RECOMMENDER SYSTEMS IN SOCIAL NETWORKING SITES

by

Huayu Li

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Computing and Information Systems

Charlotte

2018

Approved by:

_____
Dr. Yong Ge

_____
Dr. Yu Wang

_____
Dr. Jing Yang

_____
Dr. Wei Fan

ABSTRACT

HUAYU LI. Recommender Systems in Social Networking Sites. (Under the direction of DR. YONG GE)

Social networking sites have experienced an explosion in both the number of users and the amount of shared information in recent years. Thanks to the positioning function in mobile devices, e.g., GPS, location-based social networks (LBSNs) become a prominent representative of social networks. Thus, the rapid development of LBSN services has stimulated the emergence of a new line of research to develop two novel recommender systems that seek to facilitate users' location exploration and social interaction, where we refer them to location and friend recommender systems, respectively. Even with the help of large available user interaction data, it is challenging to produce accurate location recommendation and friend recommendation in LBSNs by the reason of the interdependency between human mobility and social proximity, and the heterogeneity of social link, i.e. connection between user and user, and consumption link, i.e. connection from user to location, in the whole network. To this end, in this dissertation, we propose different approaches for these two types of recommender systems with the characteristics observed in user behaviors to serve LBSNs. For location recommendation, we study human mobility and exploit human movement properties to design recommender systems in spatial, temporal, and social aspects for helping users discover the desired locations. For friend recommendation, a new approach is introduced to maximize the growth of both heterogeneous links in the whole network for helping users find potential friends to connect with. The performance evaluated on real-world datasets demonstrates the ability of our models for recommender systems in LBSNs.

ACKNOWLEDGEMENTS

I would like to thank all the members of my dissertation committee to their consistent support, constructive criticism and valuable instruction in the course of the dissertation.

I would like to express my deepest gratitude to my advisor, Prof. Yong Ge, for his excellent guidance, and persistent support throughout the time of my Ph.D. study. I am very fortunate to work under his supervision. I have benefited tremendously from his feedback, ideas, and criticism on my research, writing, and presentation skills.

Furthermore, I would like to thank UNC Charlotte for providing me with such a wonderful opportunity for my PhD study. The faculties and staffs in UNC Charlotte help me a lot in both course work and life.

Thanks as well to all my friends. Many of them having accompanied me for several years. I am greatly indebted for their affection, their support and encouragement. Moreover, my life would not have been possible without the support of my family. I really appreciate their support and encouragement very much.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Motivation

In the past decade, with the widespread adoption of Web 2.0 technology, social networking sites, such as Facebook and Foursquare, have attracted a huge attention and dramatically reshaped people's social lives [1, 2]. Smart mobile devices have provided the web users with the ability to access online services as they are on the move and, literally, from any place where Internet connectivity is present. For example, until 2016, there are 56.5% Facebook users that only login from the mobile device [1]. Advances in smart mobile devices and localization technologies, have fundamentally enhanced social networking services, allowing users to share their life experiences in the physical world via location sharing, and thus location-based social networks (LBSNs) become prominent representatives of the class of social networks.

In LBSNs, the dimension of location brings social networks back to reality, bridging the gap between the physical world and online social networking services. Thus, LBSN services, such as Facebook Places, Foursquare, Gowalla, and Dianping, are able to facilitate outdoor activities for users by providing nearby locations, e.g., restaurants, stores, cinema theaters, in a real-time fashion, and at the same time allow users to make friends online for sharing check-in experience with each other. A variety of user interaction data with these LBSN services, such as checking-in at locations and building social connection, have been accumulated, which provides unprecedented opportunities for developing both location and friend recommender systems [1, 2, 3, 4]. Indeed, it becomes crucial to have a personalized recommender system that is able to learn user preferences, and based on these preferences, to automatically filter irrel-

[1] https://expandedramblings.com/index.php/facebook-mobile-app-statistics/

evant information (i.e. friends or locations in LBSNs) or suggest useful information to this user in a timely manner. With limited human attention, finding relevant information and knowledge from the huge amount of available information is usually frustrating and extremely time-consuming. The suggestions provided by a recommender system are aimed at efficiently and effectively supporting users in various decision-making processes, such as what locations to check-in, or what friends to make, for enabling them to quickly locate desirable items without being overwhelmed by irrelevant information.

To this end, in this dissertation, we provide a new family of recommender systems with location recommendation and friend recommendation aimed at serving LBSNs [2]. Location recommender systems help users find their preferred locations, and friend recommender systems help users find potential friends to connect with in social networking sites. However, location and friend recommendations in LBSNs are extremely challenging problems, and we list here several motivating challenges:

- *Implicit-feedback.* Traditional recommender systems, e.g., movie recommendation, allow users to provide an explicit rating for an item to express their interests. A higher (or lower) rating explicitly indicates that the user likes (or dislikes) the item more. Nevertheless, in LBSNs, such explicit feedback is not always available, where the user's opinion is indirectly reflected through the observed user behaviors, such as check-in behavior. For example, one user is able to realize that checking-in a certain location is a poor experience only after visiting this location, therefore just knowing a user has visited a location does not necessarily imply that she likes it.

- *Complex nature of decision making process.* The user's movement is generally influenced by many different kinds of factors, such as geographical location, social friends, and time. For example, a user usually likes to prefer a nearby

[2] We also refer location to point-of-interest (POI).

location rather than another one far away due to short distance. Furthermore, friends have an important role in decision making process for users associated with location choice. Some of the friends have positive impact on a user's check-in at a particular location, while others may influence negatively. Even with the same group of friends, two users may be affected by these friends in different ways. Additionally, the user's interest may change over time. For example, one user likely chooses a restaurant next to her work place for convenience in week days, but she possibly goes to a bar for entertainment at weekends. These factors present another critical challenge for developing recommender systems in LBSNs.

- *Cold-start Problem.* When a new item or a new user enters the system, we do not have their historical interaction information. There is no training data for such user or item, definitely leading to the failure of machine learning models. Thus, the insufficient interaction information prevents producing reasonable recommendation for cold start users and items.

- *Heterogeneity.* There are two types of links in LBSNs: social link, connecting user and user, and consumption link which is an interaction between user and location and also refers to the check-in behavior. It is hard to have users build multiple social connections and check-in at many locations at the same time. In reality, the number of online recommended friends is limited for each individual. It is impossible to display all potential locations and friends for users together for the sake of the limited size of device screen, which as a result increases the difficulty of the growth of these two heterogeneous links. For example, a friend recommender system is capable of providing each individual with a group of users who she might be interested in. But it cannot guarantee that she would have more chances to check-in locations as well. Thus, how to enable these two

types of links to grow fast together poses the key challenge to recommender systems in LBSNs.

1.2 Contribution

In this dissertation, we focus our attention on addressing the challenges described above for recommender systems with location recommendation and friend recommendation in LBSNs. To summarize, the contributions of our research work are as follows, which will be elaborated in each chapter:

- Propose a novel ranking-based model for general recommender systems with implicit feedback datasets, which can be practically applied to both location recommendation and friend recommendation in LBSNs, by relaxing pairwise ranking into a SVM-like task where positive and negative feedbacks are separated by the soft boundaries, and design a group- and instance-level based algorithm to ease the sampling bias for optimization process. [5].

- Propose different matrix factorization models for location recommender systems in the context of LBSNs, and model geographical influence, social influence, and temporal effect observed in human movement for solving cold-start problem and producing more accurate recommendation [6, 2, 1, 4].

- Propose a bi-utility based approach for friend recommender systems to enable the heterogeneous links in LBSNs to grow faster together, and produce friend recommendation by maximizing the bi-utility of each social link, i.e. the summation over corresponding social utility and potential consumption utility, which helps users build more social connections and explore more locations at the same time.

1.3 Organization

The remainder of this dissertation is organized as follows. We first discuss prior research on recommendation techniques and give a brief literature review in Chap-

ter 2. From Chapter 3 to Chapter 5, we study different recommender systems, i.e., location recommendation and friend recommendation, in social networking sites. In Chapter 3, we introduce a ranking-based method for general recommender systems with implicit feedback datasets which can be practically applied to location and friend recommendation. In Chapter 4, we present different matrix factorization-based methods for location recommender systems by exploiting various characteristics observed in human movement. In Chapter 5, a new bi-utility based approach is elaborated to maximize the growth of heterogeneous links in LBSNs for friend recommendation systems. We finally conclude the dissertation in Chapter 6.

CHAPTER 2: LITERATURE REVIEW

Recommender systems have gained increasing popularity and attention in recent years, and are utilized in a variety of areas, e.g., movie recommendation on NetFlix, job recommendation on Linkedin, product recommendation on Amazon, and news recommendation on Yahoo [7, 8, 9, 10, 11]. POI recommendation [12, 13] and friend recommendation have become the essential tasks in location-based social networking services for facilitating human life experience, which belong to sub-categories of recommender systems. Thus, techniques of general recommender systems are also practically applicable to both location and friend recommendation, although the performance may be limited due to the specific characteristics of human mobility and social structure in location-based social networking services. In the following section, we first provide a literature review on general recommender systems, and then review approaches of POI recommendation and friend recommendation.

2.1 General Recommender Systems

In this section we briefly review some existing work on recommender systems. Existing recommendation models have mostly focused on minimizing a uniform type of recommendation recovery loss to predict the recommendation scores. Based on the type of losses, recommender systems can be grouped into three categories: point-wise methods, pairwise methods, and listwise methods, where their related work is summarized in Table 2.1.

Pointwise method, also referred to rating prediction-based method, optimizes a loss function defined on *absolute* preference score from the user by minimizing the point-wise (entry-wise) divergence of the reconstructed recommendation matrix from

the original observation matrix. With the observed explicit ratings, i.e. observed entries in matrix, probabilistic matrix factorization (PMF) minimizes the point-wise difference in terms of ℓ_2 norm between the reconstructed matrix produced by the inner product of the user and item latent representations and the input training matrix [14, 15]. In addition to exploiting latent factor model, SVD/SVD+ [16, 17] also capitalizes the advantage of neighborhood method to estimate the preference score of the user on the item for the reconstructed matrix. Factorization Machines (FM) [18, 19] is a general rating predictor by modeling all interactions between variables using factorized parameters and capable of incorporating various context information. In some other recommendation scenarios, e.g., browsing history, clicking history, purchase history, and check-in behavior, the user's true preference scores to items are not available. To address such implicit feedback issue, a weighted regularized matrix factorization (WRMF) model has been developed in [20] to recover a binary matrix by optimizing the loss for observed entries with large weights while fitting the unobserved ones with small weights. Similar to this idea, one-class collaborative filtering (OCCF) [21] only minimizes the loss by partial matrix with sampled negative examples instead of the whole matrix to improve the efficiency.

Paiwise method optimizes a loss function defined on *relative pairwise* preferences from the user by minimizing the preference structure inconsistency between the reconstructed recommendation matrix and the original matrix [22, 23, 24, 25, 26, 27]. The loss is low when the predicted order is correct, or in other words, when the more relevant item in a pair is predicted to have a higher score than the less relevant item. Bayesian personalized ranking (BPR) [28] optimizes pairwise ranking loss for the user's preference score to the observed item over the unobserved one, which actually is equivalent to maximize area under the ROC curve (AUC). The ranking loss is approximated by a smooth logistic function, and the optimization problem is solved by uniformly sampling negative example with stochastic gradient descent (SGD). To

improve the convergence rate, advanced sampling strategy [29] is designed to oversample the top ranked negative items. Instead of directly optimizing the loss of strict ranking order, RankALS [30] optimizes the relative ranking order using ℓ_2 norm for minimizing the difference between all pairs of the reconstructed relative orders and the observed true relationship. In addition, RankNet [31] employs cross-entropy loss to optimize pairwise ranking with neural networks. To care more about the top of the ranking than the bottom in case of capacity waste in improving the order of preference score at low (poor) ranks at the expense of those at the top of the ranking, LambdaRank [32] further manipulates the gradients by simply scaling with the size of the change in Normalized Discounted Cumulative Gain (NDCG).

Listwise method optimizes a loss function defined on *relative listwise* preferences from the user by minimizing the inconsistency between the predicted list and the groundtruth list [33]. In other words, it learns to model ranked list, or to model evaluation metrics that capture the quality of ranked list. Listwise methods can be classified into two groups. The first one *directly* optimizes the information retrieval (IR) measures, such as NDCG [34, 35], Mean Average Precision (MAP) [36], Mean Reciprocal Rank (MRR) [37, 38], and Graded Average Precision (GAP) [39]. Optimizing IR measures directly is difficult since they depend on the rank and are not differentiable. To avoid the computational difficulty, the core idea is to find a smooth proxy loss function that can be differentiable and used for training by approximating the "hard" version rank function with a "soft" function. The second category of listwise algorithms defines loss function as an *indirect* way to optimize the IR evaluation metrics. For instance, ListNet [40] uses the KL-divergence as a loss function by defining a probability distribution. ListRank-MF [41] employs the cross entropy loss that represents the uncertainty between training lists and output lists produced by a matrix factorization ranking model.

Table 2.1: Techniques of recommender systems classified by what to model.

Pointwise	**Pairwise**	**Listwise**
PMF [14]	BPR [28, 29]	SoftRank [34]
SVD [16]	RankALS [30]	CofiRank [35]
SVD+ [17]	RankNet [31]	TFMAP [36]
FM [18, 19]	LambdaRank [32]	CLiMF [38, 37]
WRMF [20]	LambdaMART [42]	GAPfm [39]
	EigenRank [43]	ListNet [40]
RRFM [5]	RRFM [5]	ListRank-MF [41]

2.2 Point-of-Interest Recommender Systems

In human mobility, user's check-in behavior is frequently affected by temporal effect, geographical location, and social network information. Thus, POI recommendation algorithms are classified into three categories.

The first category focuses on modeling **temporal effect** [3, 44, 45, 46, 47]. For example, a simple yet effective approach is to distinguish user's latent factors in different temporal state and then take several strategies to aggregate user check-in preference in each temporal state [48]. Then the ranking score is produced by the dot product of the corresponding user latent vector and location latent vector. Multi-center Gaussian mixture distribution has been developed in [49, 50] to capture the user's temporal preference. The author in [51] adopts the summation to fuse user's interest for POI and graphical influence, where the user's interest for POI could be learned by a User-based Collaborative Filtering extension with temporal state. Besides, a graph-based method is proposed to learn user's preference for POIs and then Breadth-first Preference Propagation is employed to search the optimal solutions [45].

The second category throws light on modeling **geographical influence** for recommendation [4, 52, 53, 54, 45, 55, 56]. Several approaches have been developed to incorporate geographical distance for location prediction. For example, [50] discovers user's check-in behavior follows a two-state (i.e. home and work) mixture of Gaussian

in geographical distance, assuming a POI that user would choose to check-in is next to either her home or work place, and then exploits such distribution to model user movement. Instead of employing the fixed two-center Gaussian mixture models, [57] further adopts the multi-center Gaussian model to form the distribution of the distance between the visited location and its center for each individual user. To avoid the difficulty of specifying a known distribution, [58] utilizes the kernel density estimation (KDE) to learn personalized check-in behavior with POI's geographical location, which is much more flexible. Furthermore, [59] further studies a mixture of adaptive Kernel density estimates to characterize a distribution between check-in probability and distances at different spatial level in order to avoid data sparsity in individual level for KDE. To exploit user based collaborative filtering, the author in [60] first proposes to use a power law distribution to estimate the check-in probability with the distance of any pair of visited POIs due to the spatial clustering phenomenon exhibited in LBSNs. The user's preference for one location is then predicted by a linear model with the combination of users' interest, social friends' interests and geographical influence. There are some other methods based on matrix factorization to produce more accurate recommendation. For example, [61] considers two types of geographical neighborhood characteristics: instance level and region level. In instance level, a user's preference for one location is modeled by a combination of her preference for this location and the nearest neighborhoods of this location. In region level, it places a group lasso penalty to learn location-specific latent vectors and capture the region effect. More important, [4] proposes a unified approach to integrate squared error loss and ranking error loss for solving location recommendation task by effectively learning fine-grained and interpretable user interest, and adaptively modeling the missing data. Specifically, each user's general interest is modeled as a mixture of her intrinsic and extrinsic interests, upon which we formulated the ranking constraints in our unified approach. Additionally, a self-adaptive location-oriented method is proposed

to capture the characteristic of missing data, and is then formulated as the squared error loss in our unified optimization objective.

The third category is elaborating **social information** for POI recommendation [1, 62, 63, 64, 65, 66, 2]. For example, with the intuition that users and their friends will share the similar interests, [67] places a social regularization term to constrain matrix factorization object functions for learning more accurate user feature vectors. [68] proposes a geo-social correlation model to capture four types of social correlations of users' check-in behaviors, i.e. local friends, distant friends, local non-friends and distant non-friends. The check-in probability is measured as a combination of these four geo-social correlations, where the corresponding coefficients are learned by a group of features in a logistic regression similar fashion. On the other hand, [69] exploits local and global social relations to assist recommendations. Specifically, in local context, it models the correlations between users and their friends by fitting the similarities between them; while in global context, it utilizes the reputation of a user in the whole social network as the weight to fit the observed ratings. The approaches proposed in [52, 60] predict the preference of the user for the POI by collaborating the preferences of her friends on this POI.

2.3 Friend Recommender Systems

Friend recommendation is a critical task in social networking sites that not only helps increase the linkage inside the network and also improves the user experience [66]. One popular research direction about friend recommendation is proximity-based method, surrogating the linkage likelihood of a potential link using the proximity between users that would be connected by the link. It can generally be grouped into nodal proximity-based methods and structural proximity-based methods. Nodal proximity-based methods usually measure the similarity between two social entities using their profiles, including demographic, geographic, and semantic characteristics by Jaccard coefficient, KL-divergence, cosine similarity and etc. Structural proximity-

based methods comprise neighborhood-based structural proximity and path-based structural proximity based methods[70, 71]. Neighborhood-based structural proximity includes Common Neighbor computed as the number of their mutual neighbors, Adamic/Adar measure assigning less weight to more connected common neighbors, Preferential Attachment where the linkage likelihood between social entities is highly correlated with their neighborhood sizes, and SimRank assuming that two social entities are similar if their neighbors are similar. Path-based structural proximity approaches contain Katz index which measures the structural proximity between social entities using the number of paths connecting them and is weighted by their lengths[72], random walk [73], and pagerank [71]. On the other hand, matrix factorization is also employed for friend recommendation [74]. For example, [75] extends ranking-based matrix factorization to model three different status of links in signed network. [76] optimizes log loss function for top-k recommendation by incorporating both the latent features and explicit features of the network. The ranking score of a potential social link between two users is produced by the summation over (1) the score estimated by the inner product of representations in latent space, and (2) the score predicted in a regression form with observed features.

CHAPTER 3: LEARNING FROM IMPLICIT FEEDBACK

3.1 Introduction

Recommender systems have been an important feature to recommend relevant items to relevant users in many online communities, e.g. Amazon, Netflix, and Foursquare. Some online systems allow users to provide an explicit rating for an item to express how much they like it. Nevertheless, many other recommender systems only have user's implicit feedback, such as browsing activity, purchasing history, watching history and check-in information. As implicit feedback becomes more and more prevalent, this type of recommender system has attracted many researchers' attention [77]. However, implicit feedback based recommender systems suffer from many challenges. For example, the sparseness of observed data, i.e., only a small percentage of user-item pairs have feedbacks, increases the difficulty to learn user's exact taste on items. Also, different from explicit feedback, only positive preference is observed in implicit feedback. In other words, we have no prior knowledge about which items users dislike. This has been a thorny issue for learning task.

In the literature, some related work has been proposed to take advantage of implicit feedback for item recommendations. For example, [20] regards user's preference for an item as a binary value, where a user's preference for the observed item and the unobserved one are viewed as one and zero, respectively. Then it fits these predefined ratings with vastly varying confidential levels based on matrix factorization framework. Although it assumes that a user prefers the observed items to unobserved ones, the quadratic in the formulation weakens their instinct ranking order. There is

no guarantee that the higher accuracy in rating prediction will result in the better ranking effectiveness [78]. For instance, the true ratings for two items are $\{1, 0.5\}$. The predicted ratings $\{0.4, 0.6\}$ and $\{1.6, 0.6\}$ have the same prediction accuracy. But in fact, they lead to totally different ranking orders of items. To this end, Rendle formulates user's consuming behavior into the pairwise ranking problem, i.e., users are much more interested in their consumed items than unconsumed ones [28]. Due to a large number of such pairs, it only samples some negative items for the learning procedure. However, there are two limitations. First, the pairwise ranking increases the number of comparisons. Second, although Rendle further improves the sampling skill by oversampling the top ranked items [29], sampling technique itself easily leads to bias. It is likely that the sampled negative item is already ranked below the positive one, which as a result has no contribution to the optimization.

To address these issues, in this chapter, we propose to relax the ranking model proposed in [28] to eliminate the pairwise ranking. Specifically, the positive and negative feedbacks are separated by the positive and negative boundaries. In fact, the unobserved implicit feedback is often a mixture of negative and missed positive data, so a slack variable is introduced to capture such characteristic, which allows some negative feedbacks and positive feedbacks to be non-separate. Furthermore, instead of sampling, a smooth and scalable algorithm is designed to learn model's parameters based on group and instance level's optimization. The proposed algorithm allows to take all the unobserved items into account for optimization, and as a result addresses the bias caused by the sampling technique in [28, 29]. Finally, the proposed model is evaluated with many state-of-the-art baseline models and different validation metrics on three real-world data sets. The experimental results demonstrate the superiority of our model for tackling implicit feedback based recommendations.

3.2 Pairwise Ranking

The recommendation task addressed in this chapter is defined as: given the consumption behaviors of N users over M items, we aim at recommending each user with top-K new items that she might be interested in but has never consumed before. Matrix factorization based models assume that $\mathbf{U} \in \mathbb{R}^{D \times N}$ and $\mathbf{V} \in \mathbb{R}^{D \times M}$ are the user and item latent feature matrices, with column vectors $\mathbf{u}_i$ and $\mathbf{v}_j$ representing the D-dimension user-specific and item-specific feature vectors of user i and item j, respectively. The predicted preference (rating) of user i for item j, denoted as $\hat{r}_{ij}$, is approximated by:

$$\hat{r}_{ij} = \mathbf{u}_i^T \mathbf{v}_j. \tag{3.1}$$

In implicit feedback datasets, only positive feedback is observed. As we lack substantial evidence on which items users dislike, their preference for an unobserved item is regarded as a mixture of negative and missing values. Along this line, [28] assumes that each user prefers the observed items over unobserved ones. Let us denote $\mathcal{M}_i^o$ as a set of items that user i has consumed and $\mathcal{M}_i^u$ as the remaining items that she never consumed. Then for user i, the ranking based on user's preference for an observed item j over an unobserved item k is given by:

$$\mathbf{u}_i^T \mathbf{v}_j > \mathbf{u}_i^T \mathbf{v}_k, \qquad \forall j \in \mathcal{M}_i^o \wedge \forall k \in \mathcal{M}_i^u. \tag{3.2}$$

For convenience we will henceforth refer to i as user, j as observed item, and k as unobserved item unless stated otherwise. Eq.(3.2) models the correlation of user's preference for each pair of observed item and unobserved one. It is actually maximizing the Area Under the ROC Curve (AUC) for matrix factorization. The optimization

problem for pairwise ranking is formulated as follows:

$$\max_{\mathbf{U},\mathbf{V}} \sum_{i=1}^{N} \sum_{j \in \mathcal{M}_i^o} \sum_{k \in \mathcal{M}_i^u} \log \sigma(\mathbf{u}_i^T \mathbf{v}_j - \mathbf{u}_i^T \mathbf{v}_k) - \frac{\lambda_u}{2} ||\mathbf{U}||_F^2 - \sum_{i=1}^{N} \left(\frac{\lambda_{v_1}}{2} \sum_{j \in \mathcal{M}_i^o} ||\mathbf{v}_j||^2 + \frac{\lambda_{v_2}}{2} \sum_{k \in \mathcal{M}_i^u} ||\mathbf{v}_k||^2 \right),$$

where $\sigma(x)$ is the sigmoid function, i.e., $\sigma(x) = 1/(1 + e^{-x})$; $|| \cdot ||_F$ is the Frobenius norm; $|| \cdot ||$ is Euclidean norm; and λ_u, λ_{v_1} and λ_{v_2} are regularization constraints. In optimization process, sampling negative items is adopted to avoid comparing with all unobserved items for each individual user. The optimal solution, as a result, can be obtained by Stochastic Gradient Descent (SGD).

3.3 Relaxed Ranking Model

Method

In implicit feedback systems, user's preference for the observed item is usually regarded as positive rating and there is no negative rating. Most research work argues that one user's preference for the observed item is supposed to be larger than that for any unobserved one [28, 20], which indicates the presence of ranking between positive and negative ratings. However, computing the pairwise ranking of a user's preference for the observed items over the unobserved ones is quite inefficient, especially when the user's historical data increases. To address this issue, we propose to relax the pairwise ranking. We treat one user's preference for an item as an point, where her positive (or negative) rating is viewed as the positive (or negative) point. Our goal is to make all positive points reside above all negative points. Inspired by the soft margin idea of SVM, we separate these two types of points by two different boundaries. In other words, user's preference for the observed items and unobserved ones are separated by the boundaries. Specifically, user's positive rating is located on or above a boundary represented as a numeric value r_+. On the other hand, her negative rating resides

on or below another boundary. It not only improves the efficiency of comparisons, but also preserves the ranking information. Along this line, we relax Eq.(3.2) for $\forall j \in \mathcal{M}_i^o \wedge \forall k \in \mathcal{M}_i^u$ in the following:

$$\begin{cases} \mathbf{u}_i^T \mathbf{v}_j \geq r_+, \\ \mathbf{u}_i^T \mathbf{v}_k \leq r_- + \xi_{ik}, \end{cases} \tag{3.3}$$

where ξ_{ik} is the slack variable for user i on the unobserved item k [1]. Even though there may be many unobserved items for a user, it does not necessarily indicate that she dislikes them. Probably she may be just unaware of them. In other words, some of the unobserved items might be those users are interested in, while others are actually those they dislike. To explicitly capture this characteristic, the slack variable ξ_{ik} in Eq.(3.3) is introduced to allow the mixture of negative feedback and positive feedback in unobserved data [2]. Hence, we apply the following constraint for ξ_{ik} as:

$$r_- + \xi_{ik} \geq r_+ \quad \Rightarrow \quad \xi_{ik} \geq r_+ - r_-. \tag{3.4}$$

It is worth to note that for each user i, pairwise ranking in Eq.(3.2) needs $|\mathcal{M}_i^o| \times |\mathcal{M}_i^u|$ comparisons; but our relaxed ranking in Eq.(3.3) only requires $M = |\mathcal{M}_i^o| + |\mathcal{M}_i^u|$ comparisons. Specifically, when $|\mathcal{M}_i^o| = M/2$, pairwise ranking needs $M^2/4$ comparisons while relaxed ranking only requires M comparisons.

However, Eq.(3.3) is a "hard" version of optimization problem due to the strict constraints. Thus, we make the constraints "soft" by introducing a plus function to penalize the violated constraints. The optimization problem with the softened

[1] Generally, r_+ and r_- can be any numerical values. In the experiments, r_+ and r_- are set as one and zero respectively, which is similar to [20].

[2] Even though the optimization error in hard boundary might lead to the overlap of negative and positive results, we consider to explicitly model this characteristics with soft boundary in this chapter.

constraints is formulated as follows:

$$\min_{\mathbf{U},\mathbf{V},\xi} \sum_{i=1}^{N} \left(\sum_{j \in \mathcal{M}_i^o} \left(-\mathbf{u}_i^T \mathbf{v}_j + r_+\right)_+ + \sum_{k \in \mathcal{M}_i^u} \left(\mathbf{u}_i^T \mathbf{v}_k - \xi_{ik} - r_-\right)_+ \right) + ||\Theta||, \tag{3.5}$$

where $(\cdot)_+$ is the plus function [79], i.e., $(x)_+ = max(x, 0)$, and $||\Theta||$ is the regularization term given by:

$$||\Theta|| = \frac{\lambda_u}{2}||\mathbf{U}||_F^2 + \frac{\lambda_v}{2}||\mathbf{V}||_F^2 + \lambda_\xi \sum_i^N ||\xi_i||_1,$$

where λ_u, λ_v and λ_ξ are the regularization constants, and $||\xi_i||_1$ is ℓ_1 norm of vector ξ_i. More important, one reason for imposing such a vector norm on ξ is that users are usually interested in a small percentage of unobserved items among all the remaining unobserved ones. It is worth to note that the formulation in Eq.(3.5) is different from a pure SVM scheme: (1) We relax the pairwise ranking problem which is adaptable for any pairwise ranking based application; (2) A novel and scalable optimization is designed for the objective function (in the next section). (3) The slack variable with ℓ_1 norm is introduced to capture the characteristics of unobserved data (i.e., the mixture of positive and negative data).

Optimization

The bottleneck for the optimization problem shown in Eq.(3.5) is the calculation of each user's ratings on all the unobserved items, which results in at least $\mathcal{O}(MND)$ time complexity. Particularly, with the increase of the item number, it becomes more and more inefficient. As sampling negative items possibly leads to bias, in this chapter we propose a smooth and scalable optimization algorithm to take all the unobserved items into consideration.

