Title

Exploiting Geographical and Temporal Patterns for Personalized POI Recommendation

Author

Kannar, Kiran

Publication Date

2018

UNIVERSITY OF CALIFORNIA SAN DIEGO

Exploiting Geographical and Temporal Patterns for Personalized POI Recommendation

A Thesis submitted in partial satisfaction of the requirements
for the degree
Master of Science

in

Computer Science

by

Kiran Kannar

Committee in charge:

Professor Julian John McAuley, Chair
Professor Sanjoy Dasgupta
Professor Sicun Gao

2018

The thesis of Kiran Kannar is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California San Diego

2018

DEDICATION

I dedicate this thesis to my mother, who is every reason for who I am today.

EPIGRAPH

Happiness can be found

in the darkest of times,

if one only remembers to

turn on the light.

— J.K. Rowling

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I thank my advisor, Prof. Julian McAuley for his invaluable advice and guidance through the course of my degree. I am immensely grateful to him for providing me an opportunity to do research, to explore my interests in human behavioral modeling and to immensely learn from him. The exposure I received to a variety of interesting problems is invaluable. I would like to also thank my thesis committee members, Prof. Sanjoy Dasgupta and Prof. Sicun Gao for their time to review my research and for their valuable comments.

I am grateful to all of the amazing opportunities I have had to learn and grow at UCSD, through classes, teaching, and events. I especially thank Prof. Yannis Papakonstantinou, Prof. Ramamohan Paturi, Prof. Andrew Kahng and Prof. Miles Jones for giving me opportunities to be teaching assistants in their classes. I am thankful to each one of them for remodeling my thinking about teaching and providing me a great deal of flexibility and involvement in the courses. I will cherish the discussions I held, the homework and exams I prepared or helped to prepare, and fondly remember the long hours spent on grading effectively.

I thank Nadyne Nawar, without whose constant encouragement and advice, my UCSD experience would have been fairly different. I am also greatly thankful to the entire CSE MS advising team for their support and guidance throughout my two years at UCSD.

I thank my fellow UCSD classmate and my friend since we were three, Janet Johnson, for being a pillar of support and strength. Thank you for listening and offering much-needed help and advice as I navigated graduate school. I also thank my fellow UCSD classmate Swathi Hoysala for motivating me to finish up this thesis!

Lastly, I thank my parents and family who provided me the inspiration and motivation to pursue my career in the field of computer science. I couldn't have gotten this far without them.

VITA

2014	B.E. in Computer Science and Engineering , R.V. College of Engineering (RVCE), Bangalore
2014	Software Engineering Intern, PayPal India, Bangalore
2014-2016	Software Engineer, Oracle India, Bangalore
2017	Software Engineering Intern, Salesforce.com, San Francisco
2018	M.S. in Computer Science, University of California San Diego (UCSD)

ABSTRACT OF THE THESIS

Exploiting Geographical and Temporal Patterns for Personalized POI Recommendation

by

Kiran Kannar

Master of Science in Computer Science

University of California San Diego, 2018

Professor Julian John McAuley, Chair

Human behavior presents various temporal and geographical patterns that can be used to model user preferences and enhance prediction in the task of POI recommendation. The task of personalized next point-of-interest (POI) recommendation in Location-based Social Networks (LBSNs) has been studied extensively in recent years. The challenge of modeling the interactions of the user, current POI, and next POI presents the need to incorporate sequential dynamics using methodologies like Markov chains and Metric embedding. Existing methods capture these interactions by decomposing them into pairwise relationships. In this thesis, we apply Personalized Ranking Metric Embedding (PRME) for personalized next POI recommendation based on user's check-in history in various LBSNs like Foursquare and Gowalla. We introduce methods to incorporate spatial and temporal patterns in this metric embedding model. Experiments conducted on the above publicly available datasets indicate superior results demonstrating the effectiveness of incorporating these behavioral patterns in the task of recommendation.

Chapter 1

Introduction

With rapid advancements in technology, a growing number of mobile applications can geo-tag information like photos and posts. Location-based Social Applications like Foursquare and Yelp are increasingly used to check-in to locations the user has visited. The check-in history represents not only the information about the locations the user has visited but also the intrinsic user preferences and his mobility patterns.

