with the following benchmarks:

- **Least Frequently Used (LFU)**: When the cache is full, discard the least recently used items first.
- **Least Recently Used (LRU)**: Purge the item with the lowest reference frequency when the cache is full.
- **Popularity Caching (PC)**: The most popular contents over the whole network will always be cached at each cell. Here we suppose the popularity of files is known and not changing during the time period we are considering.

7.6.3 Performance Comparison

Here, we divide the cells randomly into 4 groups, and set the content request prediction accuracy as 30%, 50%, 70% and 90% respectively. We investigate the performance of our proposed method with varying cache capacity ($s = 10^0, ..., 10^3$) and with various skewness parameters ($\alpha = 0.4, 0.8, 1.2, 1.6$) in terms of the averaged cache hit rate. Figure 7.5 shows the results during a high network load of 90%, which means approximately 2,250 users out of 2,500 are active in the simulation area. We can extract the following insights: (1) As the storage capacity increases, the hit rates of all the algorithms grow, but our optimal method grows the fastest. This indicates that our method is better suited for resource-limited networks. (2) The performance of all the methods increases when α turns larger, and LRU, LFU and PC are very sensitive to the skewness of the content popularity distribution. (3) In the experiments, our optimal solution always outperforms the other benchmark methods, and the performance of the popularity based method is better than LFU and LRU. Especially, when α is low, the hit rate of our method is much higher than the other three methods. For example, when $\alpha = 0.4$, the hit rate of our method can achieve 0.6, while the hit rates of the other three methods are quite low, less than 0.05. When the value of α is low, the distribution of file popularity is close to a uniform distribution. In that case, the performance of LFU, LRU and PC are more similar to random caching, which results in low hit rates, whereas our model can achieve a better result because of the incorporation of a prediction model. This indicates that our method has better performance when mobile users have a lower similarity in content requests.

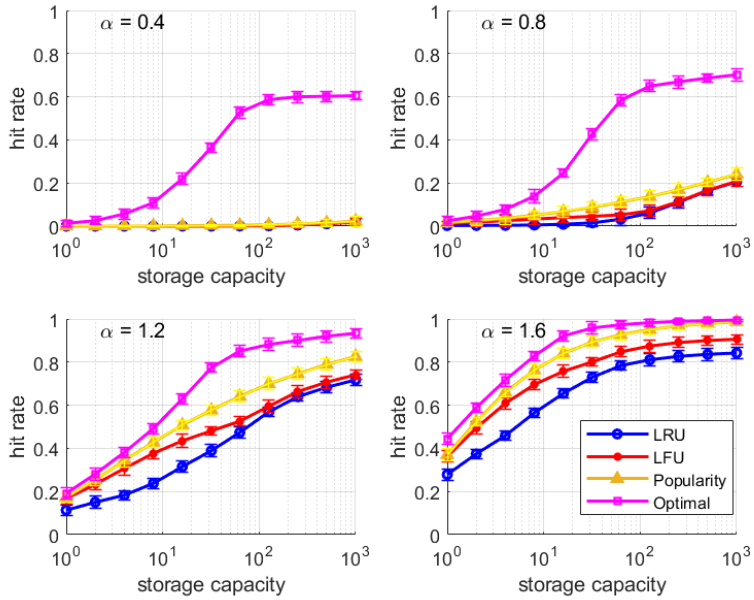


Figure 7.5: Averaged cache hit rate under different cache storage capacity ($s = 10^0, ..., 10^3$) and four different α values ($\alpha = 0.4, 0.8, 1.2, 1.6$).

(4) Our proposed method can effectively reduce the cache storage requirement. For example, if the target cache hit rate is 0.5, when $\alpha = 0.4$ and $\alpha = 0.8$, our optimal caching needs a cache capacity of around 100 files, while other benchmark methods require more than 1,000 files.

7.6.4 Parameter Selection and Performance of GA

There are four parameters in GA, i.e., population size n_P, number of crossover points k, crossover rate r_c and mutation rate r_m. Here, we mainly analyze the effect of n_P and r_c. For the mutation rate r_m, normally a small value of $0.5\% - 1\%$ is suggested, and here we select $r_m = 1\%$. The number of crossover points is selected as $k = 5$. The replacement cost constraint $Cost_r$ is selected to be $0.3sK$, where s is the cache storage of a cell and K is the total number of cells. Given different values of n_P and r_c with $\alpha = 0.8$ and $s = 100$, Table 7.1 compares three metrics including the number of generations needed to converge to the final state, the averaged hit rate and the maximum hit rate of the final generation. To ensure consistent results with less randomness, given a set of n_P and r_c

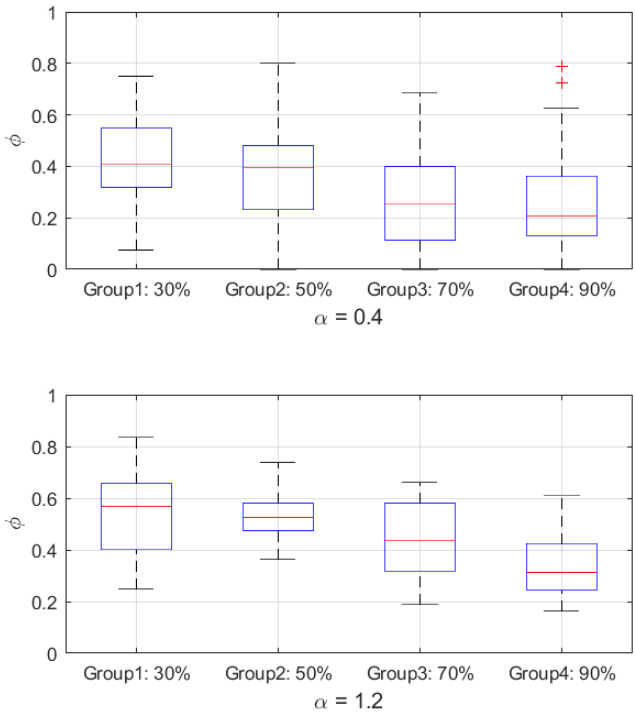


Figure 7.6: The ϕ values of cell groups with different values of prediction accuracy. For example, in Group1 the prediction accuracy is set as 30%.

Table 7.1: Parameter analysis of GA.

n_P	r_c	Generations	Average hit rate	Maximum hit rate
10	0.7	21	0.6144	0.6288
20	0.7	45	0.6506	**0.6555**
30	0.7	62	0.6516	0.6557
20	0.6	77	0.6496	0.6505
20	0.8	65	0.6515	0.6547

values, the averaged results of the GA running 20 times rather than a single run are listed in Table 7.1. A population size of 10 cannot preserve the population diversity, and the converged value is much lower than that of $n_P = 20$. As the population size n_P further increases from 20 to 30, the averaged hit rate and the maximum hit rate remain essentially unchanged. A variation in the crossover rate r_c between 0.6 and 0.8 has basically no effect on the results, and $r_c = 0.7$ leads to the fastest convergence. Thus, we select $r_c = 0.7$ in GA for the case studies presented in our work. A sample of the GA convergence curve is shown in Figure 7.7.