First, we solve the problem in a group-based optimization. The entire item set is randomly divided into G groups. It is worth to note that we adopt the random

group technique in the experiments for the sake of efficient computation. Instead of requiring the rating on each unobserved item to reside on or below the negative boundary, we softly make the average rating of the unobserved items in each group locate on or below this boundary. Suppose $\mathcal{G}_h$ is the set of all items which belong to group h, and $\mathcal{G}_h^i$ is the set of unobserved items for user i with group as h, then we have the following inequality:

$$\frac{1}{|\mathcal{G}_h^i|}\sum\nolimits_{k\in\mathcal{G}_h^i}\mathbf{u}_i^T\mathbf{v}_k \leq r_- + \xi_{ih},$$

where ξ_{ih} is the slack variable for user i on group h. Let us define the following variables:

$$\mathbf{v}_h^s \equiv \sum_{p\in\mathcal{G}_h}\mathbf{v}_p, \quad \tilde{\mathbf{v}}_{ih}^s \equiv \sum_{j\in\mathcal{M}_i^o\wedge h(j)=h}\mathbf{v}_j, \quad \bar{\mathbf{v}}_h^i \equiv \frac{1}{|\mathcal{G}_h^i|}(\mathbf{v}_h^s - \tilde{\mathbf{v}}_{ih}^s), \tag{3.6}$$

where $h(j)$ is the group index of item j. In addition, the plus function is not twice differentiable and can be smoothly approximated by the integral to a smooth approximation of the sigmoid function [80, 81] as follows:

$$(x)_+ \approx p(x,\alpha) = x + \frac{1}{\alpha}\log(1+\exp(-\alpha x)).$$

In the experiments, we find the above approximation form is slightly better than the logistic function. Hence, the objective function in Eq.(3.5) can be modified as follows:

$$\mathcal{L} = \underset{\mathbf{U},\mathbf{V},\xi}{\operatorname{argmin}}\sum_{i=1}^{N}\left\{\sum_{j\in\mathcal{M}_i^o} p\left(-\mathbf{u}_i^T\mathbf{v}_j + r_+,\alpha\right) + \sum_{h}^{G} p\left(\mathbf{u}_i^T\bar{\mathbf{v}}_h^i - \xi_{ih} - r_-,\alpha\right)\right\} + ||\Theta||, \tag{3.7}$$

where the slack variable has the constraint as $\xi_{ih} \geq r_+ - r_-$. We exploit the gradient

descent based optimization procedure to obtain the optimal solutions for $\mathbf{U}$, $\mathbf{V}$ and ξ in above problem. Their gradients are given by:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{u}_i} = -\sum_{j \in \mathcal{M}_i^o} p'_{ij}(\mathbf{U}, \mathbf{V}, \xi)\mathbf{v}_j + \sum_{h=1}^{G} \bar{p}'_{ih}(\mathbf{U}, \mathbf{V}, \xi)\bar{\mathbf{v}}_h^i + \lambda_u \mathbf{u}_i, \tag{3.8}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}_j} = -\sum_{i \in \mathcal{N}_j^o} p'_{ij}(\mathbf{U}, \mathbf{V}, \xi)\mathbf{u}_i + \sum_{i \in \mathcal{N}_j^u} \frac{\bar{p}'_{ih(j)}(\mathbf{U}, \mathbf{V}, \xi)\mathbf{u}_i}{|\mathcal{G}_{h(j)}^i|} + \lambda_v \mathbf{v}_j, \tag{3.9}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_{ih}} = -\bar{p}'_{ih}(\mathbf{U}, \mathbf{V}, \xi) + \lambda_\xi, \tag{3.10}$$

where $\mathcal{N}_j^o$ is the set of users who have consumed item j and $\mathcal{N}_j^u$ is the remaining users who have never consumed it. One caveat is that as the constraint placed on ξ guarantees it to be larger than 0 (i.e., its sign holds as positive), we can simply remove the absolute value sign to obtain its gradient. Also, $p'_{ij}(\mathbf{U}, \mathbf{V}, \xi)$ and $\bar{p}'_{ih}(\mathbf{U}, \mathbf{V}, \xi)$ are defined as the following:

$$p'_{ij}(\mathbf{U}, \mathbf{V}, \xi) = 1 - 1/(1 + \exp(\alpha(-\mathbf{u}_i^T \mathbf{v}_j + r_+)),$$
$$\bar{p}'_{ih}(\mathbf{U}, \mathbf{V}, \xi) = 1 - 1/(1 + \exp(\alpha(\mathbf{u}_i^T \bar{\mathbf{v}}_h^i - \xi_{ih} - r_-))).$$

To efficiently calculate the gradient of latent factor $\mathbf{V}$, we rewrite Eq.(3.9) in the following (see the next section):

$$\begin{aligned}\frac{\partial \mathcal{L}(\mathbf{U}, \mathbf{V}, \xi)}{\partial \mathbf{v}_j} = &-\sum_{i \in \mathcal{N}_j^o} p'_{ij}(\mathbf{U}, \mathbf{V}, \xi)\mathbf{u}_i - \sum_{i \in \mathcal{N}_j^o} \widetilde{\mathbf{v}}_{h(j)i}(\mathbf{U}, \mathbf{V}, \xi) \\ &+ \widetilde{\mathbf{v}}_{h(j)}^s(\mathbf{U}, \mathbf{V}, \xi) + \lambda_v \mathbf{v}_j,\end{aligned} \tag{3.11}$$

where $\widetilde{\mathbf{v}}_{hi}(\mathbf{U}, \mathbf{V}, \xi)$ and $\widetilde{\mathbf{v}}_h^s(\mathbf{U}, \mathbf{V}, \xi)$ are defined as follows:

$$\widetilde{\mathbf{v}}_{hi}(\mathbf{U}, \mathbf{V}, \xi) = \frac{1}{|\mathcal{G}_h^i|}\bar{p}'_{ih}(\mathbf{U}, \mathbf{V}, \xi)\mathbf{u}_i, \quad \widetilde{\mathbf{v}}_h^s(\mathbf{U}, \mathbf{V}, \xi) = \sum_{i=1}^{N} \widetilde{\mathbf{v}}_{hi}(\mathbf{U}, \mathbf{V}, \xi).$$

The algorithm detail is shown in Figure 3.1, where adaptive learning rate is used to improve convergence rate [82]. Specifically, we sample a set of unobserved items for each user to reinforce the optimization granularity. It allows us to optimize the problem in both group and instance levels. Suppose for user i, we sample a set of unconsumed items, denoted as $\mathcal{A}$, and then select those items whose predicted ratings are larger than its group's mean rating to learn parameters. The selected item l for user i satisfies the following rule:

$$\mathbf{u}_i^T \mathbf{v}_l > \mathbf{u}_i^T \bar{\mathbf{v}}_{h(l)}^i. \tag{3.12}$$

Hence the objective function in Eq.(3.7) is refined by:

$$\mathcal{L}^{(new)} = \mathcal{L} + \sum_{i=1}^{N} \sum_{q \in \mathcal{A}_i} p(\mathbf{u}_i^T \mathbf{v}_q - \xi_{ih(q)} - r_-, \alpha)),$$

where $\mathcal{A}_i$ is the set of selected items that satisfy Eq.(3.12) and are unconsumed by user i. Therefore, the gradients of $\mathbf{U}$, $\mathbf{V}$, and ξ can be obtained similarly as above inference, given by:

$$\begin{aligned}
\frac{\partial \mathcal{L}^{(new)}(\mathbf{U}, \mathbf{V}, \xi)}{\partial \mathbf{u}_i} &= \frac{\partial \mathcal{L}(\mathbf{U}, \mathbf{V}, \xi)}{\partial \mathbf{u}_i} + \sum_{q \in \mathcal{A}_i} \bar{p}'_{ih(q)}(\mathbf{U}, \mathbf{V}, \xi) \mathbf{v}_q, \\
\frac{\partial \mathcal{L}^{(new)}(\mathbf{U}, \mathbf{V}, \xi)}{\partial \mathbf{v}_j} &= \frac{\partial \mathcal{L}(\mathbf{U}, \mathbf{V}, \xi)}{\partial \mathbf{v}_j} + \sum_{i \in \mathcal{A}_j} \bar{p}'_{ih(j)}(\mathbf{U}, \mathbf{V}, \xi) \mathbf{u}_i, \\
\frac{\partial \mathcal{L}^{(new)}(\mathbf{U}, \mathbf{V}, \xi)}{\partial \xi_{ih}} &= \frac{\partial \mathcal{L}(\mathbf{U}, \mathbf{V}, \xi)}{\partial \xi_{ih}} - \sum_{q \in \mathcal{A}_{ih}} \bar{p}'_{ih}(\mathbf{U}, \mathbf{V}, \xi),
\end{aligned}$$

where $\mathcal{A}_j$ is the set of users who have never consumed item j and $\mathcal{A}_{ih}$ is the set of items that belong to group h and are unconsumed by user i.

Input: Observed user-item pairs, regularization constants λ_u, λ_v and λ_ξ, dimension D of latent space, group number G, stop criteria τ and $maxIter$, and learning rate η

Output: $\mathbf{U}^{(k)}$, $\mathbf{V}^{(k)}$, $\xi^{(k)}$

Randomly initialize $\mathbf{U}^{(0)}$ and $\mathbf{V}^{(0)}$, $k \leftarrow 1$, $\varepsilon \leftarrow \infty$;
Randomly initialize $\xi^{(0)}$ and ensure $\xi_{ih}^{(0)} \geq r_+ - r_-$;
Randomly divide items into G equal groups;
while $k \leqslant maxIter$ && $\varepsilon \geqslant \tau$ **do**
    Uniformly sample $\mathcal{A}$ unobserved items for each user i and select those items which satisfies Eq.(3.12) to join to learn latent factors;
    for $j = 1$ ***to*** M **do**
        $\mathbf{v}_j^{(k+1)} \leftarrow \mathbf{v}_j^{(k)} - \eta \frac{\partial \mathcal{L}^{(new)}(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \xi^{(k)})}{\partial \mathbf{v}_j}$;
    for $i = 1$ ***to*** N **do**
        $\mathbf{u}_i^{(k+1)} \leftarrow \mathbf{u}_i^{(k)} - \eta \frac{\partial \mathcal{L}^{(new)}(\mathbf{U}^{(k)}, \mathbf{V}^{(k+1)}, \xi^{(k)})}{\partial \mathbf{u}_i}$;
    for $(i, h) = (1, 1)$ ***to*** (N, G) **do**
        $\xi_{ih}^{(k+1)} \leftarrow \max(r_+ - r_-, \xi_{ih}^{(k)} - \eta \frac{\partial \mathcal{L}^{(new)}(\mathbf{U}^{(k+1)}, \mathbf{V}^{(k+1)}, \xi^{(k)})}{\partial \xi_{ih}})$
    $\Delta \leftarrow \mathcal{L}^{(new)}(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \xi^{(k)}) - \mathcal{L}^{(new)}(\mathbf{U}^{(k+1)}, \mathbf{V}^{(k+1)}, \xi^{(k+1)})$;
    Update η: $\eta \leftarrow \begin{cases} 1.05\eta & \text{if } \Delta > 0, \\ 0.5\eta & \text{otherwise;} \end{cases}$
    $\varepsilon \leftarrow \frac{|\Delta|}{|\mathcal{L}^{(new)}(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \xi^{(k)})|}$
    $k \leftarrow k + 1$
return $\mathbf{U}^{(k)}$, $\mathbf{V}^{(k)}$, $\xi^{(k)}$

Figure 3.1: Algorithm of Relaxed Ranking-based Factor model.

Complexity Analysis

In each iteration, updating $\mathbf{U}$, $\mathbf{V}$ and ξ dominates the major time complexity for solving the optimal problem shown in Eq.(3.7). To update $\mathbf{u}_i$, the first term of Eq.(3.8) requires time complexity as $\mathcal{O}(n_i D)$, where n_i is the number of items that user i has consumed. $\mathbf{v}_h^s$ is independent of i and can be pre-computed. Hence, computing $\tilde{\mathbf{v}}_{ih}^s$ costs $\mathcal{O}(n_{ih} D)$, where n_{ih} is the number of items that user i has consumed and belong to group h, and $n_i = \sum_h n_{ih}$. Then, the second term of updating $\mathbf{u}_i$ needs time complexity as $\mathcal{O}(n_i D + GD)$. Consequently, the computation of $\mathbf{u}_i$ is performed

in $\mathcal{O}(n_i D + GD)$. This procedure is performed over N users, so the total time is $\mathcal{O}(nD + NGD)$, where $n \equiv \sum_i n_i$.

In the procedure of optimizing $\mathbf{v}_j$, the searching direction is dependent on $\mathbf{V}^{(old)}$ obtained in the last iteration. The first term in Eq.(3.11) costs $\mathcal{O}(n_j D)$, where n_j is the number of users who have consumed item j. Similar to above, $\widetilde{\mathbf{v}}_{ih}^{(old)\,s}$ in Eq.(3.11) can be pre-calculated. On the other hand, $\widetilde{\mathbf{v}}_{hi}(\mathbf{U}, \mathbf{V}^{(old)}, \xi)$ is independent of j and could be also pre-computed. Then $\widetilde{\mathbf{v}}_h^s(\mathbf{U}, \mathbf{V}^{(old)}, \xi)$ is pre-stored in the memory due to the sum of $\widetilde{\mathbf{v}}_{hi}(\mathbf{U}, \mathbf{V}^{(old)}, \xi)$ over all users. Hence, the second term in Eq.(3.11) costs $\mathcal{O}(n_j D)$ time complexity. Thus, updating $\mathbf{v}_j$ needs running time as $\mathcal{O}(n_j D)$, and the total cost time over M items is $\mathcal{O}(nD)$, where $n = \sum_j n_j$.

Similar to update user latent matrix $\mathbf{U}$, for each slack variable ξ_{ih}, we only need $\mathcal{O}(D)$ time complexity due to the pre-computation of $\mathbf{v}_{ih}^s$. Totally, updating ξ over N users and G groups costs time complexity as $\mathcal{O}(NGD)$.

In instance-level optimization, similar to above analysis, in the worst situation, each iteration needs $\mathcal{O}(NAD)$ running time. In a summary, each iteration of our algorithm costs $\mathcal{O}(nD + N(A+G)D)$ time complexity, where n is the number of the observed entries in user-item matrix, and $\frac{N(A+G)}{n}$ is small. Therefore, the time complexity can be approximated by $\mathcal{O}(nD)$. In other words, the time complexity of each iteration for the optimization is a linear proportion to the number of observed user-item pairs.

3.4 Experimental Results

Datasets

We use three datasets to evaluate the performance of the proposed model. The first two datasets are check-in data collected from Gowalla and Foursquare. Each check-in record in the dataset includes a user ID, a location ID, a check-in frequency and a timestamp that she first time checked-in this location. As we only know user's check-in action, these two data sets are the implicit feedback based data sets. The third

data set is the movie data of MovieLens, where each user provides explicit ratings, 1 to 5 stars, for some movies. As our task focuses on the implicit feedback, similar to [28], we remove the rating score from the dataset. We utilize *consuming* to represent user's *checking-in* or *watching* behavior in these three datasets. We remove those users who have consumed less than 10 items. The detailed statistics of datasets are reported in Table 3.1.

Table 3.1: The statistics of data sets.

Dataset	#User	#Item	#Records	Sparsity
Gowalla	52,216	98,351	2,577,336	0.0399%
Foursquare	74,343	198,161	3,501,608	0.0238%
MovieLens	6,040	3,681	1,000,179	4.4986%

As recommender system targets at recommending new items for users, we split the training and testing as the following. In check-in data, for each user, we first sort the records according to the check-in timestamp; and then select the earliest 80% to train the model and use the next 20% as testing. In MovieLens dataset, we randomly select 80% data as training and use the rest as testing due to the absent timestamp.

Parameter Settings

In the experiments, the regularization constants λ_u, λ_v and λ_ξ are set as 0.01. The number D of latent space is set as 10, the initial learning rate η is 0.001, and parameter α is set as 5. We divide 100 groups for three datasets. A in check-in data and MovieLens are 150 and 400, respectively.

Evaluation Metrics

We quantitatively evaluate model performance in terms of top-K recommendation performance, i.e., Precion@K and Recall@K, and ranking performance, i.e., MAP and

AUC [2]. Formally, their definitions are shown as:

$$P@K = \frac{1}{N}\sum_{i=1}^{N}\frac{|\mathcal{S}_i(K)\cap\mathcal{T}_i|}{K}, MAP = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{j=1}^{\widetilde{\mathcal{M}}_i} p(j)\times rel(j)}{|\mathcal{T}_i|},$$
$$R@K = \frac{1}{N}\sum_{i=1}^{N}\frac{|\mathcal{S}_i(K)\cap\mathcal{T}_i|}{|\mathcal{T}_i|}, AUC = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{(j,k)\in\mathcal{E}(i)} I(\hat{r}_{ij}>\hat{r}_{ik})}{|\mathcal{E}(i)|},$$

where $\mathcal{S}_i(K)$ is a set of top-K new items recommended to user i excluding those items in training, and $\mathcal{T}_i$ is a set of items that have been consumed in the testing. $\widetilde{\mathcal{M}}_i$ is the number of the returned items in the list of user i, $p(j)$ is the precision of a cut-off rank list from 1 to j, and $rel(j)$ is an indicator function that equals to 1 if the item is in the testing, otherwise is 0. $\mathcal{E}(i)$ is defined as $\mathcal{E}(i) := \{(j,k)|j\in\mathcal{T}_i\wedge k\notin(\mathcal{T}_i\cup T_i^t)\}$, where $\mathcal{T}_i^t$ is a set of items consumed by user i in the training. $I(\cdot)$ is a indicator function, which equals to 1 if its argument is true and 0 otherwise.

Baseline Methods

To comprehensively demonstrate the effectiveness of the proposed model, named as RRFM, we compare it with the following popular recommendation models:

- **UBPR** [28] models the pairwise ranking for each pair of the observed and unobserved items, and employs SGD with uniformly sampling for optimization.
- **ABPR** [29] is similar to UBPR, but it over-samples the top ranked negative items based on a context-dependent sampling schema.
- **WRMF** [20], treats user's rating as a binary value and fits the ratings on the observed and unobserved items with different confidential values.
- **PMF** [14] regards the rating as the dot product of user-specific and item-specific feature factors. Logistic function is used as rating conversion method to avoid the data's bias.

UBPR, ABPR and WRMF are all modeling both observed and unobserved data,

while PMF only models the observed data. In the perspective of optimization, UBPR and ABPR are both ranking based models; while WRMF and PMF are both rating prediction-based models. The number of latent space in all baseline models is set the same with RRFM.

Ranking Performance Comparison

The ranking performance in terms of MAP and AUC for the proposed model with baseline models is reported in Table 3.2. We summarize the following observations.

First, the method only modeling the observed feedback (i.e., PMF), performs much worse than those taking both observed and unobserved data into consideration (i.e., other four models). Specifically, in Gowalla data, RRFM achieves result as 95.673% in terms of AUC while PMF only obtains 62.005%. Also, RRFM is 3.793% in terms of MAP while PMF is 1.357%. Different from explicit feedback, it is difficult to know user's negative preference from implicit feedback. As a result, explicit feedback based model is difficult to achieve better performance for implicit feedback. The result indicates that the unobserved data also provides meaningful information and assist to improve ranking performance.

Second, ranking-based methods (i.e., UBPR, ABPR and RRFM) are nearly better than rating prediction-based models (i.e., WRMF and PMF), particularly in terms of AUC metric. Ranking-based methods are modeling the ranking order of user's positive rating over negative rating, which actually maximizes the AUC metric; while rating prediction-based models focus on the task how to correctly predict ratings. Although WRMF regards user's positive and negative ratings as one and zero respectively, there is no guarantee for the correctness of their ranking order due to rating prediction based loss. This explains why it has poor performance in AUC.

Third, our model is superior to others, such as that in Foursquare, RRFM outperforms WRMF 100.7% and 12.9% in terms of MAP and AUC, respectively; RRFM achieves 15.2% improvement over ABPR in MAP. It happens due to two reasons.

Table 3.2: Performance comparison in terms of MAP and AUC metrics. The result is reported in percentage (%).

Gowalla Dataset					
Metric	RRFM	WRMF	ABPR	UBPR	PMF
MAP	**3.793**	2.634	3.169	3.080	1.357
AUC	**95.673**	88.748	94.732	94.642	62.005
Foursquare Dataset					
MAP	**2.302**	1.147	1.999	1.913	0.024
AUC	**95.814**	84.866	94.726	94.554	61.211
MovieLens Dataset					
MAP	**21.108**	20.528	19.659	19.512	10.159
AUC	**93.795**	91.743	92.417	92.372	85.685

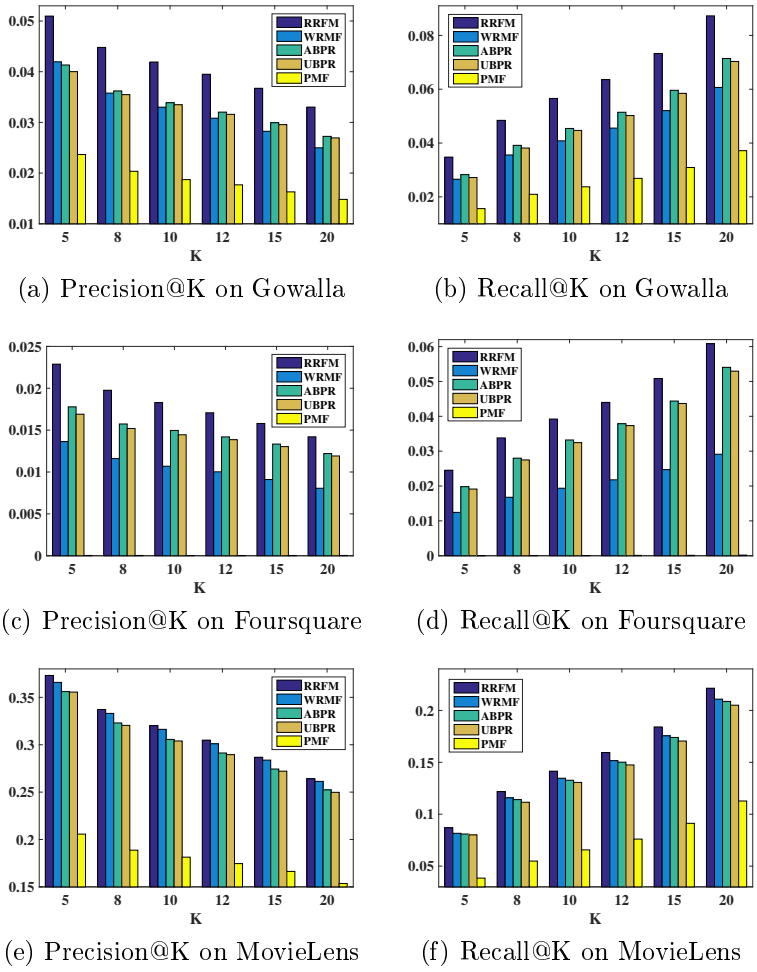
First, we separate the positive and negative ratings by the soft boundaries, where some of them are non-separate. It captures user's actual consuming behavior where she does not consume an item probably due to her dislike or unawareness. Second, we optimize the model in group and instance granularity, where group-based optimization makes parameters reach their optimal solutions in a direction towards the minimum of cost function and instance-based optimization reinforces the approximation to the exact optimization problem in Eq.(3.5). In addition, WRMF has different performance in MAP due to the much less items and much denser user-item entries in MovieLens than check-in data.

Last, ABPR performs slightly better than UBPR. Instead of uniformly sampling negative items, ABPR samples with an adaptive distribution. It exploits item tailed characteristics (i.e., over-sampling the top ranked negative items) to speed up the convergence of SGD. But with the increase of iteration, UBPR approximates to the true optimal solution as ABPR. It is the reason why it performs similarly as ABPR.

Top-K Recommendation Performance Comparison

The top-K performance of our model versus baseline methods over three datasets is plotted in Figure 3.2. Based on the observations, we summarize as follows:

First, RRFM outperforms all baseline methods. In particular, it has significant



(a) Precision@K on Gowalla (b) Recall@K on Gowalla

(c) Precision@K on Foursquare (d) Recall@K on Foursquare

(e) Precision@K on MovieLens (f) Recall@K on MovieLens

Figure 3.2: Performance comparison in terms of precision and recall on Gowalla, Foursquare and MovieLens datasets.

improvement over PMF, which contributes to the involvement of unobserved data for modeling. Its superior performance than ABPR and UBPR is beneficial from the following reasons: (1) The relaxed model allows user's preference for some unobserved items to mix with positive preference. This is based on the assumption that the unobserved data might be actually negative or missed positive value. (2) It optimizes the ranking problem based on the entire unobserved items with a smooth approximation, which addresses the bias caused by ABPR and UBPR. Furthermore, RRFM obtains much better performance than WRMF. Although WRMF also considers the ranking of positive rating over negative one, it models the recommendation problem by fitting the binary ratings, which cannot guarantee their actual ranking order. Hence, it does

not perform as well as RRMF. In MovieLens, the improvement of RRFM over WRMF is small due to that the denser observed data helps WRMF approximate to the true optimal point.

Second, ABPR and UBPR are superior to WRMF in check-in data, while they are a little worse than WRMF in MovieLens data. ABPR and UBPR actually optimize the AUC metric which evaluates the ranking of one user's preference for the testing items over the remaining unobserved items. To make user's positive ratings larger than most of the negative ones, they might sacrifice some top-K performance. It is a tradeoff between the performance in AUC and top-K. Hence although they obtain extremely superior performance in terms of AUC, it cannot guarantee that they can also perform well in top-K performance. That's why they have poor performance in terms of precision@K and recall@K in MovieLens dataset.

Last, ABPR, UBPR and PMF perform consistently with the ranking performance, where ABPR is better than UBPR, and PMF performs the worst among all models. Although the way to sample negative items has been improved, ABPR is difficult to make significant improvements due to the limitation of sampling technique. PMF's poor result further demonstrates the importance of modeling all data.

3.5 Conclusion

In this chapter, we propose a relaxed ranking-based algorithm for item recommendation with implicit feedback, and design a smooth and scalable optimization method for model's parameter estimation. The relaxed model not only avoids the pairwise ranking by separating the positive feedback and negative feedback with the soft boundaries, but also exploits the non-separate property of negative and positive feedback to capture the characteristic of unobserved data. To evaluate our model, we conduct extensive experiments with many baseline methods and evaluation metrics on the real-world data sets. The experimental results have shown our model's effectiveness.

CHAPTER 4: POINT-OF-INTEREST RECOMMENDATION

4.1 Introduction

Recent years have witnessed the rapid prevalence of location-based social network (LBSN) services such as Foursquare, Jiepang, and Facebook Places that can significantly facilitate users' outdoor activities by providing a large number of nearby Point-of-Interests (POIs) in a real-time fashion. A variety of user interaction data with these LBSN services such as searching locations, providing check-in information and posting tips after visiting a POI have been accumulated, which provides an unparalleled opportunity for developing personalized POI recommender systems [1, 83, 84, 85]. Indeed, the accurate and personalized POI recommendation is a crucial demand in LBSN services. First, given the massive locations, it is very difficult for users to find their preferred ones in an efficient way. A personalized POI recommender system would help users easily find relevant POIs without spending much time on manually searching, particularly when a user is in a new region. Also, it is very challenging for POI owners to deliver right POIs to various users. A personalized POI recommender system is able to not only ease such burden, but also attract more customers with the recommended POIs.

In the literature, recommender systems have been widely studied among various categories, such as movie recommendation on NetFlix and product (or item) recommendation on Amazon. However, it is not sufficient to directly apply these technologies on POI recommendation due to the specificity of human mobility in LBSNs. Hence, it is crucial to identify the unique characteristics of human mobility in LBSNs,

which motivates us to design the specific and more appropriate approaches for POI recommendation. In the following we list the prime properties:

- *Geographical Influence.* In LBSNs, geographical property associated with locations plays an important role in user's choices to locations [60, 53, 2]. Due to the restriction of human mobility, a user usually would like to prefer a nearby POI rather than another one far away due to short distance. Furthermore, according to the First Law of Geography, everything is related to everything else, but near things are more related than distant things [86]. The geographically close users may share similar interests and should have potential influence on check-in behaviors. Therefore, considering the geographical property of POIs enables us to capture the user preferences more precisely for POI recommendation in LBSNs.

- *Temporal Effect.* Human geographical movement exhibits strong temporal patterns in LBSNs [48]. For example, a user regularly chooses a restaurant next to her work place for convenience in week days, while she goes to a bar for entertainment at weekends. Such temporal cyclic patterns are often observed in check-in data and provide us a perspective to understand user mobility. Thus, mining the temporal patterns in check-in data in terms of where a user would like to go enables us to better model user check-in behaviors.

- *Social Influence.* Online social connections offer the user opportunities to view her friends' historical check-in behaviors. Some of the friends have positive impact on a user's check-in at a particular location, while others may influence negatively. Hence, a user's check-ins can be highly affected by a group of friends. Taking such social influence into account suggests its potential to design more advanced POI recommender systems in LBSNs.

In this chapter, in order to provide more accurate and efficient POI recommendation, we study the above properties and incorporate them into recommender systems.

Specifically, we first model geographical property and temporal effect, and propose a novel geo-temporal POI recommendation approach, which is presented in Section 4.2. In Section 4.3, we introduce the geo-social POI recommender systems by considering geographical influence and social correlations observed in human movement. In Section 4.4, the geographical property and social influence are also leveraged to tackle cold-start problems in POI recommendation.

4.2 Modeling Geo-Temporal Influence

In this section, we study geographical influence and temporal effect for POI recommendation in LBSNs. With geographical property and temporal pattern, user's preference is modeled from two aspects. Firstly, rich geographical information, such as distance and business prosperity of a region, is represented as a group of features to capture user's preference in geographical aspect. Secondly, temporal cyclic pattern and location property form a check-in event, which is introduced to characterize each individual user's check-in behavior. Then the hidden topic distribution over all check-in events for each user is exploited to model user's taste in temporal dynamic aspect.

4.2.1 Methodology

Let us first introduce the formal definition of *Geographical Feature* and *Check-in Event*, and then present a general framework to incorporate geographical features into matrix factorization and learn user preference in a more robust way. Finally, we present the specific Spatial-Temporal Probabilistic Matrix Factorization (STPMF) model and our estimation method.

4.2.1.1 Definitions

Recent works on POI recommendation have revealed that the distance of POI has great influence on user's preference for POIs [4, 60, 57, 50, 58, 59, 53]. In fact, there are other more geographical factors that also significantly affect user's check-

in decision making process. For example, the business prosperity of a region where a POI is located might affect user's preference. Users are likely to choose a POI where its surrounding environment is relatively prosperous as it often indicates high-credibility, high-quality services and crowd human activities. In addition, users are often used to visiting a certain region with the same purposes such as shopping or eating. In fact, this has been reflected in many real-world check-in data, where each POI belongs to one or more categories such as food, sights and etc. For example, we pick up a user's historical check-ins from our data and show them in Figure 5.1, where red marker represents category **nightlife** and black marker indicates category **sights**. Two regions are observed: one top red box area and one bottom red box area where there are about 2689 and 1205 services in their neighbors respectively. We can see that: (1) She would like to go to the top region for visiting sights and prefer the bottom region for nightlife; (2) She likes those POIs in rather prosperous regions. Thus, a group of geographical features are defined as follows.

Definition 1 (Geographical Features) *Geographical features of a POI include the business prosperity of the region where the POI is located, the distance of the POI, and the number of user's historical POIs in this region associated with each category.*

Then geographical features of POI j for user i could be represented with a multi-dimensional vector:

$$\mathbf{f}_{ij} = (prosperity(j), d(i,j), n_{i1}, ..., n_{iC})^T, \tag{4.1}$$

where $prosperity(j)$ is the prosperity of region where POI j is located and could be measured by the number of business services in this region. $d(i,j)$ represents the distance between the home of user i and POI j. n_{ic} is the number of POIs within category c visited by user i in this region. Note that more other geographical information could be included into $\mathbf{f}_{ij}$ if it is available for a particular data set.

User's temporal preference for check-ins follows a cyclic pattern [48, 49, 50]. Therefore, we divide the time into two parts: time slot in a day and day in a week. Specifically, time slot in a day includes morning, noon, afternoon, evening and midnight. While traditional methods simply regard each check-in as a location, we represent a check-in event with category and temporal information as follows.

Definition 2 (Check-in Event) *A check-in event is a tuple consisting of the category of POI, the check-in day in a week, and the check-in time slot in a day.*

For example, an check-in event can be represented as ⟨*Restaurant, Saturday, Evening*⟩, where *Restaurant* is the category. As can be seen, a check-in event essentially describes what a user is doing at a particular time. In above example, it indicates that the user is eating at a restaurant on Saturday evening.

