2014-09-23

Location Recommendation on Location Based Social Networks Utilizing Check-in Data and Location Category

Zhou, Dequan

UNIVERSITY OF CALGARY

Location Recommendation on Location Based Social Networks Utilizing Check-in Data and Location Category

by

Dequan Zhou

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF GEOMATICS ENGINEERING

CALGARY, ALBERTA

September 2014

Abstract

Location-based social networks (LBSNs) provide a platform for users to share their location information with each other. Location recommendation is the task of suggesting unvisited locations to the users. It aims to make satisfying recommendations of locations by utilizing the information such as users' visiting histories, user profiles and location profiles.

This thesis investigates the utilization of check-in data and location category information for location recommendation on LBSNs. A distributed crawler is developed to collect a large amount of check-in data from Gowalla for the research. Then, three ways are used to utilize the check-in data, namely, binary utilization, *FIF* utilization, and probability utilization. According to different utilizations, different Collaborative Filtering recommenders are introduced to do location recommendation. Experiments are conducted to compare the performances of different recommenders using different check-in utilizations. Location category information is utilized for location recommendation by considering the temporal and spatial patterns. A user's periodic check-in behaviors at different location categories are represented as temporal curves. A temporal influence model is used to predict similar users' check-in behaviors based on temporal curves. A geographical influence model is proposed to filter out locations that are not of interest to the user. By integrating temporal influence and geographical influence a location recommendation algorithm called *sPCLR* is proposed to recommend locations to the users at a given time of the day. Experimental results show that the *sPCLR* algorithm performs better than three existing location recommendation algorithms.

Acknowledgements

I want to sincerely thank my supervisor, Dr. Xin Wang for her consistent support and guidance during the research. Thanks to Mr. Gary Zhang from MRF Geosystems Corporation who helped me apply for the NSERC IPS scholarship. I would also like to thank Dr. Bin Wang for his kind help in the field of location recommendation. My colleague Mr. Seyyed Mohammadreza Rahimi helped me a lot during my research.

I am very grateful for my family to support me during my master's study. Thanks for their unfailing love, encouragement and patience.

Table of Contents

List of Figures and Illustrations

List of Tables

List of Symbols, Abbreviations and Nomenclature

Symbol	Definition
OSN	Online Social Network
LBSN	Location-based Social Network
POI	Point-of-Interest
CF	Collaborative Filtering
CBF	Content-based Filtering
PCLR	Probabilistic Category-based Location Recommender
PLSA	Probabilistic Latent Semantic Analysis
PMM	Periodic Mobility Model
I-CF	Item-based CF
U-CF	User-based CF

Chapter One: **Introduction**

1.1 Background

With the rapid development of online social networks (OSNs), it is very convenient for people to share posts or pictures with their friends. People tend to share more and know more about recent status of their friends. Currently, some location-based social network (LBSN) services, such as Foursquare, have emerged. LBSN services allow users to record their visiting histories at certain locations by check-ins and make it much easier for them to attach geographical contexts. Users of LBSN services explore the cities and neighbourhoods and share tips and experiences of their visits to various locations, e.g., restaurants, coffee shops, tourist attractions, etc. The research on making recommendations of new locations for the users on LBSN services receives much attention in recent years.

1.1.1 Recommender Systems

Recommender systems can make predictions or recommendations of items to users based on information gathered from multiple sources. They collect information about the users and items, and the interactions between them. Then the systems analyze the patterns and preferences of the users towards items and make recommendations accordingly. Generally, there are three types of strategies for recommender systems: content-based filtering (CBF), collaborative filtering (CF) and hybrid recommender systems. The recommender systems of content-based filtering make recommendations by analyzing the content of textual information, such as item (e.g. locations) descriptions and users'

profiles. Content based filtering methods usually are highly dependent on the domain knowledge of the features explicitly associated with the objects. The recommender systems of collaborative filtering make recommendations by analyzing users' behaviors and activities instead of textual information. They predict a user's preference towards an item based on the user's similarities to other users. Thus, CF methods do not depend on the domain knowledge. Hybrid recommender systems make recommendations using a combination of CBF and CF methods.

1.1.2 Location Recommendation on LBSNs

A location is a specific geographical point that a user may find useful or interesting. In this thesis, locations are Points-of-Interest (POI). Location recommendation is similar to regular recommendation. Regular recommendation makes recommendation of items to users. Location recommendation provides users with suggestions of unvisited locations based on some gathered information. The information can be users' visiting histories, user profiles and location profiles. The methods of conventional recommender systems can be applied to location recommendation by considering locations as items. Content-based filtering methods analyze the user profiles and location profiles to make location recommendation. Collaborative filtering methods recommend locations by analyzing users' visiting behaviors at different locations and their similarities in terms of preferences towards locations. The existing methods mainly focus on utilizing the user's visiting histories such as GPS trajectories or check-in histories to do location recommendation. The utilization of additional information such as social ties and location category information receives a lot of attention in recent research on location recommendation.

1.2 Research Gap and Problem Statement

This research focuses on the development of new location recommendation methods for location-based social networks.

1.2.1 Research Gap

Although some researchers have proposed methods that utilize the check-in data and category information in location recommendation, the main gap in the research is the lack of methods that consider the similarity between users' periodic check-in behaviors. To solve this problem, a good recommendation method should utilize the check-in data in an effective way and study users' temporal patterns for visiting location categories. The temporal patterns should be extracted to represent users' periodic behaviors. Finally, location recommendations are made based on the temporal patterns of similar users.

1.2.2 Research Objectives

The research objectives are summarized as:

1- To obtain a real-world check-in dataset that contains both users' check-in histories and location category information for the research.

2- To study the different utilizations of users' check-in histories in location recommendation methods.

3- To utilize the location category information to represent users' periodic check-in behaviors and measure their similarities.

4- To develop a new location recommendation method that effectively uses the location category information by considering both temporal and spatial patterns as well as the users' check-in histories.

5- To test the proposed method on a real-world check-in dataset and compare the precision and recall with existing location recommendation methods.

1.3 Research Contributions

The main contributions of this thesis can be summarized as:

1- A distributed data crawler is developed to acquire a large amount of real-world check-in data from Gowalla, one of the most popular LBSN services in recent years. Based on this dataset, empirical experiments are conducted to compare the performances of different location recommendation methods.

2- The different utilizations of check-in data for the location recommendation on LBSN are investigated. Particularly, three different kinds of utilizations are considered to represent check-in behaviors: the binary utilization, the *FIF* (Frequency - Inverse Frequency) utilization, and the probability utilization. It is observed that the probability utilization has the best performance.

3- Temporal patterns are extracted from the location category information and check-in data. Users' periodic check-in behaviors for different categories are represented by temporal curves. Then a coupling method is proposed to measure the difference between two temporal curves, which are further used to calculate the temporal similarity between two users in terms of periodic check-in behaviors. Based on the temporal similarity, a temporal influence model is proposed to predict the periodic check-in behaviors for a given user.

4- A new location recommendation algorithm called *sPCLR* is developed that combines the temporal influence of similar users and geographical influence of

locations. The temporal influence model makes prediction of the user's periodic check-in behaviors at different categories using a collaborative filtering approach. The geographical influence model takes account of the user's home location and measures the check-in probability of a certain location.

5- A set of experiments is conducted on the Gowalla check-in dataset to measure the performance of the *sPCLR* location recommendation algorithm. According to the experimental results, the *sPCLR* algorithm performs better than three existing location recommendation algorithms, namely PCLR, PMM and USG.

1.4 Thesis Outline

Chapter Two gives a literature review of the methods for recommender systems and location recommendation. Chapter Three describes the study of utilizations of check-in data in location recommendation. Chapter Four extends the work of chapter three by studying the utilizations of location category information and considering the temporal and spatial patterns. A location recommendation algorithm called Probabilistic Category-based Location Recommender Utilizing Temporal Influence and Geographical Influence (*sPCLR*) is proposed. Chapter Five draws conclusions and states future work of this thesis.

Chapter Two: **Related Works**

This chapter presents the literature review in the following areas: recommender systems and their classifications, recent research on location recommendation. Moreover, some of the widely used location recommendation methods are discussed.

2.1 Recommender Systems

Recommender systems get information from different sources of information and use that information to provide users with predictions and recommendation of items (Bobadilla et al., 2013). Recommender systems can be classified into three types: Content-based filtering (CBF), collaborative filtering (CF), and hybrid recommender systems. CBF methods consider domain knowledge such as the director in a movie recommendation, while CF techniques use the similarities between users or items based on rating data. Hybrid recommender systems use a combination of CBF and CF techniques to make recommendations. In the following subsections the CBF and CF techniques are discussed in more detail.

2.1.1 Content-Based Filtering

Content-based filtering techniques make recommendations of items by considering the content information. It analyzes the content of the objects intended for the recommendation. For example, when recommending movies, the content could be the title, genre, or director of the movie. Content-based filtering utilizes a series of discrete characteristics of an item to recommend additional items that have similar properties.

There are two challenging problems for Content-based filtering, 1) limited content analysis, and, 2) overspecialization (Adomavicius and Tuzhilin, 2005). Limited content analysis is due to the difficulty of collecting reliable information automatically from different sources. Since CBF creates a profile for each user or item to characterize its nature. It requires gathering external information that might not be available or easy to obtain. Limited and unreliable content could have a huge negative impact on the quality of the recommendation results. Overspecialization happens when the system only recommends items that are very similar to the items the users previously liked. When this happens, the system fails to recommend new items to a user just because they are not similar to any items liked by the user in the past.

2.1.2 Collaborative Filtering

Collaborative filtering techniques are based on collecting and analyzing a large amount of information on users' behaviors, activities or preferences. Then it makes prediction of what users will like, based on their similarity to other users. The preference or rating values that a user has towards an item is very important during the recommendation. Sometimes the rating values are collected explicitly. For example, each user can assign a rating value ranging from one to five to a movie. Sometimes the rating values are not given explicitly, implicit information could be used. For example, the number of times that a user watches a movie could be implicit information. The recommender system analyzes the relationships among users and items. Then it makes a list of recommended items for the user.

CF methods can be divided into memory-based and model-based methods (Su and Khoshgoftaar, 2009). Memory-based methods focus on computing the relationships

between items or users. Some similarity metrics are used for measuring the similarity of users and items. Recommendations are made based on the similarity of users and items. Model-based methods, on the other hand, build a model that explains the ratings by characterizing items and users. Collaborative filtering methods often suffer from three problems: cold start, scalability, and sparsity. Some of the most common collaborative filtering techniques, such as user-based recommendation, item-based recommendation, and Probabilistic Latent Semantic Analysis are summarized in chapter Three.

2.2 Location Recommendation

The research on LBSNs has become a hot area in both academia and industry since LBSNs bridges the gap between the online social networks and physical locations. Location recommendation is one of the most popular research topics on LBSNs. It is the task of suggesting unvisited locations to the users based on the information such as the users' visiting history, preferences, current time and profile of the locations and users.

Beeharee et al. (2006) and Simon et al. (2007) proposed to provide location recommendations based on a user's real-time location in mobile tour guide systems. Park et al. (2007) considered users' location histories in their systems to provide the users with more personalized location recommendations. Zheng et al. (2010) recommended locations and activities to users by utilizing multiple users' real-world location histories. However, those works were doing the location recommendation based on the GPS trajectory logs and they did not consider the check-in data.

Since there is no explicit rating in location recommendation, the recommender should imply a user's preference towards a location by utilizing the information

collected. The check-in data is one of the most important information. A check-in records the timestamp that a user visited a location.

The check-in can be considered as implicit ratings and used in different ways for location recommendation. Ye et al. (2011) transformed the numbers of check-ins into binary ratings. Then, the binary ratings were used for calculating the similarity weights between friends. In terms of the fact that social friends share more common locations than non-friends, a memory-based CF approach was proposed to predict the rating given by a user to a location using the ratings of his/her friends. Berjani and Strufe (2011) proposed two inference strategies to interpret the check-in data as user preferences. The first one is a simple binary preference definition, and the second one is based on the method of equal width intervals (EWI). The open scale of check-ins for each user is divided into intervals of equal width and the rating is derived by the index of the interval. A latent factor model is learned to predict the missing ratings using model-based CF.

Recent studies have started to consider the social relationships between users for the location recommendation on LBSNs. Ye et al. (2011) proposed a fusion framework named USG to recommend locations by using three different models: 1) a user-based collaborative filtering (CF) model, 2) a social influence model, and 3) a geographic influence model. The number of check-ins was transformed into binary ratings, which then were used for calculating the similarity weights between friends. The user-based CF model was built for predicting the preference given by a user to a location using the preferences of similar users. The social influence model was also a CF model that predicts the preference given by a user to a location using the preferences of his/her friends. The geographic influence model used a power law distribution to model the

probability of visiting locations that have certain distances from the previously visited locations of the user. Finally, the probability of a user checking into a location can be estimated as:

$$p(u,l) = (1-\alpha-\beta)\cdot U(u,l) + \alpha \cdot S(u,l) + \beta \cdot G(u,l) \quad (2.1)$$

where u is the user for the recommendation, l is a candidate location. $U(u,l)$ is the probability of the user checking into the location based on the user-based collaborative filtering recommender; $S(u,l)$ is the probability of the user checking into the location based on the social influence model; $G(u,l)$ is the spatial probability the user checking into the location based on the geographical influence model; α and β are two weight parameters to denote the relative importance of social influence and geographical influence. Since users usually go to different locations for different activities at different time, temporal constraints exist for checking into locations. However, the model did not consider the temporal information existed inside the check-in data.

The human periodic movement behaviors and temporal patterns in check-in data were studied recently. Cho et al. (2011) provided location recommendations based on the periodicity of human movements and social ties. Two models were proposed, which were called *PMM* (Periodic Mobility Model) and *PSMM* (Periodic Social Mobility Model). The method separates the user's behavior into two states: home state and work state. The user is either in home or work state at any point in time. The PMM models the probability distribution over the state of the user over time. The probability of a user checking into a location is estimated as

$$p(u,l|t) = p(u,l|home) * p(u,home|t) + p(u,l|work) * p(u,work|t) \qquad (2.2)$$

where $p(u,home|t)$ is the probability of the user in home state at time t; $p(u,l|home)$ is the probability of the user checking into location l when the user is in home state; $p(u,work|t)$ is the probability of the user in work state at time t; $p(u,l|work)$ is the probability of the user checking into location l when the user is in work state. It is defined as the summation of the probability of the user checking into the location given the user is in home state and the probability of the user checking into the location given the user is in work state. The advantage of the model is that it considers the periodicity of human movement behaviors between home and work locations. However, it does not consider other important information such as users' preferences and location related information (e.g. location category).