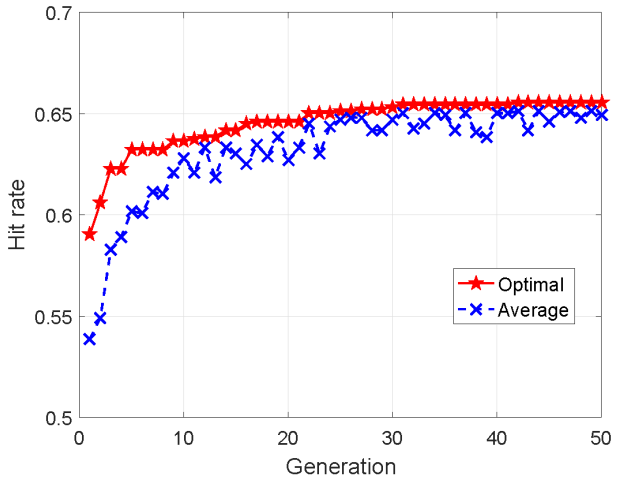


Figure 7.7: Convergence curve of GA at $n_P = 20$, $r_c = 0.7$.

7.6.5 Analysis of Optimal ϕ

The value of ϕ indicates the proportion we should cache according to popularity. Figure 7.6 shows the ϕ values of cell groups with different levels of accuracy when $\alpha = 0.4$ and $\alpha = 1.2$. Two key findings are listed as follows: (1) The ϕ value decreases as the prediction accuracy increases, which means that for cells with high predictability, we should have greater trust in the prediction result, and vice versa; (2) For the same group of cells, higher α results in larger ϕ. This indicates that when mobile users have higher similarity, the performance of caching by popularity becomes better.

7.7 Experiments and Evaluation

In this section, we evaluate our proposed model on a real-life dataset to further confirm its effectiveness. Case studies on two cities in the southern part of China are provided. We also analyze the influence of cache storage capacity and the similarity degree of mobile users on the hit rate for different methods.

7.7.1 Dataset Description

The real-life dataset, which was originally collected by China Mobile, is preprocessed to extract certain fields and protect users' privacy. In the dataset, we have 5,000 mobile users from one province in the southern part of China (nearly 80,000 cells). The total size of the data is 11.1 GBytes. Each user's behavior is recorded every 5 minutes during the period from 23:55 14/11/2015 to 23:50 05/12/2015, which means that there are 288 time slots per day and three weeks (6,048 time slots) in total. For each user, we have the cell location, service type ID (e.g., QQ, WeChat), and downlink/uplink bytes at different timestamps. Considering the limitation of the dataset, i.e., we only have the service type information in the dataset but not the content request information, in our experiments we assume that the content popularity follows the Zipf distribution. All the users' request information is obtained by simulation based on the service type, and noise is added to the prediction results.

7.7.2 Performance of Prediction Model

There have been many works in the area of prediction, whereas in our work, the main contribution is how to design the caching strategy based on popularity and prediction. However, we can still obtain an improved prediction accuracy by applying our proposed model. Here we evaluate the effectiveness of our proposed prediction model. Using datasets from three cities, the prediction accuracy of four models, i.e., the first-order, second-order, third-order Markov Model and our proposed model, are compared. Here, the prediction accuracy is defined as the ratio between the number of correct predictions and the total number of predictions for all the users.

As shown in Table 7.2, the second-order Markov Model achieves the highest prediction accuracy compared with other orders. This may be because the first-order Markov model ignores the sequential information of people's movement patterns, and the third-order Markov model that considers three previous states may suffer from the cold start problem. Thus, the second-order Markov Model is selected as the basic prediction model before applying the proposed improvement based on user clustering. The selection of the

Table 7.2: Comparison of prediction accuracy.

Model	City A	City B	City C
1^{st}-order Markov	0.56 +/- 0.20	0.53 +/- 0.17	0.59 +/- 0.22
2^{nd}-order Markov	**0.60 +/- 0.21**	0.61 +/- 0.16	0.66 +/- 0.19
3^{rd}-order Markov	0.51 +/- 0.22	0.57 +/- 0.18	0.62 +/- 0.21
Our proposed	**0.60 +/- 0.21**	**0.65 +/- 0.16**	**0.69 +/- 0.18**

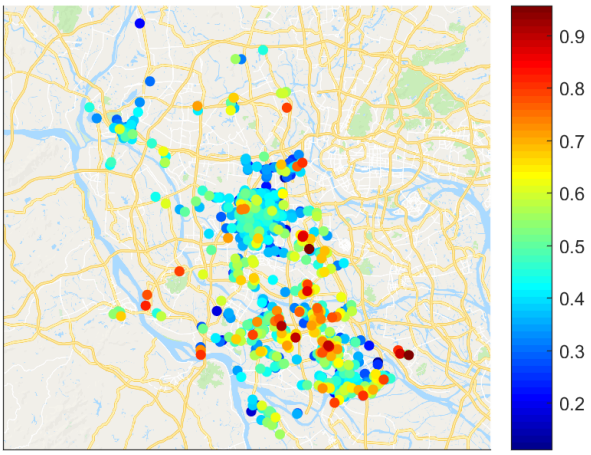


Figure 7.8: Averaged prediction accuracy of mobile users in the top 2,000 most visited cells within city A.

second-order Markov Model is consistent with the conclusion from other studies (e.g., [108]). The results also show that our proposed method outperforms the other three methods in terms of accuracy.

7.7.3 Case Study - City A

We use the data of the first two weeks as the training set, and the third week as the test set. We build a Markov model for users' movements by the data in the training set, and test the model on the test set to get our prediction result. The data in the training set is adopted to build the second-order Markov Model with user clustering improvement, and the data in the test set is applied to test our proposed model. Our first case study is on the data from city A. In city A, there are 7,796 cells in our data. Here, 1,000 cells with high visit frequency and 462 users are taken into consideration, as shown in Figure 7.8.

We set $\alpha = 1.6$, $N = 100,000$ and $s = 10$. Based on the characteristics of the data set,

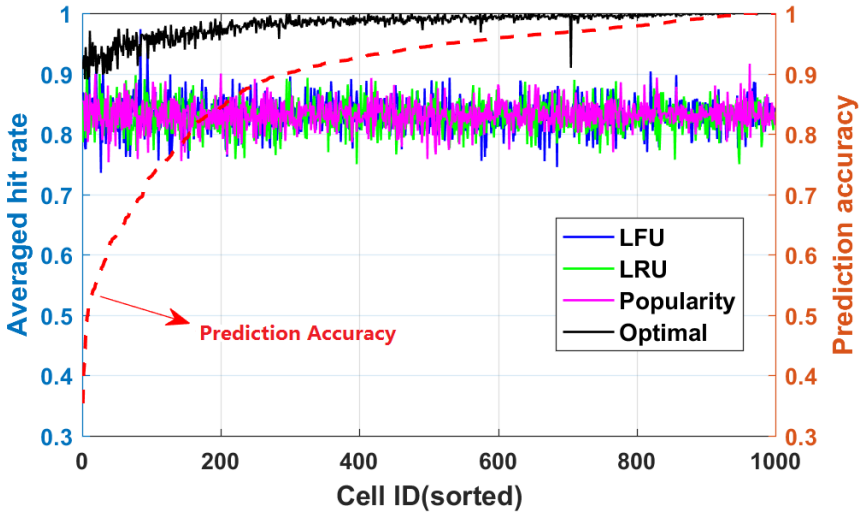


Figure 7.9: Averaged cache hit rate comparison among our method and three benchmarks for city A when $\alpha = 1.6$ and $s = 10$.

we assume that 20% of the files can be refreshed at the beginning of each time slot and the refresh rate is 5 minutes. Figure 7.9 shows the averaged cache hit rates of different methods. The selected 1,000 cells are sorted by the averaged prediction accuracy. Here we only consider the uncertainty of users' mobility, which implies that the request of each user at each time is supposed to be known.