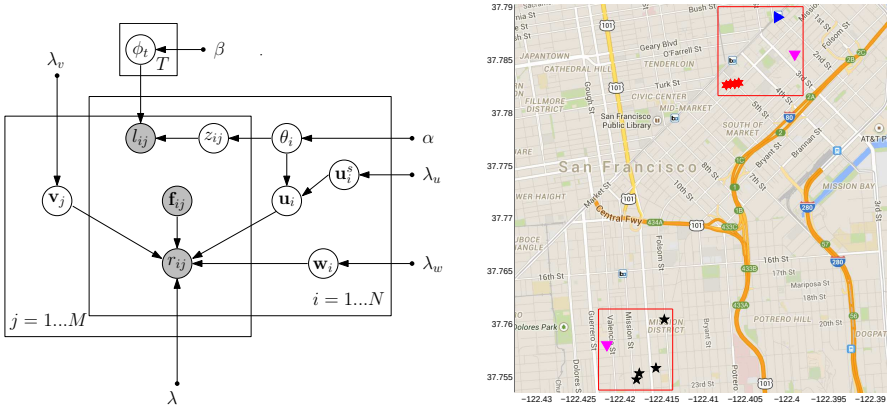


Figure 4.1: (a)The graphical model of STPMF. (b) An example of a user's historical check-ins. Red box represents a region, and each color indicates a category.

4.2.1.2 The General Framework.

We propose a general framework to generate accurate POI recommendation for N users over M locations. In reality, there are many latent factors affecting users' preference for POIs. Particularly, different from consuming movies, when users visit or check-in POIs, most of them would take into account the geographical factors [60] such as distance and region prosperity, in addition to other general factors such as

intrinsic property of POIs. Thus, based on matrix factorization method, we propose to model the rating (i.e. check-in frequency), denoted as $\hat{r}_{ij}$, of user i for POI j as a combination of the general preference and the geographical preference of user i for POI j, which is shown as:

$$\hat{r}_{ij} = \mathbf{u}_i^T \mathbf{v}_j + g_i \mathbf{w}_i^T \mathbf{f}_{ij}, \tag{4.2}$$

where $\mathbf{f}_{ij}$ represents the observed geographical features defined in section 4.2.1.1. $\mathbf{w}_i$ is the i-th user's latent preference for geographical features and is assumed to be drawn from a Gaussian distribution with zero-mean and covariance matrix as $\lambda_w^{-1}\mathbf{I}_G$, where G is the dimension of geographical feature. g_i is a weight indicating how much the i-th user's check-in decision is affected by geographical factors. The item $\mathbf{w}_i^T \mathbf{f}_{ij}$ captures the geographical effect on check-in decision. $\mathbf{u}_i$ and $\mathbf{v}_j$ are the D-dimensional user and location specific latent vectors respectively, both of which are drawn from Gaussian distribution. $\mathbf{u}_i^T \mathbf{v}_j$ is assumed to capture the i-th user's preference for the j-th POI's other intrinsic property, such as environment, price and service, that also affect the check-in decision.

Furthermore, user's general preference may change over time. Hence we argue that user's preference characteristics may consist of two parts: stationary preference which has nothing to do with time, and temporal preference that changes over time. Specifically, the i-th user's general preference could be decomposed into stationary preference $\mathbf{u}_i^s$ and temporal preference $\mathbf{u}_i^b$ as follows:

$$\mathbf{u}_i = \mathbf{u}_i^s + \mathbf{u}_i^b + \triangle_D, \tag{4.3}$$

where $\triangle_D$ is the D-dimensional Guassian noise. In other words, $\mathbf{u}_i$ can be regarded to be drawn from the Gaussian distribution with $\mathbf{u}_i^s + \mathbf{u}_i^b$ as mean and $\lambda_u^{-1}\mathbf{I}_D$ as

covariance matrix, which is shown in the following:

$$\mathbf{u}_i \sim \mathcal{N}(\mathbf{u}_i^s + \mathbf{u}_i^b, \lambda_u^{-1}\mathbf{I}_D). \tag{4.4}$$

We will adapt topic modeling technique to discover the temporal preference $\mathbf{u}_i^b$ based on the defined check-in events, which will be discussed in details in the following section. In addition, as mentioned in [46, 48], the location characteristics captured with $\mathbf{v}_j$ are inherent properties and do not change much as time goes. Therefore, we only model the dynamics for user preference. $\mathbf{u}_i^s$ is assumed to be drawn from a Gaussian distribution while $\mathbf{u}_i^b$ is a latent vector which could have various modeling methods according to application circumstance. We want to emphasize that the proposed model for incorporating geographical information is different from [54, 53, 60]. [53, 60] only model POI distance and [54] only considers an area's influence, all of which are not able to take other more geographical factors into consideration for users' check-in decision making process. However, the proposed framework is much more general, and could incorporate different kinds of geographical features into matrix factorization.

4.2.1.3 The STPMF Model.

Two recent works have been proposed to incorporate temporal effect into matrix factorization for improving the accuracy of recommendation [17, 48]. However, there are some shortcomings for these two methods. First, both of them utilize multiple user-specific latent vectors to model each individual's preference at different temporal state. Each temporal state usually corresponds to a time window. For example, [17] considers each day as a different temporal state. And as cyclic pattern is taken into account, [48] regards a pair of hour and day as a temporal state. Both of them have many different temporal states for each user. As a result, the number of observed ratings for fitting each user latent vector at each temporal state would significantly

decrease, and learning so many user-specific latent vectors for each individual makes recommender systems very inefficient. Second, both methods could not predict the rating of one user well when the corresponding temporal state does not appear in training set. But in fact, for the location check-in application, only a small set of ratings per user are observed that belong to a very limited number of temporal states. Thus these methods could not make reasonable predictions for users at new temporal states. All these limitations motivate us to find a more feasible and effective way to capture user's temporal preference.

If we take a close look at user's check-in records, we can find that there are some underlying topics that could explain the reason why users check in POIs at particular times. For example, users are very likely to frequently check in some similar locations such as bars or restaurants around evening at weekend for a similar purpose such as *party*. In other words, there will be many check-in events that have similar categories and close times and often appear together in user's check-in records. And a set of such check-in events essentially describe a purpose of check-in behavior. As each check-in event has both category and time information, such a check-in purpose (e.g., *party*) provides a possible interpretation for the reason why users would like to check in these POIs at a particular time. And each user will have different preference for check-in purposes.

Based on this assumption, for the user i, we analogize a check-in event l_{im} as a term and her historical check-ins $\{l_{im}|m = 1, ..., M_i\}$ as a document (where M_i is the number of check-in events for user i), and then employ topic modeling to model all observed check-ins. The learned topic will be a distribution over all check-in events. Each individual's check-ins are a mixture over a set of interpretable topics. As a check-in event includes check-in time, each individual's topic distribution naturally captures her temporal preference. Consequently, we develop a specific method namely Spatial-Temporal Probabilistic Matrix Factorization (STPMF) model. Its generative

process and graphical model are shown in Table 4.1 and Figure 4.1a respectively.

In the model, ϕ_t is the distribution over check-in events for the t-th topic, and drawn from $Dirichlet(\beta)$ prior. Each check-in event l_{im} is drawn from ϕ corresponding to its hidden topic z_{im} being drawn from the individual-specific mixture weights θ_i over T topics. Thus, based on Eq.(4.4), we assume that the user's general preference consists of a static one and temporal one, and is assumed to be drawn from the Gaussian distribution with $\mathbf{u}_i^s + \mathbf{X}\theta_i$ as the mean vector:

$$\mathbf{u}_i \sim \mathcal{N}(\mathbf{u}_i^s + \mathbf{X}\theta_i, \lambda_u \mathbf{I}_D), \tag{4.5}$$

where $\mathbf{X}$ is a matrix that is used to transfer user topical space into the user latent space, and $\mathbf{X}\theta_i$ indicates user's temporal preference. Also, we place the zero-mean spherical Gaussian prior on each entry of $\mathbf{X}$.

There are three advantages to adopt this method to capture users' temporal preference: (1) The extracted topic provides a novel interpretation for a user's check-in event; (2) The process of extracting topics from large numbers of check-in events is a kind of dimensionality deduction. Thus, we do not need to model multiple user-specific preferences at different temporal states for each individual; (3) It is able to address the prediction problem mentioned before, because each user has a different topic distribution learned in training data and not explicitly related to each specific temporal state. Thus unlike [17, 48], even though one users' some temporal states may be not observed in the training set, their learned topic distributions could be still used as their dynamic preference for the prediction.

4.2.1.4 Parameter Estimation.

We employ the Maximum-A-Posteriori (MAP) for the parameter estimation of STPMF model. Given the observed rating $\mathbf{R}$, geographical feature $\mathbf{f}$ and hyperparameters, maximizing the posterior is equivalent to maximizing the log-likelihood of

Table 4.1: The generative process for STPMF model.

1. For global context, draw each entry in matrix $x_{ij} \sim \mathcal{N}(0, \lambda_x^{-1})$.
2. For each user i,
   a. Draw topic proportion $\theta_i \sim Dir(\alpha)$.
   b. Draw user static preference $\mathbf{u}_i^s \sim \mathcal{N}(0, \lambda_s^{-1}\mathbf{I}_D)$.
   c. Draw user general preference $\mathbf{u}_i \sim \mathcal{N}(\mathbf{u}_i^s + \mathbf{X}\theta_i, \lambda_u^{-1}\mathbf{I}_D)$.
   d. Draw user graphical preference $\mathbf{w}_i \sim \mathcal{N}(0, \lambda_w^{-1}\mathbf{I}_G)$.
   e. Draw user graphical influence $g_i \sim \mathcal{N}(0, \lambda_g^{-1})$
   f. For each check-in event l_{im},
      i. Draw topic $z_{im} \sim Mult(\theta_i)$.
      ii. Draw check-in event $l_{im} \sim Mult(\phi_{z_{im}})$.
3. For each topic t, draw $\phi_t \sim Dir(\beta)$.
4. For each location j,
   a. Draw location latent vector $\mathbf{v}_j \sim \mathcal{N}(0, \lambda_v^{-1}\mathbf{I}_D)$.
5. For each user-location pair (i, j),
   a. Draw rating $r_{ij} \sim \mathcal{N}(\mathbf{u}_i^T\mathbf{v}_j + g_i\mathbf{w}_i^T\mathbf{f}_{ij}, \lambda^{-1})$.

latent parameters. Thus, the log-likelihood is obtained as follows:

$$\begin{aligned}\mathcal{L} = &-\lambda \sum_i^N \sum_j^M I_{ij}(r_{ij} - \mathbf{u}_i{}^T\mathbf{v}_j - g_i\mathbf{w}_i^T\mathbf{f}_{ij})^2 \\ &+ \sum_i^N \sum_m^{M_i} \log \sum_{t=1}^T \phi_{t,l_{im}}\theta_{it} + \sum_i^N \sum_t^T (\alpha_t - 1)\log\theta_{it} \\ &+ \sum_t^T \sum_l^L (\beta_l - 1)\log\phi_{tl} - ||\cdot||_F^2, \end{aligned} \tag{4.6}$$

s.t. $\forall k,\ 0 < \phi_{il}, \theta_{it} \leq 1,\ \sum_{l=1}^L \phi_{il} = 1,\ \sum_{t=1}^T \theta_{it} = 1$, where I_{ij} is the indicator function that is equal to 1 if the i-th user checks in the j-th location and equal to 0 otherwise. $|||\cdot||_F^2$ is denoted as:

$$\begin{aligned}||\cdot||_F^2 = &\lambda_u||\mathbf{U} - \mathbf{U}^s - \mathbf{X}\theta||_F^2 + \lambda_s||\mathbf{U}^s||_F^2 \\ &+ \lambda_v||\mathbf{V}||_F^2 + \lambda_w||\mathbf{W}||_F^2 + \lambda_x||\mathbf{X}||_F^2 + \lambda_g||\mathbf{g}||_F^2. \end{aligned} \tag{4.7}$$

As in Eq.(4.6) the summation over t for item $\phi_{t,l_{im}}\theta_{it}$ is within the log function, it is intractable for us to estimate θ by maximizing the log-likelihood. To overcome

this problem, We adopt the similar method proposed in [87] to obtain the relaxed lower-bound. Specifically, we first define $q(z_{im} = t) = \psi_{imt}$, and then apply Jensen's inequality. Finally, the object function with respect to θ has the following lower-bound:

$$\sum_{i}^{N}\sum_{m}^{M_i}\log\sum_{t=1}^{T}\phi_{t,l_{im}}\theta_{it} >= \sum_{i}^{N}\sum_{m}^{M_i}\sum_{t}^{T}\psi_{imt}[\log\theta_{it}\phi_{t,l_{im}} - \log\psi_{imt}],$$
$$s.t.\ \forall t,\ 0 < \psi_{imt} \leq 1, and \sum_{t=1}^{T}\psi_{imt} = 1. \tag{4.8}$$

Therefore, we can relax the object function by using Variational Inference method for latent variable Z. The lower-bound for the object function is obtained by integrating above equation back into Eq.(4.6):

$$\begin{aligned}\mathcal{L} = &-\lambda\sum_{i}^{N}\sum_{j}^{M} I_{ij}(r_{ij} - \mathbf{u}_i{}^T\mathbf{v}_j - g_i\mathbf{w}_i^T\mathbf{f}_{ij})^2 \\ &+ \sum_{i}^{N}\{\sum_{m}^{M_i}\sum_{t}^{T}\psi_{imt}[\log\theta_{it}\phi_{t,l_{im}} - \log\psi_{imt}] \\ &+ \sum_{t}^{T}(\alpha_t - 1)\log\theta_{it}\} + \sum_{t}^{T}\sum_{l}^{L}(\beta_l - 1)\log\phi_{tl} - ||\Theta||_F^2, \\ &s.t. \forall k, 0 < \phi_{il}, \theta_{it}, \psi_{imt} \leq 1, \sum_{l=1}^{L}\phi_{il} = 1, \sum_{t=1}^{T}\{\theta_{it},\ \psi_{imt}\} = 1.\end{aligned} \tag{4.9}$$

Alternating Least Squares (ALS) is a popular optimization method leading to more accurate parameter estimation and faster convergence. Thus, we utilize ALS method to compute each latent variable by fixing the other variables when maximizing the relaxed log-likelihood in Eq.(4.9). The updating equation for each variable of interest shown in Table 4.2 is obtained by setting its gradient of the relaxed log-likelihood to zero. However, it is difficult to get the close-form of variable θ. Therefore, we utilize the gradient-based searching algorithm to assist with searching its optimal solutions.

By fixing other variables, we could obtain the object function with respect to θ as:

$$\begin{aligned}\hat{\theta}_{it} = &\underset{\theta_{it}}{\operatorname{argmin}} \{\lambda_u(\mathbf{u}_i - \mathbf{u}_i^s - \mathbf{X}\theta_i)^T(\mathbf{u}_i - \mathbf{u}_i^s - \mathbf{X}\theta_i) \\ &- \sum\nolimits_{m=1}^{M_i} \psi_{imt} \log \theta_{it} - (\alpha_t - 1) log \theta_{it}\}, \\ &s.t.\ \forall t,\ 0 < \theta_{it} \leq 1,\ and \sum_{t=1}^{T} \theta_{it} = 1. \end{aligned} \tag{4.10}$$

To optimize θ, the gradient descent is employed to find the optimal solution with the following gradient:

$$\frac{\partial \mathcal{L}(\theta_{it})}{\partial \theta_{it}} = 2\lambda_u(\mathbf{X}\theta_i + \mathbf{u}_i^s - \mathbf{u}_i)(\mathbf{X})_t - \frac{\sum_{m=1}^{M_i} \psi_{imt} + \alpha_t - 1}{\theta_{it}}, \tag{4.11}$$

where $(\mathbf{X})_t$ represents the t-th column in matrix $\mathbf{X}$.

More details of the algorithm is shown in Table 4.2. In addition, $\mathbf{C}_i$ is a diagonal matrix with λI_{ij} $(j = 1, ..., M)$ as its diagonal elements, denoted as $\mathbf{C}_i = (\lambda I_{ij})_{j=1}^{M}$. Similarly, we denote $\mathbf{C}_j = (\lambda I_{ij})_{i=1}^{N}$, $\mathbf{F}_i = (g_i \mathbf{f}_{ij})_{j=1}^{G}$, $\mathbf{R}_{i,\backslash UV} = (r_{ij} - \mathbf{u}_i^T \mathbf{v}_j)_{j=1}^{M}$, and $\mathbf{R}_{i,\backslash Wf} = (r_{ij} - g_i \mathbf{w}_i^T \mathbf{v}_j)_{j=1}^{M}$.

4.2.2 Experimental Results

4.2.2.1 Datasets

In this section, we use two Foursquare data sets to evaluate the proposed STPMF model: (1) **Data 1** [88] contains the check-in history of users who live in the California, ranging from December 2009 to January 2013; (2) **Data 2** [89] contains the check-in data from September 2010 to January 2011. Each check-in record in both two data sets includes user ID, location ID and timestamp, where each location has latitude, longitude and category information. After merging the overlapped categories, we totally obtain 6 (or 7) different kinds of categories in data 1 (or data 2). Meanwhile, we remove users who have visited less than 5 locations and locations which are visited

Table 4.2: The algorithm for STPMF model.

Step 1: Randomly initialize variables.
Step 2: Execute E-Step and M-Step in each iteration repeatedly until the log-likelihood in Eq.(4.9) converges:

E-Step:

- $\mathbf{v}_j = (\mathbf{U}\mathbf{C}_j\mathbf{U}^T + \lambda_v\mathbf{I}_D)^{-1}\mathbf{U}\mathbf{C}_j\mathbf{R}_{j,\backslash Wf}$
- $\mathbf{u}_i = (\mathbf{V}\mathbf{C}_i\mathbf{V}^T + \lambda_u\mathbf{I}_D)^{-1}[\mathbf{V}\mathbf{C}_i\mathbf{R}_{i,\backslash Wf} + \lambda_u(\mathbf{u}_i^s + \mathbf{X}\theta_i)]$
- $\mathbf{w}_i = (\mathbf{F}_i\mathbf{C}_i\mathbf{F}_i^T + \lambda_w\mathbf{I}_G)^{-1}\mathbf{F}_i\mathbf{C}_i\mathbf{R}_{i,\backslash UV}$
- $\mathbf{u}_i^s = \frac{\lambda_u}{\lambda_u+\lambda_s}(\mathbf{u}_i - \mathbf{X}\theta_i)$
- $\mathbf{X} = (\lambda_u\theta\theta^T + \lambda_x\mathbf{I}_T)^{-1}\lambda_u(\mathbf{U} - \mathbf{U}^s)\theta^T$
- $\psi_{imt} = \theta_{it}\phi_{t,l_{im}}$
  Normalize ψ_{ijt}.
- $\phi_{tl} = \sum_{i,m}^{N,M_i}\psi_{imt}I(l_{im} = l) + \beta_l - 1$
  Normalize ϕ_{tl}.
- Update θ_{it}: Use gradient descent with constraints to find the optimal solution of θ_{it} with its object function shown in Eq.(4.10) and its gradient as Eq.(4.11), where the constraint is $s.t.\ \forall t,\ 0 < \theta_{it} \leq 1,\ and \sum_{t=1}^{T}\theta_{it} = 1$.

M-Step:

- $g_i = \frac{\lambda\sum_{j=1}^{M} I_{ij}(r_{ij} - \mathbf{u}_i^T\mathbf{v}_j)\mathbf{w}_i^T\mathbf{f}_{ij}}{\lambda_g + \lambda\sum_{j=1}^{M} I_{ij}(\mathbf{w}_i^T\mathbf{f}_{ij})^2}$

Step 3: After obtaining optimal $\hat{\mathbf{U}}$, $\hat{\mathbf{V}}$, $\hat{\mathbf{W}}$ and $\hat{\mathbf{g}}$, the predicted rating on i-th user for j-th location is calculated by: $r_{ij} = \hat{\mathbf{u}}_i^T\hat{\mathbf{v}}_j + \hat{g}_i\hat{\mathbf{w}}_i^T\mathbf{f}_{ij}$.

by less than 2 users. The data statistics for two data sets are reported in Table 4.3. We can observe that **Data 2** is nearly 10 times sparser than **Data 1**.

Table 4.3: The statistics of data sets.

Data Set	#User	#Location	#Checkin	Sparsity
Data 1	3,375	36,866	312,568	0.25%
Data 2	74,343	198,161	3,501,608	0.024%

In addition, we divide the space into a set of grids based on latitude and longitude. Each grid is regarded as a region. In our experiment, the size of region is about $5km \times 5km$ (i.e. 0.045 by 0.045 square degrees). We then crawl all venues for each individual region with Foursquare API. As Foursquare API returns at most 50 venues (locations) with a specified center point and searching radius, we propose a simple

method to crawl all venues in each region as follows. First, we get all venues in one specified region if Foursquare API returns less than 50 venues, and we end up searching. Otherwise we further divide this region into four small areas and repeat the above steps in each small area until each area contains less than 50 venues or its area size is less than $100m \times 100m$. Second, we merge all crawled venues in the same region according to the venue ID. We use the number of venues in each region to indicate its prosperity degree. Moreover, in the experiments we adopt the method [89] to locate user's home from her all historical check-ins for calculating the POI distance.

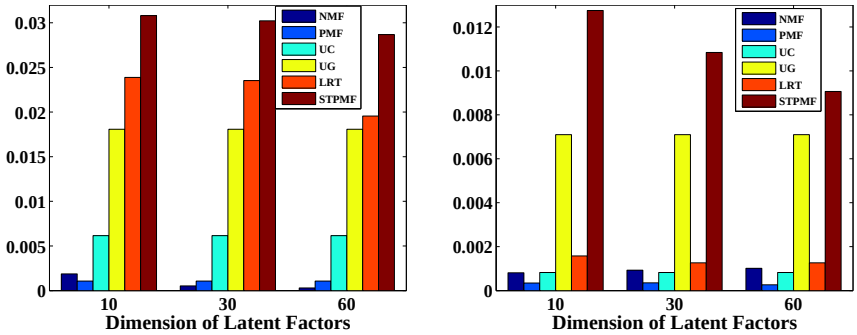
In recommendation system, we aim to recommend those unvisited locations for users. Thus, we split the training and testing data as follows: for each user, (1) aggregating the check-ins for each individual location; (2) sorting the location according to the first time that user checks in; (3) selecting the earliest 80% to train the model, and using the next 20% as testing.

4.2.2.2 Experimental Settings

Parameters λ and α are set as 0.001 and 1.01 respectively. Meanwhile, other parameters are specified as 0.01.

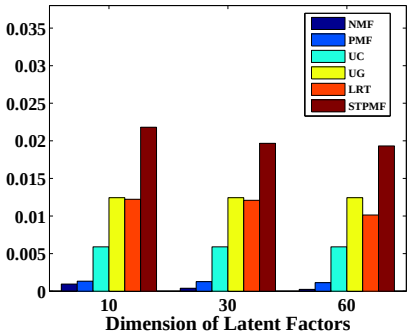
4.2.2.3 Evaluation Metrics

Precision@K and Recall@K are used to quantitatively evaluate the top-K recommendation performance. Meanwhile we also adopt MAP metric, mean of average precision (AP). AP is computed as $AP_i = \frac{\sum_{j=1}^{N} p(j) \times rel(j)}{\#relavant\ locations}$. Precision@K and Recall@K are calculated as $Precision@K = \frac{\sum_{\mathcal{M}_i \in \mathcal{M}} |\mathcal{T}_K(\mathcal{M}_i)|}{\sum_{\mathcal{M}_i \in \mathcal{M}} |\mathcal{R}_K(\mathcal{M}_i)|}$ and $Recall@K = \frac{\sum_{\mathcal{M}_i \in \mathcal{M}} |\mathcal{T}_K(\mathcal{M}_i)|}{\#relavant\ locations}$, respectively. Specifically, $\mathcal{R}_K(\mathcal{M}_i)$ are the top-K locations recommended to user i, $\mathcal{T}_K(\mathcal{M}_i)$ denotes all truly relevant locations among $\mathcal{R}_K(\mathcal{M}_i)$, $\mathcal{M}$ represents the set of locations in the testing, j is the position in the rank list, N is the number of returned items in the list, $p(j)$ is the precision of a cut-off rank list from 1 to j, and $rel(j)$ is

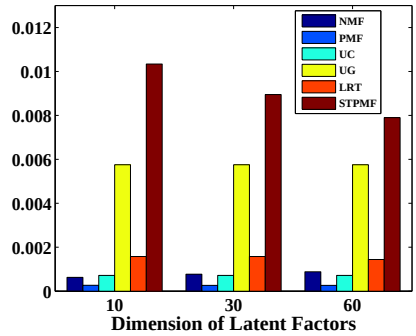


(a) Precison@5 on Data 1.

(b) Precison@5 on Data 2.



(c) Precison@10 on Data 1.



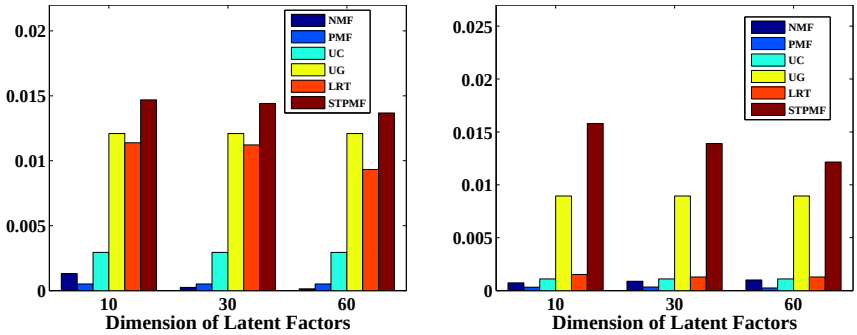
(d) Precison@10 on Data 2.

Figure 4.2: Performance comparison in term of precision with top 5 and 10 on two data sets.

an indicator function. The term **relevant locations** is defined as the locations in the testing.

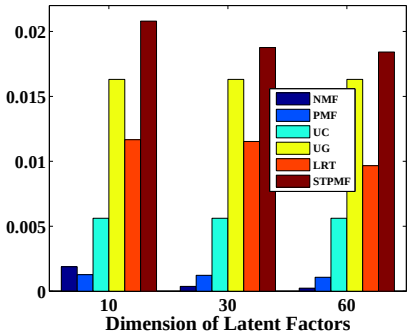
4.2.2.4 Baseline Methods

To demonstrate the effectiveness of our STPMF model, we compare it with five baseline methods: (1) **LRT** [48] that assumes users have different preference in each temporal state and use multiple user-specific latent vectors for different temporal states; (2) **UG** [60], that considers both geographical effect based on power law characteristics, and user interest based on user-based collaborative filtering for POI recommendation; (3) **PMF** [14], that assumes the user and location latent vectors to be drawn from Gaussian distribution and estimates a user's preference on a location as the dot product of user-specific and location-specific latent vector; (4) **NMF** [10], a Bayesian non-negative matrix factorization algorithm that places exponential priors

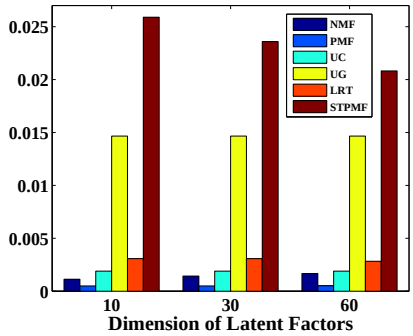


(a) Recall@5 on Data 1.

(b) Recall@5 on Data 2.



(c) Recall@10 on Data 1.



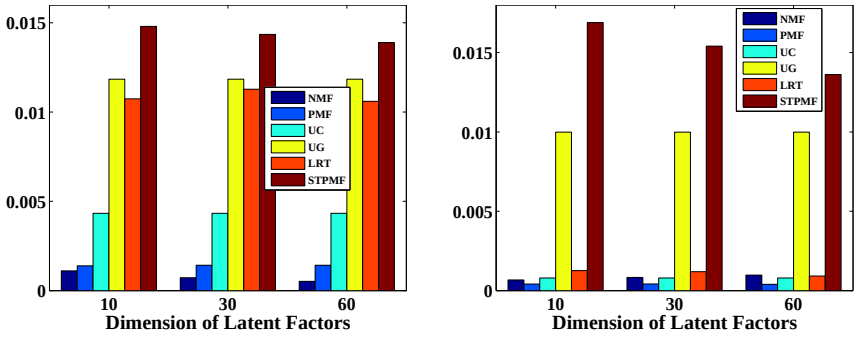
(d) Recall@10 on Data 2.

Figure 4.3: Performance comparison in term of recall with top 5 and 10 on two data sets.

on latent vectors and utilizes the Gibbs sampler to approximate the posteriors of latent variables; (5) **UCF**, user-based collaborative filtering with Pearson correlation as the similarity measurement.

4.2.2.5 Top-K Recommendation Performance

We will evaluate our STPMF model with the baseline methods. Specifically, the topic number is set to be the same value as the latent feature number. The performance of Precision@K, Recall@K and MAP for different models on two data sets are reported in Figure 4.2, 4.3, and 4.4. Totally, STPMF performs the best among all methods on these three metrics; while NMF, PMF and UCF are nearly the worst. It occurs likely because they do not model the geographical and temporal effects. UG is superior than LRT specifically on Data 2 due to (1) Most of users would visit a nearby location, which could be modeled with distance property. When recommending top



(a) (b)

Figure 4.4: Performance comparison in terms of MAP on Data 1 and Data 2.

K locations among all locations, UG would have more accurate results. (2) When data becomes much sparser, each user's user-specific latent vector for each different state in LRT would be fitted by much less observed ratings. It also illustrates the geographical effects play an more important role in POI recommendation. STPMF appropriately leverages multiple geographical features to learn user's interests, as a result it improves the recommendation accuracy. It can be observed that the performance on data 1 is worse than the one on data 2 (which is much sparser) possibly due to that a much smaller set of the observed data is employed to train the model in Data 1.

4.2.3 Conclusion

In Section 4.2, we propose a novel model for POI recommendation that incorporates both geographical influence and temporal effect into matrix factorization. We model user's preference on a POI as the summation over her geographical preference for the POI and her general interest in the POI. User's geographical preference is captured by leveraging multiple geographical features. Meanwhile user's general preference is decomposed into a static one and a dynamic one. Specifically, we analogize a check-in event as a term and one user's check-ins as a document, and employ topic modeling to model all observed check-ins. As the check-in event includes the check-in time and each topic is characterized by a distribution over check-in events, the learned topic

distribution naturally captures the dynamics of user's general interest. To this end, a specific model namely STPMF is proposed. Finally, the experimental results on real-world data sets demonstrate the improvements of the proposed model.

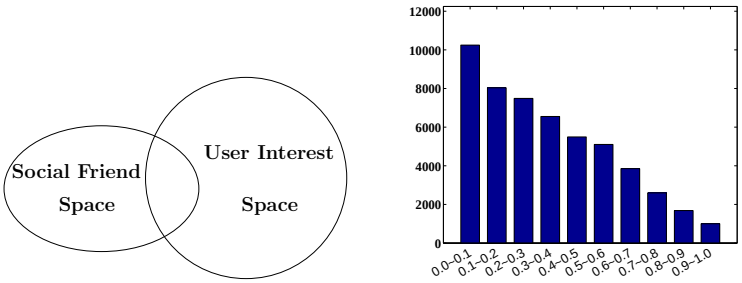
4.3 Modeling Geo-Social Influence

In this section, we study geographical and social influence for POI recommendation in LBSNs. By carefully examining the real-world check-in data, we observe that users' check-ins mainly consist of two groups of check-ins. First, 30% of check-ins are those that have been checked-in by direct friends. In other words, many users like to repeat their friends' check-ins. Second, the rest of check-ins are very similar as users' historically checked-in POIs. Therefore, we propose to divide the whole recommendation space into two parts: social friend space and user interest space. The social friend space refers to the set of POI candidates that users' friends have checked-in before. The user interest space denotes the set of POI candidates that have not been visited by their friends before, but are very similar to users' historical check-ins. We then develop different approaches to model these two spaces separately.