The utilization of location category information was studied in recent research. The category of a location usually can reflect the activities happening in that location, it represents a user's check-in behavior to some extent. Rahimi and Wang (2013) proposed two recommender algorithms called PCR (Probabilistic Category Recommender) and PCLR (Probabilistic Category-based Location Recommender) by utilizing the category information inside check-in data. The temporal and spatial check-in behaviors of users were modeled using probability distribution functions (PDF). PCLR combined the temporal category model used in PCR with a geographical influence model built on the spatial PDF to do location recommendation. By combing the temporal and spatial

components, the probability of a user *u* checking in to a location *l* at the given time *t* is defined as:

$$P(u, l|t) = S(l; h_u) * T(u, c_l|t) \quad (2.3)$$

where h_u is the home location of user *u* and c_l is the category of location *l; S* is the spatial probability of visiting the location *l* given the home location of the user; and $T(u, c_l|t)$ is the probability of checking in to the category of location *l* at given time *t* based on the temporal category model. However, the temporal model only considered the periodicity of only one user's check-in behaviors. It did not consider the check-in behaviors of similar users. If a user has only visited a few location categories before based on his/her past history, some potential locations might not be suggested to the user.

Chapter Three: **Location Recommendation Utilizing Check-in Data**[1]

This chapter presents an empirical study of recommending locations on location based social networks utilizing the check-in data.

3.1 Introduction

With the rapid development of online social networks (OSNs), people are no longer satisfied with sharing posts or pictures with their friends through OSNs. People tend to share more and know more about what their friends are doing, where they are, and whom they are with. Thanks to the location-based services associated with mobile devices, nowadays it is much easier for users to attach their geographical context by checking-in a certain location. Currently, a number of location-based social network (LBSN) services, e.g., Foursquare[2] and Gowalla[3], have emerged. Users of LBSN services are more interested in sharing tips and experiences of their visits to various locations, e.g., restaurants, stores, tourist attractions, etc. Since the exploration in cities and neighborhoods is one of the main activities in many LBSNs, how to recommend new locations to users on LBSNs becomes a novel challenge that attracts much attention of researchers recently.

Recommender systems are the natural solution for recommending locations on LBSNs. Generally, the strategies of recommender systems can be categorized into three types: content-based filtering, collaborative filtering and hybrid recommender systems.

[1] Published in Canadian AI 2012 conference (Zhou and Wang, 2012)
[2] https://foursquare.com
[3] http://gowalla.com/

Recommender systems of content-based filtering (CBF) make recommendations by analyzing the content of textual information, such as item (e.g. locations) descriptions and users' profiles, and finding regularities in the content (Pazzani, 1999). Classification algorithms are always used in the content-based recommenders. Since content-based techniques always need enough information to build a reliable classifier, they usually are highly dependent on domain knowledge about the features explicitly associated with the objects they attempt to recommend. Domain knowledge is the features or characteristics describing the nature of the users or items such as user profile. In practice, however, such domain knowledge is usually hard to collect or unavailable. Recommender systems of collaborative filtering (CF), on the other hand, do not depend on domain knowledge. CF recommenders collect and analyze users' behaviors and activities, and then predict what users will like based on their similarity to other users. Hybrid recommender systems are based on a combination of CBF and CF methods.

This chapter aims at an empirical study of recommending locations to users on a LBSN by CF recommenders. The first contribution of this chapter is that I design a distributed data crawler to acquire a large amount of real-world data from Gowalla, one of the most popular LBSN services in recent years. Based on this dataset, I attempt to recommend some unvisited locations to users only using check-in data, which is represented by a set of triples (u_i, l_j, n_{ij}) indicating that user u_i has visited location l_j n_{ij} times.

Usually, CF recommenders infer users' preferences by the rating values users gave to some items. The rating values convey the explicit opinions of users towards items so that the rating behavior is usually called the explicit feedback. In our task of location

recommendation, however, the number of check-ins n_{ij} could not explicitly reflect user u_i's preference for location l_j.

Given two check-in triples $(u_i, l_1, 1)$ and $(u_i, l_2, 100)$, for example, it is difficult to validate that user u_i likes location l_2 much more than l_1. It could be a situation that l_1 is a restaurant that u_i has visited once and likes, and l_2 is a bus station that u_i has to visit every day but does not like.

Therefore, the check-in behavior is usually called the implicit feedback in contrast to the explicit feedback. How to use the check-in behavior to recommend locations is still a challenging question for researchers, while there are few studies focusing on this topic currently. The second main contribution of this chapter is to carry out a thorough empirical study on the different utilizations of check-in behaviors for the location recommendation on LBSN. Particularly, three different kinds of utilizations are considered to present check-in behaviors: binary utilization, FIF (Frequency - Inverse Frequency) utilization, and probability utilization.

Typically, different CF recommenders have their own preferences for the types of ratings. The third contribution of the chapter is to introduce different algorithms of recommenders using different types of check-in utilizations to recommend locations on LBSN. The recommenders include user-based CF, item-based CF, and probabilistic latent semantic analysis (PLSA). Finally, a set of experiments have been conducted to compare the performances of different recommender algorithms using different types of check-in values.

The rest of this chapter is organized as follows. In Section 3.2, the details about different types of check-in values are described. In Section 3.3, the recommender

algorithms are described. Section 3.4 presents the experiments and discusses the results. Finally, the summary of this chapter is given in Section 3.5.

3.2 Check-in Data Utilization

3.2.1 Data Collection

In order to study the location recommendation on LBSN, a generic distributed data crawler is developed to collect the real-world data from Gowalla. It was a very popular LBSN launched in 2007 and was acquired by Facebook in 2012. Based on the web-based APIs of Gowalla, the data crawler consists of four main components: a set of crawler instances, a configuration server, a task engine, and a central database. The overall architecture is shown in Figure 3-1. It is designed to be distributed because it can dispatch multiple crawler instances on different machines to reduce the overall time required for collecting data and also overcome the quota limitation set by the service.

A crawler instance is an actual process for executing a crawling task. The configuration server defines configuration parameters for crawling tasks, by which it ensures that each crawler instance just collects a portion of check-in data from Gowalla. The task engine is responsible for managing all the crawler instances distributed in different machines. At the beginning, the task engine reads the parameters from the configuration server and creates a set of crawler instances, and then dispatches individual crawler instances to different machines. The parameters include the range of user IDs and location IDs for each local task. Next, each crawler instance starts to collect the check-in data using Gowalla's public web-based APIs. The status of each crawler instance is reported to the task engine periodically. Finally, the check-in data is saved into the central

database. This architecture of distributed crawlers significantly reduces the crawling times required for collecting the large portion of data from a LBSN.

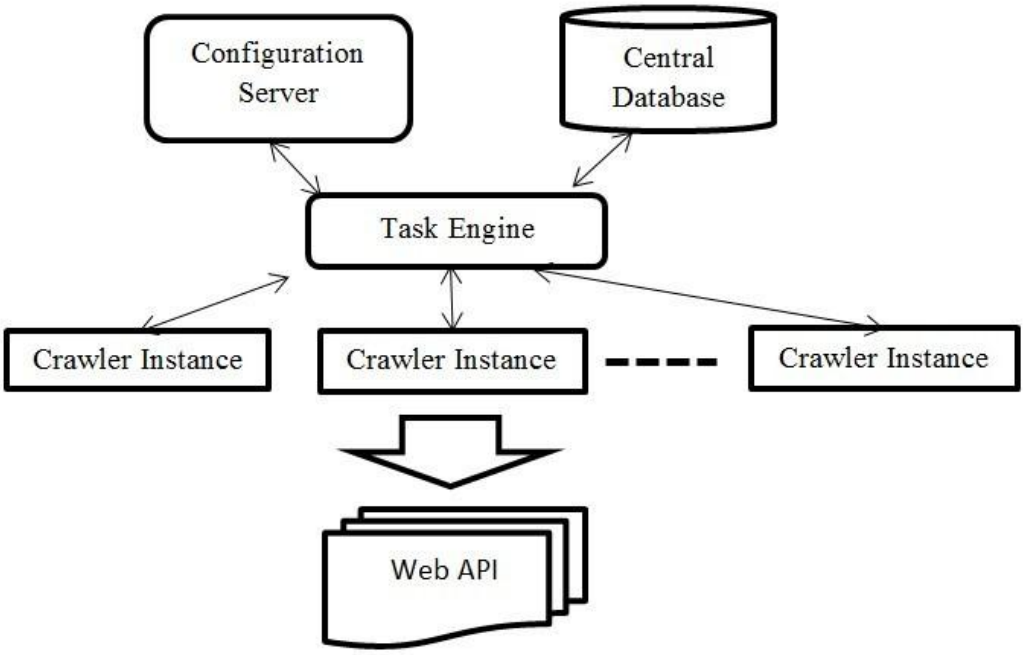


Figure 3-1 The architecture of distributed data crawler

Table 3-1 The description of the check-in data in four US cities

region	num of users	num of locations	num of check-ins	num of user-location pairs
Austin	2777	1982	102612	93767
New York	849	1551	21380	19461
San Francisco	766	1166	24160	21938
Seattle	509	864	15634	14135

A large portion of check-in data on Gowalla is collected from October 2011 to November 2011. In total, 90,978 user profiles and 143,196 locations have been acquired. The public API only provides the last 20 check-in events for a user. Each user's last 20

check-in events are obtained. The total number of user-location observations is 346,618 and the total number of check-ins is 594,474. Based on the analysis of the real world data, it is observed that most users only check-in within a certain geographically bounded region. Therefore, this chapter focuses on check-in data specific to four USA cities, namely, Austin, New York, San Francisco, and Seattle. Table 3-1 provides a description of the check-in data in these four cities. In order to filter the noises, the check-in data is preprocessed as follows by: 1) considering only the local users whose visited locations are all within the same city and, 2) ignoring the users who only visited one location as well as locations which are visited by only one user.

3.2.2 Data Utilization

Given our task of recommending locations to users without any domain knowledge from LBSN, we have to infer users' location preferences based only on the check-in data.

Following the convention of recommender algorithms, we can use a matrix $R \in \mathbb{R}^{m \times n}$ to represent the check-in data. In matrix *R*, each row is a vector associated with a user, and each column is a vector associated with a location. An entry of *R* represents the corresponding user's interest in the corresponding location, where the higher the value is, the more the user interests in the location. Suppose there are *m* users and *n* locations, the ratings matrix is shown in Table 3-2. Each row of the matrix represents a user, and each column of the matrix represents a location. Each entry of the matrix represents a user's interest in a location. For example, the entry r_{ij} denotes the interest for the user *i* in the location *j*.

Table 3-2 The ratings matrix

	I_1	…	I_j	…	I_n
U_1	$r_{1,1}$		$r_{1,j}$		$r_{1,n}$
⋮					
U_i	$r_{i,1}$		$r_{i,j}$		$r_{i,n}$
⋮					
U_m	$r_{m,1}$		$r_{m,j}$		$r_{m,n}$

This chapter proposes three ways to utilize check-in data for inferring users' location preferences. Formally, given a check-in triple (u_i, l_j, n_{ij}), we try to infer the preference value r_{ij} for user u_i to location l_j.

The first way of utilizing the check-in data is to transform the number of check-ins into a binary rating value. Particularly, if there is a triple (u_i, l_j, n_{ij}) that has been observed in the check-in data, then the preference value r_{ij} would be 1; otherwise, r_{ij} would be 0.

The main challenge of directly utilizing the number of check-ins n_{ij} as a rating value is that n_{ij} cannot reflect the user u_i's interest towards location l_j straightforwardly, since n_{ij} is a kind of implicit feedback. The prediction of u_i's location preference should be not only based on how many times u_i visited some locations, but also based on the overall check-in situations at the locations which u_i visited. Therefore, a weight schema

called FIF (Frequency - Inverse Frequency) is proposed as the second way of utilizing the check-in data, by which the number of check-ins is transformed into a scaled weight value.

The basic idea of FIF originates from TFIDF (Term Frequency - Inverse Document Frequency) which is a weight schema widely used in information retrieval and text mining for evaluating how important a word is to a document in a corpus. Different from TFIDF, the schema of FIF consists of two sub-schemas, namely, UFILF (User Frequency - Inverse Location Frequency) and LFIUF (Location Frequency - Inverse User Frequency). Specifically, UFILF is used to evaluate how significant a user's visit is towards a location, while LFIUF is used to evaluate how important a location is for a user.

The user frequency $UF(u, l)$ of a user *u* to a location *l* is defined as:

$$UF(u, l) = \frac{n(u, l)}{nc_U(l)} \tag{3.1}$$

where $n(u, l)$ is the number of check-ins by the user u in the location l, and $nc_U(l)$ is the number of check-ins at location *l* by all the users.

The inverse location frequency $ILF(u)$ of a user *u* for the entire set of locations is defined as

$$ILF(u) = \frac{N_L}{n_L(u)} \tag{3.2}$$

where N_L is the total number of locations, and $n_L(u)$ is the number of locations where the user *u* has visited. Then, UFILF is defined as

$$UFILF(u,l) = UF(u,l) \times \log(ILF(u)) \tag{3.3}$$

From Equation (3.3), we can see that a high value of UFILF is reached by a high user frequency and a low location frequency. Therefore, UFILF sets a weight of a user *u* towards a location *l* by balancing the weight value between the count of locations *u* has visited and the number of check-ins at *l* by all the users.

On the other hand, LFIUF is used to evaluate how important a location is for a user, the location frequency $LF(u,l)$ of a location *l* to a user *u* is defined as

$$LF(u,l) = \frac{n(u,l)}{nc_L(u)} \tag{3.4}$$

where $n(u,l)$ is the number of check-ins by the user u in the location l, $nc_L(u)$ is the number of check-ins by the user *u* at all locations.