The results in Figure 7.9 show that when the files are cached by popularity, LFU or LRU, the hit rates of different cells are similar, which are all around 0.82. The reason for the relatively low hit rates is that these methods cannot differentiate the predictability of different cells. By caching files using the optimal strategy in our proposed method, the hit rates improve substantially, especially for those cells with high prediction accuracies. The (average) hit rate of the optimal caching strategy found by our proposed method is 0.91.

7.7.4 Case Study - City B

In city B, there are 12,027 cells available in the data set. Here, the top 2,000 most visited cells and 1,015 users are taken into consideration. Figure 7.1 shows the prediction accuracy of each cell in the city. We use the same setting of parameters as in Section 7.7.3,

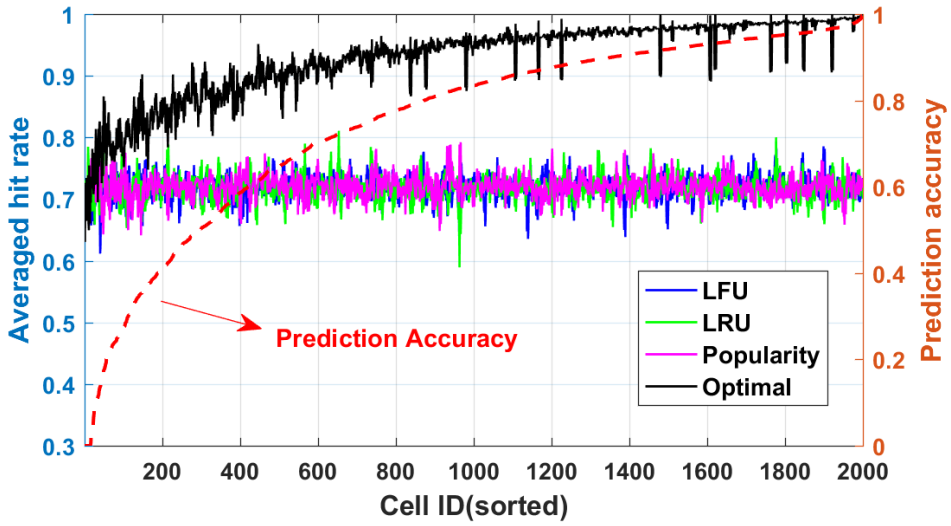


Figure 7.10: Averaged cache hit rate comparison among our method and three benchmarks for city B when $\alpha = 1.4$ and $s = 10$.

except that $\alpha = 1.4$ for city B. The average cache hit rates of our proposed method and the other three benchmark methods are obtained and compared in Figure 7.10.

The results of city B in Figure 7.10 are similar to the results of city A in Figure 7.9. In city B, our proposed method still outperforms the three benchmark methods. Compared with city A, there exists a decrease of the average cache hit rates of the three benchmark methods from 0.82 to 0.7 in city B, which can be primarily attributed to the lower α value. The accuracy of the optimal caching strategy by our proposed prediction model in city B is also lower than city A.

7.7.5 Effect of Varying Content Popularity Distribution and Varying Storage Capacity

We compare our proposed method with the three benchmark methods with varying storage sizes under four different α values ($\alpha = 0.2, 0.6, 1.0, 1.4$) for city A, as shown in Figure 7.11. We find that the hit rate turns higher as the storage size s increases, and the hit rate also increases with a larger value of α. The averaged hit rate of the optimal caching strategy obtained by our proposed method is higher than that of the other methods in all cases.

Figure 7.12 shows the average hit rates of our optimal method and the three other

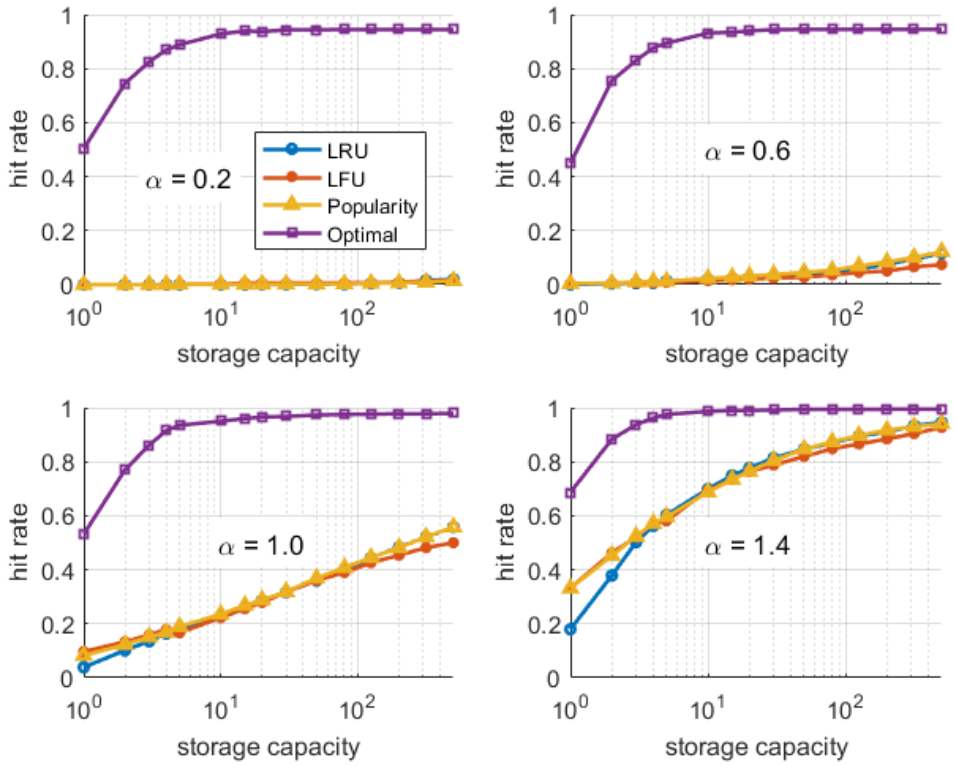


Figure 7.11: Averaged cache hit rates with varying storage limit and varying α value with perfect prediction on content requests.

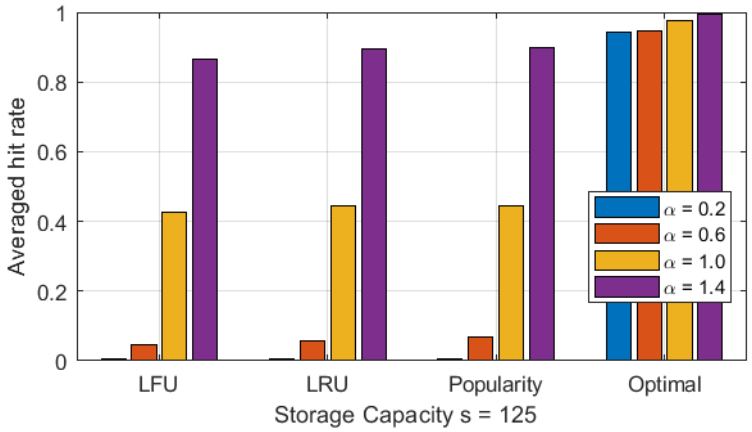


Figure 7.12: Averaged cache hit rates comparison with varying α value when $s = 125$.

benchmark methods when the cache storage capacity is $s = 125$. We can clearly see that our method is always the best among all these four methods; especially when α is small, the hit rate of our method is much higher than those of the other methods. This demonstrates again that our method can better deal with the heterogeneity of users.