With the growing popularity of these applications, personalization of recommendation has become prominent. There are various approaches that look at the past user interactions for predictive modeling of POI recommendation. In this thesis, we first consider the modeling of sequential dynamics present in these user interactions. We consider the task of next Point-of-interest (POI) recommendation and employ metric embedding techniques for learning these dynamics. We seek to employ embedding spaces that can hold the meaningful relationships that exist among the objects embedded in these spaces. We use two LBSN datasets across four cities varying in their densities to interpret what the model has learned.

State-of-art techniques using matrix factorization [33] perform well in modeling pairwise ranking, along with user preferences. On the other hand, Markov chain models like Factorized Personalized Markov Chains (FPMC) [34] model the sequential dynamics (i.e user visits a location after another) by factorizing a first-order transition matrix. FPMC models this complex interaction between *(user, current location, next location)* by modeling the pair-wise relationships between *(user, next location)* and *(current location, next location)*.

The use of Euclidean distance metric in Personalized Ranking Metric Embedding [10] provides a direct representation of relationships present in these interactions; objects close to each other are similar to each other, or users often visit these nearby locations successively. These embedding methods have become popular in fields like Natural Language Processing due to the easy interpretability of the learned models.

Our work considers exploiting the use of geographical and temporal patterns present in user movement. Users visit coffee shops in the morning before they get to work. Their commute patterns to work occur consistently during the same morning and evening hours. They visit different locations based on the time of the day. The distinctive weekend patterns also provide useful information of user preferences, depending on the day of the week. Users also tend to visit nearby locations from the current location; a coffee shop close to their office, or a pub at a short travel distance from work. Locations have temporal patterns too that can be exploited; many locations have peak hours during which they are most busy. Incorporating these influences can make recommendations meaningful and relevant.

We summarize the contributions of this thesis as follows:

- We first demonstrate the superiority of Personalized Ranking Metric Embedding (PRME) over other sequential models like Factorized Personalized Markov Chains (FPMC)[34] and pair-wise, nonsequential ranking models like Bayesian Personalized Ranking (BPR)[33].

- We introduce simple extensions to PRME by accounting for location popularities and using dual point technique to incorporate asymmetry in mobility directions.

- We then incorporate geographical and temporal patterns and investigate the performance of these models. Based on a qualitative and quantitative analysis of the obtained results, we see that the recommendations are superior in nature.

The remainder of this thesis is organized as follows: We first give a brief description of some related work in chapter 2. We describe the datasets in chapter 3. We then elaborate on the metric embedding models in chapter 4 and present the basic extensions to the model. We discuss various temporal and spatial patterns in chapter 5 and 6 and also propose methods to incorporate these influences. Chapter 7 details the experiments conducted and the results we obtained on the LBSN datasets. We conclude and present some future work in Chapter 8.

Chapter 2

Related Work

2.1 General Recommendation

One of the most popular approaches to the general item recommendation task is collaborative filtering with matrix factorization [35][20]. This technique relies on modeling user preferences by factorizing the rating matrix to user and item latent factor matrices in the same shared space. An appropriate number of latent factors (or dimensions) is chosen to be retained during the process of factorization by Singular Value Decomposition (SVD) of the rating matrix R. The optimization problem has a nice closed form solution; however, it involves computing inverses of huge matrices. Instead, Stochastic Gradient Descent (SGD) or Alternating Least Squares is often used to find the optimal configuration of parameters.

In contrast to content-based methods, collaborative filtering techniques do not rely on the explicit knowledge of item features. Rather, they model user preferences using the user and item interactions. Memory-based collaborative filtering techniques use item-item and user-user similarities to predict the rating[23]. Other collaborative filtering models like the Weighted Regularized Matrix factorization model (WR-MF) [18] [30] add weights to the loss function to increase the impact of positive feedback along with regularization to control overfitting.