Similarly, the inverse user frequency $IUF(l)$ of a location *l* can be defined as

$$IUF(l) = \frac{N_U}{n_U(l)} \tag{3.5}$$

where N_U is the total number of users, and $n_U(l)$is the number of users who have visited the location l. Then, LFIUF is defined as

$$LFIUF(u,l) = LF(u,l) \times \log(IUF(l)) \tag{3.6}$$

Comparing Equation (3.3) with Equation (3.6), we can see that LFIUF attempts to balance the weight value for a user-location pair (u,l) between the count of users who visited l and the number of check-ins by u at all locations. A high value of LFIUF can be reached by a high location frequency and a low user frequency.

Based on UFILF and LFIUF, the FIF schema for a user-location pair (u,l) can be defined as:

$$FIF(u,l) = UFILF(u,l) + LFIUF(u,l) \tag{3.7}$$

For each user, we first calculate FIF values for all the locations this user visited, and then normalize those values into (0,1). The normalized FIF values are treated as the preference values of users to corresponding locations. They are used to fill in the corresponding entries of matrix R. The empty entries of R which correspond to the unobserved user-location pairs are filled by zero.

The third way of utilizing the check-in data is to estimate the joint probability $P(u,l)$ for each pair of user and location, by which a user u's location preferences can be inferred from a set of probabilities $P(u,l_j|l_j \in L)$ where L is the set of all the locations.

The original idea of using the joint probability $P(u, l)$ in recommender systems came from Probabilistic Latent Semantic Analysis (PLSA) (Hofmann, 1999), which is one of the most popular methods for automated document indexing. The core of PLSA is a statistical model called aspect model (Saul and Pereira, 1997). Aspect model is a latent variable model for general co-occurrence data which associates a set of unobserved latent variables with each observation. In our scenarios, for example, each check-in triple (u_i, l_j, n_{ij}) will associate with a set of latent variables from $Z = \{z_i, \dots, z_k\}$. The latent variables Z contain a set of states z for each user-location pair, so that user u and location l are rendered conditionally independent. The possible set of states of z is assumed to be finite and of size k. The strategy of choosing the number of k needs to adjust the model complexity according to the amount and sparseness of available data. The state z from latent variables Z associated with an observation (u,l) is supposed to model a hidden cause. In other words, a user u visits location l because of the cause z. Each z can be considered as a hypothetical explanation for an implicit rating that is unobservable. Therefore, the joint probability $P(u, l)$ can be estimated as:

$$P(u, l) = \sum_{z \in Z} P(u, l, z) = \sum_{z \in Z} P(u, l|z)P(z) \tag{3.8}$$

The aspect model has an important assumption that users and locations are conditionally independent given latent values. Based on this assumption, Equation (3.8) can be written as

$$P(u,l) = \sum_{z \in Z} P(u|z)P(l|z)P(z) \tag{3.9}$$

Since PLSA is a kind of recommender without the need of conventional ratings, it is not necessary to fill in the matrix R. We will introduce details of how to estimate $P(u,l)$ within PLSA recommender in the next section.

3.3 Location Recommenders on LBSN

Generally, the methods of CF recommenders fall into two categories: memory-based methods and model-based methods. Memory-based methods use rating data to calculate the similarities or weights between users or items and make predictions or recommendations according to those calculated similarity values. On the other hand, model-based CF methods design and develop models using machine learning algorithms to learn complex patterns based on training data, and then make intelligent predictions for the CF tasks.

Based on the different utilizations of check-in data, we build up the corresponding CF recommenders to recommend unvisited locations to LBSN users. Specifically, two memory-based recommenders, namely, user-based recommender and item-based recommender, are created by different similarity calculations based on the binary utilization and the FIF utilization. A model-based recommender, namely, PLSA recommender, is built for the probability utilization of check-in data.

3.3.1 Memory-based Recommenders

User-based recommender and item-based recommender are two prevalent memory-based methods. Given an active user u_a, user-based recommender first calculates the similarity $sim(u_a, u_i)$ between u_a and another user u_i. Then the rating value to a location l_j, which u_a never visited, is predicted by following equation:

$$P_{u_a, l_j} = \bar{r}_a + \frac{\sum_{u_i \in U_{l_j}} (r_{i,j} - \bar{r}_i) \cdot sim(u_a, u_i)}{\sum_{u_i \in U_{l_j}} sim(u_a, u_i)} \tag{3.10}$$

where the set U_{l_j} represents all the users that have visited location l_j, $\bar{r}_a$ and $\bar{r}_i$ are the average ratings for the users u_a and u_i of all other visited locations, and $r_{i,j}$ is the rating u_i gives to l_j, and $sim(u_a, u_i)$ denotes the similarity weight between users u_a and u_i.

Finally, a list of top-*N* location recommendations is made to u_a, which consists of *N* locations having the top predicted ratings.

Item-based recommender has a similar procedure to the user-based recommender except it calculates the similarity between items. The rating prediction for a user u_a to a location l_j can be made by taking a weighted average of all the ratings of the user as following equation:

$$P_{u_a, l_j} = \bar{r}_a + \frac{\sum_{l_i \in L_{u_a}} r_{a,i} \cdot sim(l_i, l_j)}{\sum_{l_i \in L_{u_a}} sim(l_i, l_j)} \tag{3.11}$$

where L_{u_a} represents all the locations visited by u_a, $r_{a,i}$ is the rating given to location l_i by u_a, and $sim(l_i, l_j)$ denotes the similarity weight between locations l_i and l_j.

From Equations (3.10) and (3.11), we can see that the critical part for both user-based recommender and item-based recommender is the similarity calculation. Jaccard coefficient and Pearson correlation are two widely used similarity measures.

The Jaccard coefficient is defined as the size of the intersection divided by the size of the union of the two sets. Using Jaccard coefficient, the similarity between two users u_i and u_j in the user-based recommender is calculated by Equation (3.12), and the similarity between two locations l_i and l_j in item-based recommender is calculated by Equation (3.13):

$$sim(u_i, u_j) = \frac{\left| L_{u_i} \cap L_{u_j} \right|}{\left| L_{u_i} \cup L_{u_j} \right|} \tag{3.12}$$

$$sim(l_i, l_j) = \frac{\left| U_{l_i} \cap U_{l_j} \right|}{\left| U_{l_i} \cup U_{l_j} \right|} \tag{3.13}$$

where L_{u_i} and L_{u_j} denote sets of locations that users u_i and u_j visited, respectively; U_{l_i} and U_{l_j} denote sets of users who have visited locations l_i and l_j, respectively.

Pearson correlation deals with the continuous FIF variables by dividing the covariance of two variables by the product of their standard deviations. Using Pearson correlation, the similarity between two users u_i and u_j in the user-based recommender is

calculated by Equation (3.14), and the similarity between two locations l_i and l_j in the item-based recommender is calculated by Equation (3.15):

$$sim(u_i, u_j) = \frac{\sum_{l_k \in L_{u_i,u_j}} (r_{i,k} - \bar{r}_i) \cdot (r_{j,k} - \bar{r}_j)}{\sqrt{\sum_{l_k \in L_{u_i,u_j}} (r_{i,k} - \bar{r}_i)^2} \sqrt{\sum_{l_k \in L_{u_i,u_j}} (r_{j,k} - \bar{r}_j)^2}} \quad (3.14)$$

$$sim(l_i, l_j) = \frac{\sum_{u_k \in U_{l_i,l_j}} (r_{k,i} - \bar{r}_i) \cdot (r_{k,j} - \bar{r}_j)}{\sqrt{\sum_{u_k \in U_{l_i,l_j}} (r_{k,i} - \bar{r}_i)^2} \sqrt{\sum_{u_k \in U_{l_i,l_j}} (r_{k,j} - \bar{r}_j)^2}} \quad (3.15)$$

where L_{u_i,u_j} is the set of all the locations visited by both users u_i and u_j, U_{l_i,l_j} is the set of all the users who visited both locations l_i and l_j, and other notations have the same meanings as in Equation (3.10) and (3.11).

Because Pearson correlation requires an exact rating value and binary utilization only has two values (0 or 1), we use Jaccard coefficient as the similarity measures for the binary check-in utilization. Pearson correlation is used for the FIF check-in utilization.

3.3.2 Model-based Recommenders

Model-based recommenders usually learn statistical models to discover the latent patterns that are able to explain how the rating data is generated. Model-based recommenders often achieve better performance than memory-based recommenders by addressing the scalability and sparsity problems (Su and Khoshgoftaar, 2009). The PLSA recommender is a model-based recommender, in which the aspect model (Hofmann, 1999) is built for inferring the users' preferences by estimating the joint probabilities.

PLSA recommender makes the location recommendation as follows. Given an active user u, PLSA recommender first estimates the joint probability $P(u,l)$ for each location l in the data set. Then, the set of joint probabilities are ranked. Finally, N locations which have top values of joint probabilities are selected to generate a top-N recommendation list for the user u.

The essential part of PLSA recommender is to estimate the joint probability $P(u,l)$ for each user-location pair (u,l). As mentioned in Section 3.2.2, $P(u,l)$ can be estimated by Equation (3.9) in terms of the parameters of aspect model, namely, the conditional probabilities of u and l given latent variables and the prior probabilities of latent variables. Expectation Maximization (EM) algorithm is used to estimate the maximum likelihoods of those parameters. The algorithm consists of two steps: an expectation (E) step and a maximization (M) step. The details about EM algorithm used by PLSA recommender are shown in Figure 3-2.

Algorithm ExpectationMaximization(U, L, C, Z, maxIter)
// **Input:** User set U, location set L, check-in set $C = \{(u_i, l_j, n_{ij}) | u_i \in U, l_j \in L, n_{ij} \in \mathbb{R}\}$,
// latent variable set Z, and maximum iteration number maxIter
// **Output:** Parameter sets: $U|Z = \{P(u_i|z_j) | u_i \in U, z_j \in Z\}$, $L|Z = \{P(l_i|z_j) | l_i \in L, z_j \in Z\}$ and
// $Z_{prior} = \{P(z_i|z_i \in Z\}$
// **Initialization:** Randomly assign values within (0,1) to U|Z, L|Z, and Z_{prior};
// normalize each set; nIter = 0

1-**while** nIter < maxIter **do**
2- nIter = nIter + 1
3- // E-Step
4- **for** $u \in U$ **do**
5- **for** $l \in L$ **do**
6- **for** $z \in Z$ **do**
7- $P(z|u,l) = \frac{P(z)P(u|z)P(l|z)}{\sum_{z' \in Z} P(z')P(u|z')P(l|z')}$
8- **end for**
9- **end for**
10- **end for**
11- // M-Step
12- **for** $z \in Z$ **do**
13- **for** $u \in U$ **do**
14- $P(u|z) = \frac{\sum_l n_{ul} \times P(z|u,l)}{\sum_{u',l} n_{u'l} \times P(z|u',l)}$
15- **end for**
16- **for** $l \in L$ **do**
17- $P(l|z) = \frac{\sum_u n_{ul} \times P(z|u,l)}{\sum_{u,l'} n_{ul'} \times P(z|u,l')}$
18- **end for**
19- $P(z) = \frac{1}{R} \sum_{u,l} n_{ul} \times P(z|u,l), R \equiv \sum_{u,l} n_{ul}$
20- **end for**
21-**end while**

Figure 3-2The EM algorithm

3.4 Experiments

Experiments are conducted to compare the performances of different recommenders based on different utilizations of check-in data for recommending locations to users on LBSN. Specifically, there are five combinations of recommenders and utilizations evaluated in the experiments:

- User-based recommender using binary values (denoted by U-BINARY);
- Item-based recommender using binary values (denoted by I-BINARY);
- User-based recommender using FIF values (denoted by U-PEARSON);
- Item-based recommender using FIF values (denoted by I-PEARSON);
- PLSA recommender using probability values (denoted by PLSA).

The location recommendation algorithms are implemented in Java and are run on a PC with 12GB of ram and 2.7GHz CPU for the experiments. The dataset used is collected from Gowalla. It contains check-in records from four US cities, which has been described in Section 3.2.1. A check-in record indicates the event that a certain user has visited a certain location at a certain time. It contains the user ID, location ID, location latitude, location longitude, time stamp of the check-in, and location category. To evaluate the algorithms, each dataset is split into a training set and a testing set. 10% of the check-in triples are randomly selected from a dataset first. Then those triples' numbers of check-ins are marked off and their user-location pairs are put into the testing set. The remaining part of dataset forms the training set. Each recommender is trained based on the corresponding type of values which are calculated from the training set, and

then recommends top-*N* locations to the users in the testing set. For PLSA recommender, the number of latent variables is set to 100.

The performance of the location recommendation algorithms is evaluated by precision and recall. Precision and recall are widely accepted as the performance measurements for recommender systems (Su and Khoshgoftaar, 2009). Precision is the fraction of retrieved instances that are relevant, while recall is the fraction of relevant instances that are retrieved. They are defined as:

$$Precision = \frac{Number\ of\ correct\ recommendations}{Number\ of\ recommendations} \tag{3.16}$$

$$Recall = \frac{Number\ of\ correct\ recommendations}{Number\ of\ correct\ answers} \tag{3.17}$$

All of the check-in records in the testing set are considered as correct answers. If the location from the testing set appears in the top-N recommendation list for a user, it is marked as a correct recommendation. Two evaluation metrics, namely, *precision@N* and *recall@N*, are defined as follows:

$$precision@N = \frac{\sum_{u \in U_T} |TopN(u) \cap L_T(u)|}{\sum_{u \in U_T} |TopN(u)|} \tag{3.18}$$

$$recall@N = \frac{\sum_{u \in U_T} |TopN(u) \cap L_T(u)|}{\sum_{u \in U_T} |L_T(u)|} \tag{3.19}$$

where U_T is the set of users in the testing set, $TopN(u)$ is the set of top-*N* locations recommended to *u*, and $L_T(u)$ is the set of locations paired with *u* in the testing set.