If we assume that different cells have different prediction accuracies on the content prediction, then the hit rate of our algorithm will decrease slightly, since our algorithm can adaptively balance between popularity and prediction, as shown in Figure 7.13. In this experiment, we divide the cells randomly into 4 groups, and set the content request prediction accuracy as 0.3, 0.5, 0.7 and 0.9 respectively. Since the performance of LFU, LRU and PC is not affected by the prediction accuracy, in Figure 7.13, only the results of our optimal method are shown. By assuming there is noise in the prediction accuracy, the hit rate of our proposed method reduces slightly due to the decreased predictability of users.

7.8 Conclusion

In this chapter, we proposed an adaptive edge caching algorithm for mobile networks to improve users' experience and reduce cost. In our algorithm, mobile users' historical behavior in terms of mobility and content requests is applied to make predictions, and an optimization model is built based on it. In the optimization model, the optimal caching

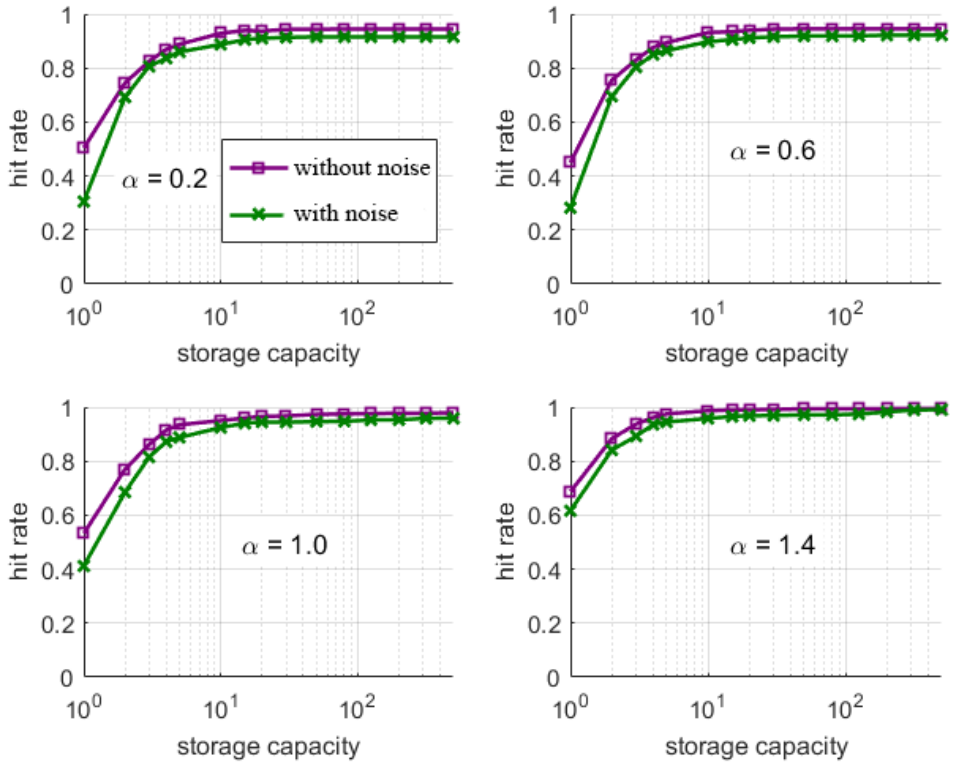


Figure 7.13: Averaged cache hit rates of our proposed method with varying storage limit and varying α value with/without perfect prediction on content requests.

strategy is obtained by adaptively dividing the cache storage into two parts, one based on popularity and the other based on prediction. What's more, a robust prediction model based on a second-order Markov model with improvement by iVAT clustering is proposed for mobile users' behavior prediction. Experiments on both simulation data and real-life data show that our algorithm outperforms three benchmarks, LFU, LRU and PC in terms of the averaged hit rate. The advantage of our model is more prominent when the storage capacity is limited and the similarities of mobile users are relatively low. The results demonstrate that our method is more suitable for resource-limited networks and more robust for the heterogeneity of mobile users' behavior.

Chapter 8
Conclusions and Future Work

8.1 Summary of Contributions

This thesis has addressed a range of challenges for knowledge discovery and data analytics on trajectory data in different application scenarios. Based on the rapid development of various kinds of location detection techniques, there is growing demand for the analysis of human movement behavior in the form of trajectories. Analyzing human trajectory data using different data mining techniques can assist a range of applications. In particular, we have focused on four research questions in three different application scenarios, namely pedestrian traffic flow anomaly detection, contrast pattern mining for multi-source data, as well as corridor identification and edge caching strategy design for mobile networks.

In Chapter 3, we focused on extracting pedestrian movement patterns and determining anomalous regions/time periods from trajectory datasets. Many existing approaches to address this problem have the limitation that they focus on the details of individual trajectories, but do not consider the characteristics of the overall trajectory distribution. Therefore, we utilized a visualization method to describe pedestrian movement distributions in terms of their origin and destination points. In order to summarize related flows of pedestrians, we transformed the origin-destination flow matrix into a dissimilarity matrix and visually clustered the most popular flows using the VAT/iVAT algorithms. This information can be useful for monitoring the safety and security of public areas. Then we proposed a clustering based method to detect abnormal time periods with anomalous pedestrian trajectory distributions, which can used to detect the occurrence and impact

of special events. We evaluated our proposed methods on one synthetic dataset and one real-life dataset, the Edinburgh informatics forum database, which demonstrated the effectiveness of our proposed algorithms.

In Chapter 4, we focused on finding patterns where there has been a major change in one type of dataset but little corresponding change in a related dataset. Our motivation is that in retail environments, store managers may be interested in finding the relationships or differences between sales data and customers' movement behavior. Therefore, we defined a new kind of contrast pattern, which is called a conditional contrast pattern. Although many different kinds of contrast patterns have been proposed in previous work, none of the methods has considered identifying contrast patterns from multiple datasets. We also proposed a conditional contrast pattern tree based method for mining these patterns. We evaluated our method compared with two baseline methods on a synthetic dataset as well as a real life retail dataset from a shop in Melbourne. The results show the efficiency and practicality of our proposed method.

In Chapters 5, 6 and 7, we focused on how to develop human mobility mining techniques for the deployment and management of large-scale mobile edge computing infrastructure in 5G networks. Specifically, in Chapter 5, we first studied the identification of underlying geographical corridors of trajectories generated in mobile networks. Compared with other types of trajectories, mobile trajectories are coarse, and their granularity varies due to the inconsistent density of cell towers. To solve this problem, we proposed a new distance measure based on Hausdorff distance to calculate the similarity between sub-trajectories by considering the distribution of cells. A two-level hierarchical trajectory clustering model was also proposed for the identification of corridors. Experiments on a real-life dataset from China Mobile demonstrate the effectiveness of our method. Our method can achieve the best performance with more than 10% improvement in clustering quality compared with state-of-the-art methods. To the best of our knowledge, this is the first published study that focuses on corridor identification of human trajectories generated from heterogeneous mobile data networks. This information could help mobile operators to identify key focus regions (i.e., corridors) in large-scale deployments of 5G networks for cost minimization.