These techniques also called as point-wise methods, however, assume that all of the data is known; Any missing or unobserved data is considered as negative. The feedback can be explicit interactions like item ratings given by the user. Implicit feedback relies on user clicks, views and purchase history. In contrast to point-wise methods, pair-wise methods like Bayesian Personalized Ranking (BPR) [33] make the assumption that users prefer the observed interactions over the unobserved interactions, and optimize the pair-wise ranking of pairs of positive and negative samples. These methods are preferred over Matrix Factorization (MF) as SVD overfits as the number of dimensions increases.

BPR assumes that users prefer items they have interacted with over all items that they have not. BPR optimizes for correctly ranking such pairs to provide each user with a personalized total ranking $>_u$ of all items. The optimization criterion maximizes the posterior probability $P(\Theta| >_u)$ over model parameters Θ. The ordering of every pair (i, j) for each user u is independent of all other pairs. The individual probability for this particular pair preference is $P(i >_u j|\Theta)$. The loss function is approximated by the differentiable sigmoid $\sigma(x)$, which gives BPR-OPT, the optimization criterion:

$$\text{BPR-OPT} = \sum_{(u,i,j) \in D_s} \ln \sigma(\hat{x_{uij}}) - \lambda ||\Theta||^2 \tag{2.1}$$

where D_s is the training data of triples (u, i, j), with i and j as the positive and negative items respectively. $\hat{x_{uij}} = x_{ui} - x_{uj}$ can be estimated using the individual components x_{ui} obtained through standard collaborative filtering techniques illustrated above. The optimization problem uses SGD along with random uniform sampling of triples with replacement for maximization. BPR is directly optimized for personalized ranking.

There are variants of BPR proposed for various scenarios: Factorized Personalized Markov Chains (FPMC) [34] proposes S-BPR, which adapts pairwise ranking on the sequential check-in history of the user. He et.al. propose Visual-BPR (VBPR) [16] incorporate visual signals into BPR-MF to consider the visual appearance of the items for recommendation. Pan et.al. propose GBPR [31] to extend the BPR-MF algorithm for incorporating group preferences by using a group pairwise assumption. Other kinds of loss functions like WARP loss [42] have also been used with some success.

2.2 Sequential Dynamics

One of the most common techniques to handle sequences of item interactions is Markov Chains (MC)[36]. Factorized Personalized Markov Chain (FPMC) [34] is a combination of matrix factorization and Markov chains, modeling user preferences as well as sequential dynamics. However the use of the inner product in the model does not allow sufficient generalization; the knowledge of close pairs (i, j) and (j, k) does not guarantee the closeness of the third pair (i,j). One approach to this problem is learning a suitable distance metric that follows this triangular inequality assumption, through metric learning to rank methods [28][49]. Another modeling technique, Collaborative Metric Learning (CML)[17] uses metric learning to rank to learn a distance metric that pulls positive items closer and pushes negative or irrelevant items further away from the user using margin loss.

Some of the recent work in this area has been in the use of metric embeddings for the task of recommendation. Chen et.al. proposed Logistic Markov Embeddings (LME) [7] for playlist prediction, using the distances between songs in the metric space. Pairwise Ranking Metric Embedding (PRME) [10] models use pairwise BPR loss to optimize the maximum a posterior.

Another approach to POI recommendation presents a unified space for user preferences and sequential dynamics. The GME-S model [44] modells user preferences using time-decayed latent embeddings and uses the information from successive POI check-in pairs to weigh each transition, thereby incorporating sequential dynamics. Another recent research work, TransRec

[15] models the third-order interactions using a unified space; the method adopts techniques from knowledge graph embeddings [6] with a user as a ”relation” capturing transitions from one item to another. The authors demonstrate the power of TransRec over models like FPMC and PRME, especially in sparse datasets, with fewer parameters in comparison. However, in this thesis, we use separate spaces to model different components of user interactions and therefore use PRME as a base model. Other knowledge graph embeddings using translation-based methods [41],[22],[43] have also been known to work well.

One of the most common ways to identify sessions in sequences is to consider the inactivity time as the gap between two sessions. There has been some research on identifying sessions based on temporal rhythms in user activities; identifying valleys in inter-activity density distributions provides a better inactivity threshold than choosing an arbitrary threshold that does not consider goal-directed user behavior [12]. Another useful macroscopic behavioral pattern in consumption sequences is repeat consumption and abandonment; modeling these patterns [5][1] is subject to availability of data on repeat consumption.