4.3.1 Methodology

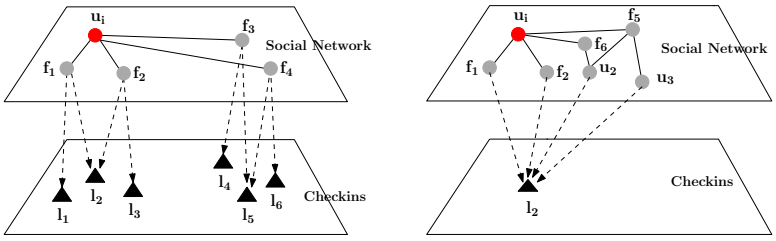
Our goal is to recommend the new (unvisited) locations for users. Based on the assumption that a new recommended POI for one user is either one of her friends' historical ones or similar to her own historical ones, we divide the whole recommendation space into social friend space and user interest space. In this section, we first introduce the social friend space and design a new Social Friend Probabilistic Matrix Factorization (SFPMF) model. Then we propose another novel User Interest Probabilistic Matrix Factorization (UIPMF) model for user interest space. Finally, we present the recommendation strategies and models' estimations.

Notations. Suppose there are totally N users and M locations. $\mathcal{M}$ denotes the set of all locations, i.e., $|\mathcal{M}| = M$. $\mathcal{M}_i$ then refers to the set of all locations checked-in by



(a) (b)

Figure 4.5: (a) Relationship between social friend space and user interest space. (b) The x-axis is the ratio of check-ins visited by friends; the y-axis is the number of corresponding users.



(a) (b)

Figure 4.6: (a) An example of user's social networks and check-ins. The circle is user and the triangle is location. The red circle is the target user. The solid line indicates friendship and the dashed line indicates check-in behavior. (b) An example of users checking-in the same location and their social networks.

user i , and $\mathcal{M}_i^{c_j}$ the set of locations that user i checked-in before location j and have the same category as c_j. $\mathcal{F}_i$ is the set of all friends of user i. Ψ_j denotes the set of all users who have checked-in location j. Γ_i represents the set of all locations checked-in by friends of user i. The terms **location** and **POI** are used interchangeably. **Friends** indicate **direct friendship** in Section 4.3.

4.3.1.1 Social Friend Space

In this section, we introduce the framework of SFPMF model, and present a P^3MF model to compute the check-in preference propagation probability for users.

4.3.1.1.1 The SFPMF Framework

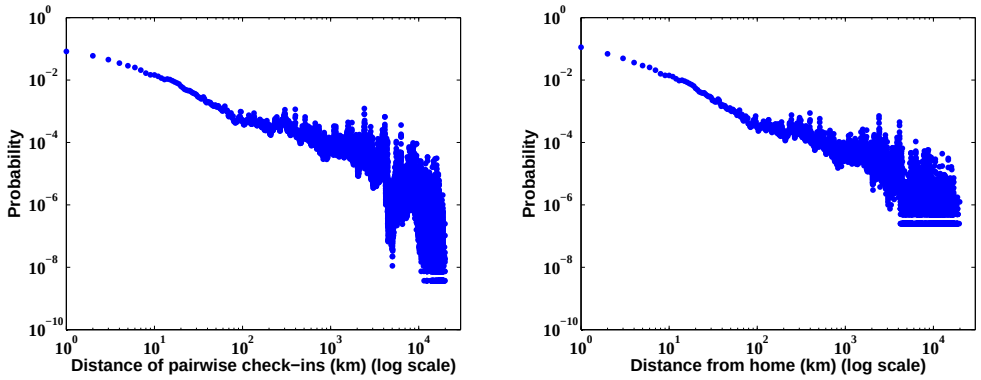
Many works have demonstrated the importance of social network in recommender system [67, 62]. To examine the influence of friends on users' check-in behaviors, we

depict the histogram of the ratio of check-ins that repeat friends' historical ones in Figure 4.5b. There are two surprising observations: (1) Over 50% users like to repeat those locations checked-in by their friends prior to their first check-in at them; (2) More than 30% check-ins are those that have been visited by friends. The results exhibit an important check-in behavior trend that users are probable to choose a POI from a set of POIs that her friends have visited before. Therefore, we believe the social network is a significant factor that affects user's decision on POIs. Furthermore, it is also crucial to recommend a new POI for one user from a collection of POIs having been checked-in by her friends, because it is able to encourage users to have more check-ins due to the trust to their friends and the similar POI taste as them. The inherent characteristics of check-in data and the evident benefit motivate us to recommend users with their friends' historically checked-in POIs. The problem in social friend space can be formally defined as:

Definition 3 (Problem in Social Friend Space) *Given a set of candidate locations* Γ_i*, i.e.* $\{l : l \in (\cup_{k \in \mathcal{F}_i} \mathcal{M}_k) \setminus \mathcal{M}_i)\}$*, which have been checked-in by the friends of user* i *but are new for her, the prediction is to find the location that user* i *would most likely prefer to check-in at the next time.*

For example, Figure 4.6a shows that the target user u_i has friends $\{f_1, f_2, f_3, f_4\}$ who have checked-in locations $\{l_1, l_2, l_3, l_4, l_5, l_6\}$, and these locations are never visited by the target user before. We aim to predict the probabilities of these POIs that she would check-in and recommend the POI with the highest probability for her.

In social friend space, we assume one user would repeat a location that her friends have checked-in before mainly due to the preference propagation in the whole social network. For each location j, the set of visitors Ψ_j are regarded as check-in preference injectors. They would likely tell their friends about this location, who might also tell their friends, and finally "everyone" knows. Thus, a check-in event is propagated in the whole network in such word-of-mouth way with these visitors as the initial



(a) (b)

Figure 4.7: Probability of check-ins as a function of distance of pairwise check-in distance (Left) and from home (Right).

preference injectors. For example, in Figure 4.6b, $\{f_1, f_2, u_2, u_3\}$ are the visitors of location l_2 and will propagate the preference for this location in the whole network. The target user u_i may be influenced by f_1, f_2, u_2 and u_3 with different probabilities. Specifically, u_i knows the location l_2 from u_2 likely due to (1) She hears from u_2 directly although they are not friends explicitly in the online social network but they are real friends off line; (2) u_2 tells f_6, and f_6 tells her then. *The weight that the user is influenced by the check-in event about a POI naturally determines the probability that she prefers to check-in this POI.* Based on this analysis, we propose the Social Friend Probabilistic Matrix Factorization (SFPMF) model. Let us define P_{iv}^I as the probability that the preference is propagated from the user v to the user i. Therefore the probability that the user i chooses to check-in the location j is obtained by:

$$P_{ij}^S = 1 - \prod_{v \in \Psi_j} (1 - P_{iv}^I). \tag{4.12}$$

Note that the influence cannot be simply computed as the summation over all the influences propagated by the initial injectors because these injectors may have correlations [90].

Different from traditional online product consuming, one user has a very small

chance to go to check-in a far away location due to the limited transportation even though she is interested in it. For instance, a user living at California would not go to check-in a restaurant in New York for the sake of long distance. In the example shown in Figure 4.6a, user u_i has more chance to visit the locations in the left side than those in the right side, because the locations in the left side are much closer to her. Thus, a POI's distance significantly affects the user's check-in decision process. Some works [60] propose to leverage a power law distribution to model check-in probability and distance of any pair of visited POIs based on the observation shown in Figure 4.7a. However, computing the distance of each POI with user's all historical POIs is inefficient, especially when the number of POIs is tremendous. To address this issue, we first adopt the method in [89] to locate user's home location from her all historical check-ins. We find that user's check-in probability and the distance between POI and her home also follow a power law distribution shown in Figure 4.7b. Thus let us define the probability of a user to check-in a d-km far away POI as:

$$Pr(d) = a \cdot d^b, \tag{4.13}$$

where a and b are the parameters of power law distribution, and could be learned by maximum likelihood estimation. Then the probability of user i to check-in location j due to the geographical influence is defined as:

$$P_{ij}^G = Pr(d(j, h_i)), \tag{4.14}$$

where h_i is the home location of user i, and $d(j, h_i)$ indicates the distance between the POI j and the home of user i.

Table 4.4: The objective functions for the proposed two models.

$$\underset{\mathbf{X},\mathbf{Y}}{\operatorname{argmin}} \lambda_z \sum_{i=1}^{N}\sum_{v=1}^{N} I_{iv}(z_{iv} - \beta \mathbf{x}_i^T \mathbf{y}_v + (1-\beta) \sum_{f \in \mathcal{F}_{v,-i}} g_{vf} \mathbf{x}_i^T \mathbf{y}_f)^2 + \lambda_x ||\mathbf{X}||_F^2 + \lambda_y ||\mathbf{Y}||_F^2 \quad (4.16)$$

$$\underset{\mathbf{U},\mathbf{V}}{\operatorname{argmin}} \lambda_r \sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(r_{ij} - \alpha \mathbf{u}_i^T \mathbf{v}_j + (1-\alpha) \sum\nolimits_{l \in \mathcal{M}_i^{c_j}} s_{jl} \mathbf{u}_i^T \mathbf{v}_l)^2 + \lambda_u ||\mathbf{U}||_F^2 + \lambda_v ||\mathbf{V}||_F^2 \quad (4.17)$$

Then Eq.(4.12) is refined with the geographical influence:

$$P_{ij}^S \propto Pr(d(j, h_i))(1 - \prod_{v \in \Psi_j} (1 - P_{iv}^I)). \quad (4.15)$$

Specifically, we further introduce a Preference Propagation Probabilistic Matrix Factorization (P^3MF) model to compute the probability P_{iv}^I in the following section.

4.3.1.1.2 The P^3MF Model

Different from traditional probabilistic matrix factorization [14], we construct a user-user matrix, where each rating z_{iv} indicates the preference propagated from user v to user i. The observed rating z_{iv} is a function of frequency that user i repeats the check-ins of her friend v, which will be discussed in Section 4.3.2.4.1. Let $\mathbf{X} \in \mathbb{R}^{Z \times N}$ and $\mathbf{Y} \in \mathbb{R}^{Z \times N}$ be the latent user and factor feature metrics, with column vectors $\mathbf{x}_i$ and $\mathbf{y}_v$ representing the Z-dimensional user-specific and factor-specific feature vectors of user i and user v, respectively. Specifically, the factor feature vector captures the properties of user v, such as age and activity level, and the user feature vector indicates the user's preference for corresponding properties.

The process of the preference propagation from user v to user i consists of two parts: (1) User v influences user i directly; (2) The friends of user v (not including user i if they are also friends) are influenced by her, and then affect user i. To model this propagation process, we define the predicted rating $\hat{z}_{iv}$ that user i is influenced

by user v as:

$$\hat{z}_{iv} = \beta \mathbf{x}_i^T \mathbf{y}_v + (1-\beta) \sum_{f \in \mathcal{F}_{v,-i}} g_{vf} \mathbf{x}_i^T \mathbf{y}_f, \tag{4.18}$$

where $\beta \in [0,1]$ is the tuning parameter to control the direct influence from friend v, and $\mathcal{F}_{v,-i}$ indicates the set of friends of user v excluding user i. g_{vf} is the information transmission probability from user v to her friend f. As the connection between friends is undirected, we define g_{vf} as:

$$g_{vf} = \frac{1}{|\mathcal{F}_{v,-i}|}.$$

It is worth noting that g_{vf} is different from z_{fv}, where the latter one is the preference propagated from user v to user f which contains both direct and indirect influence ways.

The rating is assumed to be drawn from a Gaussian distribution with the mean as shown in Eq.(4.18) and the precision as λ_z. We also place zero-mean spherical Gaussian priors on user and factor feature vectors with the precision as λ_x and λ_y, respectively. Therefore, based on a Maximum-a-Posteriori (MAP) estimation, we obtain the objective function about $\mathbf{X}$ and $\mathbf{Y}$ in Eq.(4.16), where $||\cdot||_F$ denotes the Frobenius norm, and I_{iv} is the indicator function that is equal to 1 if user i checks-in a POI that her friend v checked-in before and equal to 0 otherwise.

In P^3MF model, we address the preference propagation influence in social network based on user's check-in behavior. The preference propagated from user v to user i is dependent on the influence of himself and her friends. Recursively, the influence of the friend is further affected by her friends. Different from [65] which focuses on modeling the trust propagation for the user feature vector, we directly factorize the preference propagation influence into latent user and factor feature vectors. In addition, the check-in preference propagation is also affected by the geographical distance, i.e. the

closer two users live physically, the more possibly they will interplay. Therefore, the check-in preference P_{iv}^I propagated from user v to user i is obtained based on $\hat{z}_{iv}$ and geographical influence in Eq.(4.14):

$$P_{iv}^I \propto Pr(d(h_i, h_v))g(\hat{z}_{iv}), \tag{4.19}$$

where $g(.)$ is logistic function to bound the value to $[0, 1]$, and $Pr(.)$ is the geographical influence.

4.3.1.2 User Interest Space

In user interest space, we recommend users with the POIs that have not been visited by their friends before but are similar to their own historical check-ins. Therefore, the problem is formally defined as:

Definition 4 (Problem in Social Friend Space) *Given a set of candidate locations $\{l : l \in \mathcal{M} \setminus (\Gamma_i \cup \mathcal{M}_i)\}$, which have not been checked-in by the friends of user i and are new for her, the prediction is to find the location that user i would most likely prefer to check-in at the next time.*

Probabilistic matrix factorization factorizes the observed rating into user and location latent space, and leverages them for rating prediction [14]. Similarly we have a user-location matrix, where the observed rating R_{ij} is a function of frequency that user i checked-in the location j. Let $\mathbf{U} \in \mathbb{R}^{D \times N}$ and $\mathbf{V} \in \mathbb{R}^{D \times M}$ be the latent user and location feature matrices, with column vectors $\mathbf{u}_i$ and $\mathbf{v}_j$ representing the D-dimensional user-specific and location-specific feature vectors of user i and location j, respectively. In traditional PMF, it assumes user's preference for a POI is the dot product of this user's and this POI's latent vectors. However, it ignores the strong correlations among user's all POIs. A user may prefer to choose a POI that is very similar to her historical ones. Therefore, we propose a User Interest Probabilistic Matrix Factorization (UIPMF) model to characterize the user's preference for a POI

by using her preferences for the historical POIs that have the same category as this POI. In other words, user i has her special preference for the POI j, and at the same time, she is also influenced by her historical POIs that have the same category as this POI. Thus, the predicted rating denoted as $\hat{r}_{ij}$ of user i for POI j is defined as:

$$\hat{r}_{ij} = \alpha \mathbf{u}_i^T \mathbf{v}_j + (1-\alpha) \sum\nolimits_{l \in \mathcal{M}_i^{c_j}} s_{jl} \mathbf{u}_i^T \mathbf{v}_l, \tag{4.20}$$

where $\alpha \in [0, 1]$ is the tuning parameter. The closer a pair of POIs are, the stronger correlation should be taken into account for a user's POI decision making process. Hence, the similarity between POI j and POI l denoted as s_{jl} is measured by leveraging power law property in Eq.(4.13) and normalized as:

$$s_{jl} = \frac{Pr(d(j,l))}{\sum_{p \in \mathcal{M}_i^{c_j}} Pr(d(j,p))},$$

where $d(j,l)$ is the distance between POI j and POI l.

Similarly, the rating is drawn from a Gaussian distribution with the mean as shown in Eq.(4.20) and the precision as λ_r. We also place zero-mean spherical Gaussian priors on user and location feature vectors with the precision as λ_u and λ_v, respectively. Therefore, the object function is obtained through Maximum-a-Posteriori estimation in Eq.(4.17).

As discussed earlier, the check-in decision making process of user i on POI j is significantly affected by the POI's geographical distance. Thus, similar to Eq.(4.15), the probability that user i prefers to check-in POI j is given as:

$$P_{ij}^U \propto P_{ij}^G g(\hat{r}_{ij}). \tag{4.21}$$

4.3.1.3 Strategies for POI Recommendations

In this section, we propose that the recommendation space could be divided into two parts: social friend space and user interest space, and their relationship is shown in Figure 4.5a. Evidently, these two spaces might have overlap because the POIs that friends have checked-in are also possibly similar to users' historical check-ins. Thus, we adopt the following two strategies for POI recommendations.

- **Separated Recommendation.**

  We build two different recommender systems. One is to adopt SFPMF model to recommend POIs that users' friends have checked-in before. Another one is to adopt UIPMF model to recommend POIs that have not been visited by friends but are very similar to their historical check-ins.

- **Integrated Recommendation.**

  We propose an Integrated Social Friend and User Interest model (ISU) to integrate SFPMF model and UIPMF model for recommendation in the whole space. The integrated probability P_{ij}^R that user i will check-in POI j is formally defined as follows:

$$P_{ij}^R = \gamma P_{ij}^S + (1-\gamma) P_{ij}^U, \tag{4.22}$$

  where $\gamma \in [0,1]$ is the tuning parameter to align two recommendation spaces. Specifically, P_{ij}^S is computed only on the candidate POIs that user' friends have checked-in, while P_{ij}^U is calculated for all POIs.

4.3.1.4 Parameter Estimation

Both P^3MF model and UIPMF model are matrix factorization based models, and their objective functions are given in Eq.(4.16) and Eq.(4.17). Alternating Least Squares (ALS) is a popular optimization method with accurate parameter estimation and fast convergence rate. Thus, we utilize ALS method to compute each latent

variable by fixing the other variables when minimizing the object function. As two models have similar formulations, their optimizations are also similar. The optimization process is executed as follows:

- Randomly initialize each variable of interest.
- Update each of them with the updating equation iteratively until the object function converges.

The Optimization for the P^3MF Model: For the simplicity of inference, we first define the following variables:

$$\tilde{\mathbf{y}}_v^i = \beta \mathbf{y}_v + (1-\beta) \sum_{f \in \mathcal{F}_{vi}} g_{vf} \mathbf{y}_f,$$
$$w_\beta = \beta^2 + (1-\beta)^2 \sum_{f \in \mathcal{F}_{vi}} g_{vf}^2,$$
$$\bar{z}_{iv} = z_{iv} - \beta \mathbf{x}_i^T \mathbf{y}_v - (1-\beta) \sum_{f \in \mathcal{F}_{vi}} g_{vf} \mathbf{x}_i^T \mathbf{y}_f,$$
$$f(\bar{z}_{iv}) = \beta \bar{z}_{iv} + (1-\beta) \sum_{f \in \mathcal{F}_{vi}} g_{vf} \bar{z}_{if},$$

where $\mathcal{F}_{vi}$ is the set of common friends of user v and i. Then the updating equations for $\mathbf{X}$ and $\mathbf{Y}$ are obtained as:

$$\mathbf{x}_i = [\lambda_x \mathbf{I}_Z + \lambda_z \sum_{v=1}^{N} I_{iv} \tilde{\mathbf{y}}_v^i \tilde{\mathbf{y}}_v^{iT}]^{-1} \lambda_z \sum_{v=1}^{N} I_{iv} z_{iv} \tilde{\mathbf{y}}_v^i,$$
$$\mathbf{y}_v = [\lambda_z \mathbf{I}_Z + \lambda_z \sum_{i=1}^{N} I_{iv} w_\beta \mathbf{x}_i \mathbf{x}_i^T]^{-1} \lambda_z \sum_{i=1}^{N} I_{iv} [f(\bar{z}_{iv}) + w_\beta \mathbf{x}_i^T \mathbf{y}_v] \mathbf{x}_i.$$

The Optimization for the UIPMF Model: We can obtain the similar defini-

tions about $\tilde{\mathbf{v}}_j^i$, w_α, $\bar{r}_{ij}$ and $f(\bar{r}_{ij})$:

$$\tilde{\mathbf{v}}_j^i = \alpha \mathbf{v}_j + (1-\alpha) \sum_{l \in \mathcal{M}_i^{c_j}} s_{jl} \mathbf{v}_l,$$

$$w_\alpha = \alpha^2 + (1-\alpha)^2 \sum_{l \in \mathcal{M}_i^{c_j}} s_{jl}^2,$$

$$\bar{r}_{ij} = r_{ij} - \alpha \mathbf{u}_i^T \mathbf{v}_j - (1-\alpha) \sum_{l \in \mathcal{M}_i^{c_j}} s_{jl} \mathbf{u}_i^T \mathbf{v}_l,$$

$$f(\bar{r}_{ij}) = \alpha \bar{r}_{ij} + (1-\alpha) \sum_{l \in \mathcal{M}_i^{c_j}} s_{jl} \bar{r}_{il}.$$

Then the updating equations for $\mathbf{U}$ and $\mathbf{V}$ are shown as:

$$\mathbf{u}_i = [\lambda_u \mathbf{I}_D + \lambda_r \sum_{j=1}^{M} I_{ij} \tilde{\mathbf{v}}_j^i \tilde{\mathbf{v}}_j^{iT}]^{-1} \lambda_r \sum_{j=1}^{M} I_{ij} r_{ij} \tilde{\mathbf{v}}_j^i,$$

$$\mathbf{v}_j = [\lambda_v \mathbf{I}_D + \lambda_r \sum_{i=1}^{N} I_{ij} w_\alpha \mathbf{u}_i \mathbf{u}_i^T]^{-1} \lambda_r \sum_{i=1}^{N} I_{ij} [f(\bar{r}_{ij}) + w_\alpha \mathbf{u}_i^T \mathbf{v}_j] \mathbf{u}_i.$$

4.3.2 Experimental Results

In this section, we will evaluate our models with the real-world dataset.

4.3.2.1 The Experimental Setup

Dataset. In the experiments, we use Gowalla dataset [66] to evaluate the performance of our models, which contains check-in data ranging from January 2009 to August 2010. Each check-in record in the dataset includes user ID, location ID and timestamp, where each location has latitude, longitude and category information. Meanwhile the dataset has undirected friendship information. Specifically, we have the creation timestamp for each friendship, which is different from most of datasets used in recent research works. In addition, we remove users who have visited less than 5 locations and more than 1000 locations, and locations which are visited by less than 5 users. The data statistics are shown in Table 4.5.

Table 4.5: The statistics of data sets.

#User	#Location	#Checkin	Sparsity
61,578	178,062	3,257,029	0.0297%
#Train	#Test	#Test (SFS)[1]	#Test (UIS)[2]
2,581,882	675,147	167,546	507,601

In recommendation system, we aim to recommend those unvisited locations for users. Therefore, we split the training and testing data as follows: for each individual user, (1) aggregating the check-ins for each individual location; (2) sorting the location according to the first time that user checks in; (3) selecting the earliest 80% to train the model, and using the next 20% as testing. With the dynamic information, we use the social network at the end date of training data for both training and testing. Specifically, there are on average 8.29 friends for each user, and for those users who have friends, there are on average 556.32 locations that their friends have visited before. The observed rating for SFPMF model is a function of frequency that user repeats the check-ins of her friends, and we then obtain $372,502$ ratings in the training.

Experimental Settings. In the experiments, the parameter β, α and γ are set as 0.1, 0.8, and 0.01, respectively. The parameters λ_t and λ_z are set as 0.0001 and the other regularization parameters are set as 0.01. The dimensions of latent factors (i.e. Z and D) in SFPMF and UIPMF models are set as the same. We discuss the rating conversion methods for MF based models in Section 4.3.2.4.1.

4.3.2.2 Evaluation Metrics

As POI recommender system only recommends the limited POIs for users, we quantitatively evaluate recommendation models in terms of top-K recommendation performance i.e. Precision@K and Recall@K metrics. We also adopt MAP metric, the mean of the average precision (AP) over all locations in the testing, to evaluate

[1] We select those check-ins in the test which friends have checked-in before to evaluate models in social friend space.

[2] We select those check-ins in the test which friends have not checked-in before to evaluate models in user interest space.

models' performance. Formally, they are defined as:

$$Precision@K = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathcal{S}_i(K)\cap T_i}{K},$$

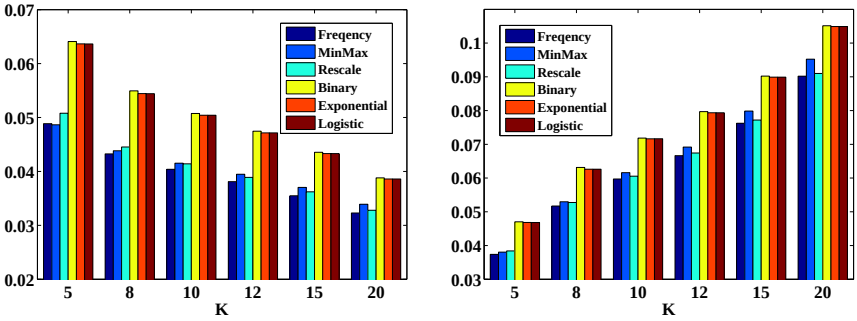
$$Recall@K = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathcal{S}_i(K)\cap \mathcal{T}_i}{|\mathcal{T}_i|},$$

$$MAP = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{j=1}^{\hat{M}_i} p(j)\times rel(j)}{|\mathcal{T}_i|},$$

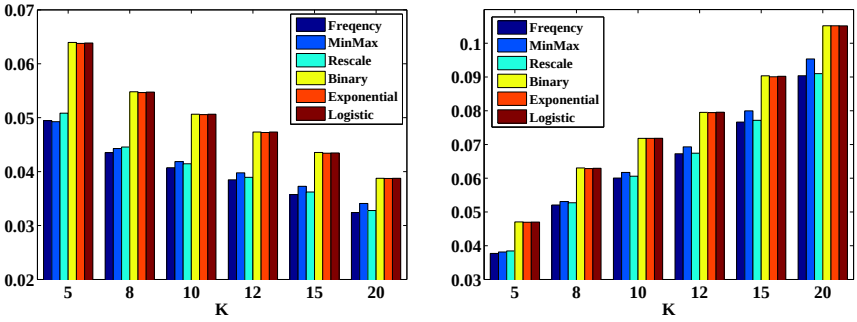
where $\mathcal{S}_i(K)$ is a set of top-K POIs recommended to user i excluding those POIs in the training, $\mathcal{T}_i$ is a set of locations that are checked-in by user i in the testing. $\hat{\mathcal{M}}_i$ is the number of the returned locations in the list for user i, $p(j)$ is the precision of a cut-off rank list from 1 to j, and $rel(j)$ is an indicator function that equals to 1 if the location is appearing in the testing, otherwise equals to 0.

4.3.2.3 Baseline Methods

To comparatively demonstrate the effectiveness of the proposed models, we compare them with five recommendation models: (1) **USG** [60], taking geographical influence, social network and user interest into account for POI recommendation; (2) **LOCABAL** [69], capturing two types of social relations, i.e. the local friends and the users with high global reputations, for recommendation based on matrix factorization; (3) **RegPMF** [67], assuming that users and their friends share similar interests in the preference and placing a social regularization term on learning latent user feature vectors; (4) **PMF** [14] that assumes the user and location latent vectors to be drawn from Gaussian distribution and estimates a user's preference for a location as the dot product of user-specific and location-specific latent vector; (5) **UC**, user-based collaborative filtering that adopts cosine similarity as the similarity measurement between users.



(a) Precison@K (10-D) (b) Recall@K (10-D)



(c) Precision@K (60-D) (d) Recall@K (60-D)

Figure 4.8: Performance comparison for different rating conversions on UIPMF with different dimensions.

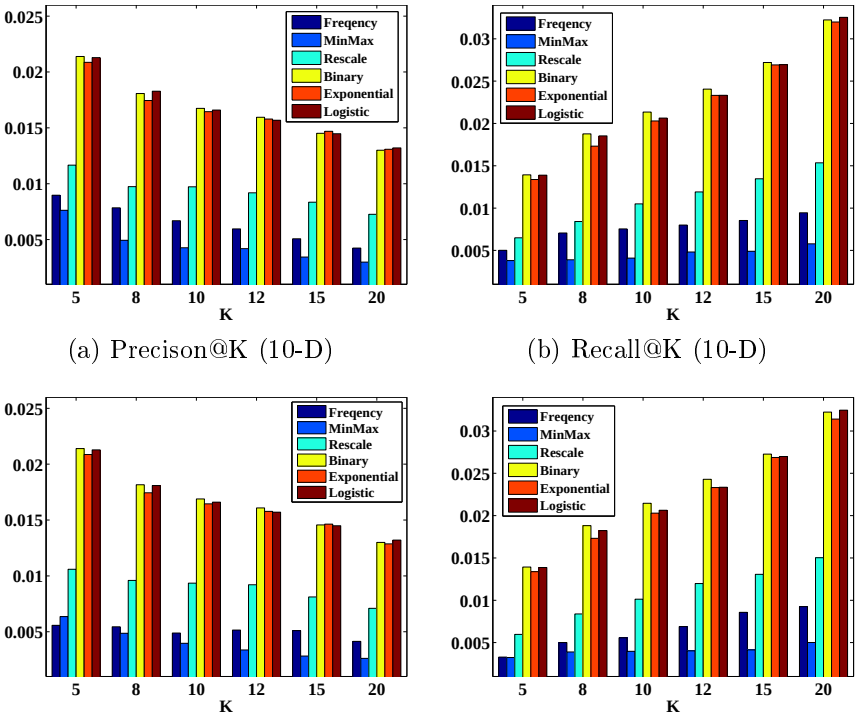
4.3.2.4 Performance Comparison

First, we evaluate different rating conversion methods for MF based models, and then compare the performance of the proposed models with baseline methods in social friend space, user interest space and the whole recommendation space.

4.3.2.4.1 Performance Comparisons of Rating Conversion Methods

In the literature, various rating conversion methods are proposed to fit Matrix Factorization (MF) based models for POI recommendation due to the bias of check-in data (i.e. majority ratings are very small and small percentages of ratings are extremely high). We formally compare the following six methods with MF based models:

- **Logistic:** Logistic function $\frac{1}{1+(e^x)^{-1}}$ is commonly used in recommendation to map each matrix entry into $[0, 1]$.



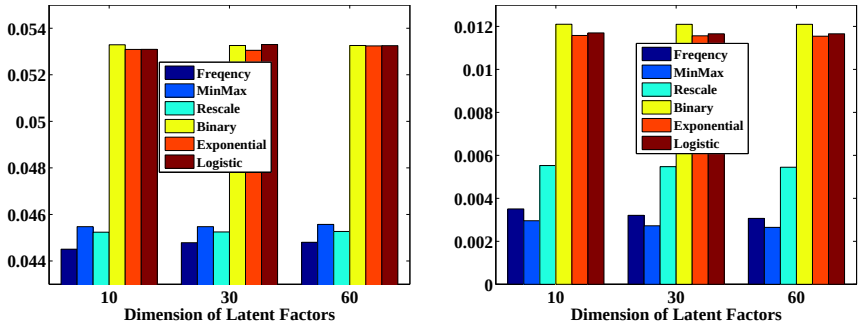
(a) Precison@K (10-D) (b) Recall@K (10-D)

(c) Precision@K (60-D) (d) Recall@K (60-D)

Figure 4.9: Performance comparison for different rating conversions on PMF with different dimensions.