In Chapter 6, to identify the significant changes in traffic flows in mobile networks, we utilized contrast mining techniques to highlight corridors that are sufficiently different between different time periods. To measure the difference, an improved distance measure based on a modified Hausdorff distance and earth movers' distance is proposed to calculate the dissimilarity between the identified corridors, which considers the heterogeneity of mobile networks. To further extract the significantly different corridors, we formulate the definition of contrast corridors of mobile users' movement. Both synthetic and real-life datasets are applied to validate the effectiveness of our proposed method. Contrast corridors can be effectively detected from trajectories in mobile networks, and our method outperforms others by an average 20% improvement in the F1 score. The identification of these contrast corridor patterns could provide insights to help the management and orchestration of 5G network resources.

Then in Chapter 7, we studied how to design edge caching strategies for mobile networks, i.e., determining what files to pre-fetch at which cell and at what time. Considering the heterogeneity of users' content preferences and mobility, we focused on the problem of how to utilize the historical data of users to improve the cache hit rate with limited storage size. We identified that cells located in different places have different degrees of predictability, which indicates that caching solely based on the prediction model may not be appropriate all the time. To overcome this problem, we propose to integrate a personalized prediction model into the edge caching problem, and also provide an optimization model that can adaptively decide the preferences for popularity and prediction results. We solved the optimization problem using a heuristic strategy based on genetic algorithms. Simulations are provided to evaluate the performance of our proposed algorithm. Experiments on a real-life dataset from China Mobile also show that our algorithm outperforms existing schemes, especially when the cache capacity is low and the distribution of content popularity is skewed.

8.2 Future Work

We now discuss some of the future research directions, which are motivated by this thesis or are open challenges for trajectory data mining.

Human Behavior Prediction in Mobile Networks Human behavior prediction (including next place and next request) is a powerful tool for data delivery and location services in mobile networks. For example, in the edge caching problem, the improvement of prediction accuracy can also help with a higher hit rate. In this thesis, we predict users' movements and content requests separately, but there may exist a correlation between these two behaviors. Therefore, a joint prediction model for users' movement and content request behavior may be more useful.

Utilization of Identified Patterns One open challenge for data mining is how to utilize the patterns identified from data to help with the final decision making tasks. For example, in Chapter 5, we proposed a method for corridor identification. It would be interesting to study how to utilize the corridors of mobile users identified from their historical trajectories. One possible application is to help with the prediction model.

Temporal Analysis Our proposed models for edge caching and corridor identification are built based on historical data, and all the time periods are taken into consideration as a whole, which means we assume users' behavior patterns are the same during different time periods. One potential research direction is to incorporate the temporal characteristics into our proposed model. In the future, we may further improve our proposed models by considering various temporal characteristics, e.g., the effects of different time periods of the day or different days of the week.

Real-time Data Processing Currently our proposed models are built based on historical data. All the data are processed offline. However, many applications, such as traffic monitoring and event management, require the results to be presented in real time or near real time. Depending on the deployment environment, mobile phone network data

may not be available in real time and the quantity data can be massive. Therefore, it is necessary to propose a streaming platform to process large volumes of data in real time.

Bibliography

[1] A. Afiq, M. Zakariya, M. Saad, A. Nurfarzana, M. Khir, A. Fadzil, A. Jale, W. Gunawan, Z. Izuddin, and M. Faizari, "A review on classifying abnormal behavior in crowd scene," Journal of Visual Communication and Image Representation, vol. 58, pp. 285–303, 2019.

[2] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic, "Discovering clusters in motion time-series data," in Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'03), vol. 1, 2003, pp. I–I.

[3] H. Alt and M. Godau, "Computing the fréchet distance between two polygonal curves," International Journal of Computational Geometry & Applications, vol. 5, pp. 75–91, 1995.

[4] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," SIGMOD Rec., vol. 28, no. 2, pp. 49–60, 1999.

[5] G. Atluri, A. Karpatne, and V. Kumar, "Spatio-temporal data mining: A survey of problems and methods," ACM Computing Surveys, vol. 51, no. 4, p. 83, 2018.

[6] D. Bachir, G. Khodabandelou, V. Gauthier, M. El Yacoubi, and E. Vachon, "Combining bayesian inference and clustering for transport mode detection from sparse and noisy geolocation data," in Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'18), 2018, pp. 569–584.

[7] D. Bachir, G. Khodabandelou, V. Gauthier, M. E. Yacoubi, and J. Puchinger, "Inferring dynamic origin-destination flows by transport mode using mobile phone data," Transportation Research Part C: Emerging Technologies, vol. 101, pp. 254–275, 2019.

[8] T. Bäck, Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. New York, USA: Oxford University Press, Inc., 1996.

[9] S. D. Bay and M. J. Pazzani, "Detecting change in categorical data: Mining contrast sets," in Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99), 1999, pp. 302–306.

[10] I. Ben-Gal, S. Weinstock, G. Singer, and N. Bambos, "Clustering users by their mobility behavioral patterns," ACM Transaction on Knowledge Discovery from Data, vol. 13, no. 4, pp. 45:1–45:28, 2019.

[11] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'94), 1994, pp. 359–370.

[12] P. C. Besse, B. Guillouet, J. Loubes, and F. Royer, "Review and perspective for distance-based clustering of vehicle trajectories," IEEE Transactions on Intelligent Transportation Systems, vol. 17, pp. 3306–3317, 2016.

[13] J. C. Bezdek and R. J. Hathaway, "Vat: a tool for visual assessment of (cluster) tendency," in Proc. International Joint Conference on Neural Networks (IJCNN'02), 2002, pp. 2225–2230.

[14] Z. Boukhers, Y. Wang, K. Shirahama, K. Uehara, and M. Grzegorzek, "Convoy detection in crowded surveillance videos," in Proc. International Workshop on Human Behavior Understanding, 2016, pp. 137–147.

[15] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in ACM sigmod record, vol. 29, no. 2, 2000, pp. 93–104.

[16] F. Calabrese, G. Di Lorenzo, L. Liu, and C. Ratti, "Estimating origin-destination flows using mobile phone location data," IEEE Pervasive Computing, vol. 10, no. 4, pp. 36–44, 2011.

[17] F. Calabrese, L. Ferrari, and V. D. Blondel, "Urban sensing using mobile phone network data: A survey of research," ACM Computing Surveys, vol. 47, no. 2, pp. 25:1–25:20, 2014.

[18] H. Cao, N. Mamoulis, and D. W. Cheung, "Discovery of periodic patterns in spatiotemporal sequences," IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 4, pp. 453–467, 2007.

[19] ——, "Mining frequent spatio-temporal sequential patterns," in Proc. IEEE International Conference on Data Mining (ICDM'05), 2005, p. 8.

[20] L. Cao and M. Grabchak, "Smoothly truncated levy walks: Toward a realistic mobility model," in Proc. IEEE International Performance Computing and Communications Conference (IPCCC'14), 2014, pp. 1–8.

[21] U. Celikkan, G. Somun, U. Kutuk, I. Gamzeli, E. D. Cinar, and I. Atici, "Capturing supermarket shopper behavior using smartbasket," in Proc. International Conference on Digital Information Processing and Communications, 2011, pp. 44–53.

[22] S. Chawla, Y. Zheng, and J. Hu, "Inferring the root cause in road traffic anomalies," in Proc. IEEE International Conference on Data Mining (ICDM'12). IEEE Computer Society, 2012, pp. 141–150.