Other kinds of recommender systems [37] adopt exploration-exploitation technique through Markov Decision Processes (MDP)[4], the downside of which is the requirement of a deployed recommender system or a simulation of interactions in the recommender system. Similar to Markov Chains, MDP-based recommender systems look at transitions between states, but in the scope of an optimal policy that give long-term benefit.

2.3 Temporal Dynamics

Temporal dynamics can be crucial in understanding the evolving user preferences, in terms of short-term and long-term dynamics. Some popular techniques use decaying time weights [9] or exploitation of temporal signals in different ways in real-world systems like Netflix [19]. Sequential dynamics does not automatically include time. Sequence presents an order and modeling sequential dynamics focuses on learning the intrinsic relationships present in the order. However, human mobility is heavily influenced by the time of the activity. As we shall see in chapter 5, the incorporation of these signals presents opportunities to learn various aspects of temporal influence. Chapter 5 also goes through some of the related work in more detail.

2.4 Geographical Patterns

Geographical location and distance heavily influence human mobility patterns; users tend to have a lot of check-in activity in their home region [8]. Additionally, the distance between successive check-in pairs is usually small. Some of the recent work range from modeling the influence of distance directly for POI recommendation [47] to identifying out-of-town regions that a user is likely to visit [32]. The GTAG model (Geographical-Temporal influences aware Graph) [51] encodes session nodes that connect POIs and users, along with edges between neighboring POIs. The authors also propose a preference propagation algorithm to provide recommendations within 6 propagation steps. A more detailed discussion of the past work is given in chapter 6, where we discuss modeling some of the geographical patterns observed in the datasets.

Other influences include the use of categorical information in improving prediction accuracy, either by explicitly mining user preferences over categories [24] or by a two step-process of category prediction and further item recommendation [13]. In [3], the hierarchy of category (primary and secondary category levels) is used to construct each user's weighted personal preferences across these category levels. The knowledge of friendship or social circles can also improve prediction accuracies [47][8]. A detailed survey of many of these techniques in terms of incorporating various influences, methodologies, and tasks is performed in [53]. An experimental evaluation of these techniques is illustrated in [25].

Chapter 3

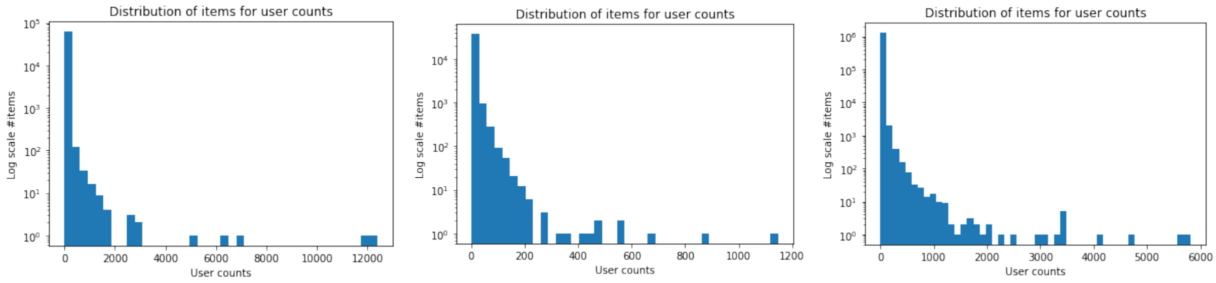
Datasets

We use variants of Foursquare and Gowalla datasets collected by researchers in the past. In this chapter, we briefly describe the general characteristics of the datasets. An elaborate discussion on specific patterns is made in subsequent chapters that elaborate on exploiting these patterns for improving personalized recommendation. Foursquare is a local mobile application that provides personalized recommendations of locations in the vicinity of a user based on their check-in history, preferences and their interaction with the application. Gowalla is another location-based social network that allows users to check into locations they visited. Gowalla was acquired by Facebook and was shut down in 2012.