Figures 3-3, 3-4, 3-5 and 3-6 show the results of *precision@N* and *recall@N* (*N*=5,10,15,20) for all recommenders in all four cities. According to the results, we observe that U-BINARY always performs better than I-BINARY. U-PEARSON performs better than I-PEARSON when *N* is small, but I-PEARSON outperforms U-PEARSON with the increase of *N*. U-BINARY usually outperforms U-PEARSON and I-PEARSON when *N* is small, but I-PEARSON outperforms U-BINARY in some datasets when *N* is large. Finally, PLSA always outperforms other recommenders on all the datasets. The reason that the performance of memory-based recommenders such as I-PEARSON varies is maybe because the number of users in each dataset is different.

From the results, we can see that the memory-based recommenders do not work better than the model-based recommender for this task. As mentioned in Section 3.3, memory-based recommenders are highly dependent on the similarity calculations. Even though the FIF utilization attempts to balance between the check-in frequencies at locations and the appearance frequencies of users, the correlations calculated by Pearson method cannot represent the similarities between users or items. The performances of U-PEARSON and I-PEARSON demonstrate that the quantity of check-ins may not be a good indicator for the location preference of a user. The binary utilization ignores the number of check-ins by only considering the appearance of a user at a certain location. As a type of implicit feedback, however, the user appearance cannot accurately reflect how interesting a location is to a user. PLSA, on the other hand, infers users' location preferences using the probability utilization. PLSA introduces the latent variables in the model, which essentially represent the potential relationships between users and locations. Instead of directly calculating similarities based on the quantity of check-ins,

PLSA estimates the joint probabilities of users and locations, which interprets the check-in behaviors of users in a better way. The experiments show that PLSA using the probability utilization works better for recommending location on LBSN.