[23] L. Chen and R. Ng, "On the marriage of lp-norms and edit distance," in Proc. International Conference on Very Large Data Bases (VLDB'04), 2004, pp. 792–803.

[24] X. Chen and L. Lu, "A new algorithm based on shared pattern-tree to mine shared emerging patterns," in Proc. IEEE International Conference on Data Mining Workshops (ICDMW'11), 2011, pp. 1136–1140.

[25] X. Chong, W. Liu, P. Huang, and N. I. Badler, "Hierarchical crowd analysis and anomaly detection," J. Vis. Lang. Comput., vol. 25, no. 4, pp. 376–393, 2014.

[26] Cisco, "Cisco visual networking index: Forecast and trends," 2019.

[27] Y. Cui and S. S. Ge, "Autonomous vehicle positioning with GPS in urban canyon environments," IEEE Transactions on Robotics and Automation, vol. 19, no. 1, pp. 15–25, 2003.

[28] K. Deb, "An efficient constraint handling method for genetic algorithms," Computer Methods in Applied Mechanics and Engineering, vol. 186, pp. 311–338, 2000.

[29] K. Deep, K. P. Singh, M. L. Kansal, and C. Mohan, "A real coded genetic algorithm for solving integer and mixed integer optimization problems," Applied Mathematics and Computation, vol. 212, pp. 505–518, 2009.

[30] G. Dong and J. Bailey, Contrast data mining: Concepts, algorithms, and applications, 1st ed. Chapman & Hall/CRC, 2012.

[31] G. Dong and J. Li, "Efficient mining of emerging patterns: Discovering trends and differences," in Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99), 1999, pp. 43–52.

[32] X. Dong, "Predicting LTE throughput using traffic time series," ZTE Communications, 2015.

[33] J. C. Dunn, "A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters," Journal of Cybernetics, vol. 3, pp. 32–57, 1973.

[34] T. Eiter and H. Mannila, "Computing discrete fréchet distance," Technical University of Vienna, Tech. Rep., 1994.

[35] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in Proc. International Conference on Knowledge Discovery and Data Mining (KDD'96), 1996, pp. 226–231.

[36] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases," in Proc. ACM International Conference on Management of Data (SIGMOD'94), 1994, pp. 419–429.

[37] H. Fan and Kotagiri Ramamohanarao, "Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers," IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 6, pp. 721–737, 2006.

[38] Z. Feng and Y. Zhu, "A survey on trajectory data mining: Techniques and applications," IEEE Access, vol. 4, pp. 2056–2067, 2016.

[39] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," IEEE Communications Magazine, pp. 94–100, 2017.

[40] S. Gaffney and P. Smyth, "Trajectory clustering with mixtures of regression models," in Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99), vol. 99, 1999, pp. 63–72.

[41] C. Gao, L. Duan, G. Dong, H. Zhang, H. Yang, and C. Tang, "Mining top-k distinguishing sequential patterns with flexible gap constraints," in Proc. International Conference on Web-Age Information Management. Springer, 2016, pp. 82–94.

[42] M. García-Borroto, J. F. Martínez-Trinidad, and J. A. Carrasco-Ochoa, "Fuzzy emerging patterns for classifying hard domains," Knowledge and Information Systems, vol. 28, no. 2, pp. 473–489, 2011.

[43] A. García-Vico, C. J. Carmona, D. Martín, M. García-Borroto, and M. del Jesus, "An overview of emerging pattern mining in supervised descriptive rule discovery: taxonomy, empirical study, trends, and prospects," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2018.

[44] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory pattern mining," in Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'07), 2007, pp. 330–339.

[45] J. Gudmundsson, A. Thom, and J. Vahrenhold, "Of motifs and goals: Mining trajectory data," in Proc. International Conference on Advances in Geographic Information Systems (SIGSPATIAL'12), 2012, pp. 129–138.

[46] J. Gudmundsson and M. van Kreveld, "Computing longest duration flocks in trajectory data," in Proc. ACM International Symposium on Advances in Geographic Information Systems. ACM, 2006, pp. 35–42.

[47] T. D. Gwiazda, Crossover for single-objective numerical optimization problems, 2006, vol. 1.

[48] J. Han, J. Pei, and M. Kamber, Data mining: concepts and techniques. Elsevier, 2011.

[49] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," Data Mining and Knowledge Discovery, vol. 8, no. 1, pp. 53–87, 2004.

[50] F. Hausdorff, Grundzüge der mengenlehre. American Mathematical Soc., 1978, vol. 61.

[51] J. H. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press, 1992.

[52] J. Hu and N. Zhong, "Organizing multiple data sources for developing intelligent e-business portals," Data Mining and Knowledge Discovery, vol. 12, no. 2, pp. 127–150, 2006.

[53] S. K. Hui, P. S. Fader, and E. T. Bradlow, "Research note - the traveling salesman goes shopping: The systematic deviations of grocery paths from TSP optimality," Marketing Science, vol. 28, no. 3, pp. 566–572, 2009.

[54] H. Hwangbo, J. Kim, Z. Lee, and S. Kim, "Store layout optimization using indoor positioning system," International Journal of Distributed Sensor Networks, vol. 13, no. 2, pp. 1–13, 2017.

[55] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen, "Discovery of convoys in trajectory databases," Proceedings of the VLDB Endowment, vol. 1, no. 1, pp. 1068–1080, 2008.

[56] S. Jiang, J. Ferreira, and M. C. Gonzalez, "Activity-based human mobility patterns inferred from mobile phone data: A case study of singapore," IEEE Transactions on Big Data, vol. 3, no. 2, pp. 208–219, 2017.

[57] M. Kholod, K. Takai, and K. Yada, "Clockwise and anti-clockwise directions of customer orientation in a supermarket: Evidence from rfid data," in Proc. International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, 2011, pp. 304–309.

[58] K. Laasonen, "Clustering and prediction of mobile user routes from cellular data," in Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD'05), 2005, pp. 569–576.

[59] J. S. Larson, E. T. Bradlow, and P. S. Fader, "An exploratory look at supermarket shopping paths," International Journal of Research in Marketing, vol. 22, no. 4, pp. 395–414, 2005.

[60] J. Lee and J. Han, "Trajectory clustering: A partition-and-group framework," in Proc. ACM Special Interest Group on Management of Data (SIGMOD'07), 2007, pp. 593–604.

[61] J. Lee, J. Han, and X. Li, "Trajectory outlier detection: A partition-and-detect framework," in Proc. IEEE International Conference on Data Engineering (ICDE'08), 2008, pp. 140–149.

[62] J.-G. Lee, J. Han, X. Li, and H. Cheng, "Mining discriminative patterns for classifying trajectories on road networks," IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 5, pp. 713–726, 2011.

[63] G. Li, C.-J. Chen, S.-Y. Huang, A.-J. Chou, X. Gou, W.-C. Peng, and C.-W. Yi, "Public transportation mode detection from cellular data," in Proc. ACM International

Conference on Information and Knowledge Management (CIKM'17), 2017, pp. 2499–2502.

[64] J. Li, K. Ramamohanarao, and G. Dong, "The space of jumping emerging patterns and its incremental maintenance algorithms," in Proc. International Conference on Machine Learning (ICML'00). Morgan Kaufmann Publishers Inc., 2000, pp. 551–558.

[65] S. Li, J. Xu, M. van der Schaar, and W. Li, "Popularity-driven content caching," in Proc. IEEE International Conference on Computer Communications (INFOCOM'16), 2016, pp. 1–9.