3.1 Foursquare

We use the Foursquare check-in dataset obtained by authors of [46]. The dataset [1] is a collection of check-ins over a period of 10 months (approximately April 2012 to February 2013). Since Foursquare check-ins are personal and therefore not visible to public API, the authors gather the check-in data through Twitter shares. The authors filter out users who have not checked-in at least once a week, and items which cannot be associated with a Foursquare category. The details

[1] Foursquare: `https://sites.google.com/site/yangdingqi/home/foursquare-dataset`



(a) FourSquare (TKY)

(b) FourSquare (NYC)

(c) Gowalla (SNAP)

Figure 3.1: Distribution of items with respect to user check-in counts in Foursquare and Gowalla datasets (log-scale)

of the dataset collection process are outlined in the authors' previous work[45].

The dataset statistics are outlined in table 3.1. We see that while all users have a high minimum threshold of 100 check-ins due to the stringent filtering in data collection process, the items have a long-tail distribution. This is confirmed by the distributions of number of users who have visited the locations in figures 3.1a, 3.1b. The #user count and #venue count fields represent the number of locations visited by a user and the number of users who visited a location and respectively.

Table 3.1: Summary of datasets

Measure	**Foursquare (TKY)**	**Foursquare (NYC)**	**Gowalla (SNAP)**
#Check-ins	573,703	227,428	6,442,892
#Users	2,293	1,083	107,092
#Venues	61,858	38,333	1,280,969
#User counts (max, min)	(2991, 100)	(2697,100)	(2175,1)
#Venue counts (max, min)	(12372, 1)	(1147, 1)	(5811,1)
Avg. #items /user	250.20	209.99	60.16
Avg. #users/item	9.27	5.93	5.03

We further filter the datasets to have at least 5 check-ins from each user and 5 check-ins for each location. We then have 2,293 users and 15,177 venues in the FourSquare (TKY) dataset, and 1,083 users and 9,989 venues in the FourSquare (NYC) dataset.

We also look at the top 10 categories with the most check-ins in the Foursquare data. The data reported in table 3.2 is on the thresholded data, and the category listed is the lowest level category in the category hierarchy, and not the primary category description available directly in the dataset.[2]

Table 3.2: Top 10 categories with most check-ins in Foursquare datasets.

Foursquare (TKY)	**Foursquare (NYC)**
Train Station	Home (private)
Metro Station	Office
Noodle House	Metro Station
Convenience Store	Coffee Shop
Japanese Restaurant	Bar
Electronics Store	Train Station
Grocery Store	Gym
Shopping Mall	Park
Coffee Shop	Neighborhood
Cafe	Grocery Store

3.2 Gowalla

We use the Gowalla dataset [3] obtained by authors of [8]. The public check-in data is collected over the period of February 2009 to October 2010. The dataset is also long-tailed as seen in figure 3.1c. We create two subsets specific to two cities, San Francisco (SF) and San Diego (SD), filtered on the geographical coordinates. The dataset statistics are outlined in table 3.3.We further filter out users and items with the same threshold of 5 as above. The Gowalla SNAP dataset does not have categorical information. The subsets have much less user interaction with items as indicated by the low average statistics.

[2] `https://developer.foursquare.com/docs/resources/categories`

[3] http://snap.stanford.edu/data/loc-gowalla.html

Table 3.3: Summary of Gowalla sub-datasets in two cities - San Francisco and San Diego

Measure	**Gowalla (SF)**	**Gowalla (SD)**
#Check-ins	185,120	31,989
#Users	4,355	1,218
#Venues	8,720	2,500
#User counts (max, min)	(1925, 1)	(1618, 1)
#Venue counts (max, min, avg)	(5662, 1)	(969, 1)
Avg. #items /user	28.35	18.59
Avg. #users/item	10.93	5.26

The behavioural rhythms of users and locations in these datasets is vastly different to each other. We explain this in detail in subsequent chapters, as we illustrate incorporating these dynamics into the task of POI recommendation.

Chapter 4

Models for Personalized POI Recommendation

In this section, we present some preliminaries about the models we implemented, the optimization criterion and the parameter learning techniques used. Our model extensions to incorporate temporal and geographical patterns are inspired by the PRME model [10], which is also explained in this section. Henceforth, we use locations, POI, and items interchangeably.