- **Exponential [48]:** A mapping function $\frac{1}{1+x^{-1}}$ is used to bound each matrix entry into $[0, 1]$.
- **Binary:** It has two values: 0 and 1. The rating is assigned to 1 if user has check-in at this POI, and assigned to 0 otherwise.
- **Rescale [13]:** Due to the power law distribution of user-location check-in numbers, we could obtain a five-point scale rating with check-in frequency: converting one check-in to 2, two check-ins to 3, three check-ins to 4, and four or more check-ins to 5.
- **MinMax [91]:** It is defined as $\frac{x-1}{max-1}$, where max is the maximum frequency value.
- **Frequency:** Rating is the number of user-location check-ins.

where x is the number of user-location check-ins. It is worth to noting that for Rescale and Frequency, we use the ratings after minus mean value to fit models; other kinds

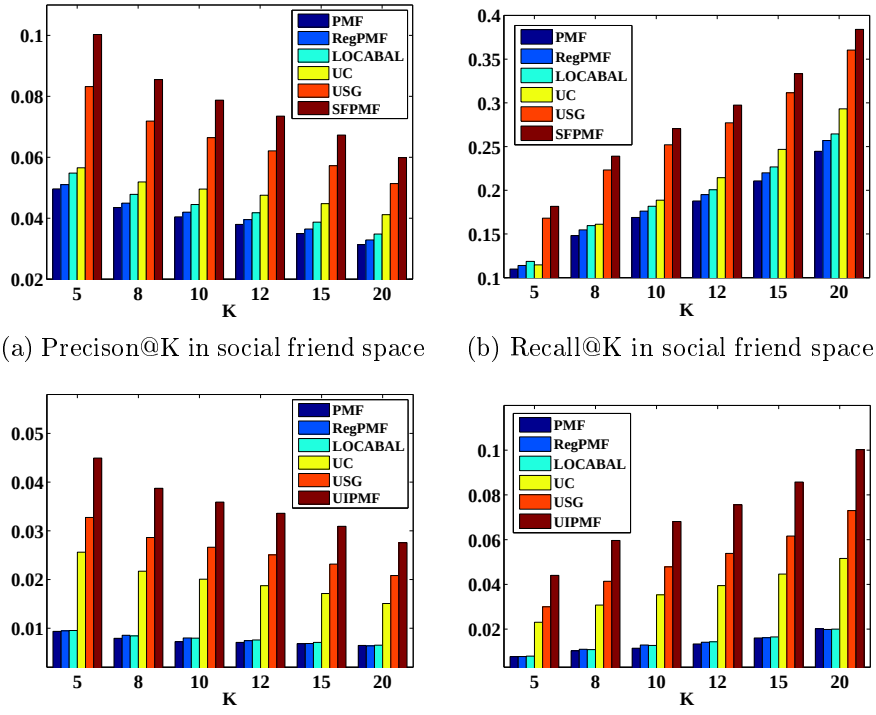


(a) MAP on UIPMF (b) MAP on PMF

Figure 4.10: Performance comparison for different rating conversions in terms of MAP with different dimensions.

of ratings are used to fit the models directly. Due to the limited space, we only report the performance of different rating conversion methods on both UIPMF and PMF in terms of precision@K, recall@K and MAP in Figure 4.8, Figure 4.9 and Figure 4.10.

Based on the results, we summarize as following: (1) Two models perform almost consistent with different rating conversion methods, indicating that matrix factorization based models will have consistent performance with different rating conversion methods. (2) Frequency, MinMax and Rescale perform much worse than others, suggesting the bias in check-in data would affect the model's performance. Even though MinMax bounds the rating to $[0, 1]$, many zero ratings are possible to explain its bad performance. (3) Logistic, Exponential and Binary methods have very similar performance and are much superior than others. It happens possibly because they constrain the ratings in $[0, 1]$ to avoid the large fluctuation of ratings. Surprisingly Binary performs very well under this group of regularization parameters. But we observe that this method pronely leads to over-fitting in high dimension under other parameter settings. In the following experiments, we will adopt logistic function as rating conversion method to fit matrix factorization based methods because it is widely used in recommendation and obtains good performance. We only report the performance with the latent factor dimension as 10 due to the similar performance in different dimensions.



(a) Precison@K in social friend space

(b) Recall@K in social friend space

(c) Precision@K in user interest space

(d) Recall@K in user interest space

Figure 4.11: Performance comparison in terms of precision@K and recall@K in social friend space and user interest space.

4.3.2.4.2 Performance Comparison in Social Friend Space

We compare our SFPMF model with the baseline methods in social friend space, where the testing check-ins are only those that friends have visited before (see Test(SFS) in Section 4.3.2.1). We first compute the probabilities for each user and her candidate locations with different models, and then recommend the top-K locations with the highest probabilities. The candidate locations are those that users' friends have checked-in before. The performances in terms of Precision@K, Recall@K and MAP are shown in Figure 4.11a, Figure 4.11b and at the top of Table 4.6.

It can be observed that all models perform consistently with different metrics. We find that as K increases, the precision decreases while the recall increases. Totally PMF has the worst performance among all methods, indicating that traditional matrix factorization is difficult to work well on check-in data due to the difference

Table 4.6: Performance comparison in terms of MAP in social friend space and user interest space.

Social Friend Space					
SFPMF	USG	UC	LOCABAL	RegPMF	PMF
0.16825	0.14947	0.11111	0.10406	0.10017	0.09689
User Interest Space					
UIPMF	USG	UC	LOCABAL	RegPMF	PMF
0.04538	0.02996	0.02303	0.00684	0.00666	0.00647

between user's behaviors on products consuming and check-ins. The POI decision making process is more affected by social network and geographical influence. Both RegPMF and LOCALBAL perform better than PMF. Their improvements indicate that social network is a factor affecting the performance in POI recommender system. Leveraging the assumption that users and their friends share similar interests does improve the recommendation accuracy. It is surprising that UC also works well. It is possible that users' similar interests can help to estimate more accurate ratings in our check-in data. Although USG takes the geographical influence, social network and user interest into account for recommendation, it is not as good as our SFPMF model. It demonstrates the framework of SFPMF and the modeling approach of user's preference propagation assist to improve the prediction accuracy. Furthermore, it illustrates that utilizing the characteristics of social network, i.e. preference propagation from one person to another, can appropriately model user's check-in decision making process in social friend space.

4.3.2.4.3 Performance Comparison in User Interest Space

We evaluate the performance of UIPMF model versus various baseline methods in user interest space, where the testings are excluding those that are checked-in by friends in the testing (see Test(UIS) in Section 4.3.2.1). The performances in terms of Precision@K, Recall@K and MAP are shown in Figure 4.11c, Figure 4.11d and at the bottom of Table 4.6.

From the results, we can see RegPMF and LOCALBAL are only a little better

Table 4.7: Performance comparison in terms of Precision, Recall and MAP in the whole recommendation space.

	Precision@5	Precision@8	Precision@10	Precision@15	
ISU	**0.06464**	**0.05548**	**0.05139**	**0.04414**	
USG	0.03651	0.03488	0.03377	0.03137	
UC	0.02895	0.02744	0.02668	0.02524	
LOCABAL	0.02193	0.01866	0.01735	0.01488	
RegPMF	0.02142	0.01846	0.01715	0.01448	
PMF	0.02129	0.01828	0.01660	0.01448	
	Recall@5	Recall@8	Recall@10	Recall@15	MAP
ISU	**0.04784**	**0.06400**	**0.07312**	**0.09174**	**0.05379**
USG	0.02916	0.04387	0.05255	0.07106	0.02847
UC	0.02330	0.03469	0.04159	0.05743	0.02427
LOCABAL	0.01426	0.01906	0.02198	0.02773	0.01231
RegPMF	0.01393	0.01891	0.02182	0.02707	0.01190
PMF	0.01389	0.01854	0.02063	0.02695	0.01170

than PMF, but they are much worse than others. It shows the traditional modeling methods with social network fail to achieve accurate recommendations in user interest space. In this space, since users and friends have no common check-ins in the testing, social network does not work in USG model. Hence in fact USG only captures user's interests and geographical influence. Its superior performance exhibits that geographical influence play an important role in location recommendation. The significant improvements of UIPMF compared to USG demonstrate that utilizing user's historical interests to seek a new POI similar to previous ones, as well as exploiting the spatial clustering phenomenon of check-in data are helpful for recommendation.

4.3.2.4.4 Performance Comparison in the Whole Recommendation Space

As social friend space and user interest space might have overlap, we can integrate them (namely ISU) to make recommendations in the whole space. We evaluate the performance of ISU model with baseline models in terms of precision@K, recall@K and MAP. We do the test in the whole testing data. The performances are reported in Table 4.7. We summarize the main results as the followings:

- PMF performs the worst among all the models. The sparseness of data may be one reason why it has such bad performance. UC performs better than PMF. Although the data is very sparse, it might have the tendency in our check-in data that similar users have similar interests in the preference of POIs. Thus, UC achieves good performance in our check-in data.
- Both RegPMF and LOCALBAL are better than PMF. It shows social network is helpful for location prediction. Although both of them assume users and their friends have similar interests, they have different modeling approaches. The better performance of LOCALBAL than that of RegPMF reflects that considering the local and global effect of friends is a superior approach to utilize social network information for recommendation.
- USG obtains a better performance than LOCALBAL, RegPMF, UC and PMF, which indicates that both social network and geographical influence can benefit POI recommender system. However, it is worse than the proposed model possibly due to the weak connection of a new POI and one user's historical POIs. Thus it clearly shows our models' effectiveness.
- ISU performs much better than other baseline models, illustrating the effectiveness of (1) modeling user's repeating behaviour for her friends' historical POIs; (2) capturing the influence of user's own historical check-ins on new POIs. It also indicates that both social network and geographical influence contribute to POI recommendation together.

4.3.3 Conclusion

We investigate a novel Point-of-Interest recommender system in Section 4.3. Specifically, we divide the recommendation space into social friend space and user interest space. In social friend space, the problem is formulated as recommending one user with new POIs that her friends have checked-in before. A novel SFPMF model is proposed to factorize the preference propagation influence into user and factor fea-

ture vectors. In user interest space, the problem is defined as recommending one user with new POIs that have not been visited by her friends but are very similar to her historical ones. Then UIPMF model is developed to capture the connection between one user's preference for a new POI and her preference for historically visited POIs. Finally, experimental results on a real-world dataset effectively demonstrate the improvement of the proposed models over several baseline methods based on many validation metrics.

4.4 Addressing Cold-Start Problem

In this section, we exploit geographical properties and social correlations to design advanced and efficient recommender systems for addressing cold-start issues in POI recommendation on LBSNs. To mitigate data sparsity and tackle cold-start problems, we introduce three types of friends for each user based on geo-social correlations: social friends, location friends and neighboring friends. The social friends of a user refer to the set of users who are socially connected with this user in LBSNs. The location friends of a user denote the set of users who check-in the same locations as this user does. The neighboring friends of a user are those users who are geographically close to this user. Then we novelly incorporate their historical check-ins into matrix factorization model with different loss functions. In a nutshell, in the following section we:

- empirically analyze the correlations between users and their three type of friends using two check-in datasets.
- design two approaches to learn a set of locations for each individual user that her friends have checked-in before and she is most interested in.
- develop matrix factorization based models via different loss functions with the learned potential check-ins, and correspondingly propose two scalable optimization methods.

- propose three different recommendation strategies for standard recommendation, new location recommendation, and new user recommendation.

4.4.1 Notation and Definition

4.4.1.1 Notation

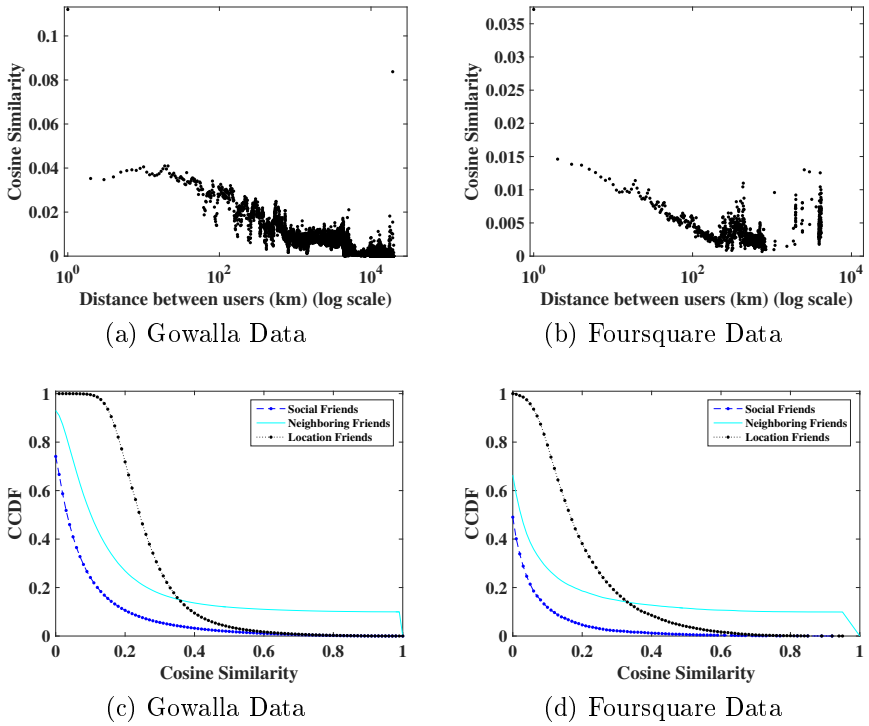
Suppose there are N users and M locations. For convenience we will henceforth refer to i as user, f as friend and j as the location unless stated otherwise. Suppose there are C kinds of categories and the category of location j is denoted as c_j. For user i, $\mathcal{F}_i$ denotes a set of friends which will be further defined and explained in Section 4.4.1.2, $\mathcal{M}_i^o$ is a set of locations checked-in by her, $\mathcal{M}_i^p$ is a set of potential locations learned in Section 4.4.3, and $\mathcal{M}_i^u$ is the remaining unvisited locations. r_{ij} is the check-in frequency of user i on location j. In addition, all column vectors are represented by bold lower case letters, all matrices are represented by bold upper case letters, and a numeric value is denoted by lower case letter. A predicted value is denoted with a ˆ(hat) over it. The terms *location* and *POI* are used interchangeably.

4.4.1.2 Definition

To better understand users' check-in behaviours, we examine the check-in data collected from Gowalla and Foursquare (Details can be found in Section 4.4.5). To clarify the relation between the similarity of pairwise users and their physical distance, we plot their relations in Figure 4.12a and Figure 4.12b. The physical distance of two users refers to the distance between their home locations, and the similarity of user i and user f is measured by cosine similarity, given by:

$$Sim_u(i,f) = (\sum_{j \in \mathcal{M}_i^o} r_{ij}^2 \sum_{j \in \mathcal{M}_f^o} r_{fj}^2)^{-\frac{1}{2}} \sum_{j \in \mathcal{M}_i^o \cap \mathcal{M}_f^o} r_{ij} r_{fj}. \tag{4.23}$$

Based on the observation, we find that the physically closer two users live, the more



(a) Gowalla Data (b) Foursquare Data

(c) Gowalla Data (d) Foursquare Data

Figure 4.12: (a) ∼ (b) Cosine similarity as a function of distance between users' home locations. (c) ∼ (d) Complementary Cumulative Distribution Function (CCDF) of cosine similarity between friends.

similar their POI interests are. It motivates us to leverage neighboring friends, who are physically neighbors, to learn user's interest in POIs. In addition, social friends who build connections online share the similar interests in POI decisions [67, 57, 68, 69]. Users who check-in similar locations are treated as location friends, and also might have similar tastes. Thus, three types of friends of user i, i.e., neighboring friends, social friends and location friends, might affect her check-in activity, defined as:

Definition 5 (Social Friends) *The social friends of user i are the set of users who have socially connected with her in LBSNs, which is denoted as $\mathcal{F}_i^s$.*

Definition 6 (Location Friends) *Given a set of locations $\mathcal{M}_i^o$, which have been checked-in by user i, her location friends, denoted as $\mathcal{F}_i^l$, are the set of users who have also checked-in these locations, i.e., $\mathcal{F}_i^l = \bigcup_{j \in \mathcal{M}_i^o} \Psi_j$, where Ψ_j is the set of users who also have checked-in location j.*

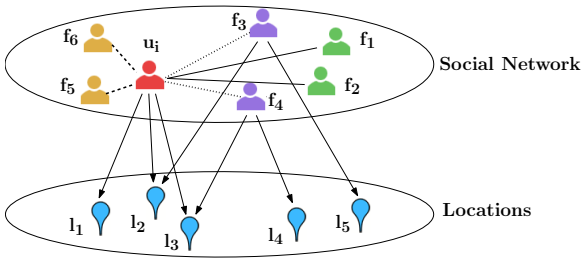


Figure 4.13: The user u_i's social network and check-ins.

Definition 7 (Neighboring Friends) *Given the home location of user i, the neighboring friends are the set of users who live physically closest to her and denoted as* $\mathcal{F}_i^n$.

In the example of Figure 4.13, the target user u_i has checked-in locations $\{l_1, l_2, l_3\}$. $\{f_1, f_2\}$ are her social friends who socially connect with her online. User f_3 has check-ins at locations l_2 and l_5, and user f_4 has check-ins at locations l_3 and l_4. f_4 and f_5 have common POIs with user u_i, i.e., l_2 and l_3, respectively. Thus, both of them are the location friends of user u_i. In addition, f_5 and f_6 are the target user's neighboring friends due to their physically short distance to her. Thus, $\{f_1, \cdots, f_6\}$ are regarded as the friends of user i. Here, the **friends** of user i are defined as:

$$\mathcal{F}_i = \mathcal{F}_i^s \cup \mathcal{S}(\mathcal{F}_i^l) \cup \mathcal{S}(\mathcal{F}_i^n), \tag{4.24}$$

where $\mathcal{S}(\mathcal{F}_i^l)$ is the set of S most similar friends with the highest cosine similarities and $\mathcal{S}(\mathcal{F}_i^n)$ is the set of S physically nearest friends with the shortest distance among their homes[3]. To examine the correlation between friends, we report the complementary cumulative distributions of their similarities in Figure 4.12c and Figure 4.12d on Gowalla and Foursquare, respectively. There are over 5%, 20% and 40% pairs of social, neighboring and location friends which have similarities larger than 0.2. Particularly, the friends' correlation is much stronger in Gowalla than Foursquare. The observation shows the importance of friends in LBSNs and motivates us to use friends' historical

[3]In the experiment, we set S as 10.

check-ins to improve recommendation accuracy.

4.4.2 Recommendation Framework

The recommendation task in Section 4.4 is defined as: given users' historical checked-in locations, we aim at recommending each user with top-K locations that she might be interested in but has not visited before. We propose a two-step recommendation framework. Specifically, in the first step, we learn a set of potential locations from three types of friends, which will be introduced in Section 4.4.3. In the second step, we incorporate the learned potential locations of each individual into matrix factorization model with different error loss functions, which will be presented in Section 4.4.4. At last, we introduce different recommendation strategies for standard recommendation, location cold-start recommendation and user cold-start recommendation.

4.4.3 Learning Potential Locations

Social network plays an important role in recommendations [67, 60, 68, 69]. However, only leveraging the historical locations of social friends cannot successfully model user's preference for locations due to that it is difficult to appropriately model the preference of users who have no social friends, not mention to tackle user cold-start problem (i.e., a user has never checked-in any location before). To address these issues, we will exploit the characteristics of three types of friends: social friends, location friends and neighboring friends. The earlier section has shown their significance in LBSNs, i.e., friends would share the similar preferences for POIs. In other words, users might be interested in those locations which have been checked-in by their friends, and have a high probability to check-in them next time. However, the extremely large number of these locations will lead to the inefficiency of computation with the increase of locations, and the inaccuracy of prediction with the increase of noise. Hence, the problem in this section is to find the most potential locations for

the target user, defined as:

Definition 8 (Problem of Potential Locations) *Given the set of locations* $\mathcal{M}_i^f = \bigcup_{f \in \mathcal{F}_i} \mathcal{M}_f^o \backslash \mathcal{M}_i^o$, *that the friends of target user* i *have checked-in before but she never visits, the problem is to find top* S *most potential locations that she might be interested, denoted as* $\mathcal{M}_i^p$.

To obtain the potential locations of each user i, we propose two methods, i.e., *Linear Aggregation* and *Random Walk*, to estimate the probability p_{ij}^{pot} of this user on each location j that her friends have checked-in. Then we rank them by the estimated probabilities and select S locations with the highest probabilities[4]. The learned potential locations will assist to make accurate recommendation in Section 4.4.4.

4.4.3.1 Linear Aggregation

In this section, we propose *Linear Aggregation* method, denoted as LA, to predict the probability p_{ij}^{pot} that user i prefers location j which has been visited by her friends. Suppose $Sim(i, f; j)$ is the similarity between user i and friend f on the preference for location j. A location is possibly checked-in by more than one friends, so we define p_{ij}^{pot} as:

$$p_{ij}^{pot} \propto \max_{f \in \mathcal{F}_i^j} \{Sim(i, f; j)\},$$

where $\mathcal{F}_i^j$ is the set of user i's friends who have checked-in location j. The similarity $Sim(i, f; j)$ incorporates two parts: (1) the similarity of user interest, and (2) the similarity of geographical location. The similarity of user interest can be measured by cosine similarity in Eq.(4.23). Since a user's check-in probability and the distance from her home to the corresponding location follow a power law distribution [2], we exploit this characteristic to model geographical similarity. Hence, we define the

[4]In the experiments, we set S as 500.

probability that a user checks-in a location d-km far away as the following:

$$Pr_G(d) = a \cdot d^b, \tag{4.25}$$

where a and b are the parameters of power law distribution and could be learned by maximum likelihood estimation. Then the probability of user i to check-in a POI j due to the geographical influence is normalized as:

$$p_{ij}^G = \frac{Pr_G(d(h_i, j))}{Pr_G(d_{min})}, \tag{4.26}$$

where h_i is the home location of user i, and $d(h_i, j)$ indicates the distance between the home location of user i and the POI j, and d_{min} is the minimum distance. The distance could be computed by Haversine formula with latitude and longitude. Thus, $Sim(i, f; j)$ is the linear aggregation of similarities on both user interest and geographical location, given by

$$Sim(i, f; j) = \zeta \; Sim_u(i, f) + (1 - \zeta) \; p_{ij}^G,$$

where $\zeta \in [0, 1]$ is a tuning parameter to control the importance of the similarity of user interest.

4.4.3.2 Random Walk

Random walk with restart has successfully measured the correlation between two nodes in a graph [64, 92]. In this section, we propose a *Random Walk* method, denoted as *RW*, to learn the probability p_{ij}^{pot} of user i on location j which has been visited by her friends. We construct a directed graph with two kinds of nodes: the users (i.e., the target user and her friends), and the locations checked-in by her and her friends. Let $\mathbf{y}$ be a column vector where y_i refers to the probability that the random walk is at node i. Also let $\mathbf{A}$ be the column normalized transition matrix where a_{ij} denotes the

probability that node i jumps to node j. Here we consider three types of transition probabilities: (1) the probability between users measured by the cosine similarity in Eq.(4.23); (2) the probability from each user to each location which is one if the user checks-in the corresponding location and otherwise is zero; (3) the similarity between a pair of locations (j and k) measured by the normalized power-law function which is defined as the following:

$$Sim_G(j,k) = \frac{Pr_G(d(j,k))}{Pr_G(d_{min})}, \tag{4.27}$$

where $d(j,k)$ is the distance between these two locations and the power law parameters are learned with the check-in probabilities and corresponding distances of pairwise locations. Hence, the iteration equation for updating the steady-state probability of each node is given as follows:

$$\mathbf{y} = (1-\beta)\mathbf{A}\mathbf{y} + \frac{\beta}{|\mathcal{M}_i^o \cup \mathcal{M}_i^f| + |\mathcal{F}_i| + 1}\mathbf{x}, \tag{4.28}$$

where $\mathbf{x}$ is the column vector of zeros with the elements corresponding to the target user and her checked-in locations as one, and $\beta \in [0,1]$ is the restart probability to return to the target user and her checked-in locations. The steady-state probability is achieved by recursively performing Eq.(4.28) until convergence. Thus, the probability p_{ij}^{pot} is the steady-state probability corresponding to the location j.

4.4.4 Recommendation Models

In Section 4.4.3, for each individual user, we have learned the potential locations from her friends' information. In this section, the learned potential locations are utilized to make accurate recommendation and address user cold-start problem. Overall, for each user i, we have her three kinds of locations: observed locations $\mathcal{M}_i^o$, potential locations $\mathcal{M}_i^p$ and other unobserved locations $\mathcal{M}_i^u$.

We build our recommendation models by leveraging the widely-used matrix factorization techniques [14, 9, 15, 20, 6], where both user and location are mapped into latent low-dimension spaces. Let $\mathbf{U} \in \mathbb{R}^{D \times N}$ and $\mathbf{V} \in \mathbb{R}^{D \times M}$ be the latent user and location feature matrices, with column vectors $\mathbf{u}_i$ and $\mathbf{v}_j$ representing the D-dimensional user-specific and location-specific feature vectors of user i and location j, respectively. A typical prediction for the preference of user i to location j is taken by an inner product of latent vectors, i.e., $\hat{p}_{ij} = \mathbf{u}_i^T \mathbf{v}_j$, where $\mathbf{P} \in \mathbb{R}^{N \times M}$ is the preference matrix.

However, in LBSNs the category information of POIs affects user's check-in decision making process. Users are often used to visiting those POIs which belong to the same category due to their specific hobbies. For example, users who like eating would have a much higher probability to choose a new POI relevant to *food* next time, but they have much less chance to check-in a POI about *sight*. Thus, the preference of category is another important factor to affect user's decision on a new POI. Here, we introduce the category feature matrix $\mathbf{Q} \in \mathbb{R}^{N \times C}$, where each entry q_{ic} indicates the preference of user i to category c. Hence, the preference of user i for location j is refined as follows:

$$\hat{p}_{ij} = (q_{ic_j} + \varepsilon)\mathbf{u}_i^T \mathbf{v}_j, \tag{4.29}$$

where c_j is the category of location j and ε is a tuning parameter to indicate that user has a small probability to prefer one location with another category.

Many of the recent works suppose to only model the observed rating, which is adapted to explicit feedback datasets. However, the check-in dataset is implicit feedback dataset, where we do not have explicit feedback for user's preference to locations. In other words, we lack substantial evidence on which location the user dislikes. To address user cold start problem and tackle data sparseness problem, we propose to

model the observed preference, potential preference and unobserved preference of users for location, simultaneously. Let j, k, h denote the observed location, potential location and unobserved location, respectively. The loss function of general form is given as follows:

$$\underset{\mathbf{U},\mathbf{V},\mathbf{Q}}{\operatorname{argmin}} \sum_i E_i(p_{ij}, p_{ik}, p_{ih}, \hat{p}_{ij}, \hat{p}_{ik}, \hat{p}_{ih}) + \Theta(\mathbf{U}, \mathbf{V}, \mathbf{Q}), \tag{4.30}$$
$$\forall j \in \mathcal{M}_i^o, \forall k \in \mathcal{M}_i^p, \forall h \in \mathcal{M}_i^u,$$

where $E_i(\cdot)$ is the loss function for the observed, potential and unobserved preference of user i for locations, and $\Theta(\cdot)$ is a regularization term with ℓ_2 norm which is defined as follows:

$$\Theta(\mathbf{U}, \mathbf{V}, \mathbf{Q}) = \frac{\lambda_u}{2}||\mathbf{U}||_F^2 + \frac{\lambda_v}{2}||\mathbf{V}||_F^2 + \frac{\lambda_q}{2}||\mathbf{Q}||_F^2, \tag{4.31}$$

where λ_u, λ_v and λ_q are the regularization constants. We develop two different types of models which use different loss function for $E_i(\cdot)$, i.e., the square error based and the ranking error based loss functions, and will be described in the next two sections, respectively.

4.4.4.1 The Square Error based Model

In this section, we present the Augmented Square error based Matrix Factorization (*ASMF*) model constrained with the square error loss function and its optimization method.

4.4.4.1.1 The ASMF Model

Due to the similar interests between friends, one user might have opportunity to visit those potential locations that her friends have visited before but she never checks-in. We treat each individual user's check-ins as an indication of positive, potential and negative preference associated with different confidence. One user has a high

confidence for the positive preference to their checked-in POIs. However, she will have a low confidence for the potential preference to those potential locations and the negative preference to other unvisited locations. Correspondingly, we augment the binary preference variable p_{ij} to a ternary value as follows:

$$p_{ij} = \begin{cases} 1 & \text{if } j \in \mathcal{M}_i^o, \\ \alpha & \text{if } j \in \mathcal{M}_i^p, \\ 0 & \text{otherwise}, \end{cases} \tag{4.32}$$

where $\alpha \in [0, 1]$ is a potential preference constant, indicating user i has a probability α to choose an unvisited location j that her friends have visited before.

Therefore, we propose the augmented square error based matrix factorization model ($ASMF$) to compute the loss $E_i(\cdot)$ by using the squared error loss function with the ternary variable defined in Eq.(4.32), given by:

$$E_i(\cdot) = \sum_{j=1}^{M} w_{ij} \left(p_{ij} - \hat{p}_{ij}\right)^2, \tag{4.33}$$

where $\mathbf{W}$ is the confidential matrix with element w_{ij} as the confidential weight for user i to location j, given by:

$$w_{ij} = \begin{cases} 1 + \gamma \times r_{ij} & \text{if } j \in \mathcal{M}_i^o, \\ 1 & \text{otherwise}, \end{cases} \tag{4.34}$$

where γ is the tuning parameter.

4.4.4.1.2 The Parameter Estimation

In *ASMF* model, based on the Eq.(4.29), Eq.(4.30) and Eq.(4.33), the matrices $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{Q}$ are learned by minimizing the following regularized optimization problem:

$$\mathcal{L} = \min_{\mathbf{U},\mathbf{V},\mathbf{Q}} \sum_{i=1}^{N}\sum_{j=1}^{M} w_{ij}\left(p_{ij} - (q_{ic_j} + \varepsilon)\mathbf{u}_i^T\mathbf{v}_j\right)^2 + \Theta(\mathbf{U},\mathbf{V},\mathbf{Q}), \tag{4.35}$$

To solve the above optimization problem, we adopt Alternating Least Squares (ALS) [15] optimization method due to the accurate parameter estimation and fast convergence rate. We perform ALS method to compute each latent variable by fixing the other variables when minimizing the objective function. The updating formulas with respect to $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{Q}$ are given as follows:

$$\mathbf{u}_i = (\lambda_u \mathbf{I}_D + \sum_j w_{ij}\tilde{q}_{ic_j}^2 \mathbf{v}_j\mathbf{v}_j^T)^{-1}\sum_j \mathbf{w}_{ij}\tilde{q}_{ic_j}p_{ij}\mathbf{v}_j, \tag{4.36}$$

$$\mathbf{v}_j = (\lambda_v \mathbf{I}_D + \sum_i w_{ij}\tilde{q}_{ic_j}^2 \mathbf{u}_i\mathbf{u}_i^T)^{-1}\sum_i w_{ij}\tilde{q}_{ic_j}p_{ij}\mathbf{u}_i, \tag{4.37}$$

$$\mathbf{q}_{ic} = (\sum_{j\in\mathcal{N}_c} w_{ij}(p_{ij} - \varepsilon)\mathbf{u}_i^T\mathbf{v}_j)/(\lambda_q + \sum_{j\in\mathcal{N}_c} w_{ij}(\mathbf{u}_i^T\mathbf{v}_j)^2), \tag{4.38}$$

where $\mathbf{I}_D$ is the D-dimension unit matrix, $\mathcal{N}_c$ is the set of locations with category c, and $\tilde{q}_{ic_j}$ is equal to $q_{ic_j} + \varepsilon$. The detailed algorithm is reported in Algorithm 1. Specifically, we place the non-negative constraints on $\mathbf{Q}$ and project the negative variables to 0 in each iteration.