[66] X. Li, W. Hu, and W. Hu, "A coarse-to-fine strategy for vehicle motion trajectory clustering," in Proc. International Conference on Pattern Recognition (ICPR'06), vol. 1, 2006, pp. 591–594.

[67] X. Li, K. Zhao, G. Cong, C. S. Jensen, and W. Wei, "Deep representation learning for trajectory similarity computation," in Proc. IEEE International Conference on Data Engineering (ICDE'18), 2018, pp. 617–628.

[68] Z. Li, B. Ding, J. Han, and R. Kays, "Swarm: Mining relaxed temporal moving object clusters," Proceedings of the VLDB Endowment, vol. 3, no. 1-2, pp. 723–734, 2010.

[69] M. Lin and W.-J. Hsu, "Mining GPS data for mobility patterns: A survey," Pervasive and Mobile Computing, vol. 12, pp. 1–16, 2014.

[70] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: design aspects, challenges, and future directions," IEEE Communications Magazine, vol. 54, pp. 22–28, 2016.

[71] D. Liu, C. Yang, and V. C. M. Leung, "When exploiting individual user preference is beneficial for caching at base stations," in Proc. IEEE International Conference on Communications Workshops (ICC Workshops'18), 2018, pp. 1–6.

[72] L. Liu, A. Biderman, and C. Ratti, "Urban mobility landscape: Real time monitoring of urban mobility patterns," in Proc. International Conference on Computers in Urban Planning and Urban Management. Citeseer, 2009, pp. 1–16.

[73] W. Liu, X. Chong, P. Huang, and N. I. Badler, "Learning motion patterns in unstructured scene based on latent structural information," Journal of Visual Languages & Computing, vol. 25, no. 1, pp. 43–53, 2014.

[74] E. Loekito and J. Bailey, "Fast mining of high dimensional expressive contrast patterns using zero-suppressed binary decision diagrams," in Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06), 2006, pp. 307–316.

[75] E. H. Lu, V. S. Tseng, and P. S. Yu, "Mining cluster-based temporal mobile sequential patterns in location-based service environments," IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 6, pp. 914–927, 2011.

[76] X. Lu, C. Wang, N. Karamzadeh, A. Croitoru, and A. Stefanidis, "Deriving implicit indoor scene structure with path analysis," in Proc. ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness. ACM, 2011, pp. 43–50.

[77] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," IEEE Journal on Selected Areas in Communications, vol. 35, pp. 1076–1089, 2017.

[78] B. Majecka, "Statistical models of pedestrian behaviour in the forum," Master's thesis, University of Edinburgh, 2009.

[79] H. Mao, Y.-Y. Ahn, B. Bhaduri, and G. Thakur, "Improving land use inference by factorizing mobile phone call activity matrix," Journal of Land Use Science, vol. 12, no. 2-3, pp. 138–153, 2017.

[80] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "Mobile edge computing: Survey and research outlook," CoRR, pp. 1–30, 2017.

[81] R. Mijumbi, J. Serrat, J. L. Gorricho, S. Latré, M. Charalambides, and D. Lopez, "Management and orchestration challenges in network functions virtualization," IEEE Communications Magazine, pp. 98–105, 2016.

[82] M. Nanni and D. Pedreschi, "Time-focused clustering of trajectories of moving objects," Journal of Intelligent Information Systems, vol. 27, no. 3, pp. 267–289, 2006.

[83] D. Oosterlinck, D. F. Benoit, P. Baecke, and N. V. de Weghe, "Bluetooth tracking of humans in an indoor environment: An application to shopping mall visits," Applied Geography, vol. 78, pp. 55–65, 2017.

[84] L. X. Pang, S. Chawla, W. Liu, and Y. Zheng, "On detection of emerging anomalous traffic patterns using GPS data," Data and Knowledge Engineering, vol. 87, pp. 357–373, 2013.

[85] T. Pei, S. Sobolevsky, C. Ratti, S.-L. Shaw, T. Li, and C. Zhou, "A new insight into land use classification based on aggregated mobile phone data," International Journal of Geographical Information Science, vol. 28, no. 9, pp. 1988–2007, 2014.

[86] R. Pierdicca, D. Liciotti, M. Contigiani, E. Frontoni, A. Mancini, and P. Zingaretti, "Low cost embedded system for increasing retail environment intelligence," in Proc. IEEE International Conference on Multimedia & Expo Workshops (ICMEW'15). IEEE, 2015, pp. 1–6.

[87] K. Poularakis and L. Tassiulas, "Cooperation and information replication in wireless networks," Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 374, no. 2062, p. 20150123, 2016.

[88] L. Qi, Y. Qiao, F. B. Abdesslem, Z. Ma, and J. Yang, "Oscillation resolution for massive cell phone traffic data," in Proc. the First Workshop on Mobile Data. ACM, 2016, pp. 25–30.

[89] Y. Qiao, Y. Cheng, J. Yang, J. Liu, and N. Kato, "A mobility analytical framework for big mobile data in densely populated area," IEEE Transactions on Vehicular Technology, vol. 66, pp. 1443–1455, 2017.

[90] Y. Qiao, Z. Si, Y. Zhang, F. B. Abdesslem, X. Zhang, and J. Yang, "A hybrid markov-based model for human mobility prediction," Neurocomputing, vol. 278, pp. 99–109, 2018.

[91] L. Qiu and G. Cao, "Popularity-aware caching increases the capacity of wireless networks," in Proc. IEEE Conference on Computer Communications (INFOCOM'17), 2017, pp. 1–9.

[92] T. Ramkumar, S. Hariharan, and S. Selvamuthukumaran, "A survey on mining multiple data sources," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 3, no. 1, pp. 1–11, 2013.

[93] S. Rashidi and S. Sharifian, "A hybrid heuristic queue based algorithm for task assignment in mobile cloud," Future Generation Computer Systems, vol. 68, pp. 331–345, 2017.

[94] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in Proc. IEEE International Conference on Computer Vision (ICCV'98), 1998, pp. 59–66.

[95] A. Sawas, A. Abuolaim, M. Afifi, and M. Papagelis, "Tensor methods for group pattern discovery of pedestrian trajectories," in Proc. IEEE International Conference on Mobile Data Management (MDM'18), 2018, pp. 76–85.

[96] S. A. Shad and E. Chen, "Cell oscillation resolution in mobility profile building," ArXiv, vol. abs/1206.5795, 2012.

[97] B. Shen, Q. Zheng, X. Li, and L. Xu, "A framework for mining actionable navigation patterns from in-store rfid datasets via indoor mapping," Sensors, vol. 15, no. 3, pp. 5344–5375, 2015.

[98] A. Sieben, J. Schumann, and A. Seyfried, "Collective phenomena in crowds—where pedestrian dynamics need social psychology," PLOS ONE, vol. 12, no. 6, pp. 1–19, 2017.

[99] P. Skogster, V. Uotila, and L. Ojala, "From mornings to evenings: is there variation in shopping behaviour between different hours of the day?" International Journal of Consumer Studies, vol. 32, no. 1, pp. 65–74, 2008.

[100] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," Science, vol. 327, pp. 1018–1021, 2010.

[101] L. Sun, W. Luo, L. Yao, S. Qiu, and J. Rong, "A comparative study of funnel shape bottlenecks in subway stations," Transportation Research Part A: Policy and Practice, vol. 98, pp. 14 – 27, 2017.

[102] K. Takai and K. Yada, "Relation between stay-time and purchase probability based on rfid data in a japanese supermarket," in Proc. International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, 2010, pp. 254–263.