4.1 Problem Formulation

Given a set of users, $\mathcal{U}$ who interact with a set of locations,$\mathcal{L}$, we model the next POI prediction problem as follows:
Every user $u \in \mathcal{U}$ marks a subset of locations as visited i.e. $L^u = (L^u_1, L^u_1, ...L^u_{|L^u|})$. These locations are ordered according to the time they are visited. Given these sequences of all users, we need to generate recommendations for each user by predicting what location each user is most likely to go to. All notations used in this thesis are summarized in table 4.1

Table 4.1: Notations

Notation	Explanation
$\mathcal{U}, \mathcal{L}$	Set of users and locations
u, i, j	user $u \in \mathcal{U}$, locations $i, j \in \mathcal{L}$
L^u	check-in history sequence of user u
K	Number of dimensions in embedding spaces
P	User preference space
S	Sequential space
T	Temporal space
G	Geographical space
A	Weekday space
B	Weekend space
C	User time space
X_*	Embedding in respective metric space P, S, T, G, A, B, U
$X_e(i), X_x(i)$	Entry and exit vectors of location i in dual point sequential space
$D^*(a,b)$	Squared Euclidean distances between points a and b in respective spaces
$d(a,b)$	Geographical distance between two locations a and b
α	Weight of user preference space towards transition probability
ν	Weight of sequential space
γ_{jt}	Weight of temporal space parameterized by location j and time t
γ_{ij}	Weight of geographic space parameterized by successive check-in location pair (i, j)
W_{LT}	Location-Time graph
W_{LL}	Location-Location graph
u_t	user embedding at time t in user time space C
β_j	Popularity of location j
τ	Time threshold within which two check-ins as sequential j
$\Delta_t(i, j)$	Time difference between the check-in times of two consecutive check-ins at locations i, j
$\hat{R}$	Regions formed by DBSCAN of location data
r_a	a specific region a

4.2 Modeling sequential information using Markov Chains

We treat user history as an ordered sequence. We can further divide this sequence into multiple sessions based on various criteria including time gap between two successive check-ins, and the distance between the two locations. While the user preferences are learned over the entire user history, we can assume that the transition from one location to another does not depend upon a check-in long past in time. This property of the transition depending only on the recent past is referred to as the *Markov property*[36].

In a stochastic model that describes a sequence, the transition probability to a location depends only on present check-in location i.e. the future predictions can be based entirely on present state, independent of the past. We can therefore model the transition from location l_t to l_{t+1} for a user u as the transition probability $P(l_{t+1}|l_t)$ instead of $P(l_{t+1}|L^u)$, where $L^u = \{l_1, l_2, \cdots, l_t\}$. This is a first order Markov chain where,

$$P(l_{t+1}|L^u) = P(l_{t+1}|l_t) \tag{4.1}$$

Higher order Markov chains can be modeled with a memory of the order m to produce m-order Markov chains. A memory of order m indicates that the future prediction relies on past m locations i.e.

$$P(l_{t+1}|L^u) = P(l_{t+1}|l_t, l_{t-1}, \cdots, l_{t-m+1}) \tag{4.2}$$

Modeling higher order Markov chains is hard due to the state space explosion by the curse of dimensionality. We do not have sufficient data to learn the transition probabilities of all of these states. However, as we shall see, modeling pairwise transitions performs well in the task of ranking a list of items.

4.2.1 Factorized Personalized Markov Chains

Factorized Personalized Markov Chain (FPMC) algorithm [34] uses Tucker Decomposition [38] to decompose user-item interactions into a combination of pairwise interactions i.e. a user triple (u, i, j) is factorized to interactions between the user and items (u, i) and (u, j), and between the current item and previous item (i, j).

FPMC employs BPR for optimization in item recommendation, which makes (u, i) interaction redundant for a quadruplet (u, i, j, k) where the positive and negative samples are (u, i, j)

and (u,i,k) respectively. Subsequently, the simplified model only considers the interactions between the user and next item and the interaction between the items in successive check-in pair. Factorization of the transition matrix reduces the number of parameters from $|L|^2$ to $2K|L|$, where K is the number of latent dimensions. Full parameterization is hard to achieve as the transition data is highly sparse.