Complexity Analysis. The complexity of direct computation is $\mathcal{O}(NMD^2)$ which is extremely inefficient particularly with the increase of locations and users. To improve the efficiency, we design the following updating strategies. For updating $\mathbf{u}_i$, we employ the similar trick in [20], i.e., $\sum_j w_{ij}\tilde{q}_{ic_j}^2\mathbf{v}_j\mathbf{v}_j^T = \sum_j \tilde{q}_{ic_j}^2\mathbf{v}_j\mathbf{v}_j^T + \sum_j \gamma r_{ij}\tilde{q}_{ic_j}^2\mathbf{v}_j\mathbf{v}_j^T$. The first term can be written as $\sum_j \tilde{q}_{ic_j}^2\mathbf{v}_j\mathbf{v}_j^T = \sum_c \tilde{q}_{ic}^2\sum_{j\in\mathcal{N}_c}\mathbf{v}_j\mathbf{v}_j^T$. For each category c, $\sum_{j\in\mathcal{N}_c}\mathbf{v}_j\mathbf{v}_j^T$ is independent of i and already pre-computed, so

Algorithm 1: ASMF Optimization

Input: $\mathbf{W}$, $\mathbf{P}$, λ_u, λ_v, λ_q, α, ε, τ, $maxIter$
Output: $\mathbf{U}^{(t)}$, $\mathbf{V}^{(t)}$, $\mathbf{Q}^{(t)}$

Randomly initialize $\mathbf{U}^{(0)}$ and $\mathbf{V}^{(0)}$, $t \leftarrow 1$, $\omega \leftarrow \infty$
Initialize $\mathbf{Q}^{(0)}$ by using Eq.(4.38)
while $t \leqslant maxIter$ && $\omega \geqslant \tau$ **do**
    Update $\mathbf{U}^{(t)}$ by using Eq.(4.36)
    Update $\mathbf{V}^{(t)}$ by using Eq.(4.37)
    Update $\mathbf{Q}^{(t)}$ by using Eq.(4.38)
    $\omega \leftarrow \frac{|\mathcal{L}(\mathbf{U}^{(t)},\mathbf{V}^{(t)},\mathbf{Q}^{(t)}) - \mathcal{L}(\mathbf{U}^{(t-1)},\mathbf{V}^{(t-1)},\mathbf{Q}^{(t-1)})|}{|\mathcal{L}(\mathbf{U}^{(t-1)},\mathbf{V}^{(t-1)},\mathbf{Q}^{(t-1)})|}$
    $t \leftarrow t + 1$
end
return $\mathbf{U}^{(t)}$, $\mathbf{V}^{(t)}$, $\mathbf{Q}^{(t)}$

the time complexity of this term is $\mathcal{O}(CD^2)$ and C is usually very small. The cost time of the second term is $\mathcal{O}(n_i D^2)$, where the potential part can be pre-computed and n_i is the number of observed locations for which $r_{ij} > 0$. In addition, the inverse of a $D \times D$ matrix costs $\mathcal{O}(D^3)$. Consequently, the re-computation of $\mathbf{u}_i$ is performed in time $\mathcal{O}(CD^2 + n_i D^2 + D^3)$. This procedure is performed over each user, so the total time is $\mathcal{O}(NCD^2 + nD^2 + ND^3)$, where n is defined as $n = \sum_i n_i$.

Similarly, when updating $\mathbf{v}_j$, we have $\sum_i w_{ij} \tilde{q}^2_{ic_j} \mathbf{u}_i \mathbf{u}_i^T = \sum_i \tilde{q}^2_{ic_j} \mathbf{u}_i \mathbf{u}_i^T + \sum_i \gamma r_{ij} \tilde{q}^2_{ic_j} \mathbf{u}_i \mathbf{u}_i^T$. For each category c, the first term is independent of j and was already pre-computed. Thus, the total cost time over M locations is $\mathcal{O}(nD^2 + MD^3)$.

To update q_{ic}, we can rewrite the expression as $\sum_{j \in \mathcal{N}_c} w_{ij} (\mathbf{u}_i^T \mathbf{v}_j)^2 = \sum_{j \in \mathcal{N}_c} (\mathbf{u}_i^T \mathbf{v}_j)^2 + \sum_{j \in \mathcal{N}_c} \gamma r_{ij} (\mathbf{u}_i^T \mathbf{v}_j)^2$. The first term is written as $\sum_{j \in \mathcal{N}_c} (\mathbf{u}_i^T \mathbf{v}_j)^2 = \mathbf{u}_i^T (\sum_{j \in \mathcal{N}_c} \mathbf{v}_j \mathbf{v}_j^T) \mathbf{u}_i$, where $\sum_{j \in \mathcal{N}_c} \mathbf{v}_j \mathbf{v}_j^T$ was already pre-computed, so it costs $\mathcal{O}(D^2)$. The second term costs $\mathcal{O}(Dn_{ic})$, where n_{ic} is the number of locations for which $r_{ij} > 0$ and belongs to category c. The total complexity of updating $\mathbf{Q}$ is $\mathcal{O}(NCD^2 + Dn)$.

In a summary, for each iteration of optimization, the total time is $\mathcal{O}(nD^2)$, where $n > max\{M, N\}D$, and $n > CN$ are usually satisfied. In other words, the time complexity of one optimization iteration is in linear proportion to the number of

observed check-ins.

4.4.4.2 The Ranking Error based Model

In this section, we present the Augmented Ranking error based Matrix Factorization ($ARMF$) model constrained with the ranking error loss and its optimization method.

4.4.4.2.1 The ARMF Model

In check-in dataset, we only have a user's check-in record and do not know how much she dislikes a location. In other words, an unvisited location does not necessarily indicate the user dislikes it. The unobserved data actually is a mixture of negative preference for locations and missing values. It motivates us to consider a ranking error based loss function for modeling the ranking order of user's preference for observed locations, potential locations and unobserved locations. We assume that the user prefers an observed location over all potential locations, and at the same time she prefers a potential location over all other unobserved locations. Thus, for user i, the ranking order of her preference over an observed location $j \in \mathcal{M}_i^o$, a potential location $k \in \mathcal{M}_i^p$ and an unobserved location $h \in \mathcal{M}_i^u$ is given as the following:

$$\begin{cases} \hat{p}_{ij} > \hat{p}_{ik} \\ \hat{p}_{ik} > \hat{p}_{ih} \end{cases} \Rightarrow \begin{cases} (q_{ic_j} + \varepsilon)\mathbf{u}_i^T\mathbf{v}_j > (q_{ic_k} + \varepsilon)\mathbf{u}_i^T\mathbf{v}_k \\ (q_{ic_k} + \varepsilon)\mathbf{u}_i^T\mathbf{v}_k > (q_{ic_h} + \varepsilon)\mathbf{u}_i^T\mathbf{v}_h \end{cases}. \tag{4.39}$$

To this end, we propose the augmented ranking error based matrix factorization ($ARMF$) to compute the loss $E_i(\cdot)$ by using the ranking error loss function, given by,

$$E_i(\cdot) = -\sum_{j \in \mathcal{M}_i^o} \sum_{k \in \mathcal{M}_i^p} \ln \sigma(\hat{p}_{ij} - \hat{p}_{ik}) - \sum_{k \in \mathcal{M}_i^p} \sum_{h \in \mathcal{M}_i^u} \ln \sigma(\hat{p}_{ik} - \hat{p}_{ih}), \tag{4.40}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic sigmoid function which is introduced to penalize the violated constraints in Eq.(4.39). As we can see in Eq.(4.40), the error function

does not focus on predicting the right value, but on the ordering of the preference for observed, potential and unobserved locations.

4.4.4.2.2 The Parameter Estimation

In ARMF model, based on the Eq.(4.29), Eq.(4.30) and Eq.(4.40), the matrices $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{Q}$ are learned by minimizing the following regularized optimization problem:

$$\underset{\mathbf{U},\mathbf{V},\mathbf{Q}}{\operatorname{argmin}} - \sum_{i} \left(\sum_{j \in \mathcal{M}_i^o} \sum_{k \in \mathcal{M}_i^p} \ln \sigma(\hat{p}_{ij} - \hat{p}_{ik}) + \sum_{k \in \mathcal{M}_i^p} \sum_{h \in \mathcal{M}_i^u} \ln \sigma(\hat{p}_{ik} - \hat{p}_{ih}) \right) + \Theta(\mathbf{U}, \mathbf{V}, \mathbf{Q}). \tag{4.41}$$

As there is no close-form for each variable with ALS approach, a Stochastic Gradient Descent (SGD) using the boostrap sampling with replacement algorithm is employed to solve the optimization problem in Eq.(4.41). The optimization algorithm is iteratively performed by sampling a tuple (i, j, k, h) and updating the corresponding variables, where i is a user, $j \in \mathcal{M}_i^o$ is her observed location, $k \in \mathcal{M}_i^p$ is her potential location, and $h \in \mathcal{M}_i^u$ is other unobserved location. More details of optimization are provided in Algorithm 2. Specifically, we define $g'(x) = \sigma(x) - 1$.

Complexity Analysis. The run time of sampling a tuple (i, j, k, h) is quite small in each update and can be neglected. Hence, the complexity of the optimization algorithm is $\mathcal{O}(mD)$, where m is the total iteration number. In the experiments, m is proportional to the number of observed check-ins.

4.4.4.3 Incorporating Geographical Influence

Different from online product consuming, a POI's geographical distance significantly affects the user's check-in decision making process. One user would have a small probability to check-in a location far away, even though she is interested in it. In the example shown in Figure 4.13, user u_i has more chance to check-in the locations in the left side than those in the right side. It motivates us to incorporate

Algorithm 2: ARMF Optimization

Input: λ_u, λ_v, λ_q, η, $maxIter$
Output: $\mathbf{U}$, $\mathbf{V}$, $\mathbf{Q}$

Randomly initialize $\mathbf{U}$ and $\mathbf{V}$, $\mathbf{Q}$, $t \leftarrow 1$
while $t \leqslant maxIter$ **do**
  Randomly sample a (i, j, k, h), where i is a user, and j, k, h are her one observed, potential, and unobserved location
  $\tilde{\mathbf{v}}_j^i \leftarrow (q_{ic_j} + \varepsilon)\mathbf{v}_j$
  $\tilde{\mathbf{u}}_i^j \leftarrow (q_{ic_j} + \varepsilon)\mathbf{u}_i$
  $\tilde{p}_{ijk} \leftarrow g'(\hat{p}_{ij} - \hat{p}_{ik})$
  $\tilde{p}_{ikh} \leftarrow g'(\hat{p}_{ik} - \hat{p}_{ih})$
  $\mathbf{u}_i \leftarrow \mathbf{u}_i - \eta(\tilde{p}_{ijk}(\tilde{\mathbf{v}}_j^i - \tilde{\mathbf{v}}_k^i) + \tilde{p}_{ikh}(\tilde{\mathbf{v}}_k^i - \tilde{\mathbf{v}}_h^i) + \lambda_u \mathbf{u}_i)$
  $\mathbf{v}_j \leftarrow \mathbf{v}_j - \eta(\tilde{p}_{ijk}\tilde{\mathbf{u}}_i^j + \lambda_v \mathbf{v}_j)$
  $\mathbf{v}_k \leftarrow \mathbf{v}_k - \eta((\tilde{p}_{ikh} - \tilde{p}_{ijk})\tilde{\mathbf{u}}_i^k + \lambda_v \mathbf{v}_k)$
  $\mathbf{v}_h \leftarrow \mathbf{v}_h - \eta(-\tilde{p}_{ikh}\tilde{\mathbf{u}}_i^h + \lambda_v \mathbf{v}_h)$
  $\mathbf{q}_{ic_j} \leftarrow q_{ic_j} - \eta(\tilde{p}_{ijk}\mathbf{u}_i^T\mathbf{v}_j + \lambda_q q_{ic_j})$
  $\mathbf{q}_{ic_k} \leftarrow q_{ic_k} - \eta((\tilde{p}_{ikh} - \tilde{p}_{ijk})\mathbf{u}_i^T\mathbf{v}_k + \lambda_q q_{ic_k})$
  $\mathbf{q}_{ic_h} \leftarrow q_{ic_h} - \eta(-\tilde{p}_{ikh}\mathbf{u}_i^T\mathbf{v}_h + \lambda_q q_{ic_h})$
  $t \leftarrow t + 1$
end
return $\mathbf{U}$, $\mathbf{V}$, $\mathbf{Q}$

the geographical influence into user's decision on POIs. Thus, the probability that user i prefers a POI j is:

$$\hat{p}_{ij} \propto p_{ij}^G \times \sigma(\hat{p}_{ij}) \Rightarrow \hat{p}_{ij} \propto p_{ij}^G \times \sigma((q_{ic_j} + \varepsilon)\mathbf{u}_i^T\mathbf{v}_j), \tag{4.42}$$

where p_{ij}^G is the geographical influence shown in Eq.(4.26).

4.4.4.4 Recommendation Strategies

Our goal is to recommend unvisited locations for users which they might be interested in. For each individual user, we first predict the probability that this user would check-in each unvisited location and then recommend the top-K locations with the highest probabilities for her. In particular, we adopt the following strategies for recommendation.

- **Standard Recommendation.** Similar to traditional recommendation, we con-

sider to recommend the existing users with the existing locations. After learning the model from training data, we exploit Eq.(4.42) to predict the probability that one user prefers each unvisited location.

- **New User Recommendation.** When new users enter the system, we consider to recommend them with the existing locations. First, we need to re-train the models with these new users by leveraging the historical check-ins of their social friends and neighboring friends. As new users do not have check-ins, they do not have location friends but they have neighboring friends. After the latent factors are learned, Eq.(4.42) is employed for recommendations.
- **New Location Recommendation.** When new locations enter the system, we consider to recommend existed users for them. By utilizing the neighboring location characteristics, the probability that user i checks-in a new location j is defined as:

$$\hat{p}_{ij} \propto p_{ij}^{G} \times \sigma \left(\frac{\sum_{l \in \hat{\psi}_j} Sim_G(j,l) \hat{p}_{il}}{\sum_{l \in \hat{\psi}_j} Sim_G(j,l)} \right),$$

where $\hat{\psi}_j$ is the set of S nearest neighboring locations of location j in the training data and in the experiments S is set as 10. The advantage to exploit the similarity of neighboring locations is that we can handle new locations as soon as they are generated in the system, without needing to re-train the model and estimate new parameters.

4.4.5 Experimental Results

In this section, we evaluate the proposed models with baseline methods on two real-world data sets.

4.4.5.1 Experimental Setup

Datasets. In this section, we use Gowalla and Foursquare datasets to evaluate the performance of the proposed models. Gowalla contains check-in data ranging from January 2009 to August 2010, and Foursquare includes the check-in data of users

who live in California, ranging from December 2009 to June 2013. Each check-in record in the datasets includes a user ID, a location ID and a timestamp, where each location has latitude, longitude and category information. Totally, there are 262 and 10 categories in Gowalla and Foursquare, respectively. Also, data sets have undirected friendship information and user's home information[5].

To evaluate model's cold-start recommendation performance, for each data set, we divide it in three steps. First, we remove those users who have visited less than 10 locations and those locations which are visited by less than 10 users. These check-ins are used to evaluate our model's performance for standard POI recommendation. In recommendation system, we aim to recommend those unvisited locations for users. Therefore, we split the training and testing data as follows: for each individual user, (1) aggregating the check-ins for each location; (2) sorting the location according to the first time that the user checked-in; (3) selecting the earliest 80% to train the model and using the next 20% as testing. Second, in the rest of check-ins (i.e., locations that are visited by less than 10 users and not included in the training), we use those check-ins whose locations are visited by users in the training data to evaluate the model's performance for new location recommendation. Third, in the rest of check-ins (i.e., users who have visited less than 10 locations), we use those check-ins where users are not in training data to evaluate user cold-start recommendation performance. The data statistics are shown in Table 4.8.

Experimental Settings. In the experiments, the parameters β, λ_u, λ_v, ζ, η and ε are set to 0.15, 0.015, 0.015, 0.5, 0.001, and 0.1, respectively. In Gowalla dataset, α, and λ_q are set to 0.3 and 500. In Foursquare dataset, α and λ_q are set to 0.1 and 300. The latent feature dimension is set as 10.

[5]Our model can be applied in the general check-in datasets. The home location can be estimated by using the existing approach in [89, 50]

Table 4.8: The statistics of data sets.

Standard Recommendation						
Data Set	#User	#Location	#Checkin	#Train	#Test	Sparsity
Gowalla	52,216	98,351	2,577,336	2,049,630	527,706	0.0399%
Foursquare	2,551	13,474	124,933	100,033	24,900	0.2910%

	New Location Recommendation		New User Recommendation	
Data Set	#New Location	#Test	#New User	#Test
Gowalla	78,881	568,937	9,326	79,153
Foursquare	93,311	119,876	1,221	17,964

4.4.5.2 Evaluation Metrics

As POI recommender system only recommends the limited locations for users, we quantitatively evaluate our models versus other models in terms of ranking performance, i.e., Precision@K and Recall@K metrics. MAP metric, the mean of the average precision (AP) over all locations in the testing, is also adopted in the experiments to evaluate models' performance. They are formally defined as follows:

$$Precision@K = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathcal{S}_i(K)\cap\mathcal{T}_i}{K},$$

$$Recall@K = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathcal{S}_i(K)\cap\mathcal{T}_i}{|\mathcal{T}_i|},$$

$$MAP = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{j=1}^{\hat{m}_i} p(j)\times rel(j)}{|\mathcal{T}_i|},$$

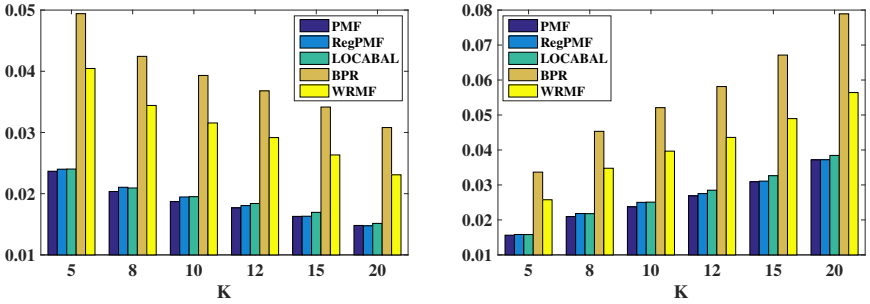
where $\mathcal{S}_i(K)$ is a set of top-K unvisited locations recommended to user i excluding those locations in the training, and $\mathcal{T}_i$ is a set of locations that are visited by user i in the testing. $\hat{m}_i$ is the number of the returned locations in the list for user i, $p(j)$ is the precision of a cut-off rank list from 1 to j, and $rel(j)$ is an indicator function that equals to 1 if the location is visited in the testing, otherwise equals to 0.

4.4.5.3 Baseline Methods

To comparatively demonstrate the effectiveness of our models, we compare them with the following seven models:

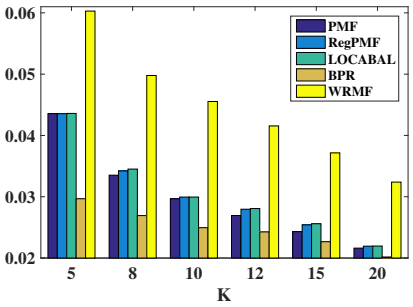
- **USG** [60], which combines geographical influence, social network and user interest with collaborative filtering;
- **IRenMF** [61], which models geographical influence by incorporating neighboring characteristics into weighted matrix factorization in both instance level and region level;
- **LOCABAL** [69], which models two types of social relations: social friends and the users with high global reputations, in the framework of matrix factorization;
- **RegPMF** [67], which models the influence of social network by placing a social regularization constraint on learning user-specific feature vectors between friends;
- **PMF** [14], which minimizes the square error loss only using the observed check-ins based on matrix factorization.
- **WRMF** [20], which minimizes the square error loss by assigning both observed and unobserved check-ins with different confidential values based on matrix factorization;
- **BRP** [28], which optimizes the ordering of the preference for the observed location and the unobserved location.

In this section, we develop two methods to learn the potential locations for each user (i.e., LA, RW) and then design two loss functions: $ASMF$ and $ARMF$. Thus we consider the following combinations: $ASMF + LA$, $ASMF + RW$, $ARMF + LA$, $ARMF + RW$, which are denoted as **ASMF-LA**, **ASMF-RW**, **ARMF-LA**, **ARMF-RW**, respectively.

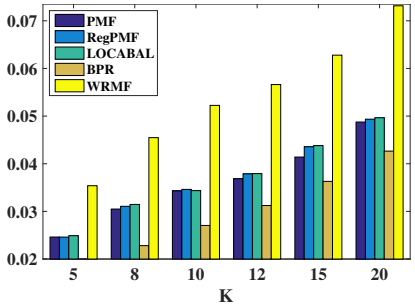


(a) Precision@K on Gowalla

(b) Recall@K on Gowalla



(c) Precision@K on Foursquare



(d) Recall@K on Foursquare

Figure 4.14: The performance comparison of standard recommendation of basic methods in terms of precision@K and recall@K.

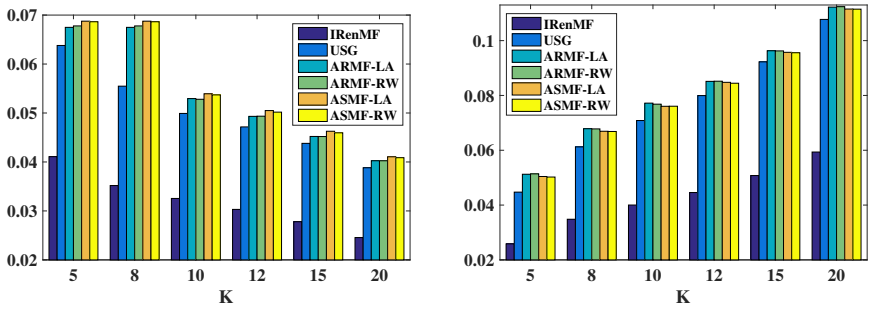
4.4.5.4 Performance Comparison

In this section, we evaluate the proposed models for standard recommendation, new location and new user recommendation in terms of Precision@K, Recall@K and MAP.

4.4.5.4.1 Performance of Standard Recommendation

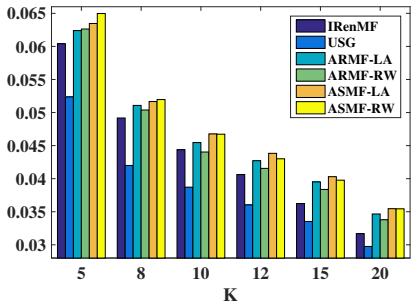
The performance comparison of our models and baseline models in terms of Precision@K, Recall@K, and MAP are shown in Figure 4.14, Figure 4.15 and Table 4.9.

Modeling observed check-ins v.s. modeling all check-ins. From the results, we can see that *WRMF* and *BPR* almost outperform *LOCABAL*, *RegPMF* and *PMF*. Even though *LOCABAL* and *RegPMF* incorporate social network into matrix factorization, the sparseness of data due to only modeling the observed check-ins results in their poor performance. Both *LOCABAL* and *RegPMF* are slightly superior to *PMF*. One possible explanation is that social network assists to make more accurate recom-

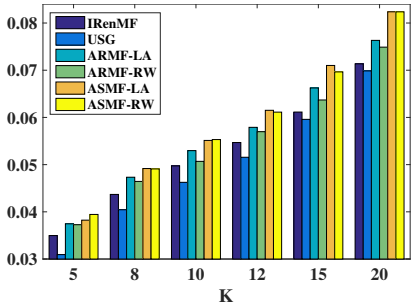


(a) Precision@K on Gowalla

(b) Recall@K on Gowalla



(c) Precision@K on Foursquare



(d) Recall@K on Foursquare

Figure 4.15: The performance comparison of standard recommendation of our models and other models in terms of precision@K and recall@K.

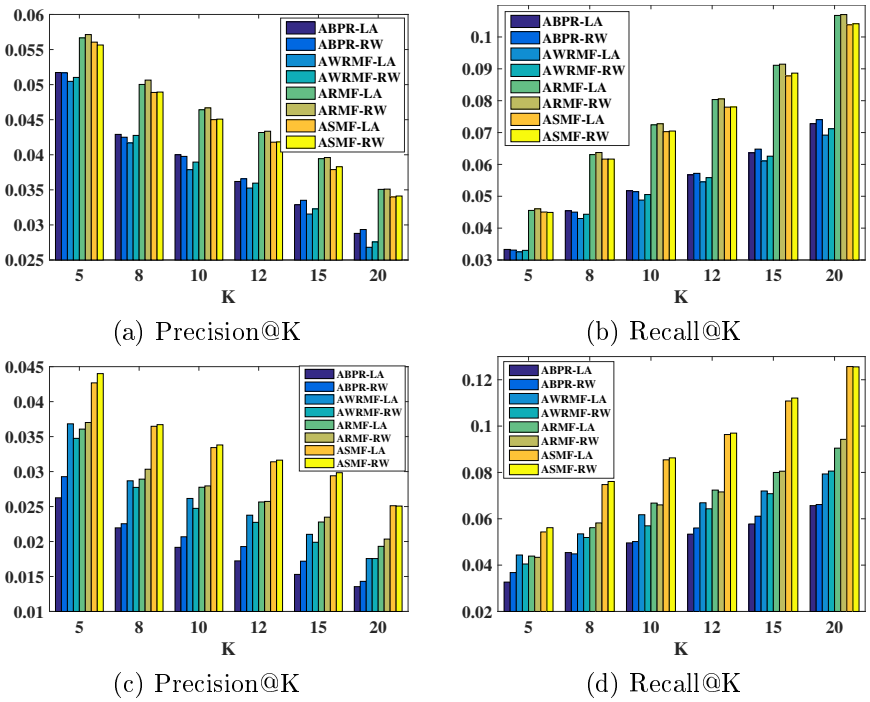
mendation. Different from them, *WRMF* not only utilizes the observed check-ins, but also models negative preference for all unvisited locations with a low confidence. But *BPR* easily leads to bias by only sampling some unvisited locations, which explains why it performs not good in Foursquare data set.

Our models v.s. baseline models. Our models achieve the best performance in both data sets with all evaluation metrics, illustrating the superiority of our approaches. Although *USG* exploits social influence, geographical effect and user interest, its simple linear combination results in the poor performance. As Gowalla covers a much larger area than Foursquare, the clustering result is not good, leading to the worse performance of *IRenMF* in Gowalla than in Foursquare. *ARMF* and *ASMF* have different performance in two datasets which is consistent with performance of *WRMF* and *BPR*. Their similar performance in Gowalla is due to the much more evident spatial clustering phenomenon. Two approaches to learn potential lo-

Table 4.9: The performance comparison of standard recommendation of our models and baseline methods in terms of MAP.

Data Set	ASMF-RW	ASMF-LA	ARMF-RW	ARMF-LA	USG	IRenMF
Gowalla	0.05700	0.05713	0.05715	0.05705	0.05205	0.02554
Foursquare	0.04167	0.04064	0.03857	0.03907	0.03464	0.03683
Data Set	WRMF	BPR	LOCABAL	RegPMF	PMF	
Gowalla	0.02470	0.03652	0.01446	0.01388	0.01357	
Foursquare	0.03626	0.01923	0.02344	0.02325	0.02288	

cations perform similarly. But *LA* is more efficient than *RW* because it does not require any matrix operation. In addition, the better performance of our models in Gowalla than in Foursquare is for the sake of (1) the stronger correlation in Gowalla which is reflected in Figure 4.12c and Figure 4.12d, and (2) the more detailed category in Gowalla than in Foursquare, where Gowalla has 262 kinds of categories while Foursquare only has 10 different categories.



(a) Precision@K (b) Recall@K

(c) Precision@K (d) Recall@K

Figure 4.16: The performance comparison of new user recommendation in terms of Precision@K and Recall@K on Gowalla dataset (top) and Fousquare dataset(bottom).

Table 4.10: The performance comparison of new location recommendation.

	P@5	P@8	P@10	R@5	R@8	R@10	MAP
Gowalla Data Set							
ASMF-RW	0.08956	0.08543	0.08320	0.06032	0.08029	0.09259	0.08424
ASMF-LA	0.08967	0.08549	0.08323	0.06020	0.08035	0.09268	0.08430
ARMF-RW	0.09463	0.08946	0.08666	0.06457	0.08576	0.09822	0.08768
ARMF-LA	0.09456	0.08927	0.08643	0.06449	0.08564	0.09794	0.08766
USG	0.08632	0.07826	0.07407	0.05448	0.07645	0.08885	0.07578
IRenMF	0.00073	0.00094	0.00104	0.00033	0.00057	0.00079	0.00271
Foursquare Data Set							
ASMF-RW	0.04195	0.04276	0.04171	0.00419	0.00697	0.00843	0.02036
ASMF-LA	0.04171	0.04257	0.04230	0.00411	0.00680	0.00860	0.02040
ARMF-RW	0.04061	0.04085	0.04057	0.00382	0.00664	0.00823	0.02010
ARMF-LA	0.04022	0.04000	0.04002	0.00457	0.00701	0.00856	0.02051
USG	0.03551	0.03594	0.03452	0.00268	0.00561	0.00714	0.01592
IRenMF	0.00401	0.00339	0.00314	0.00038	0.00050	0.00055	0.00346

4.4.5.4.2 Performance of New POI Recommendation

In this section, we evaluate the model performance of addressing location cold-start issue. To recommend new locations, we predict the check-in probability for each new location and then recommend the top-K locations with the highest probabilities. Note that, among all the baseline methods, only *USG* and *IRenMF* can be applied here. Since new locations are never checked-in by any users, *USG* is reduced to only model the geographical influence. In addition, the latent location vectors for new locations are not learned in the training, so one user's preference for a new location in *IRenMF* model is actually dependent on her preferences for this location's neighborhoods. The model performance in terms of precision, recall and MAP is shown in Table 4.10.

Based on the results, we can observe that *IRenMF* performs the worst among all methods on both datasets. Although taking advantage of the similarities of neighboring locations, *IRenMF* fails to appropriately model user's check-in behaviours. It happens likely due to that it does not well exploit the inherent characteristics of geographical distance. On the other hand, our models and *USG* utilize the power-law distribution to capture the spatial clustering phenomenon for user's check-in activi-

Table 4.11: The performance comparison of new user recommendation in terms of MAP.