[103] P.-N. Tan, Introduction to data mining. Pearson Education India, 2018.

[104] S. Theodoridis and K. Koutroumbas, Pattern Recognition, Fourth Edition, 2008.

[105] R. Trasarti, R. Guidotti, A. Monreale, and F. Giannotti, "Myway: Location prediction via mobility profiling," Information Systems, vol. 64, pp. 350–367, 2017.

[106] C.-Y. Tsai, M.-H. Li, and R.-J. Kuo, "A shopping behavior prediction system: Considering moving patterns and product characteristics," Computers & Industrial Engineering, vol. 106, pp. 192–204, 2017.

[107] C. Tuduce and T. Gross, "A mobility model based on wlan traces and its validation," in Proc. Annual Joint Conference of the IEEE Computer and Communications Societies., vol. 1, 2005, pp. 664–674 vol.1.

[108] S. Uppoor, "Understanding and exploiting mobility in wireless networks," Ph.D. dissertation, Lyon, INSA, 2013.

[109] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in Proc. IEEE International Conference on Data Engineering (ICDE'02), 2002, pp. 673–684.

[110] J. Wang, J. Tang, Z. Xu, Y. Wang, G. Xue, X. Zhang, and D. Yang, "Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach," in Proc. IEEE Conference on Computer Communications (CCC'17), 2017, pp. 1–9.

[111] L. Wang, U. T. V. Nguyen, J. C. Bezdek, C. A. Leckie, and R. Kotagiri, "ivat and avat: Enhanced visual analysis for cluster tendency assessment," in Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'10), 2010, pp. 16–27.

[112] R. Wang, X. Peng, J. Zhang, and K. B. Letaief, "Mobility-aware caching for content-centric wireless networks: modeling and methodology," IEEE Communications Magazine, vol. 54, pp. 77–83, 2016.

[113] X. Wang, Z. Zhou, Z. Yang, Y. Liu, and C. Peng, "Spatio-temporal analysis and prediction of cellular traffic in metropolis," in Proc. International Conference on Network Protocols (ICNP'17), 2017, pp. 1–10.

[114] X. Wang, C. Leckie, H. Xie, and T. Vaithianathan, "Discovering the impact of urban traffic interventions using contrast mining on vehicle trajectory data," in Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'15), 2015, pp. 486–497.

[115] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14), 2014, pp. 25–34.

[116] A. Witayangkurn, T. Horanont, Y. Sekimoto, and R. Shibasaki, "Anomalous event detection on large-scale GPS data from mobile phones using hidden markov model and cloud platform," in Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'13), 2013, p. 1219.

[117] R. Wu, G. Luo, J. Shao, L. Tian, and C. Peng, "Location prediction on trajectory data: A review," Big Data Mining and Analytics, vol. 1, no. 2, pp. 108–127, 2018.

[118] W. Wu, Y. Wang, J. B. Gomes, D. T. Anh, S. Antonatos, M. Xue, P. Yang, G. E. Yap, X. Li, S. Krishnaswamy, J. Decraene, and A. S. Nash, "Oscillation resolution for mobile phone cellular tower data to enable mobility modelling," in Proc. IEEE International Conference on Mobile Data Management (MDM'14), 2014, pp. 321–328.

[119] X. Xiao, Y. Zheng, Q. Luo, and X. Xie, "Finding similar users using category-based location history," in Proc. International Conference on Advances in Geographic Information Systems, 2010, pp. 442–445.

[120] F. Xu, Y. Lin, J. Huang, D. Wu, H. Shi, J. Song, and Y. Li, "Big data driven mobile traffic understanding and forecasting: A time series approach," IEEE Transactions on Services Computing, vol. 9, no. 5, pp. 796–805, 2016.

[121] A. Yaeli, P. Bak, G. Feigenblat, S. Nadler, H. Roitman, G. Saadoun, H. J. Ship, D. Cohen, O. Fuchs, S. Ofek-Koifman, and T. Sandbank, "Understanding customer behavior using indoor location analysis and visualization," IBM Journal of Research and Development, vol. 58, no. 5/6, pp. 3:1–3:12, 2014.

[122] P. Yang, N. Zhang, S. Zhang, L. Yu, J. Zhang, and X. Shen, "Dynamic mobile edge caching with location differentiation," in Proc. IEEE Global Communications Conference (GLOBECOM'17), 2017, pp. 1–6.

[123] Y. Ye, Y. Zheng, Y. Chen, J. Feng, and X. Xie, "Mining individual life pattern based on location history," in Proc. International Conference on Mobile Data Management: Systems, Services and Middleware, 2009, pp. 1–10.

[124] S. Yi, H. Li, and X. Wang, "Understanding pedestrian behaviors from stationary crowd groups," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15), 2015, pp. 3488–3496.

[125] G. Yuan, P. Sun, J. Zhao, D. Li, and C. Wang, "A review of moving object trajectory clustering algorithms," Artificial Intelligence Review, vol. 47, pp. 123–144, 2017.

[126] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," IEEE Communications Surveys Tutorials, pp. 1–1, 2019.

[127] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing. ACM, 2018, pp. 231–240.

[128] J. Zhang, "Multi-source remote sensing data fusion: status and trends," International Journal of Image and Data Fusion, vol. 1, no. 1, pp. 5–24, 2010.

[129] N. Zhang, K. Zheng, and M. Tao, "Using grouped linear prediction and accelerated reinforcement learning for online content caching," in Proc. IEEE International Conference on Communications Workshops (ICC Workshops'18), 2018, pp. 1–6.

[130] S. Zhang, P. He, K. Suto, P. Yang, L. Zhao, and X. Shen, "Cooperative edge caching in user-centric clustered mobile networks," IEEE Transactions on Mobile Computing, vol. 17, pp. 1791–1805, 2018.

[131] S. Zhang, N. Zhang, P. Yang, and X. Shen, "Cost-effective cache deployment in mobile heterogeneous networks," IEEE Transactions on Vehicular Technology, vol. 66, pp. 11 264–11 276, 2017.

[132] X. Zhang, C. Wang, Z. Li, J. Zhu, W. Shi, and Q. Wang, "Exploring the sequential usage patterns of mobile internet services based on markov models," Electronic Commerce Research and Applications, vol. 17, pp. 1–11, 2016.

[133] Y. Zhang, "User mobility from the view of cellular data networks," in Proc. IEEE International Conference on Computer Communications (INFOCOM'14), 2014, pp. 1348–1356.

[134] Y. Zheng, "Trajectory data mining: an overview," ACM Transactions on Intelligent Systems and Technology (TIST), vol. 6, no. 3, p. 29, 2015.

[135] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: Concepts, methodologies, and applications," ACM Transactions on Intelligent Systems and Technology, vol. 5, no. 3, pp. 38:1–38:55, 2014.

[136] H. Zhu, J. Luo, H. Yin, X. Zhou, J. Z. Huang, and F. Zhan, "Mining trajectory corridors using fréchet distance and meshing grids," in Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'10), 2010, pp. 228–237.

[137] N. Zygouras and D. Gunopulos, "Discovering corridors from GPS trajectories," in Proc. International Conference on Advances in Geographic Information Systems (SIGSPATIAL'17), 2017, pp. 61:1–61:4.

[138] ——, "Corridor learning using individual trajectories," in Proc. IEEE International Conference on Mobile Data Management (MDM'18), 2018, pp. 155–160.