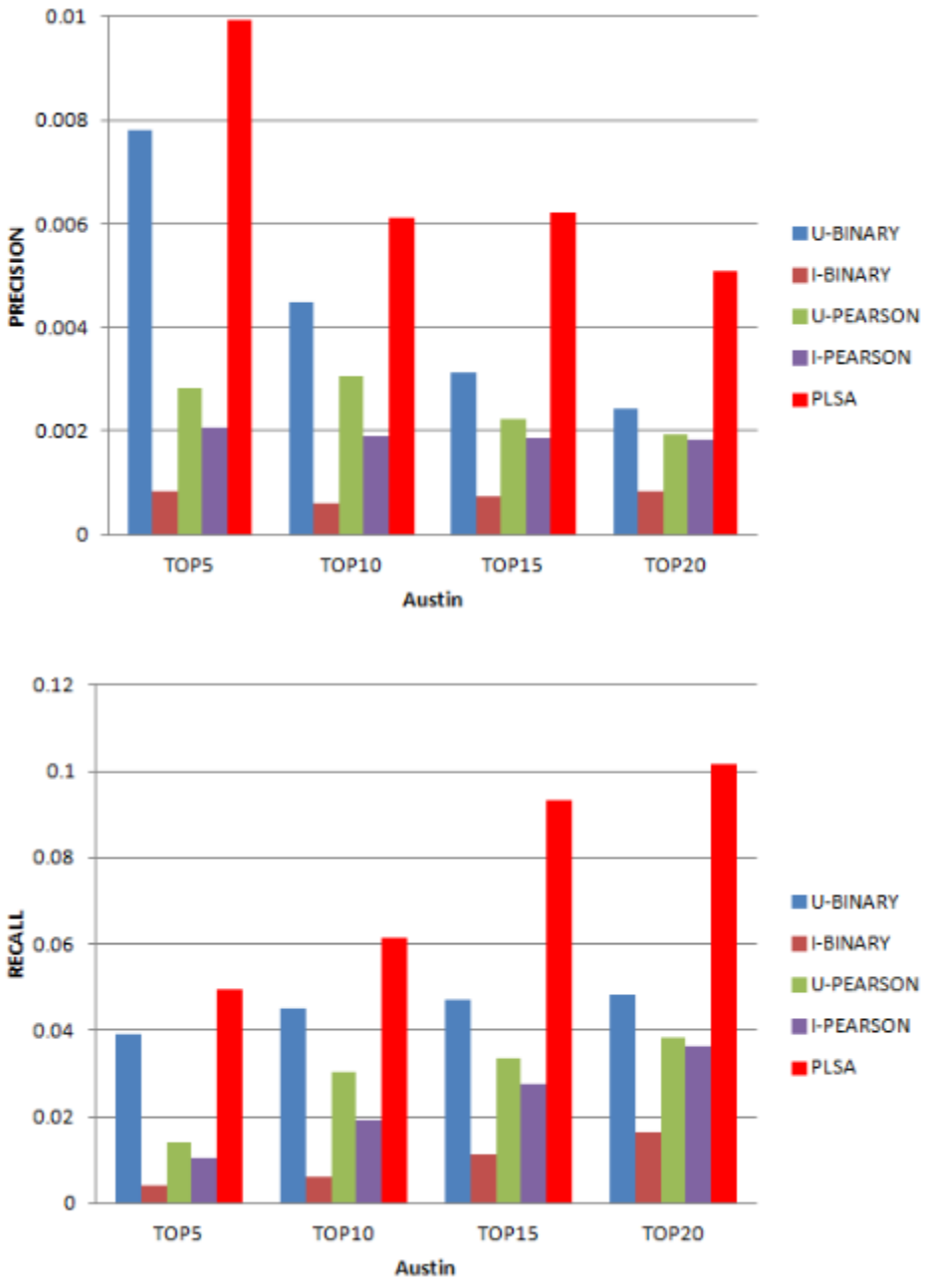


Figure 3-3 The results for recommending locations in Austin

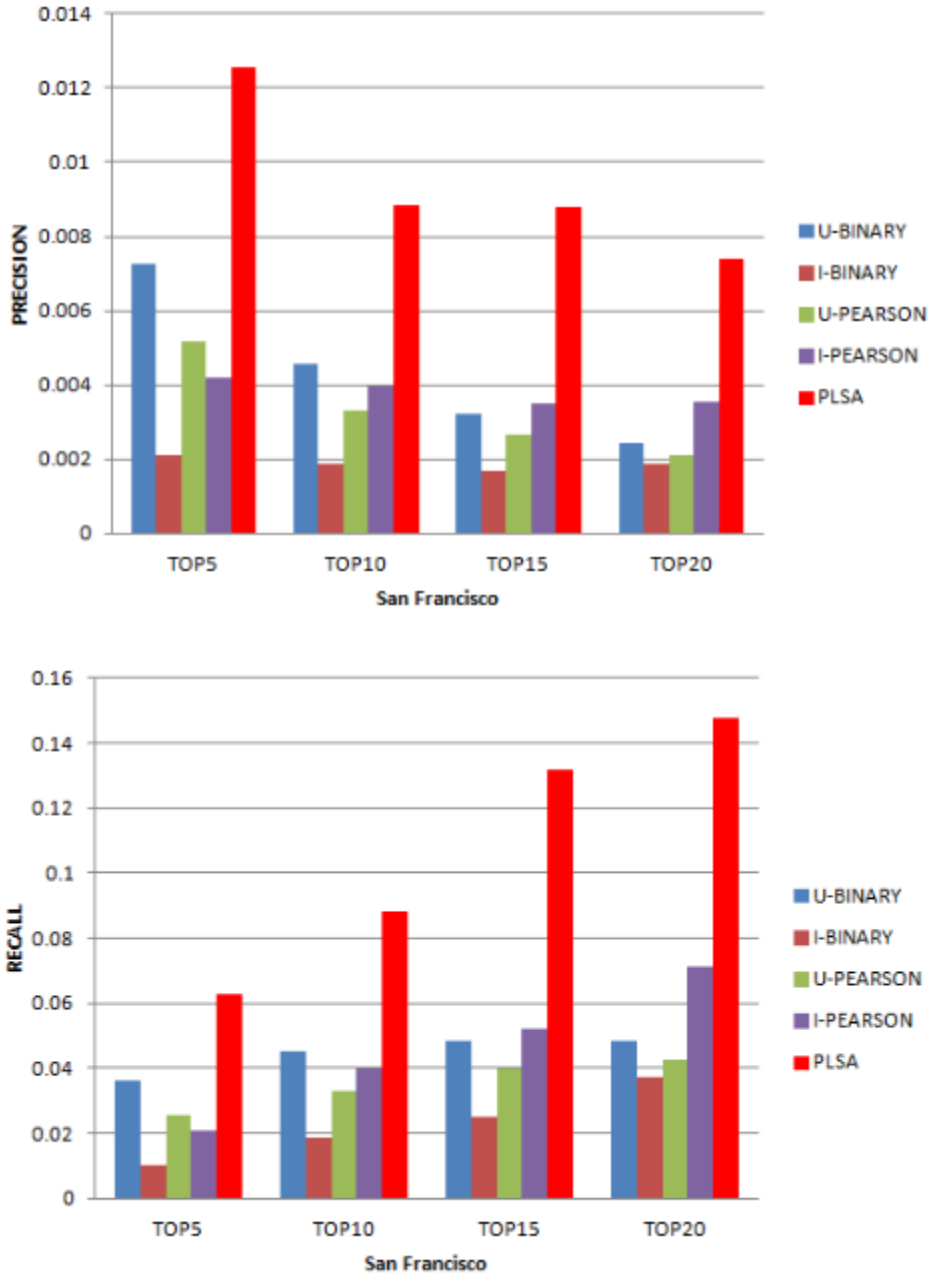


Figure 3-4 The results for recommending locations in San Francisco

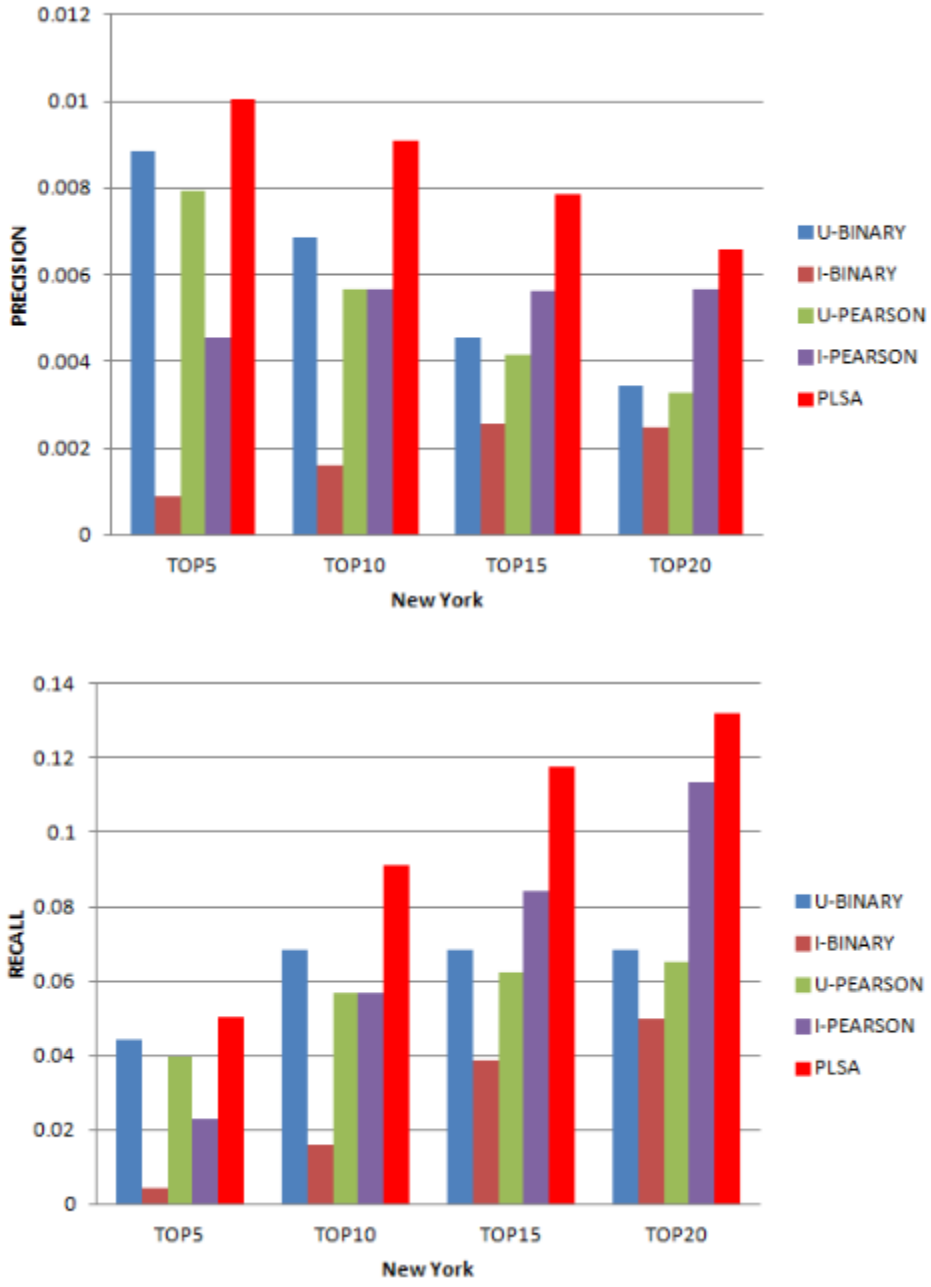


Figure 3-5 The results for recommending locations in New York

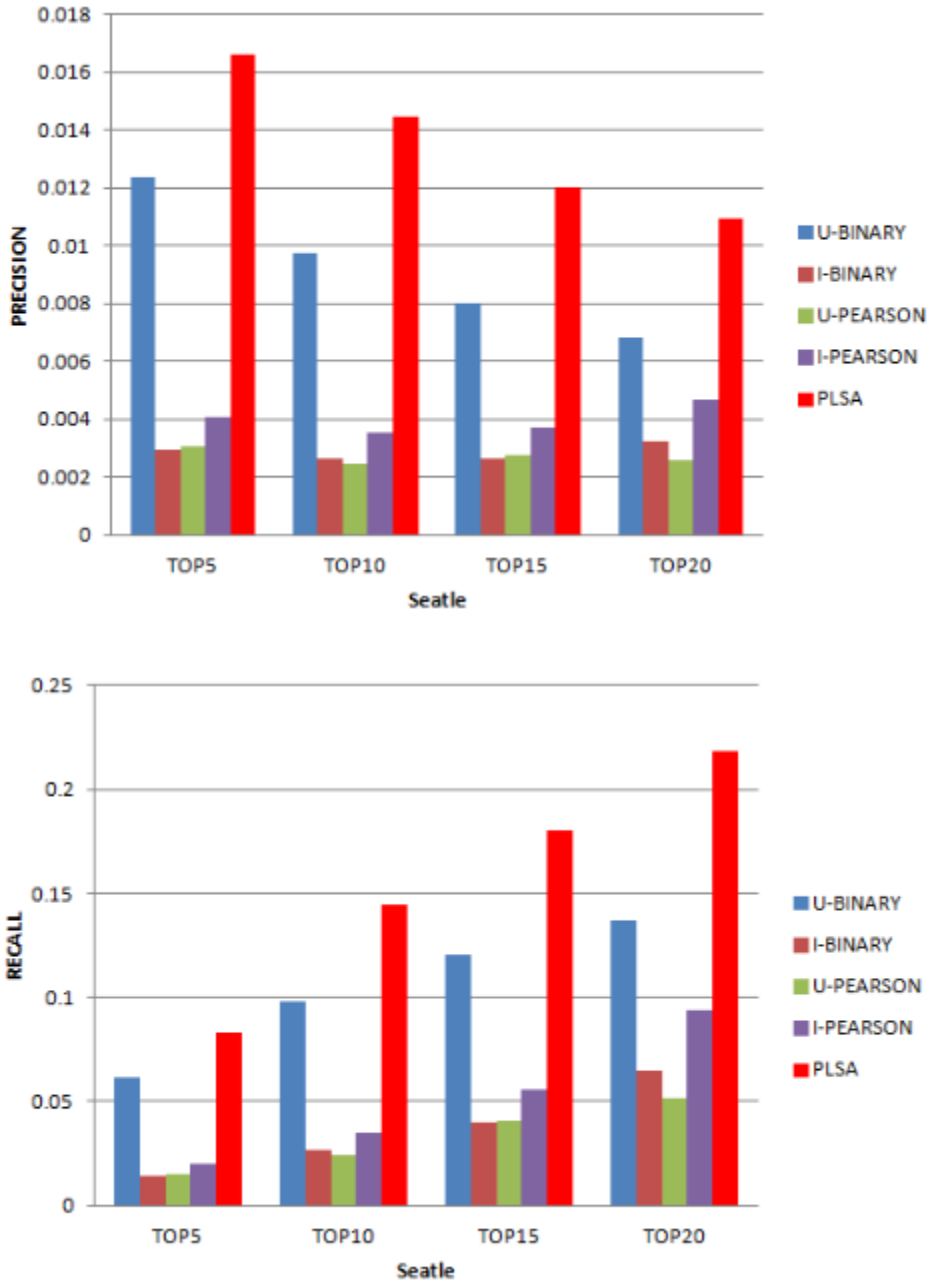


Figure 3-6 The results for recommending locations in Seattle

3.5 Summary

The development of location-based social networking (LBSN) services is growing rapidly these days. Users of LBSN services tend to visit various locations and share their exploration experiences to their friends. This chapter aims at a study of recommending locations to users on LBSNs by collaborative filtering (CF) recommenders based only on users' check-in data. A distributed crawler is developed to collect a large amount of real-world check-in data from Gowalla. Then, three approaches are proposed to utilize the check-in data, namely, the binary utilization, the FIF utilization, and the probability utilization. According to different utilizations, different CF recommenders are introduced, namely, user-based, item-based and probabilistic latent semantic analysis (PLSA), to do the location recommendation. Finally, a set of experiments is conducted to compare the performances of different recommenders using different check-in utilizations. The experimental results show that PLSA recommender with the probability utilization outperforms other combinations of recommenders and utilizations for recommending locations to users on LBSN.

Chapter Four: **Location Recommendation Utilizing Location Category**

This chapter presents a study of recommending locations on location based social networks utilizing the check-in data and location category information. The research in Chapter three does not consider spatial and temporal information from check-in data. This chapter extends the work of Chapter three by integrating spatial and temporal information in location recommendation. A location recommendation algorithm called *sPCLR* is proposed to recommend locations to the users at a given time of the day.

4.1 Introduction

The online social networks provide a platform for people to share information about their current statuses with friends. Some location-based social network (LBSN) services are emerging nowadays. By checking in to certain locations, users can attach geographical information with the posts and share them with each other on LBSNs. The location recommendation service provides suggestions of unvisited locations to the users on LBSNs based on their visiting histories and location related information such as location categories.

Location recommendation has attracted a lot of attention from both industry and academia. The existing methods mainly focus on utilizing the check-in histories and social ties within users' check-in data. Locations can be assigned to different classes as categories according to their shared characteristics. Category information is seldom utilized in the location recommendation. The category of a location usually can reflect the activities happening in that location, so it represents a user's check-in behavior to some

extent. When recommending locations to the user, the category information might be further exploited to suit his/her preferences. For example, if a user visits Chinese restaurants frequently based on his check-in history, that means he might like Chinese food very much. This implicit preference might influence the user's check-in behavior. The user is likely to visit another Chinese restaurant in the future. Therefore, this implicit preference indicated by location category information should be considered when doing location recommendations.

In addition to the category information, the temporal and spatial information of the check-in should be also considered in location recommendation. The temporal information such as the time of the check-in represents a user's periodic behavior. For example, a user usually visits coffee shops at 8AM in the morning and visits fitness centers at 7PM in the evening. That means the user likes drinking coffee in the morning and likes working out in the evening. When the recommendation choices contain a coffee shop and a fitness center, the fitness center should have higher priority to be recommended to the user if the time for the recommendation is in the evening. The spatial information such as the geographic position also has influence on a user's check-in behaviors. For example, users tend to visit locations that are close to their homes or offices.

This chapter investigates how to utilize the location category information to represent the hidden temporal patterns inside the check-in data. The problem is approached from an integration-based perspective. The integration of the temporal influence and the geographical influence is considered in location recommendation.

The temporal influence exploits the users' periodic check-in behaviors using a collaborative filtering approach. In a recommender system, collaborative filtering approach can inter a user's implicit preference by aggregating the behaviors of similar users. People might have similar periodic patterns for a location category. In the research, we assume that users who have similar temporal check-in patterns will have influence on each other's choice. For example, John and Mike usually visit coffee shops in the morning. They might be considered as having common interests and their check-in behaviors towards other places might influence each other. The temporal curves are used to represent a user's periodic check-in behaviors at different categories. Each temporal curve has a sequence of probability values representing how likely the user might visit a location category during each hour of the day. Based on the difference between two temporal curves, the temporal similarity is calculated for measuring the similarity between two users in terms of periodic check-in behavior. Since traditional similarity measure such as cosine similarity or Pearson correlation cannot be applied directly to temporal curves, a coupling method is proposed to represent the difference between temporal curves and calculate the temporal similarity. After similar users are obtained according to the temporal similarity, the periodic behavior of a given user can be predicted by a weighted summation of the periodic behaviors of his similar users.

The geographical influence exploits the geographical clustering phenomenon of user check-in activities on LBSNs. Since the check-in activities of users record their interactions with locations, the geographical proximities of locations will influence a user's check-in behavior. It is observed that a user tends to visit locations closer to their homes or offices (Rahimi and Wang, 2013). The geographical influence models a user's

probability of checking in to a location by considering the distance from the location to the user's home.

This chapter combines the temporal influence and geographical influence into location recommendation. Specifically, the contributions in this study include:

- Temporal patterns are investigated and extracted from location category information and check-in data. Temporal curves are proposed to represent users' periodic check-in behaviors for different categories.
- A coupling method is introduced to measure the difference between two temporal curves. It is further used to calculate the temporal similarity between two users in terms of periodic check-in behaviors. According to the temporal similarity, a temporal influence model is built that can predict the periodic behavior of a given user by a weighted summation of the periodic behaviors of his similar users.
- A location recommendation algorithm called sPCLR is proposed that integrates the temporal influence and geographical influence. The temporal influence model predicts the user's periodic check-in behaviors according to the temporal similarities between the user and other users. The geographical influence model measures the probability of checking in to a location by considering the user's home location.
- A set of experiments is conducted on a LBSN check-in dataset in order to evaluate the sPCLR location recommendation algorithm. The performance of sPCLR is compared with three existing location recommendation algorithms in terms of precision and recall.

The chapter is organized as follows. In Section 4.2, the temporal curve and temporal similarity are described. In section 4.3, the details of the location recommendation algorithm is described. Section 4.4 presents the experiments and discusses the results. Finally, the summary is given in Section 4.5.

4.2 User Temporal Curves and Temporal Similarity

In this section, we will extract user check-in behavior from category information, which is then represented by user temporal curves. The similarity between two users can be depicted based on calculating the distance between two temporal curves.

4.2.1 User Temporal Curves

The category of a location usually reflects the activities happening in that location. For example, if the category of a location is a coffee shop, the user will probably have a coffee in that location. Users tend to do similar activities during the same time of the day. Based on a temporal analysis on check-in data from Gowalla, it is observed that users tend to have a periodic behavior for visiting similar types of locations (Rahimi and Wang, 2013). For example, if a user usually visits a coffee shop at 8AM, then his probability of visiting another coffee shop at 9AM is higher than the probability of visiting it at 7PM. Rahimi and Wang (2013) proposed a temporal probability distribution function to quantify the probability of checking in to different categories at different times of the day. It first forms a subset of the check-ins that consists of check-ins for a certain category from a user. Then it produces a plot of the frequency of check-in pairs over given time differences in the subset. Figure 4-1 shows an example of the plot using 1-hour time window. A function is defined that can transform a time difference to a check-in

probability based on the plot. Then a temporal probability distribution function for a given set of check-ins is defined as:

$$TP(t;\mu) = F(\lfloor t - \mu \rfloor) \tag{4.1}$$

where t is the time we compute the probability for; μ is the average time of the check-ins in the subset; F is the function that transforms a time difference into a check-in probability based on check-in history.

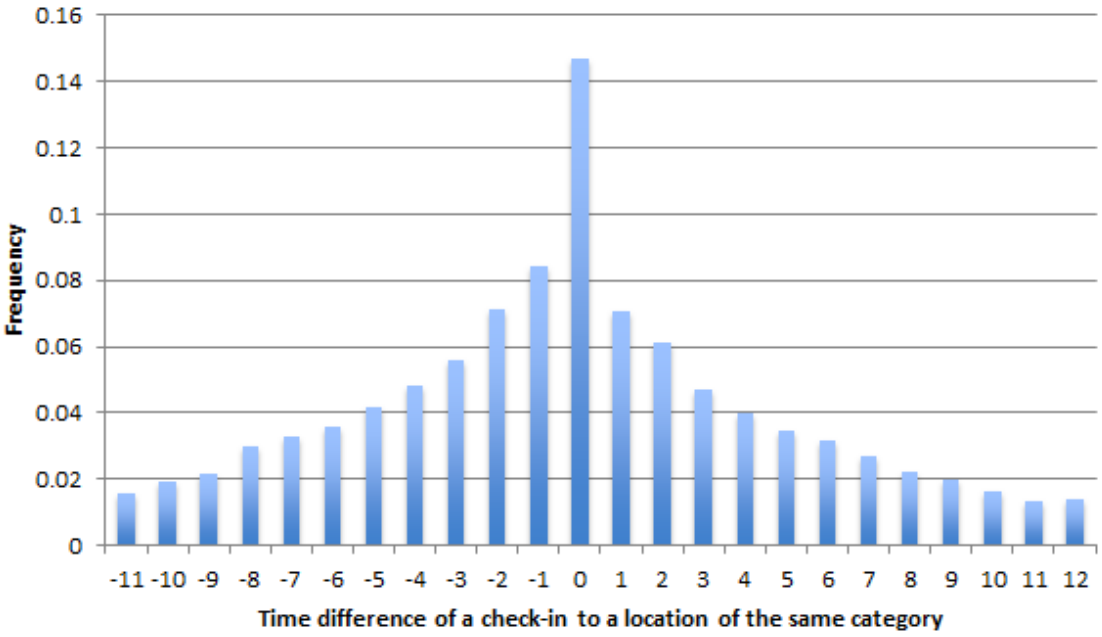
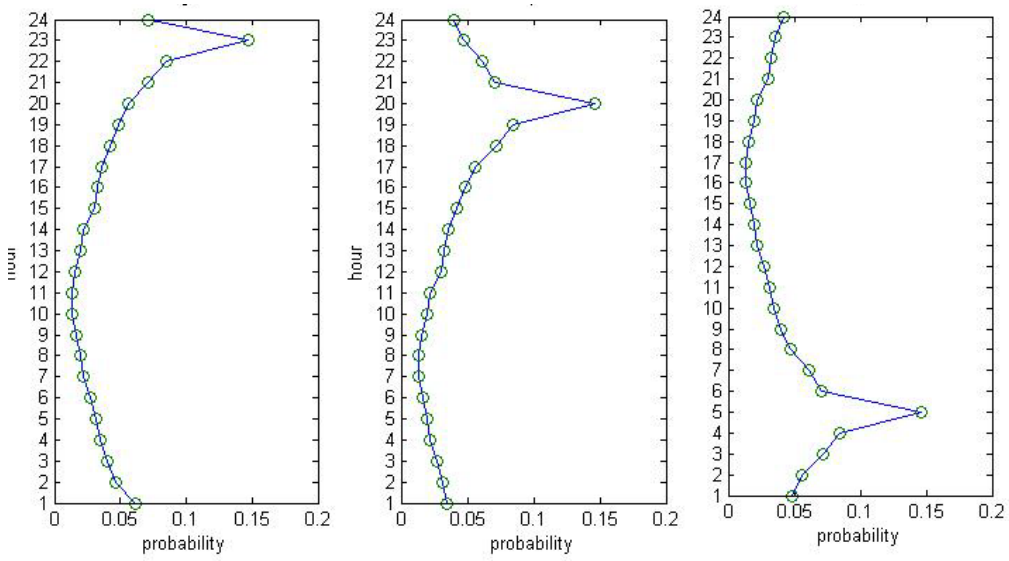


Figure 4-1 Example plot of the frequency of check-ins of the same category to the time difference using 1-hour time window

Based on the temporal probability distribution function, the probability of checking in to a specified category at 24 hours of the day for a user can be obtained. Thus, a user's check-in behavior for a category can be represented as a list of probability values.