Data Set	ASMF-RW	ASMF-LA	ARMF-RW	ARMF-LA
Gowalla	0.05442	0.05427	0.05589	0.05562
Foursquare	0.04831	0.04836	0.03770	0.03718
Data Set	AWRMF-RW	AWRMF-LA	ABPR-RW	ABPR-LA
Gowalla	0.03021	0.02921	0.02423	0.02396
Foursquare	0.03242	0.03683	0.02971	0.02813

ties, which is based on the observation over data. Therefore, they have much better performance than *IRrenMF* model in location cold-start recommendation. Also, our models gain superior performance over *USG*. A possible reason is that a user's preference latent vector has been learned in the training so that her preference on the target location's neighborhoods can be accurately predicted. However, *USG* only leverages the geographical similarity between a new location and her historical POIs as prediction. In addition, the performance of *ASMF* and *ARMF* is consistent with earlier experimental results.

4.4.5.4.3 Performance of New User Recommendation

In this section, we evaluate model's recommendation performance for user cold-start problem. When a new user enters the system, we do not have her historical check-in information. As a result, her latent vector cannot be learned and all of these baseline methods could not address this problem. The proposed models elaborate the historical check-ins of a user's neighboring friends (and social friends if she has) to learn her preference vector. Thus, they can be adopted to cope with user cold-start problem. As the proposed augmenting framework could be adapted to *WRMF* and *BPR* based models, we construct the following baseline methods with the similar loss functions in Eq.(4.40) and Eq.(4.33): (1) *WRMF+LA*, denoted as *AWRMF-LA*; (2) *WRMF+RW*, denoted as *AWRMF-RW*; (3) *BPR+LA*, denoted as *ABPR-LA*; (4) *BPR+RW*, denoted as *ABPR-RW*. The precision, recall and MAP of these models

over two datasets are shown in Figure 4.16 and Table 4.11.

From the results, we find that all models have good performance to address user cold-start problem. The augmenting approach with friends' historical check-ins significantly benefits the location recommendation, in particular user cold-start recommendation. Meanwhile, the successful application of the augmenting strategy in *WRMF* and *BPR* demonstrates that the proposed augmenting strategy can be easily applied in any square error and ranking error based matrix factorization models. Moreover, our models perform much better than the baseline approaches for the sake of exploiting geographical influence and category information. *ARMF* and *ASMF* perform consistently as above results. Overall, we can see that our models can handle user cold-start problem very well.

4.4.6 Conclusion

In Section 4.4, we propose a two-step framework for POI recommendation problem, which considers the check-in information of three types of friends, i.e., social friends, location friends and neighboring friends. Specifically, in the first step, we design two approaches to learn the locations that a user's friends have checked-in before and she is most interested in. In the second step, we develop matrix factorization based models with two different error loss functions using the learned potential locations. Specifically, the square error based loss extends a binary preference to a ternary variable for the observed check-ins, potential check-ins and other unobserved check-ins, and the ranking error based loss models the ranking of user's preference for her visited locations, potential locations, and unvisited locations. Finally, experimental results on two real-world data sets clearly validate the improvement of our models over many baseline methods based on different validation metrics.

CHAPTER 5: FRIEND RECOMMENDATION

5.1 Introduction

The online social network of friends and contacts has been a key of social networking sites since their beginning. In recent years, the prevailing online social network services, such as Facebook, Foursquare, Jiepang, and Dianping, provide novel ways for people to communicate and build virtual communities [1]. These online social network services, also named as location-based social networks (LBSNs), not only facilitate users to build social relationship, but also provide opportunity for users to explore their desired point-of-interests in a real-time fashion [2]. With their popularity, a large amount of user engagement data has been accumulated through these platforms, such as online friendship establishing and information sharing, which enables the potential of research to work on the task of friend recommendation in LBSNs.

Friend recommendation is a crucial approach to help users discover new friends and interesting information. Unfortunately, the heterogeneity of links in LBSNs and the data sparsity issue possess great challenges for research work to help online users explore new friendships. First, different from pure social network which only has social relationship between users, such as Twitter, LBSNs comprise two heterogeneous links: social link and consumption link. A social link refers to the relationship between two users, and a consumption link indicates a user's check-in activity at a particular location. Actually, it is hard to have users check-in at many locations and build multiple social connections at the same time by the reason of limited positions in websites to show potential locations and friends for each individual user. In other words, listing and showing all potential locations and friends to users is unrealistic, as a result, stunting the growth of heterogeneous links in LBSNs. For example, a friend

recommender system only suggests new friends to users to expand their social circle. It cannot guarantee they will explore new places and have check-in activities as well. Second, recommender systems suffer from another challenge caused by the extremely sparse data. In real systems, there are over millions of users. However, the number of each user's friends is usually very small. It is not sufficient to use such sparse data to successfully train machine learning models.

Recently a variety of friend recommendation approaches have been developed [71], however, they produce recommendation results by only maximizing the single utility, i.e., social utility, without considering consumption utility. The social utility (or consumption utility) is defined as the benefit that one user gets from a newly added social link (or consumption link). In fact, a newly added social link may lead to more consumption links. To enable these two types of links to grow faster together, in this chapter, we propose to make friend recommendations by maximizing the bi-utility of social link. Maximizing the bi-utility of social link is to maximize both social utility and consumption utility of a social link. Specifically, the consumption utility of a social link for a user is defined as the utility of the potential new consumption links between this user and the locations that the corresponding new friend has checked-in before. It is modeled by the probability that she will check-in at the locations visited by this new friend after they become friends. Social utility of a social link is the likelihood of the creation of such social link, which is similar to conventional approaches. Upon these, we propose the Maximum Bi-utility of Social Links (MBSL) model for friend recommendation in LBSNs, and design a scalable optimization algorithm to learn optimal parameters.

5.2 Notation and Problem Definition

Social friendship and user check-in behavior in LBSNs are inextricably intertwined. There are two types of links: consumption link and social link. A consumption link is defined as a user's check-in behavior on a location depicted by the solid line with

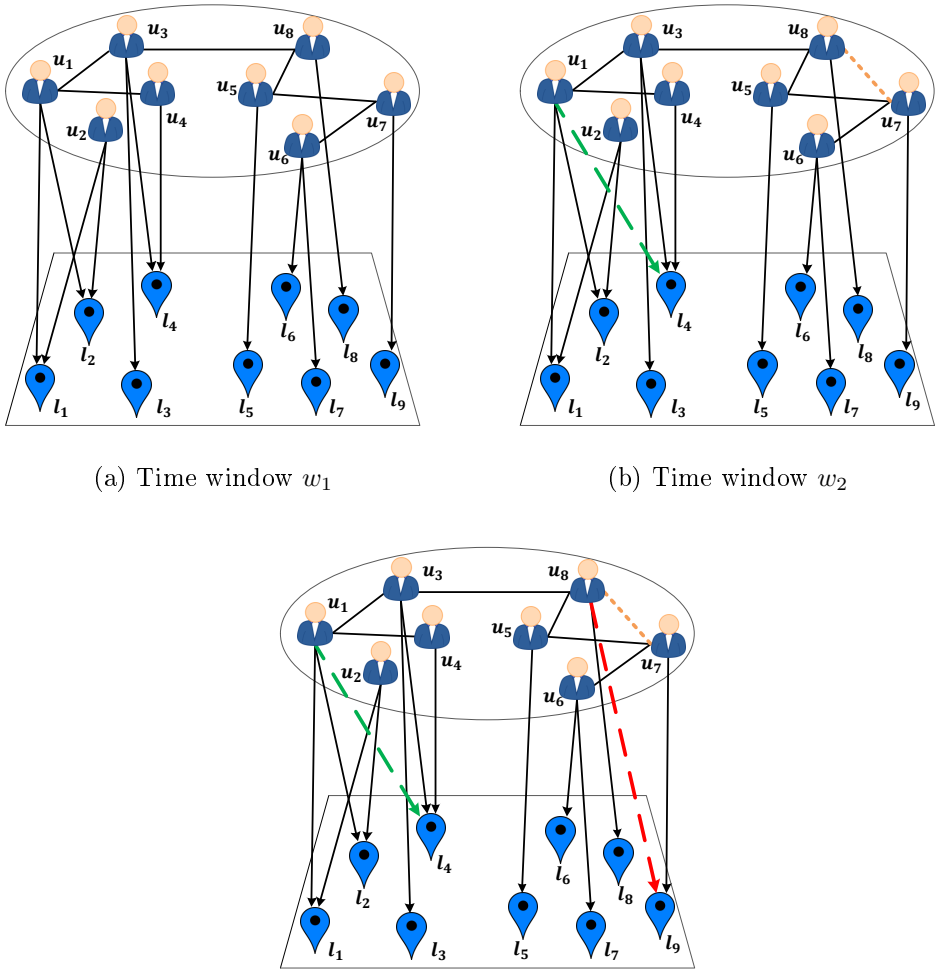
Table 5.1: Mathematical notations.

Symbol	Description
N	the number of users
M	the number of locations
W	the number of time windows
i	the index of user
k	the index of friend
j	the index of location
w	the index of time window
$\mathcal{N}^w$	the set of users who have check-ins in time window w
t_{ij}^w	timestamp of user i to location j in time window w
t_{ik}^w	timestamp of user i with friend k in time window w
$\mathcal{M}_i^w$	the set of user i's checked-in locations in time window w
$\mathcal{F}_i^w$	the set of user i's new friends in time window w
$\mathcal{H}_i^w$	the set of user i's cumulative locations by the end of w
$\mathcal{S}_i^w$	the set of user i's cumulative friends by the end of w
$\mathcal{V}_j^w$	the set of location j's visitors by the end of w
$\mathcal{M}_i$	the set of all locations checked-in by user i
$\mathcal{F}_i$	the set of user i's all friends
$\mathcal{W}_i$	the set of active time windows where user i has checked-in actions (or builds social connections)

arrow in Figure 5.1a, and a social link is a social connection between users like the solid line in Figure 5.1a. The consumption links form a consumption network and the social links form a social network. The network is denoted as both of social network and consumption network.

Notations. To investigate the network's dynamic changes, we collect the network information for W periodically consecutive time windows, where each time window w for $w = 1, \cdots, W$ refers to time period $(t_{w-1}, t_w]$. Specifically, we define $t_0 = -\infty$. For convenience we will henceforth refer to w as *time window*, i as *user*, k as *friend* and j as *location* unless stated otherwise. More notations are reported in Table 5.1.

Problem Definitions. In this chapter, we intend to explore friend recommendation task. The consumption utility (or social utility) is defined as the benefit that a user gets when she adds a consumption link (or a social link) [93]. Traditional friend



(a) Time window w_1

(b) Time window w_2

(c) Time window w_3

Figure 5.1: (a) $\sim$ (c) are the examples of three consecutive time windows. The colors other than black indicate the newly added links.

recommender systems only generate recommendation results by only maximizing the single social utility. However, a newly added social link also possibly leads to more consumption links. Different from traditional approaches, we propose to make recommendation by maximizing the the bi-utility of a social link, which allows these two types of links grow faster together in the whole network.

Specifically in **friend recommendation problem**, the consumption utility of a social link denotes as the utility of new check-ins by the influence of this social link. Friends possibly affect a user's check-in decision making process, which allows her to

have more check-ins. Hence, we define the bi-utility of a social link as follows:

Definition 9 *The bi-utility of a social link is the summation of its corresponding social utility and potential consumption utility by the influence of this social link.*

For example, during time window w_1 in Figure 5.1a, we recommend u_8 for new friend u_7. In the next time window w_2, she makes friend with this user, which generates the corresponding social utility. Then in w_3, she checks-in location l_9 due to this newly established friendship. As her friend u_7 has checked-in location l_9 before, she is possible to check-in this location by the influence of this friend. Therefore, *the task of friend recommendation in this chapter is to recommend the target user with the new friend which she is interested in by maximizing the bi-utility created by this friendship.*

5.3 Methodology

We aim at recommending users with new friends by maximizing the bi-utility of the corresponding social link so that they can not only have more friends but also have more check-ins. We first define the bi-utility of a social link, and then introduce the proposed model and its parameter estimation.

5.3.1 The Bi-utility of a Social Link

In this chapter, the goal of friend recommendation is that users are able to make more friends online and at the same time will have more check-ins by repeating friends' historical check-ins. In other words, we want a social link to be able to generate multiple consumption links. Different from conventional recommendation methods, this approach could make the whole network grow much more rapidly. Based on the Definition 9, the bi-utility r_{ik}^w of a social link from user i to user k during time window w is defined as:

$$r_{ik}^w = r^s(e_i^w = k) + r^c(e_i^w = k), \tag{5.1}$$

where $r^s(e_i^w = k)$ is the social utility, and $r^c(e_i^w = k)$ is the consumption utility by the influence of this social link and observed in the next time window $w+1$. Here we assume the utility of a single social link is β_s and the utility of a single consumption link is α_s. Hence the observed bi-utility r_{ik}^w consists of two parts: (1) the social utility and (2) the consumption utility that user i repeats user k's checked-in locations in the next time window $w+1$, which is defined as follows:

$$r_{ik}^w = \beta_s + \alpha_s \sum\nolimits_{j \in \mathcal{M}_i^{w+1} \wedge j \in \mathcal{H}_k^w} r^c(c_i^{w+1} = j | e_i^w = k), \tag{5.2}$$

where $r^c(c_i^{w+1} = j|e_i^w = k)$ is the probability that user i checks-in location j in the next time window $w+1$ due to that her friend k has checked-in before. Those friends who have checked-in this location are also possible to influence her. For example, in Figure 5.1b, user u_1 checks-in location l_4 likely due to that her friends u_3 and u_4 have checked-in before and then influence her. We consider the time factor on modeling the influence strength. Hence, we assume that (1) only the increased friendships in time window w will affect the new check-in actions in the next time window $w+1$; (2) the influence strength is exponentially damped by time. Therefore, the normalized probability is defined as follows:

$$r^c(c_i^{w+1} = j | e_i^w = k) = \frac{|t_{ij}^{w+1} - t_{ik}^w|^{-\beta}}{\sum_{o \in \mathcal{F}_i^w} |t_{ij}^{w+1} - t_{io}^w|^{-\beta}}, \tag{5.3}$$

where β is the damping factor and $\mathcal{F}_i^w$ is the set of user i's friends who build social connection with user i during time window w, and have checked-in location j before.

5.3.2 Model Presentation

The predicted bi-utility $\hat{r}_{ik}^w$ of a social connection from user i to user k at a specific time t_{ik}^w during time window w consists of (1) the potential social utility that this link directly generates due to the influence of social network and (2) the potential

consumption utility of the check-in actions that user i repeats user k's historical check-ins after they build connections, which are influenced by the whole network, i.e., both social network and consumption network. It is defined as follows:

$$\hat{r}_{ik}^{w} = \underbrace{P(e_i^w = k|\widetilde{\mathcal{S}}_{ik}^w, \tilde{\mathcal{S}}_{k\leftarrow i}^w)}_{\text{Social Preference}}(\beta_s + \alpha_s z_i \underbrace{\sum_{j\in\mathcal{H}_{k,/i}^w} P(c_i^w = j|\mathcal{H}_i^w, \mathcal{H}_k^w, \mathcal{S}_i^w, \mathcal{S}_k^w)}_{\text{Check-in Preference}}), \tag{5.4}$$

where $\mathcal{H}_{k,/i}^w = \{l|l \in \mathcal{H}_k^w \wedge l \notin \mathcal{H}_i^w\}$ is the set of locations that user k has checked-in but user i never checked-in before the end of time window w, and $\tilde{\mathcal{S}}_{k\leftarrow i}^w$ indicates the set of friends that user k made before the specific time t_{ik}^w during the time window w. $P(e_i^w = k|\widetilde{\mathcal{S}}_{ik}^w, \tilde{\mathcal{S}}_{k\leftarrow i}^w)$ is the probability that user i would like to make friends with user k, which is assumed to be influenced by the structure of social network. In fact, after building a new social connection, some users would like to concern about the check-in behaviors of this friend, and then they might visit the corresponding locations; While some others do not care about it and would not produce corresponding consumption links. To model this kind of behavior, we introduce a variable $z_i \geq 0$ to indicate the weight that user i wants to repeat her friend's checked-in locations after building a social connection. $P(c_i^w = j|\mathcal{H}_i^w, \mathcal{H}_k^w, \mathcal{S}_i^w, \mathcal{S}_k^w)$ is the probability that user i prefers to check-in location j before the end of time window w, which is assumed to be influenced by both consumption network and social network. To model their influence, we decompose this probability as follows:

$$\begin{aligned}
&P(c_i^w = j|\mathcal{H}_i^w, \mathcal{H}_k^w, \mathcal{S}_i^w, \mathcal{S}_k^w) \\
&= r_s P(c_i^w = j|\mathcal{H}_i^w, \mathcal{H}_k^w) + (1 - r_s)P(c_i^w = j|\mathcal{S}_i^w, \mathcal{S}_k^w) \\
&= r_s \underbrace{P(c_i^w = j|\mathcal{H}_i^w)}_{\text{Personal Check-in Interest}} + (1 - r_s)\underbrace{P(c_i^w = j|\mathcal{S}_i^w, \mathcal{S}_k^w)}_{\text{Social Influence}},
\end{aligned} \tag{5.5}$$

where the first part captures user i's personal interest in location j, which is only dependent on her own historical check-ins, and the second part captures the social

Table 5.2: The social features for user i and k in time window w.

Features	**Descriptions**
Common Neighbors	$\lvert\mathcal{S}_i^w \cap \mathcal{S}_k^w\rvert$
Adamic/Adar	$\sum_{z \in \mathcal{S}_i^w \cap \mathcal{S}_k^w} \frac{1}{\log \lvert\mathcal{S}_z^w\rvert}$
Jaccard Similarity	$\frac{\lvert\mathcal{S}_i^w \cap \mathcal{S}_k^w\rvert}{\lvert\mathcal{S}_i^w \cup \mathcal{S}_k^w\rvert}$
Cosine Similarity	$\frac{\lvert\mathcal{S}_i^w \cap \mathcal{S}_k^w\rvert}{\lVert\mathcal{S}_i^w\rVert \times \lVert\mathcal{S}_k^w\rVert}$
Common Ratio	$\frac{\lvert\mathcal{S}_i^w \cap \mathcal{S}_k^w\rvert}{\lvert\mathcal{S}_i^w\rvert \times \lvert\mathcal{S}_k^w\rvert}$

influence on user's repeating behavior. It reveals the interplay of consumption network and social network on a consumption link's formation. $\gamma_s \in [0, 1]$ is a tuning parameter that controls the contribution of personal check-in interest and social influence. It is worth to note that each part of Eq.(5.4) and Eq.(5.5) can be modeled with many existing approaches. We will introduce how to model each of them in details in the following sections.

5.3.2.1 Modeling Social Preference

In social network, the generation of a social link is dependent on two users' social structures [71]. If they have similar social structures, they will have a large probability to build social connection with each other. Motivated by this, G-dimensional observed social features between user i and k in time window w, denoted as $\mathbf{q}_{ik}^w$, can be incorporated into a linear regressor $\mathbf{p}_i \cdot \mathbf{q}_{ik}^w$ for modeling their similarity, where $\mathbf{p}_i$ is the G-dimensional weight of the linear regressor for user i. It is worth to note that here the social structure $\mathbf{q}_{ik}^w$ is observed at the particular time t_{ik}^w. In the experiments, we will use the observed features in Figure 5.2. Actually $\mathbf{q}_{ik}^w$ is fully defined based on empirical dataset, which can be extended to include other features. In addition, we introduce another H-dimensional latent matrix $\mathbf{X}$, and use $\mathbf{x}_i \cdot \mathbf{x}_k$ to capture other latent similarity between user i and k [74, 75]. Formally, the probability that user i

builds social connection with k by the reason of social structure is defined as follows:

$$P(e_i^w = k|\widetilde{\mathcal{S}}_{ik}^w, \tilde{\mathcal{S}}_{k\leftarrow i}^w) \propto \mathbf{p}_i \centerdot \mathbf{q}_{ik}^w + \mathbf{x}_i \centerdot \mathbf{x}_k. \tag{5.6}$$

5.3.2.2 Modeling Personal Check-in Interest

We factorize the user's check-in preference into user and location latent spaces [14, 15]. Let $\mathbf{U}^w \in \mathbb{R}^{D\times|\mathcal{N}^w|}$ and $\mathbf{V} \in \mathbb{R}^{D\times M}$ be the latent user and location feature matrices, with column vectors $\mathbf{u}_i^w$ and $\mathbf{v}_j$ representing the D-dimensional user i's feature vector and location j's feature vector, respectively. User's preference may change over time which is time-dependent. To capture the dynamics of a user's interest, we introduce $\mathbf{u}_i^w$ for each time window w to indicate her corresponding check-in preference. Hence, the check-in preference of user i for location j in time window w is modeled as follows:

$$P(c_i^w = j|\mathcal{H}_i^w) \propto \mathbf{u}_i^w \centerdot \mathbf{v}_j. \tag{5.7}$$

Location characteristics captured by $\mathbf{V}$ are inherent properties and do not change much as time goes, so we assume location characteristics to be time-independent [68, 46]. In LBSNs, user's check-in preference in each time window is correlated with her previous ones. In other words, $\mathbf{U}^w$ is dependent on each $\mathbf{U}^h$ for $h \in \mathcal{W}_i \wedge h < w$. Inspired by the autoregressive model [94, 95, 96, 97], the preference of user i in time window w is defined as follows:

$$\mathbf{u}_i^w = \sum_{h<_i w} \varphi_i^{hw} \mathbf{u}_i^h + \varepsilon_h, \tag{5.8}$$

where $h <_i w$ is defined as $\{h|h \in \mathcal{W}_i \wedge h < w\}$, and ε_h is the D-dimensional Gaussian noise. φ_i^{hw} is the temporal similarity that models the temporal correlation of user i's check-in preferences between time window h and w. We leverage the cosine similarity

of user's check-in behaviors to quantitatively measure the value of φ_i^{hw}, given by:

$$\varphi_i^{hw} = \frac{|\mathcal{M}_i^w \cap \mathcal{M}_i^h|}{\|\mathcal{M}_i^w\| \times \|\mathcal{M}_i^h\|}, \tag{5.9}$$

where $\|\cdot\|$ is the l_2 norm. Specifically, for the testing data, we use $\mathbf{u}_i^{|\mathcal{W}_i|}$ in her last time window to do prediction.

5.3.2.3 Modeling Social Influence

User i would like to repeat friend k's checked-in location j due to two factors: their common check-ins and geographical distances. The more common check-ins they have, the larger probability to repeat friend's check-ins she has. Also, her check-in behavior is affected by geographical influence. For example, in Figure 5.1a, users in the left side would have small opportunities to check-in far away locations like in the right side. The check-in probability due to the social influence is defined as:

$$P(c_i^w = j|\mathcal{S}_i^w, \mathcal{S}_k^w) := o_{ikj}^w = \frac{|\mathcal{H}_i^w \cap \mathcal{H}_k^w|}{|\mathcal{H}_i^w \cup \mathcal{H}_k^w|} \times P_s(d(h_i, h_j)), \tag{5.10}$$

where we use the power-law distribution $P_s(d(h_i, h_k))$ to capture the correlation between user's check-in preference and geographical influence [60, 2].

The MBSL Model

Based on Eq.(5.4), Eq.(5.6), Eq.(5.7), and Eq.(5.10), we can obtain the predicted bi-utility $\hat{r}_{ik}^w$ that user i builds social connection with user k during time window w as follows:

$$\hat{r}_{ik}^w = (\mathbf{p}_i \cdot \mathbf{q}_{ik}^w + \mathbf{x}_i \cdot \mathbf{x}_k)(\widetilde{\beta}_s + z_i \sum_{j \in \mathcal{H}_{k,/i}^w} \widetilde{\alpha}_{s1} \mathbf{u}_i^w \cdot \mathbf{v}_j + \widetilde{\alpha}_{s2} o_{ikj}^w), \tag{5.11}$$

where $\widetilde{\alpha}_{s1} = \widetilde{\alpha}_s \gamma_s$, $\widetilde{\alpha}_{s2} = \widetilde{\alpha}_s(1 - \gamma_s)$, $\widetilde{\beta}_s$ is the weight of social utility and $\widetilde{\alpha}_s$ is the weight of consumption utility.

Based on above equation, we propose the Maximum Bi-utility of Social Links (MBSL) model for friend recommendation by taking both observed and unobserved bi-utilities into consideration. The objective function of MBSL model is given by:

$$\min \sum_{i} \sum_{w \in \mathcal{W}_i} \sum_{k \in \widetilde{\mathcal{F}}_i^w} a_{ik}^w (r_{ik}^w - \hat{r}_{ik}^w)^2 + \|\mathbf{\Theta}\|^2, s.t. \ \forall z_i \geq 0, \tag{5.12}$$

where the unobserved bi-utility is 0, and $\|\mathbf{\Theta}\|^2$ is the regularization term defined with $\lambda_\star$ as regularization parameters:

$$\begin{aligned} \|\mathbf{\Theta}\|^2 = \lambda_w \sum\nolimits_i \sum\nolimits_{w \in \mathcal{W}_i} \|\mathbf{u}_i^w - \sum\nolimits_{h <_i w} \varphi_i^{hw} \mathbf{u}_i^h\|^2 + \lambda_u \|\mathbf{U}\|_F^2 \\ + \lambda_v \|\mathbf{V}\|_F^2 + \lambda_p \|\mathbf{P}\|_F^2 + \lambda_x \|\mathbf{X}\|_F^2 + \lambda_z \|\mathbf{z}\|^2. \end{aligned} \tag{5.13}$$

$\widetilde{\mathcal{F}}_i^w$ is defined to incorporate unobserved friends given in Eq.(5.14), where $\bar{\mathcal{F}}_i$ is the complementary set of $\mathcal{F}_i$.

$$\widetilde{\mathcal{F}}_i^w = \begin{cases} \mathcal{F}_i^w \cup \bar{\mathcal{F}}_i & \text{if } w = |\mathcal{W}_i|, \\ \mathcal{F}_i^w & \text{otherwise.} \end{cases} \tag{5.14}$$

a_{ik}^w is the confidence value. The friendship is the binary data, so its definition with parameter δ_s is given by:

$$a_{ik}^w = \begin{cases} 1 + \delta_s & k \in \mathcal{F}_i^w, \\ 1 & \text{otherwise.} \end{cases}$$

5.3.3 Parameter Estimation

In this section, we adopt Alternating Least Square (ALS) algorithm for the parameter estimation of MBSL model. The updating equation for each variable in each iteration is given in Table 5.3. Specifically, $\widetilde{\mathcal{F}}_i^*$ indicates $\{w, k | w \in \mathcal{W}_i \wedge k \in \widetilde{\mathcal{F}}_i^w\}$ and

$\mathcal{O}_j$ indicates $\{i, w, k | i \in \mathcal{N}_j \wedge w \in \mathcal{W}_i \wedge k \in \widetilde{\mathcal{F}}_i^w\}$. We place $(z_i)_+ = \max\{z_i, 0\}$ for z_i to guarantee its non-negative properties. Also, we have defined $\widetilde{\lambda}_i^w$ and $\widetilde{\mathbf{u}}_i^w$ as follows,

$$\widetilde{\lambda}_i^w = \lambda_u + \lambda_w(1 + \sum\nolimits_{a >_i w} (\varphi_i^{aw})^2), \tag{5.15}$$

$$\widetilde{\mathbf{u}}_i^w = \lambda_w \sum\nolimits_{h \neq_i w} (\varphi_i^{hw} - \sum\nolimits_{a >_i \max\{w,h\}} \varphi_i^{ha} \varphi_i^{wa}) \mathbf{u}_i^h, \tag{5.16}$$

where $h \neq_i w$ indicates $h \in \mathcal{W}_i \wedge h \neq w$. In addition, some notations are used in the following,

$$\begin{aligned}
\mathbf{v}_{ik}^w &= \sum\nolimits_{j \in \mathcal{H}_{k,/i}^w} \mathbf{v}_j, \\
o_{ik}^w &= \sum\nolimits_{j \in \mathcal{H}_{k,/i}^w} o_{ikj}^w, \\
\widetilde{\mathbf{q}}_{ik}^w &= (\beta_s + z_i \hat{r}_{ik}^{cw}) \mathbf{q}_{ik}^w, \\
\widetilde{\mathbf{u}}_{ik}^w &= \widetilde{\alpha}_{s1} z_i \hat{r}_{ik}^{sw} \mathbf{u}_i^w, \\
\hat{r}_{ik}^{vw} &= r_{ik}^w + \widetilde{\mathbf{u}}_{ik}^w \centerdot \mathbf{v}_j - \hat{r}_{ik}^w, \\
\hat{r}_{ik}^{cw} &= \widetilde{\alpha}_{s1} \mathbf{u}_i^w \centerdot \mathbf{v}_{ik}^w + \widetilde{\alpha}_{s2} o_{ik}^w, \\
\hat{r}_{ik}^{uw} &= r_{ik}^w - \hat{r}_{ik}^{sw} (\beta_s + \widetilde{\alpha}_{s2} z_i o_{ik}^w), \\
\hat{r}_{ik}^{sw} &= \mathbf{p}_i \centerdot \mathbf{q}_{ik}^w + \mathbf{x}_i \centerdot \mathbf{x}_f, \\
\widetilde{\mathbf{v}}_{ik}^w &= \widetilde{\alpha}_{s1} g_i \hat{r}_{ik}^{sw} \mathbf{v}_{ik}^w, \\
\widetilde{\mathbf{x}}_{ik}^w &= (\beta_s + z_i \hat{r}_{ik}^{cw}) \mathbf{x}_k.
\end{aligned}$$

Complexity Analysis. To make the optimization scalable,we utilize the sampling method to sample m unobserved friends and pre-calculate some variables in each iteration. In each iteration, the running times are $\mathcal{O}(NG^3 + \widetilde{n} \max\{G^2, H, D\})$, $\mathcal{O}(NH^3 + \widetilde{n} \max\{H^2, G, D\})$, $\mathcal{O}(N_c D^3 + \widetilde{n}_c \max\{D^2, G, H\})$, $\mathcal{O}(MD^3 + \widetilde{n}_c \max\{D^2, G, H\})$, and $\mathcal{O}(\widetilde{n} \max\{G, H, D\})$, for updating variables $\mathbf{P}$, $\mathbf{X}$, $\mathbf{U}$ and $\mathbf{V}$, $\mathbf{z}$, respectively. $\widetilde{n} = n + Nm$ and n is the number of observed bi-utilities. N_c is the number of users

Table 5.3: The optimal solutions for MBSL model.

$$\mathbf{u}_i^w = (\lambda_i^w \mathbf{I}_K + \sum_{k \in \widetilde{\mathcal{F}}_i^w} a_{ik}^w \widetilde{\mathbf{v}}_{ik}^w \widetilde{\mathbf{v}}_{ik}^{w\mathrm{T}})^{-1} (\sum_{k \in \widetilde{\mathcal{F}}_i^w} a_{ik}^w \hat{r}_{ik}^{uw} \widetilde{\mathbf{v}}_{ik}^w + \widetilde{\mathbf{u}}_i^w),$$

$$\mathbf{v}_j = (\lambda_u \mathbf{I}_K + \sum_{i,w,k \in \mathcal{O}_j} a_{ik}^w \widetilde{\mathbf{u}}_{ik}^w \widetilde{\mathbf{u}}_{ik}^{w\mathrm{T}})^{-1} \sum_{i,w,k \in \mathcal{O}_j} a_{ik}^w \hat{r}_{ik}^{vw} \widetilde{\mathbf{u}}_{ik}^w.$$

$$\mathbf{p}_i = (\lambda_p \mathbf{I}_G + \sum_{w,k \in \widetilde{\mathcal{F}}_i^*} a_{ik}^w \widetilde{\mathbf{q}}_{ik}^w \widetilde{\mathbf{q}}_{ik}^{w\mathrm{T}})^{-1} \sum_{w,k \in \widetilde{\mathcal{F}}_i^*} a_{ik}^w (r_{ik}^w - \hat{r}_{ik}^{uw} \mathbf{x}_i \centerdot \mathbf{x}_k) \widetilde{\mathbf{q}}_{ik}^w,$$

$$\mathbf{x}_i = (\lambda_x \mathbf{I}_H + \sum_{w,k \in \widetilde{\mathcal{F}}_i^*} a_{ik}^w \widetilde{\mathbf{x}}_{ik}^w \widetilde{\mathbf{x}}_{ik}^{w\mathrm{T}})^{-1} \sum_{w,k \in \widetilde{\mathcal{F}}_i^*} a_{ik}^w (r_{ik}^w - \hat{r}_{ik}^{uw} \mathbf{p}_i \centerdot \mathbf{q}_{ik}^w) \widetilde{\mathbf{x}}_{ik}^w,$$

$$z_i = (\lambda_z + \sum_{w,k \in \widetilde{\mathcal{F}}_i^*} a_{ik}^w \hat{r}_{ik}^{zw2})^{-1} \sum_{w,k \in \widetilde{\mathcal{F}}_i^*} a_{ik}^w (r_{ik}^w - \beta_s \hat{r}_{ik}^{sw}) \hat{r}_{ik}^{sw} \hat{r}_{ik}^{cw}.$$

whose friends have check-ins. $\widetilde{n}_c$ is the number of observed bi-utilities and sampled unobserved utilities for users whose friends have check-ins, and we have $\widetilde{n}_c <= \widetilde{n}$. To simplify the expression, we assume $\widetilde{n}_c \approx \widetilde{n}$. Totally, each iteration costs the running time as $\mathcal{O}(\widetilde{n} \max\{D^2, G^2, H^2\})$, where we have $n > \max\{MD, ND, NG, NH\}$. In the experiments, $Nm < n$ is satisfied, so the complexity is $\mathcal{O}(n \max\{D^2, G^2, H^2\})$ which is in linear proportion to the number of observed bi-utilities.