Definition 4.1: A ***User Temporal Curve*** *U* for category *j* consists of a sequence of probability values, denoted as $U^j = (u_1^j, u_2^j, \dots, u_m^j, \dots, u_{24}^j)$, where u_m^j is the check-in probability for category *j* in hour *m*, and $1 \leq m \leq 24$. The sum of check-in probability over all hours is equal to 1.

A temporal curve has a sequence of probability values representing how likely the user might visit a location category during each hour of the day. The larger the check-in probability, the more likely the user will visit the location category at that time window of the day. Therefore, a user's check-in behavior can be represented by a list of temporal curves where each curve models the temporal pattern for one category during a day. In order to produce the temporal curve for a certain location category, the user should have visited the location category before. Figure 4-2 shows an example of the user temporal curves for three different users towards one category.



(a) User 1 (b) User 2 (c) User 3

Figure 4-2 An example of the user temporal curves for three different users

The X axis is the check-in probability, and the Y axis is the time window of a day. We divide a whole day into 24 windows and each window indicates one hour of a day. Based on the assumption that a user will visit the location category once in a day, the three curves have the same average value of 1/24 for the X axis. If the traditional aggregation approaches such as average function is used, the three curves will be considered as the same. But it is obvious that they have different value distributions since the three curves have different shapes. User 1 and User 2 are more likely to visit the category in the evening, whereas User 3 is more likely to visit the category in the early morning. Considering probability values and their distributions, the curve for User 1 and curve for User 2 is more similar, whereas the curve for User 1 and curve for User 3 is less similar. Therefore, based on the check-in behaviors at the category, User 1 and User 2 should be considered as similar users while User 1 and User 3 should not be considered as similar users. The example illustrates that the distribution of values can have impact on the similarity measurement between temporal curves.

4.2.2 Curve Coupling

Since the distribution of probability values can have impact on the similarity measurement between two user temporal curves. A method is proposed based on coupling to measure the distance between user temporal curves by exploiting values and distributions.

The coupling or pairing provides a solution to align two sequences of values. The coupling results contain representative values chosen from both sequences. When calculating the similarity between two user temporal curves, the sequential constraints

need to be satisfied. This chapter proposes a coupling method to represent the similarity with a sequence of matched pairs.

Definition 4.2: A ***Curve Coupling*** between two user temporal curves U and V for category j, denoted as $C(U^j, V^j)$, is a sequence

$$(u^j_{a_1}, v^j_{b_1}), (u^j_{a_2}, v^j_{b_2}), \ldots, (u^j_{a_n}, v^j_{b_n})$$

of distinct pairs from $u^j \times v^j$ such that $a_1 \geq 1$, $b_1 \geq 1$, $a_n \leq 24$, $b_n \leq 24$, and for all $i = 1, \ldots, n$ we have $a_{i-1} < a_i$, and $b_{i-1} < b_i$, where $u^j_{a_1}$ is the check-in probability for category j at hour a_1; $n = |C(U^j, V^j)|$ denotes the number of matched pairs for category j.

The curve coupling selects the most representative values from two user temporal curves and forms a list of matched pairs. The sequential constraints are satisfied in the process by setting the condition on the sequence order. More than one curve coupling result can be produced between two user temporal curves. Here, we give the definition of the best curve coupling based on the number of matched pairs and aggregate distance.

Definition 4.3: A ***Best Curve Coupling*** between two user temporal curves U and V for category j, denoted as $MC(U^j, V^j)$, is a curve coupling that satisfies following conditions:

(1) $argmax(|C(U^j, V^j)|)$

(2) $argmin(\sum_{(p,q)\in C(U^j,V^j)} dist(p,q))$

where U^j and V^j are two user temporal curves for category j; $C(U^j, V^j)$ is a curve coupling between U^j and V^j; (p,q) is an element of $C(U^j, V^j)$; *dist* is a symmetric distance function.

The first condition makes sure that the number of matched pairs is maximized. The second condition makes sure that the total distance between each pair is minimized. In this chapter, the distance is termed as the global total distance of the matched pairs.

Note that it is a NP-hard problem to find the best curve coupling. To solve this problem, a heuristic method is proposed. The detailed steps are listed as follows. Firstly, a starting hour from one of the curves is chosen as the starting point for matching. Here, we start the matching from the hour which has the maximum probability value. To make sure that the similarity between two curves is symmetric, the coupling is started from the curve which has a larger peak value. Secondly, a list of candidate coupling results is obtained. Finally, the best coupling result is selected based on the definition. Figure 4-3 shows an example of the best coupling result between two temporal curves U and V. The matched pairs are connected by lines. Because the peak value from curve V is larger than U, the matching is started from V.

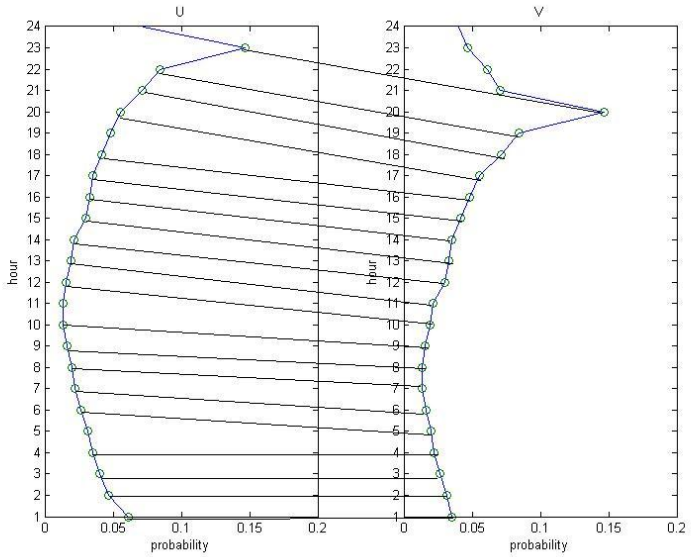


Figure 4-3 An example of the best coupling result between two curves

Each coupling result contains a list of matched pairs that satisfies the sequential constraints. The hour in a user temporal curve can be considered as a sequence number. Once a matched pair is selected, the coupling process can only go in one direction with increasing or decreasing sequence number. We call the coupling with sequence increment as upward uni-directional coupling. Similarly, we call the coupling with sequence decrement as downward uni-directional coupling. In order to organize all possible consecutive pairs which satisfy the sequential constraints, two coupling trees are constructed. It is straightforward that each path of the tree from the root node to a leaf node represents a candidate uni-directional coupling result, which is a sequence of matched pairs. Each node in the tree represents an element of a coupling result, which is a pair of values. The root node is the first matched pair, while the leaf node is the last matched pair in a sequence. The order of each instance will be kept in the coupling process. The combination of downward and upward uni-directional coupling trees forms a final bi-directional coupling result. The advantage of constructing two coupling trees is that the coupling could start with an arbitrary sequence number in a user temporal curve. And the coupling process can run concurrently in two directions.

Figure 4-4 outlines the framework of the heuristic for finding the best curve coupling. Four input arguments are provided: the U and V are the two user temporal curves for coupling; the j is the category; the *numCandidates* controls how many candidate coupling results will be acquired for comparison. In lines 1 to 7, the algorithm first chooses the starting point for the matching. The maximum values from the two curves are compared. The starting point is selected from the curve that has larger maximum value. In lines 9 to 17, several candidate curve coupling results are obtained by

building coupling trees. Each coupling result contains the list of matched pairs from an upward coupling tree and a downward coupling tree. Lines 20 to 31 describe how to construct a downward coupling tree while lines 32 to 43 describe how to construct an upward coupling tree. Finally, the best curve coupling result is selected from the candidates according to Definition (4.3).

```
Algorithm BestCurveCoupling (U, V, j, numCandidates)
// U is the user temporal curve for user u
// V is the user temporal curve for user v
// j is the location category for calculating the coupling.
// numbCandidates is the maximum number of candidate coupling results

1- P = the sequence of values of U for category j;
2- Q = the sequence of values of V for category j;
3- if(peak value in V is larger than U):
4-    P = the sequence of values of V for category j;
5-    Q = the sequence of values of U for category j;
6- Choose the starting point from P;
7- StartingPairs = get numCandidates starting pairs from (P,Q);
8- CandidateList = new List<CouplingResult>;
9- for each pair(p,q ∈StartingPairs do
10-     UpTree = new UpTree;
11-     DownTree = new DownTree;
12-     // construct a tree with upward sequence
13-     FindCandidateUp(k,root(UpTree), p, q, seq(p)+1);
14-     // construct a tree with downward sequence
15-     FindCandidateDown(k,root(DownTree), p, q, seq(p)-1);
16-     CouplingResult = list of pairs from UpTree and DownTree;
17-     add CouplingResult to CandidateList;
18- FinalResult = the CouplingResult from CandidateList according to Def. 4.3;
19- return FinalResult;

20- FindCandidateDown(k, curNode, p, q, nextSeq)
21-      if nextSeq equals the minimum sequence
22-        return;
23-      topK = find top k values from Q that have least distance from p;
24-      if topK is not empty
25-         for each tq∈topK do
26-           childNode = form a new tree node using p and tq;
27-           add childNode as the child node of the curNode;
28-           FindCandidateDown(k,childNode,p,tq,seq(p)-1);
29-      else
30-         // ignore this level, go to next level
31-         FindCandidateDown(k,curNode,p,q,nextSeq-1);

32- FindCandidateUp(k, curNode, p, q, nextSeq)
33-     if nextSeq equals the maximum sequence
34-       return;
35-      topK = find top k values from Q that have least distance from p;
36-     if topK is not empty
37-        for each tq∈topK do
38-           childNode = form a new tree node using p and tq;
39-           add childNode as the child node of the curNode;
40-           FindCandidateUp(k,childNode,p,tq,seq(p)+1);
41-      else
42-         // ignore this level, go to next level
43-         FindCandidateUp(k,curNode,p,q,nextSeq+1);

44- root (Tree)
45-      return the root node of the Tree;

46- seq (q)
47-     return the sequence of q;
```

Figure 4-4 A heuristic for finding the best curve coupling

4.2.3 Temporal Similarity

After the coupling result is acquired based on best curve coupling, we can define the distance measurement between two user temporal curves, which will further be transformed into similarity.

Definition 4.4: Given two user temporal curves *U* and *V* for category *j*, and a best curve coupling between *U* and *V* for category *j*, the A***verage Coupling Distance*** *cdist(*U^j,V^j*)* between *U* and *V* for category *j* is calculated as

$$cdist(U^j, V^j) = \frac{\sum_{(p,q)\in MC(U^j,V^j)} dist(p,q)}{|MC(U^j, V^j)|} \tag{4.2}$$

where U^j and V^j are two user temporal curves for category *j*; *dist* is a symmetric distance function; and $MC(U^j, V^j)$ is the best curve coupling between U^j and V^j.

The average coupling distance is the global total distance divided the number of matched pairs. The smaller the distance between two user temporal curves is, the more similar they will be. Since a user temporal curve represents the check-in behavior of a user for a category, by combining all the categories a user has visited, each user can be represented by a list of temporal curves. We can measure the similarity between two users by calculating the distance between the temporal curves belonging to them.

Definition 4.5: The ***Temporal Similarity*** *tsim(u,v)* between two users u and v is calculated as

$$tsim(\mathrm{u},\mathrm{v}) = \frac{\sum_{j \in C} (1 - cdist'(U^j, V^j))}{|C|} \quad (4.3)$$

where C is a set of categories visited by both users u and v; U^j and V^j are two user temporal curves for category j; $cdist'(U^j,V^j)$ is the normalized average coupling distance between U^j and V^j.

In order to have a similarity value ranging from 0 to 1, the average coupling distance is normalized into the range of 0 and 1 first. If two users have visited some same categories, they share common interests in some way. The coupling distance is used to differentiate the periodic behavior between two users for a category. The temporal similarity between two users is calculated as considering the average coupling distance between the temporal curves for all common categories. If the similarity value is larger, that means the two users are more similar in terms of periodic check-in behavior.

4.3 Probabilistic Category-based Location Recommendation Utilizing Temporal Influence and Geographical Influence

In this section, a new location recommendation algorithm called Probabilistic Category-based Location Recommendation Utilizing Temporal Influence and Geographical Influence (*sPCLR*) is proposed. *sPCLR* combines the temporal influence of similar users and geographical influence of locations to improve location recommendation.

4.3.1 Temporal Influence

In a recommender system, collaborative filtering approach can infer a user's implicit preference by aggregating the behaviors of similar users. In the research, we assume that

users who have similar temporal check-in patterns will have influence on each other's choice towards visiting a location. The temporal curves are used to represent a user's periodic check-in behaviors at different categories. If two users have similar temporal curves, that means they might share a lot of common interests, and have correlated check-in behaviors. For example, User 1 always visits coffee shop at 8PM at night, User 2 always visits coffee shop at 9PM at night, and User 3 usually visits coffee shop at 11AM in the morning. When we calculate the temporal similarity between two users, the temporal similarity between User 1 and User 2 is larger, while the temporal similarity between User 1 and User 3 is smaller. One user's check-in behavior and preference might provide good recommendations for his similar users due to their potential common interests.

Because each temporal curve has a sequence of values, traditional similarity measure such as cosine similarity or Pearson correlation cannot be applied directly to temporal curves. The temporal similarity is used for measuring the similarity between users in terms of the periodic check-in behavior. After similar users are obtained according to temporal similarity, we can predict the periodic behavior of a given user by a weighted summation of the periodic behaviors of his similar users. We can predict the check-in probability of a user *u* visiting a category *c* at time *t* by following equation

$$\hat{T}(u,c|t) = \frac{\sum_{v \in U_c} tsim(u,v) * T(v,c|t)}{\sum_{v \in U_c} tsim(u,v)} \tag{4.4}$$

where $T(v,c|t)$ denotes the probability of user v checking in a location of category c at time t; U_c denotes the set of users who have visited category c; and $tsim(u,v)$ denotes the temporal similarity between user u and v.