5.4 Experimental Results

5.4.1 Experimental Setup

Datasets. In this chapter, we use Gowalla and Dianping datasets to evaluate the performance of the proposed models. We monthly collect user's check-in and friendship information, where Gowalla is ranging from May to August in 2010, and Dianping is collected from May to August in 2015 (which focus on two districts). Each check-in record in the datasets includes a user ID, a location ID, and a timestamp, where each location has latitude and longitude information. Each friendship also has a creation timestamp. The friendship in Gowalla is undirected; while Dianping is a directed social network where we focus on the relationship between users and their followees. We remove the users who have less than 2 followees. In addition, we calculate users' home information with their activity locations according to the

Table 5.4: The statistics of data sets.

Dataset	#User	#Location	#Record	Sparsity
Location Recommendation				
Gowalla	49,139	282,212	2,272,703	0.0164%
Dianping	40,319	2,418	280,405	0.2876%
Friend Recommendation				
Gowalla	34,873	152,787	640,756	0.0527%
Dianping	33,024	2,127	144,237	0.0132%

proposed method [89]. The data statistics are shown in Table 5.4.

Experimental Settings. In the experiments, $\widetilde{\alpha}_s$, $\widetilde{\beta}_s$, γ_s, λ_u, λ_v, λ_x and δ_s are set as 0.5, 1, 0.5, 5, 15, 15 and 2, respectively. The other parameters are set as 0.015, and the sampling size is 50.

5.4.2 Evaluation Metrics

Testing Methodology. Different from traditional methods which use boolean value in friend recommendation as testing rating, we calculate the defined bi-utility of a social link shown in Eq.(5.2) which is used as the rating in testing data. Thus we will use this newly labeled bi-utility of link as benchmark to evaluate the model performance. In the experiments, α_s, and β_s are 1 and β is 0.3.

Evaluation Metrics. The models are quantitatively evaluated in terms of prediction accuracy, top-K recommendation performance, and ranking accuracy.

Prediction Accuracy. The Root Mean Square Error (RMSE) [15, 98] is exploited to directly measure the prediction accuracy for the unobserved bi-utilities in the testing.

Top-K Recommendation. As recommender system only recommends the limited friends for users, we adopt Precision@K and Recall@K metrics to evaluate model's recommendation performance, which are defined as:

$$P@K = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathcal{S}_i(K)\cap\mathcal{T}_i}{K}, \quad R@K = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathcal{S}_i(K)\cap\mathcal{T}_i}{|\mathcal{T}_i|},$$

where $\mathcal{S}_i(K)$ is a set of top-K new friends recommended to user i excluding those in training, and $\mathcal{T}_i$ is a set of relevant friends which are defined as the following steps: for each individual, (1) sorting the friends in testing according to their bi-utility values; (2) selecting 90% ones with the maximum values.

Ranking Accuracy. To evaluate model's ranking accuracy, MAP and AUC are adopted as measurements:

$$MAP = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{j=1}^{m_i} p_i(j) \times r_i(j)}{|\mathcal{T}_i|}, AUC = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{(j,k)\in\mathcal{E}(i)} I(\hat{r}_{ij} > \hat{r}_{ik})}{|\mathcal{E}(i)|},$$

where m_i is the number of returned links in the list for user i, $p_i(j)$ is the precision of a cut-off rank list from 1 to j, and $r_i(j)$ is an indicator function that equals to 1 if corresponding link in position j is relevant, otherwise is 0. $\mathcal{E}(i)$ is defined as $\mathcal{E}(i) := \{j, k | j \in \mathcal{T}_i \wedge k \notin \mathcal{T}_i \cup \mathcal{T}_i^t\}$, where $\mathcal{T}_i^t$ is a training data of user i. $I(\cdot)$ is an indicator function which is equal to 1 if the argument is true and 0 otherwise.

5.4.3 Baseline Methods

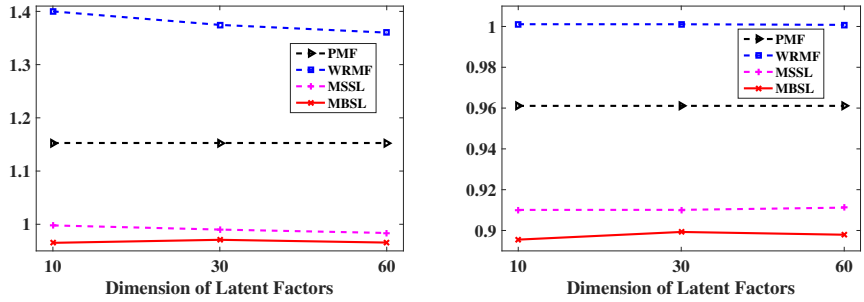
To evaluate the performance of MBSL model, we adopt the following baseline methods: (1)**PA** [71], leveraging preferential attachment as similarity score and assuming that users with more friends have a larger chance to become friends; (2)**AA** [71], taking Adamic/Adar as similarity measurement which weights the common friends by the number of their corresponding friends; (3)**PMF** [14], factorizing the user and friend into user and feature latent vectors and regarding their dot product as similarity score; (4)**WRMF** [20], assigning different confidential value for both observed and unobserved friendship. In addition, we construct another method based on MBSL model, which only maximizes the single social utilities (i.e., $\widetilde{\alpha}_s$ is 0) and is denoted as **MSSL**.

5.4.4 Performance Comparison

In this section, we evaluate the MBSL model with baseline methods for friend recommendation in terms of prediction accuracy, top-K recommendation, and ranking performance. In MBSL model, D set as 10. In the following experiments, the number of latent vector refers to H. As the friendship's formation in LBSNs is affected by the geographical influence, we also incorporate the power-law property into MBSL and MSSL models by the similar multiplication operation proposed in [2].

5.4.4.1 Prediction Accuracy Comparison

Figure 5.2 shows the prediction accuracy comparison of various models in terms of RMSE on both two datasets. As both PA and AA estimate a similarity score for a pair of users, we do not compare them in this section. To compare the values of RMSE, we do not consider the geographical influence for MBSL and MSSL model. From the results, we can see that MBSL is the best while WRMF is the worst. WRMF models the user's preference for observed friends and unobserved friends. As the number of unobserved friends is much larger than the number of observed ones, it actually spends much effort on optimizing the preference for the unobserved friends. It explains why it obtains worse result in RMSE than PMF. Both MSSL and MBSL far outperform other approaches. It illustrates that the social utilities can be more accurately modeled by considering the social structure and the underlying similarity between a pair of users. However, MBSL performs much better than MSSL. It indicates that only modeling the single social utilities is hard to achieve precise prediction for the newly constructed bi-utilities of social links. In addition, we also observe that the RMSE value is quite smaller in Dianping data than in Gowalla data, indicating that users are more active in Gowalla than in Dianping.

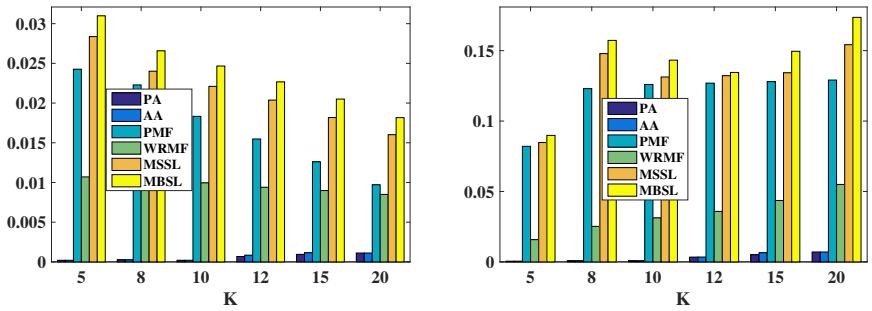


(a) Gowalla (b) Dianping

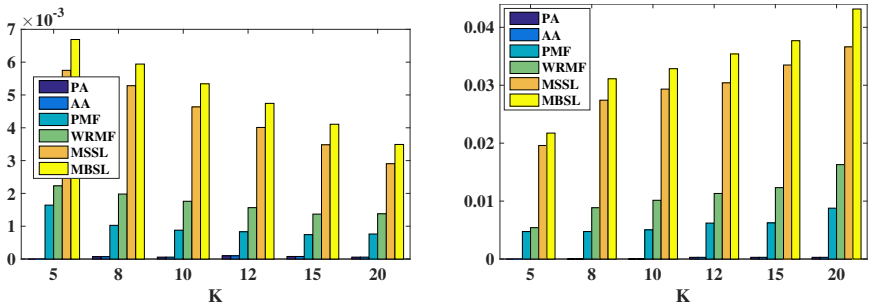
Figure 5.2: Performance comparison for friend recommendation in terms of RMSE on two datasets.

5.4.4.2 Top-K Recommendation Performance

To evaluate the top-K recommendation performance of the proposed model, we compare the performance of different models in terms of Presion@K and Recall@K in Figure 5.3, where the number of latent factor is set as 10. From the results, it can be observed that PA and AA perform the worst among all models. Although PA and AA can achieve a good prediction in [71], they fail to make the accurate recommendation. The major reason is that many pairs of users might have the same similarity score which as a result is difficult to rank them. Therefore, these two methods achieve very poor performance. WRMF and PMF have different performance in these two data sets for the sake of different sparsity degree of datasets. Both MSSL and MBSL are much better than other baseline methods due to two reasons: (1) they incorporate the observed social structure into modeling; (2) they sample the unobserved friends for modeling the unobserved utilities which actually guarantees the ranking property. The superior performance of MBSL further demonstrates that the proposed model benefits recommendation accuracy. In addition, the precision value is smaller in Dianping than in Gowalla. It happens because its data size is very small so that the number of relevant friends for each individual is also very small.



(a) Precision@K on Gowalla (b) Recall@K on Gowalla

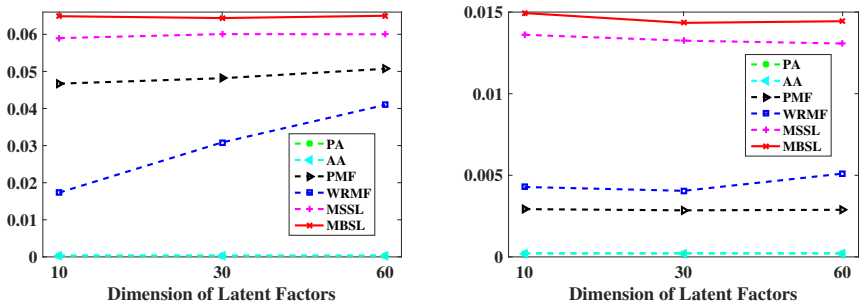


(c) Precision@K on Dianping (d) Recall@K on Dianping

Figure 5.3: Performance comparison for friend recommendation in terms of precision@K and recall@K.

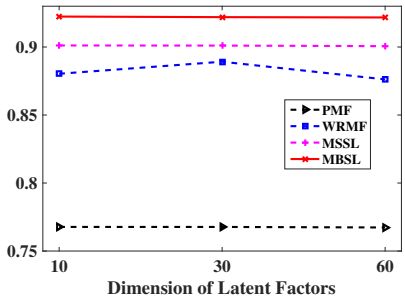
5.4.4.3 Ranking Performance Comparison

We show the ranking performance of the proposed model versus various baseline methods in terms of MAP and AUC with different dimension of latent factors in Figure 5.4 on both datasets. As PA and AA obtain about 0.48 for AUC in Gowalla which is much worse than others, we do not plot the result in the figure. The performance is consistent with the result of other metrics compared in above section. PA and AA perform the worst while MBSL is the best among all approaches. MBSL and MSSL perform much better than others. It happens likely due to the sampling of unobserved friends used to fit the unobserved utilities. The sampling method can help to not only make a low RMSE value but also achieve a high ranking performance. It proves the effectiveness of the proposed method for modeling the social utility. In addition, MBSL achieves a better performance than MSSL. It happens due to that MBSL is modeling the bi-utilities of social links while MSSL only models the single social

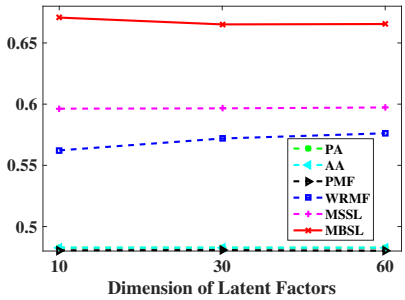


(a) MAP on Gowalla

(b) MAP on Dianping



(c) AUC on Gowalla



(d) AUC on Dianping

Figure 5.4: Performance comparison for friend recommendation in terms of MAP and AUC.

utilities. Consequently, all experimental results have proved the effectiveness of the proposed models.

5.5 Conclusion

In this chapter, we propose a novel model, namely MBSL, for friend recommendation by maximizing the bi-utility of the social link aiming at enabling both social links and consumption links to grow faster together. MBSL models the bi-utility of a social link as the summation of it social utility and the consumption utility of potential check-in behaviors between the target user and locations that the corresponding new friend has checked-in before. Finally, experimental results based on two real-world data sets clearly demonstrate the improvement of our model over baseline methods with several validation metrics.

CHAPTER 6: CONCLUSION

In this dissertation, we study two types of recommender systems in LBSNs: location recommendation and friend recommendation, helping users to explore new interesting information and expand their social circles. We investigate recommendation approaches from general and specific perspectives.

First, we propose an item recommendation approach for the general recommender system which can be practically applied to both location and friend recommender systems, and does not require additional resource for solving implicit feedback problem. Specifically, we design a scalable approach to significantly reduce the comparison scale and simultaneously preserve the pairwise ranking property by separating the positive feedback and negative feedback with the soft boundaries.

Second, we develop task-specific recommender systems for location and friend recommendation accordingly by taking advantage of human mobility pattern and social interaction nature as follows:

For location recommendation, we study various properties about user movement observed in real-world data, including geographical influence, social correlation, and temporal effect, upon which we design three different recommendation approaches. Firstly, a geo-temporal personalized recommendation approach is proposed to incorporate geographical influence and temporal effect together into latent factor models. It models a user's preference from geographical aspect with a group of observed geo-features, and captures the temporal dynamics of user's interest by modeling check-in data in a unique way. Secondly, we propose a geo-social personalized recommendation approach by dividing the whole recommendation space into two parts: social

friend space and user interest space. We then develop separate models for both spaces to recommend POIs accordingly. Thirdly, with geo-social correlation, we introduce three types of friends for each user, i.e., social friends, location friends, and neighboring friends, in LBSNs, and then develop a two-step framework to exploit these information to improve recommendation accuracy and address cold-start issue.

For friend recommendation, we study the correlation between user's check-in behavior and social relationship, and propose a novel biutility-based friend recommendation approach to enable two heterogeneous links, i.e., social link and consumption link, to grow faster together in LBSNs. To achieve this goal, we produce recommendation by maximizing the bi-utility of each social link, consisting of the corresponding social utility and consumption utility. It significantly overcomes the shortcoming of traditional approaches, i.e., only considering its single utility. Specifically, the consumption utility of a social link for a user is modeled as the potential between this user and those locations that the corresponding new friend has checked-in before. The social utility of a social link is modeled as the likelihood of the creation of this link.

REFERENCES

[1] H. Li, Y. Ge, R. Hong, and H. Zhu, "Point-of-interest recommendations: Learning potential check-ins from friends," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 975–984, 2016.

[2] H. Li, R. Hong, S. Zhu, and Y. Ge, "Point-of-interest recommender systems: A separate-space perspective," in *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pp. 231–240, 2015.

[3] H. Li, Z. W. Richang Hong, and Y. Ge, "A spatial-temporal probabilistic matrix factorization model for point-of-interest recommendation," in *Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016*, pp. 117–125, 2016.

[4] H. Li, Y. Ge, D. Lian, and H. Liu, "Learning user's intrinsic and extrinsic interests for point-of-interest recommendation: A unified approach," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pp. 2117–2123, 2017.

[5] H. Li, R. Hong, D. Lian, Z. Wu, M. Wang, and Y. Ge, "A relaxed ranking-based factor model for recommender system from implicit feedback," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 1683–1689, 2016.

[6] H. Li, H. Zhu, Y. Ge, Y. Fu, and Y. Ge, "Personalized TV recommendation with mixture probabilistic matrix factorization," in *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*, pp. 352–360, 2015.

[7] H. Steck, "Training and testing of recommender systems on data missing not at random," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pp. 713–722, 2010.

[8] B. M. Marlin and R. S. Zemel, "Collaborative prediction and ranking with non-random missing data," in *Proceedings of the 2009 ACM Conference on Recommender Systems, RecSys 2009, New York, NY, USA, October 23-25, 2009*, pp. 5–12, 2009.

[9] S. Ruslan and M. Andriy, "Bayesian probabilistic matrix factorization using markov chain monte carlo," in *ICML*, 2008.

[10] M. N. Schmidt, O. Winther, and L. K. Hansen, "Bayesian non-negative matrix factorization," in *ICASS*, 2009.

[11] H. Li, M. R. Min, Y. Ge, and A. Kadav, "A context-aware attention network for interactive question answering," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pp. 927–935, 2017.

[12] Z. Yu, H. Xu, Z. Yang, and B. Guo, "Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints," *IEEE Trans. Human-Machine Systems*, vol. 46, no. 1, pp. 151–158, 2016.

[13] D. Yang, D. Zhang, Z. Yu, and Z. Wang, "A sentiment-enhanced personalized location recommendation system," in *HT*, 2013.

[14] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pp. 1257–1264, 2007.

[15] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[16] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *KDD*, pp. 426–434, 2008.

[17] Y. Koren, "Collaborative filtering with temporal dynamics," in *KDD*, pp. 447–456, 2009.

[18] S. Rendle, "Factorization machines," in *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pp. 995–1000, 20140.

[19] S. Rendle, "Factorization machines with libfm," *ACM TIST*, vol. 3, no. 3, pp. 57:1–57:22, 2012.

[20] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, pp. 263–272, 2008.

[21] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*, pp. 502–511, 2008.

[22] S. Balakrishnan and S. Chopra, "Collaborative ranking," in *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012, Seattle, WA, USA, February 8-12, 2012*, pp. 143–152, 2012.

[23] C. Dhanjal, R. Gaudel, and S. ClÃ©menÃ§on, "Collaborative filtering with localised ranking," in *Proceedings of the Twenty-Ninth AAAI Conference on Ar-*

tificial Intelligence, January 25-30, 2015, Austin, Texas, USA., pp. 2554–2560, 2015.

[24] J. Weston, H. Yee, and R. J. Weiss, "Learning to rank recommendations with the k-order statistic loss," in *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pp. 245–248, 2013.

[25] D. Park, J. Neeman, J. Zhang, S. Sanghavi, and I. S. Dhillon, "Preference completion: Large-scale collaborative ranking from pairwise comparisons," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pp. 1907–1916, 2015.

[26] W. Pan and L. Chen, "Gbpr: group preference based bayesian personalized ranking for one-class collaborative filtering," in *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pp. 2691–2697, 2013.

[27] H. Li, Y. Ge, H. Zhu, H. Xiong, and H. Zhao, "Prospecting the career development of talents: A survival analysis perspective," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pp. 917–925, 2017.

[28] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pp. 452–461, 2009.

[29] S. Rendle and C. Freudenthaler, "Improving pairwise learning for item recommendation from implicit feedback," in *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*, pp. 273–282, 2014.

[30] G. TakÃ¡cs and D. Tikk, "Alternating least squares for personalized ranking," in *Sixth ACM Conference on Recommender Systems, RecSys '12, Dublin, Ireland, September 9-13, 2012*, pp. 83–90, 2012.

[31] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender, "Learning to rank using gradient descent," in *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, pp. 89–96, 2005.

[32] C. J. C. Burges, R. Ragno, and Q. V. Le, "Learning to rank with nonsmooth cost functions," in *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pp. 193–200, 2006.

[33] S.-H. Yang, B. Long, A. J. Smola, H. Zha, and Z. Zheng, "Collaborative competitive filtering: learning recommender using context of user choice," in *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*, pp. 295–304, 2011.

[34] M. Taylor, J. Guiver, S. Robertson, and T. Minka, "Softrank: Optimizing non-smooth rank metrics," 2008.

[35] M. Weimer, A. Karatzoglou, Q. V. Le, and A. J. Smola, "COFI RANK - maximum margin matrix factorization for collaborative ranking," in *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pp. 1593–1600, Curran Associates, Inc., 2007.

[36] Y. Shi, K. Alexandros, B. Linas, M. A. Larson, A. Hanjalic, and O. Nuria, "Tfmap: Optimizing map for top-n context-aware recommendation," in *The 35th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012*, pp. 155–164, 2012.

[37] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic, "Climf: Collaborative less-is-more filtering," in *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pp. 3077–3081, 2013.

[38] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic, "Climf: learning to maximize reciprocal rank with collaborative less-is-more filtering," in *Sixth ACM Conference on Recommender Systems, RecSys '12, Dublin, Ireland, September 9-13, 2012*, pp. 139–146, 2012.

[39] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, and A. Hanjalic, "Gapfm: optimal top-n recommendations for graded relevance domains," in *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pp. 2261–2266, 2013.

[40] Z. Cao, T. Qin, T. Liu, M. Tsai, and H. Li, "Learning to rank: from pairwise approach to listwise approach," in *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, pp. 129–136, 2007.

[41] Y. Shi, M. Larson, and A. Hanjalic, "List-wise learning to rank with matrix factorization for collaborative filtering," in *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, pp. 269–272, 2010.

[42] C. J. C. Burges, "From ranknet to lambdarank to lambdamart: An overview," tech. rep., Microsoft Research, 2010.

[43] N. N. Liu and Q. Yang, "Eigenrank: a ranking-oriented approach to collaborative filtering," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pp. 83–90, 2008.

[44] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: Successive point-of-interest recommendation," in *IJCAI*, pp. 2605–2611, 2013.

[45] Q. Yuan, G. Cong, and A. Sun, "Graph-based point-of-interest recommendation with geographical and temporal influences," in *CIKM*, pp. 659–668, 2014.

[46] M. Ye, K. Janowicz, C. Mülligann, and W. Lee, "What you are is when you are: The temporal dimension of feature types in location-based social networks," in *GIS*, pp. 102–111, 2011.

[47] S. Hanhuai and B. Arindam, "Generalized probabilistic matrix factorizations for collaborative filtering," in *ICDM*, 2010.

[48] H. Gao, J. Tang, X. Hu, and H. Liu, "Exploring temporal effects for location recommendation on location-based social networks," in *RecSys*, 2013.

[49] H. Gao, J. Tang, X. Hu, and H. Liu, "Modeling temporal effects of human mobile behavior on location-based social networks," in *CIKM*, 2013.

[50] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: User movement in location-based social networks," in *KDD*, pp. 1082–1090, 2011.

[51] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. Magnenat-Thalmann, "Time-aware point-of-interest recommendation," in *SIGIR*, 2013.

[52] M. Ye, P. Yin, and W.-C. Lee, "Location recommendation for location-based social networks," in *GIS*, pp. 458–461, 2010.

[53] B. Liu, Y. Fu, Z. Yao, and H. Xiong, "Learning geographical preferences for point-of-interest recommendation," in *KDD*, pp. 1043–1051, 2013.

[54] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "Geomf: Joint geographical modeling and matrix factorization for point-of-interest recommendation," in *KDD*, 2014.

[55] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo, "Mining user mobility features for next place prediction in location-based services," in *ICDM*, 2012.

[56] X. Li, G. Cong, X.-L. Li, T.-A. N. Pham, and S. Krishnaswamy, "Rank-geofm: A ranking based geographical factorization method for point of interest recommendation," in *SIGIR*, 2015.

[57] C. Cheng, H. Yang, I. King, and M. R. Lyu, "Fused matrix factorization with geographical and social influence in location-based social networks," in *AAAI*, 2012.

[58] J. Zhang and C. Chow, "igslr: Personalized geo-social location recommendation - a kernel density estimation approach," in *SIGSPATIAL*, pp. 334–343, 2013.

[59] M. Lichman and P. Smyth, "Modeling human location data with mixture of kernel densities," in *KDD*, 2014.

[60] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *SIGIR*, pp. 325–334, 2011.

[61] Y. Liu, W. Wei, A. Sun, and C. Miao, "Exploiting geographical neighborhood characteristics for location recommendation," in *CIKM*, pp. 739–748, 2014.

[62] H. Wang, M. Terrovitis, and N. Mamoulis, "Location recommendation in location-based social networks using user check-in data," in *SIGSPATIAL*, pp. 374–383, 2013.

[63] H. Gao, J. Tang, and H. Liu, "Exploring social-historical ties on location-based social networks," in *ICWSM*, 2012.

[64] I. Konstas, V. Stathopoulos, and J. M. Jose, "On social networks and collaborative recommendation," in *SIGIR*, 2009.

[65] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *RecSys*, 2010.

[66] S. Scellato, A. Noulas, and C. Mascolo, "Exploiting place features in link prediction on location-based social networks," in *KDD*, pp. 1046–1054, 2011.

[67] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *WSDM*, 2011.

[68] H. Gao, J. Tang, and H. Liu, "gscorr: Modeling geo-social correlations for new check-ins on location-based social networks," in *CIKM*, pp. 1582–1586, 2012.

[69] J. Tang, X. Hu, H. Gao, and H. Liu, "Exploiting local and global social context for recommendation," in *IJCAI*, 2013.

[70] D. Wang, D. Pedreschi, C. Song, and F. Giannotti, "Human mobility, social ties, and link prediction," in *KDD*, 2011.

[71] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Am. Soc. Inf. Sci. Technol.*, 2007.

[72] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[73] L. Backstrom and J. Leskovec, "Supervised random walks: predicting and recommending links in social networks," in *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM*, pp. 635–644, 2011.

[74] M. Jiang, P. Cui, R. Liu, Q. Yang, F. Wang, W. Zhu, and S. Yang, "Social contextual recommendation," in *CIKM*, 2012.

[75] D. Song, D. A. Meyer, and D. Tao, "Efficient latent link recommendation in signed networks," in *KDD*, 2015.

[76] D. Song, D. A. Meyer, and D. Tao, "Top-k link recommendation in social networks," in *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pp. 389–398, 2015.

[77] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay, "Accurately interpreting clickthrough data as implicit feedback," in *SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, August 15-19, 2005*, pp. 154–161, 2005.

[78] N. N. Liu and Y. Qiang, "Eigenrank: A ranking-oriented approach to collaborative filtering," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pp. 83–90, 2008.

[79] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, "1-norm support vector machines," in *Advances in Neural Information Processing Systems 16 (NIPS)*, pp. 49–56, 2003.

[80] Y.-J. Lee and O. L. Mangasarian, "Ssvm: A smooth support vector machine for classification," tech. rep., Computational Optimization and Applications, 1999.

[81] C. Chen and O. L. Mangasarian, "Smoothing methods for convex inequalities and linear complementarity problems," *Mathematical Programming*, vol. 71, pp. 51–69, 1993.

[82] G. Orr, "Momentum and learning rate adaptation," 1999.

[83] H. Yin, Y. Sun, B. Cui, Z. Hu, and L. Chen, "Lcars: A location-content-aware recommender system," in *KDD*, 2013.

[84] J. Bao, Y. Zheng, and M. F. Mokbel, "Location-based and preference-aware recommendation using sparse geo-social networking data," in *SIGSPATIAL*, pp. 199–208, 2012.

[85] Y. Zheng and X. Xie, "Learning travel recommendations from user-generated gps traces," in *TIST*, 2011.

[86] W. Tobler, "A computer movie simulating urban growth in the detroit region," *Economic Geography*, pp. 234–240, 1970.

[87] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *KDD*, 2011.

[88] G. Huiji and L. Huan, "Location-based social network data repository," 2014.

[89] Z. Cheng, J. Caverlee, K. Lee, and D. Z. Sui, "Exploring millions of footprints in location sharing services," in *ICWSM*, 2011.

[90] G. Li, S. Chen, J. Feng, K. lee Tan, and W. syan Li, "Efficient location-aware influence maximization," in *SIGMOD*, 2014.

[91] B. Liu and H. Xiong, "Point-of-interest recommendation in location based social networks with topic and location awareness," in *SDM*, 2013.

[92] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *ICDM*, pp. 613–622, 2006.

[93] J. Wang and Y. Zhang, "Utilizing marginal net utility for recommendation in e-commerce," in *SIGIR*, 2011.

[94] H. Yu, N. Rao, and I. S. Dhillon, "Temporal regularized matrix factorization," *CoRR*, vol. abs/1509.08333, 2015.

[95] R. S. Tsay, *Analysis of Financial Time Series*. Wiley-Interscience, 2005.

[96] E. Richard, S. Gaïffas, and N. Vayatis, "Link prediction in graphs with autoregressive features," *J. M. L. R.*, 2014.

[97] Z. Wang, P. Chakraborty, S. R. Mekaru, J. S. Brownstein, and J. Ye, "Dynamic poisson autoregression for influenza-like-illness case count prediction," in *KDD*, pp. 1285–1294, 2015.

[98] H. Li, R. Lin, R. Hong, and Y. Ge, "Generative models for mining latent aspects and their ratings from short reviews," in *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pp. 241–250, 2015.