Users who have visited the same category show similar taste towards a location category, and they might influence the check-in behavior of each other. The temporal similarity is served as the weight for the impact of the similar user's behavior on the active user. If two users are more similar in terms of temporal similarity, they will have more influence on each other's periodic check-in behavior.

4.3.2 Geographical Influence

The check-in activities of users record their interactions with locations, and the geographical proximities of locations will influence a user's check-in behavior. It is observed that a user tends to visit locations closer to their homes or offices.

Figure 4-5 shows a logarithmic scale plot of the check-in frequency to the distance from user's home location. This example is based on the real-world check-in data acquired from Gowalla, one of the popular LBSNs. When there is a large range of quantities, logarithmic scale makes it easy to compare values which cover a large range. It is observed that the plot can be separated into two parts by the 50km point. When the distance from home is greater than 50km, the check-in frequency varies randomly. When the distance from home is less than 50km, there is a relationship between the check-in frequency and the distance to user’s home. When the distance of the location to a user's home increases, the user has less probability to visit that location. Our explanation is that 50km is the human reaching distance for the dataset. When the distance of the location to user’s home is within the human reaching distance, there is potential relationship between

the check-in probability and distance to user's home. On the other hand, when the distance of the location to user's home is outside the human reaching distance, the user usually tends not to visit that location. Only when the user is on a trip, he will visit a location outside the human reaching distance.

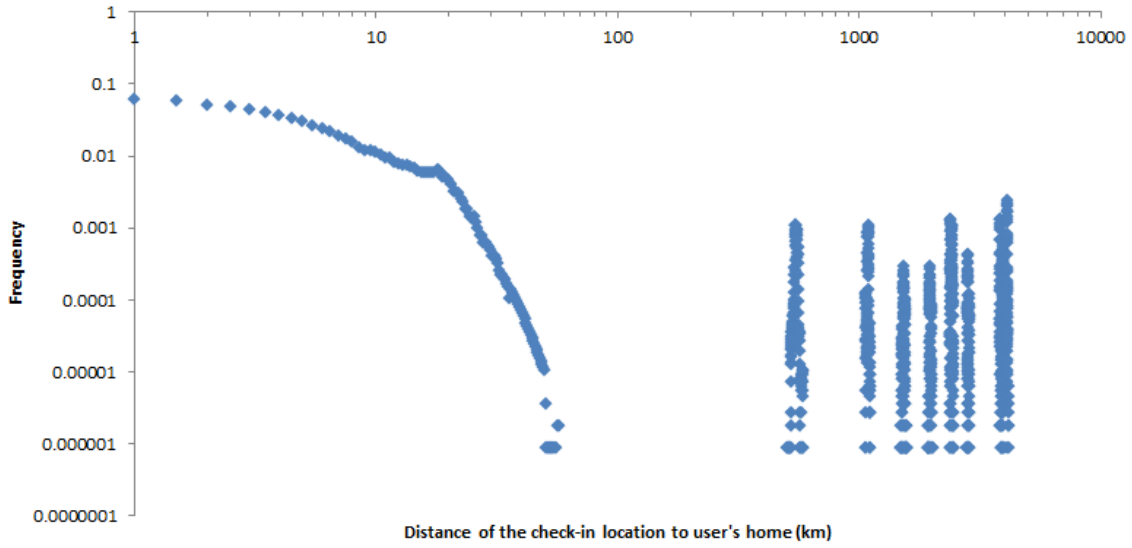


Figure 4-5 Logarithmic scale plot of the check-in frequency to the distance from user's home

The home locations of users are usually not given in the check-in dataset. They can be calculated based on the assumption that locations of check-ins are centered at the user's home location. To find the home location, we first divide the earth surface into small non-overlapping regions and group the check-ins based on the regions. The region with the maximum number of check-ins is considered as the one containing the user's home location. The average location of the check-ins inside the region is selected as the user's home location.

After the home location is defined for each user, the relationship between check-in frequency and distance from home can be analyzed. Based on the observation that the user tends not to visit a location that is farther than the human reaching distance, we can infer a user's check-in probability to a location by utilizing the geographical relationship. A spatial probability function (*SP*) for a check-in dataset is defined as follows:

$$SP(l;h) = \begin{cases} 1\,, & distance(l,h) \leq R \\ 0\,, & distance(l,h) > R \end{cases} \tag{4.5}$$

where l is the location for which we want to find the probability of check-in; h is the home location of the user; *distance*(l,h) is the distance from the location to the user's home; and R is the human reaching distance based on the check-in dataset.

Equation (4.5) models the relationship between a user's check-in probability and the distance of the location to user's home based on the geographical influence existing within the check-in dataset. It is used to indicate the intention that a user has towards a certain location. If the distance of the location to the user's home is larger than the human reaching distance, the user will not consider choosing this location. The human reaching distance can be obtained from the plot of check-in frequency to the distance from user's home. The main purpose of the spatial probability function is to filter out those locations that are not of interest to the user. If a location is far away from the user's home, it should not be recommended as a suggestion in location recommendation. The spatial probability function will be further used as the spatial component of our location recommendation algorithm.

4.3.3 Probabilistic Category-based Location Recommender Utilizing Temporal Influence and Geographical Influence

This subsection proposes a new location recommendation algorithm called Probabilistic Category-based Location Recommender Utilizing Temporal Influence and Geographical Influence (*sPCLR*). *sPCLR* improves the location recommendation by considering both temporal and spatial components within users' check-in behaviors.

The temporal component utilizes the temporal influence of users' check-in behaviors. It models a user's probability of checking in to a location by considering similar users' check-in probability. The similarity of periodic check-in behaviors is calculated by considering the difference of temporal curves using a curve coupling method. Temporal curves represent a user's periodic check-in behavior at different location categories.

The spatial component utilizes the geographical influence of locations. It models a user's probability of checking in to a location by considering the distance from the location to the user's home. If the location is closer to the user's home, it is more likely to be visited by the user. If a location is far away from the user's home, that location should not be recommended. The spatial component filters out those locations that are not of interest to the user.

By combing the temporal and spatial components, the probability of a user *u* checking in to a location *l* at the given time *t* is defined as:

$$P(u, l|t) = SP(l; h_u) * \hat{T}(u, c_l|t) \tag{4.6}$$

where h_u is the home location of user u and c_l is the category of location l; SP is the spatial probability of visiting the location l given the home location of the user; and $\hat{T}(u, c_l|t)$ is the probability of checking in to the category of location l at given time t based on temporal similarity.

Figure 4-6 shows the pseudo code for the *sPCLR* location recommendation algorithm. It accepts three input parameters 1) The user to whom we are going to make recommendation, 2) the time for the recommendation, and 3) the number of candidate locations. It first calculates the user's check-in probability for every location for the specified time slice using the formula in Equation (4.6). Then it returns the top-k candidate locations to the user as recommendations in terms of the check-in probability.

```
Algorithm sPCLRecommender (u, t, k)
// u is the user to whom we are going to give recommendation.
// t is the time for which we are going to give recommendation.
// k is the number of locations we want to be recommended to the user

1-   for each (l in locations) do
2-      find probability p (u, l| t) using formula in Equation (4.6);
3-      Add l and corresponding probability to a priority queue of locations;
4-   end for
5-   return k-top locations from the priority queue of locations;
```

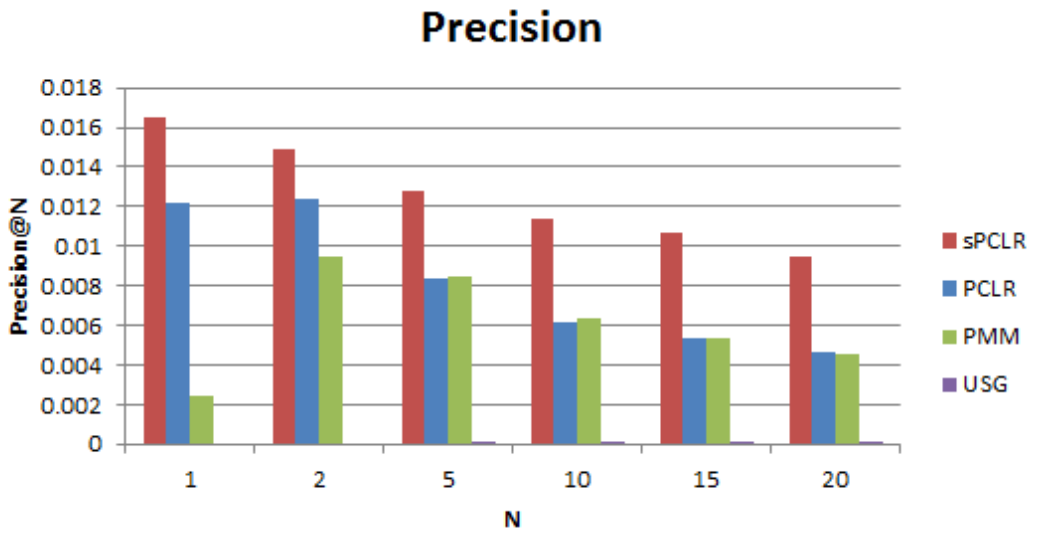
Figure 4-6 Pseudo code for the *sPCLR* location recommendation algorithm

4.4 Experiments

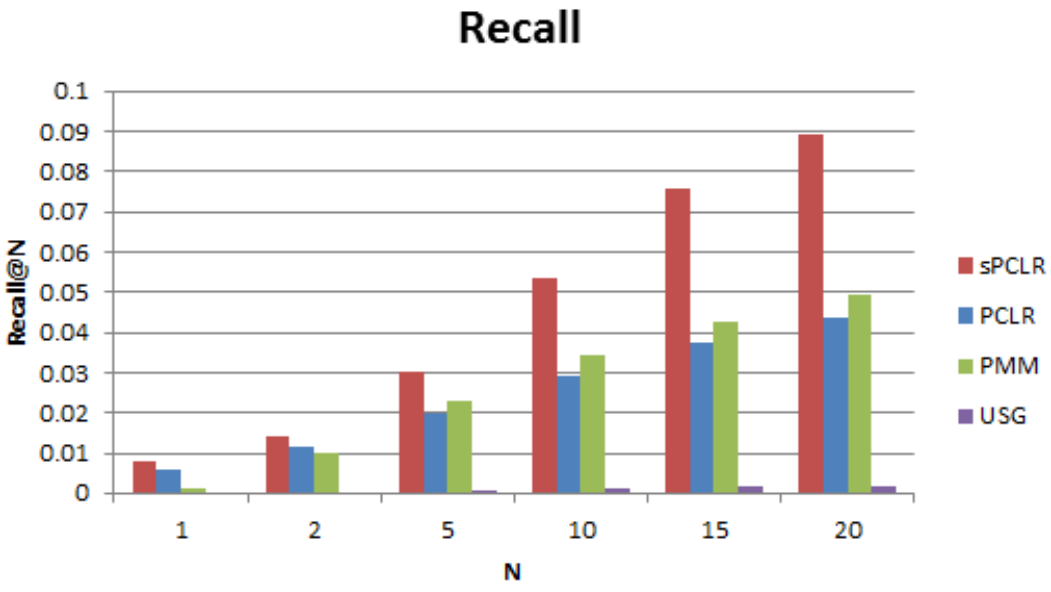
In order to evaluate the performance of the proposed location recommendation algorithm, 5417 users are randomly selected from the dataset. It contains 40242 check-in records in total for the selected users. The check-in data from Gowalla contains 225 categories. Each category is a word that represents the characteristics of the location such as coffee shop, restaurant etc. The data is divided into training and testing datasets. To do so, we randomly pick one of the check-in records of each user to form the testing dataset. The remaining records form the training dataset. That means the testing dataset contains 5417 check-in records, and the training dataset contains 34825 check-in records. We randomly generated 5 groups of different training and testing datasets and run experiments on them. The average performance of the five runs is reported as the final performance. Five computers with 12GB of RAM and 3.2 GHz CPU are used for calculating the temporal similarity in order to reduce the overall time.

The performance of the *sPCLR* location recommendation algorithm is evaluated using precision and recall. It is compared with three existing location recommenders: 1) Probabilistic Category-based Location Recommender (PCLR) proposed by Rahimi and Wang (2013), 2) Periodic Mobility Model (PMM) proposed by Cho et al. (2011) and 3) USG model proposed by Ye et al. (2011). They are selected because the models used by them are most related to the proposed location recommender. Each recommender is trained based on the corresponding check-in records from the training dataset. Then the performance is reported by recommending top-N locations to the users in the testing set. Different N values are reported: 1,2,5,10,15 and 20. The human reaching distance for

sPCLR is set to 50km. Figure 4-7 shows the precision and recall results for different recommendation algorithms.



(a) Precision for sPCLR, PCLR, PMM and USG



(b) Recall for sPCLR, PCLR, PMM and USG

Figure 4-7 Performance comparison for location recommendation algorithms

From Figure 4-7, we can see that sPCLR recommendation algorithm performs better than all other algorithms in terms of both precision and recall values. Although both sPCLR and PCLR consider a similar geographical influence model, sPCLR performs better than PCLR. This shows that the temporal influence of users' periodic check-in behavior based on category information improves the location recommendation. PCLR models the users' periodic check-in behavior based on the temporal probability distribution function. sPCLR utilizes the temporal probability distribution function to form temporal curves as a way of representing the periodic check-in behavior. It measures the temporal similarity between users based on temporal curves. Then a user's periodic check-in preference towards a certain location is predicted by his/her similar users' preferences using a collaborative filtering approach. Because PCLR predicts a user's periodic check-in preference based only on the user's past visiting history, it might fail to suggest some potential locations if the user only visited a few locations before. On the contrary, sPCLR predicts a user's periodic check-in preference based on his/her similar users. If the user has not visited a location but his/her similar users have visited that location, the location can still be suggested as a potential location. By broadening the potential candidate suggestions, sPCLR performs better than PCLR. The performance result also shows that temporal curves are a valid way for representing the periodic check-in behavior. We also observe that PMM and PCLR have similar performance and both of them outperform USG. This could be because that, different from USG, PMM and PCLR take into account the periodic model of human movements within the check-in behaviors.

4.5 Summary

Location recommendation provides suggestions of unvisited locations to the users for the rapidly growing location-based social networks. The service is based on the users' visiting histories and location related information such as location categories. In this chapter, a location recommendation algorithm called sPCLR is proposed. It makes suggestions of locations to the users at a given time of the day by utilizing location category information. The algorithm considers both temporal and spatial components. The temporal component utilizes the temporal influence of similar users' check-in behaviors. Temporal curves are extracted from location category information to represent a user's periodic check-in behaviors at different location categories. Based on the difference between temporal curves, temporal similarity is introduced to measure the similarity of users' periodic check-in behaviors. According to the user's similar users in terms of temporal similarity, a temporal influence model makes prediction of a user's periodic check-in behaviors for different locations. The spatial component utilizes the geographical influence of locations and filters out those locations that are not of interest to the user. The performance of sPCLR is compared with three existing location recommendation algorithms on a real-world dataset. Experimental results show that the sPCLR algorithm performs better than all other three algorithms.

Chapter Five: **Conclusions and Future Work**

This chapter draws conclusions from this thesis and provides suggestions for future work.

5.1 Conclusions

This thesis investigates location recommendation on LBSNs utilizing check-in data and location category information.

The first part of the thesis studies how to recommend locations to users on LBSNs by CF recommenders based only on the user check-in data without any domain knowledge. In order to conduct a comprehensive evaluation of recommenders, a distributed crawler is designed to obtain a large quantity of check-in data in four major USA cities from Gowalla. Based on the crawled real-world data, three ways are introduced to utilize the check-in data, namely, the binary utilization, the FIF utilization, and the probability utilization. Then, three different CF recommenders are designed to combine with those three kinds of check-in utilizations, namely, the user-based recommender, the item-based recommender and PLSA recommender. To compare the performances of different combinations of recommenders and check-in utilizations, a set of experiments have been conducted on the check-in dataset. The experimental results show that the PLSA recommender with the probability utilization performs the best.

The second part of the thesis proposes a location recommendation algorithm called sPCLR. It provides suggestions of locations to the users at a given time of the day by utilizing location category information. The algorithm consists of temporal and spatial components. The temporal component uses the temporal similarity to find out users that

have similar periodic check-in behaviors. A collaborative filtering approach is used to predict a user's preference towards a location by the preferences of his/her similar users. The spatial component filters out those locations that the user tends not to visit. A set of experiments are conducted to compare the performance of sPCLR with three existing location recommendation algorithms on a real-world dataset. Experimental results show that the sPCLR location recommendation algorithm performs better than all other algorithms in terms of precision and recall.

To put it in a nutshell, the contributions of this thesis are:

1- A crawler is designed to obtain a large portion of real world check-in data from Gowalla. The collected dataset is used for evaluating the performances of different location recommendation algorithms in the research.

2- An empirical study on the different utilizations of check-in data for location recommendation on LBSN is carried out. Three different kinds of utilizations are used to infer a user's check-in preference towards locations: binary utilization, *FIF* (Frequency - Inverse Frequency) utilization, and probability utilization. According to the experiments, the probability utilization performs the best.

3- Location category information is investigated to discover users' temporal patterns. Temporal curves are introduced to represent users' periodic check-in behaviors for different categories. A coupling method is proposed to measure the difference between two temporal curves. The temporal similarity between two users in terms of periodic check-in behaviors is calculated based on temporal curves. According to temporal similarity, a temporal influence model is built to predict the periodic

check-in behavior for a given user by considering the periodic behaviors of his/her similar users.

4- A new location recommendation algorithm called *sPCLR* is proposed to suggest unvisited locations to users. It combines the temporal influence of similar users and geographical influence of locations. The temporal influence model utilizes a collaborative filtering approach to make predictions. The geographical influence model predicts the probability of a user visiting a location by considering the distance of that location to user's home.

5- According to a set of experiments conducted on the real-world dataset, the performance of *sPCLR* outperforms three existing location recommendation algorithms, namely PCLR, PMM and USG.

5.2 Future Work

Several improvements and extensions to this thesis are listed as follows:

1- Evaluating the models on larger check-in datasets. Currently we only used one check-in dataset, Gowalla. The dataset is relatively small in terms of the number of users and check-ins.

2- Integrating the domain knowledge such as social ties between users to improve the modeling of temporal influence. Friends tend to have similar behaviors in terms of check-in activity because they might share a lot of common interests. For example, two friends may hang out and drink a coffee together sometimes, or a user might go to watch a movie recommended by his friends. The social connections can be exploited in location recommendation.

3- Investigating other performance measurements such as the running time of the location recommendation algorithms. In order to provide effective and efficient suggestions to the users, the running time of the algorithms is an important feature for the recommender system too. The time complexity of the curve coupling algorithm will be analyzed.

Appendix A. DATASET

The experiments of this thesis are based on a check-in dataset collected from Gowalla, an online location-based social network. The dataset contains information such as locations, users and check-ins. This appendix includes sample entries of these tables. Table A-1 shows some sample users from the Gowalla dataset. Personal information such as user name and Facebook ids are hidden because of privacy protection issues. Table A-2 shows some sample locations from the Gowalla dataset. Table A-3 shows some sample check-ins from the Gowalla dataset. Each check-in record means that one user visits one location at a certain time.

Table A-1 Sample users from the Gowalla dataset.

Id	First name	Last name	Hometown	Friends count	Photos count	Facebook id	Twitter id	Twitter username
1	?	?	Austin, Texas	385	137	?	?	?
2	?	?	Austin, Texas	792	1124	?	?	?
3	?	?	Austin, Texas	85	201	?	?	?
5	?	?	Austin	531	1042	?	?	?

Table A-2 Sample locations from the Gowalla dataset

id	Name	Checkins count	Created at	Lat	Lng	Category	City	Region	Country
8938	Broadway Cafe	498	2008-12-15T00:22:49Z	39.05282	-94.5903	2	Austin	TX	US
8964	Latte Land	557	2008-12-23T22:42:59Z	39.04105	-94.5947	1	Kansas City	MO	US
8972	Chipotle	157	2009-01-08T18:07:58Z	39.05722	-94.6056	363	Kansas City	MO	US
8988	Thai Chili	191	2009-01-15T00:46:27Z	32.94296	-97.1306	926	Austin	TX	US
8990	Chipotle	251	2009-01-16T00:40:56Z	32.85315	-97.1899	21	Austin	TX	US

Table A-3 Sample check-ins from the Gowalla dataset

id	User id	Location id	Created at	Photos count	Comments count
88085	120	11441	2009-09-26T17:22:51+00:00	0	0
89415	2938	30106	2009-09-27T00:43:47+00:00	0	0
110000	515	34756	2009-10-03T18:04:22+00:00	0	0
111805	6716	87258	2009-10-04T00:28:48+00:00	0	0
112510	4696	36374	2009-10-04T04:48:47+00:00	0	0

Appendix B. PUBLICATIONS DURING THE PROGRAM

Conference Papers:

Zhou D., Wang B., Rahimi S.M., and Wang X. (2012). A Study of Recommending Locations on Location-based Social Network by Collaborative Filtering. *The 25th Canadian Conference on Artificial Intelligence* (CAI 2012). Toronto, ON, Canada, May 28 - 30, 2012, pp.255- 266

Wang B., Rahimi S. M., **Zhou D.**, and Wang X. (2012). Expectation-Maximization Collaborative Filtering with Explicit and Implicit Feedback. *The 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (PAKDD 2012). Springer LNCS, Kuala Lumpur, Malaysia, May 29 - Jun 1, 2012, pp.604-616

Zhou D., and Wang X. (2014). Probabilistic Category-based Location Recommendation Utilizing Temporal Influence and Geographical Influence. *The 2014 IEEE International Conference on Data Science and Advanced Analytics* (DSAA'2014). Shanghai, China, Oct 30 - Nov 1, 2014 (Accepted)

REFERENCES

Adomavicius G., and Tuzhilin A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-art and Possible Extensions, *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734-749

Balabanovic M., and Shoham Y. (1997). Content-based Collaborative Recommendation. *Communications of the ACM*, vol. 40, no. 3, pp. 66-72

Bao J., Zheng Y., and Mokbel M. (2012). Location-based and Preference-aware Recommendation Using Sparse Geo-Social Networking Data. *In: 20th ACM SIGSPATIAL International Conference on Advances in GIS*. Redondo Beach, California

Beeharee A., and Steed A. (2007). Exploiting Real World Knowledge in Ubiquitous Applications. *Personal and Ubiquitous Computing Archive*, vol. 11, no. 6, pp. 429-437

Berjani B., and Strufe T. (2011). A Recommendation System for Spots in Location-based Online Social Networks. *In Proceedings of the 4th Workshop on Social Network Systems*, pp. 4:1-4:6. Salzburg, Austria

Bobadilla J., Ortega F., Hernando A., and Gutiérrez A. (2013). Recommender Systems Survey. *Knowledge-Based Systems*, vol. 46, pp. 109-132

Burke R. (2002). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, vol. 12, no. 4, pp. 331-370

Cheng Z., Caverlee J., Lee K., and Sui D. (2011). Exploring Millions of Footprints in Location Sharing Services. *In: 5th International Conference on Weblogs and Social Media*, Barcelona,Spain, pp. 81–88

Cho E., Myers S. A., and Leskovec J. (2011). Friendship and Mobility: User Movement In Location-Based Social Networks. *In Proceedings of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. San Diego, California, USA, pp. 1082-1090

Eagle N., and Pentland A. (2009). Eigenbehaviors: Identifying Structure in Routine. *Behavioral Ecology and Sociobiology*, vol. 63, pp. 1057-1066

Herlocker J.L., and Konstan J.A., Borchers A., and Riedl J. (1999). An Algorithmic Framework for Performing Collaborative Filtering. *In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR* '99), ACM, New York, NY, USA, pp. 230-237

Hofmann T. (1999). Probabilistic Latent Semantic Indexing. *In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA, pp. 50-57

Hofmann T. (2004). Latent Semantic Models for Collaborative Filtering. *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 89-115

Hofmann T., Puzicha J., and Jordan M.I. (1999). Unsupervised Learning from Dyadic Data. *In Advances in Neural Information Processing Systems*.

Koren Y., Bell R.M., and Volinsky C. (2009). Matrix Factorization Techniques for Recommender Systems. *IEEE Computer*, vol. 42, no. 8, pp. 30-37

Li Z., Ding B., Han J., Kays R., and Nye P. (2010). Mining Periodic Behaviors for Moving Objects. *In: 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, pp. 1099-1108

Li Z., Wang J., and Han J. (2012). Mining Event Periodicity from Incomplete Observations. *In: 18thACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Beijing, China, pp. 444-452

Liu N.N., Xiang E.W., Zhao M., and Yang Q. (2010). Unifying Explicit and Implicit Feedback for Collaborative Filtering. *In Proceedings of the 19th ACM International Conference on Information and knowledge management* (CIKM '10). ACM, New York, NY, USA, pp. 1445-1448

Melville P., Mooney R.J., and Nagarajan R. (2002). Content-Boosted Collaborative Filtering for Improved Recommendations. *In AAAI/IAAI*, pp. 187–192

Park M.H., Hong J.H. and Cho S.B. (2007). Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices. *In Proceedings of the 4th International Conference on Ubiquitous Intelligence and Computing*, Hong Kong, China, pp. 1130–1139

Pazzani M. J. (1999). A Framework for Collaborative, Content-Based and Demographic Filtering. *Artificial Intelligence Review*, vol. 13, no. 5-6, pp. 393-408

Rahimi S.M., and Wang X. (2013). Location Recommendation Based on Periodicity of Human Movement and Location Categories. *The 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (PAKDD 2013), Gold coast, Australia, 14-17 April 2013

Sarwar B.M., Karypis G., Konstan J.A., and Riedl J. (2001). Item-based Collaborative Filtering Recommendation Algorithms. *In WWW*, pp. 285–295

Saul L., and Pereira F. (1997). Aggregate and Mixed-Order Markov Models for Statistical Language Processing. *In Proceedings of the 2nd International Conference on Empirical Methods in Natural Language Processing*, pp. 81-89

Simon R., and Frőhlich P. (2007). A Mobile Application Framework for the Geospatial Web. *The 16th International Conference on World Wide Web*, Banff, Alberta, Canada, pp. 381–390

Su X., and Khoshgoftaar T. M. (2009). A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*.

Wang J., and Prabhala B. (2012). Periodicity Based Next Place Prediction. *In: Workshop on Mobile Data Challenge by Nokia*, Newcastle, UK

Wang J., de Vries A.P., and Reinders M.J.T. (2006). Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. *In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR* '06), ACM, New York, NY, USA, pp. 501-508

Ye M., Janowicz K., Mülligann C., and Lee W.C. (2011). What You Are Is When You Are: the Temporal Dimension of Feature Types in Location-based Social Networks. *In Proceedings of ACMGIS* 2011, New York, NY, USA, pp. 102–111

Ye M., Ying P., Lee W.C. and Lee D.L. (2011). Exploiting Geographical Influence for Collaborative Point-of-Interest Recommendation, *In Proceedings of the ACM International Conference on Research & Development on Information Retrieval* (SIGIR'11), Beijing, China, pp. 325-344

Zheng V.W., Cao B., Zheng Y., Xie X., and Yang Q. (2010). Collaborative Filtering Meets Mobile Recommendation: A User-centered Approach. *In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI 2010, Atlanta, Georgia, USA

Zheng V.W., Zheng Y., Xie X., and Yang Q. (2010). Collaborative Location and Activity Recommendations with GPS History Data. *The 19th International Conference on World Wide Web*, Raleigh, North Carolina, USA, pp. 1029-1038

Zhou D., Wang B., Rahimi S.M., and Wang X. (2012). A Study of Recommending Locations on Location-based Social Network by Collaborative Filtering. *The 25th Canadian Conference on Artificial Intelligence* (CAI 2012). Toronto, ON, Canada, May 28 - 30, 2012, pp.255-266

Zhou Y., Wilkinson D., Schreiber R., and Pan R. (2008). Large-Scale Parallel Collaborative Filtering for the Netflix Prize. *In Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management* (AAIM '08), Berlin, Heidelberg, pp. 337-348