For each user u who has interacted with item i, the probability of transition to item j is

$$P(j|u,i) \propto \langle \vec{X}_u, \vec{X}_j \rangle + \langle \vec{Y}_i, \vec{Y}_j \rangle \tag{4.3}$$

where $\vec{X}_u$ is the user latent factor representation, and $\vec{X}_j, \vec{Y}_i, \vec{Y}_j$ are item latent factor representations respectively. We observe that the transition is dependent on two inner products: the user-item interaction representing user's general preferences (MF), and the item-item interaction representing the sequential dynamics (MC).

However, the Achilles' heel of FPMC is in the use of the inner product, which does not satisfy triangular inequality. If two locations i and j are nearby, and j is also close to location k, we expect i and k to be nearby as well. The inner product does not guarantee this property, limiting the performance of FPMC.

4.3 Personalized Ranking Metric Embedding

Due to the limitations of the inner product, one methodology tries to find good metrics using learning to rank models. Another methodology uses metric embeddings in the Euclidean space. The distance in the Euclidean space can be directly interpreted as the closeness or similarity between pairs of points in the space.

4.3.1 Metric Embeddings and Partition Functions

A finite metric space is a set of points with non-zero distances between any two distinct points in the space satisfying the triangular inequality. A metric space (X,d) is defined as:

X : set of points in the metric space

$d : X \times X \rightarrow \mathbb{R}^{\geq 0}$

satisfying the three properties $\forall x, y, z \in X$

(1) $d(x,y) = 0 \iff x = y$

(2) $d(x,y) = d(y,x)$ (symmetry)

(3) $d(x,y) + d(y,z) \geq d(x,z)$ (Triangular Inequality)

If we can find a metric space where we can encode all items, then we can take advantage of triangular inequality in determining the closeness of unknown pairs in data. For example, assuming x and y are nearby and y and z are nearby, we can ascertain that x and z are guaranteed to be nearby. The inner product does not satisfy this property. Therefore, we use a distance metric that allows us to take advantage of the triangular inequality property, thereby alleviating data sparsity.

The metric space is a vector space over $\mathbb{R}^d$ where d is the number of dimensions of the space. The most common embedding metric is the l_2-metric that creates the Euclidean space. The distance metric $d(x,y) = \|X(x) - X(y)\|_2$ i.e the l_2 norm

Some of the recent metric embedding work is rooted in using the Boltzmann distribution to approximate to the most likely distribution of the given data. The Boltzmann distribution is expressed in the form of probability distribution in statistical mechanics. The distribution minimizes the free energy of states of a system, thereby maximizing their probability.

It is expressed as:

$$p_i = \frac{e^{-\beta E_j}}{\sum_{k=1}^{M} e^{-\beta E_k}} \tag{4.4}$$

where M is the number of states a system can be in, E_j is the energy of the state j, and β is the inverse temperature (i.e $\beta = 1/T$). The denominator is the canonical partition function, which provides the probabilities of the various states in a particular canonical ensemble. The partition function as seen above also acts as the normalizing constant. It can be shown that through this distribution, we use an energy model where the total energy of the system is a constant [27].

Feng et.al [10] developed PRME, a novel pairwise metric embedding algorithm that represents POIs in a low dimensional Euclidean latent space and ranks potential next POIs using the Euclidean distance metric. The distance between two POIs in the metric space directly measures the strength of their transition (or sequential relation) i.e the transition probability.

PRME uses two latent spaces for modeling the components of user interactions. The first space, P, learns the user preferences by modeling the user-item interactions in this metric space. The second space S models the item-item interactions i.e. the sequential dynamics in this metric space. The embeddings in the respective spaces are K_P and K_S-dimensional vectors $X_P(u), X_P(i), X_S(i)$. For simplicity, we use the notation K to represent the dimensions in both spaces.

The transition probabilities in each space follow Boltzmann distribution. The transition probability between a pair of items (i, j) can be expressed as,

$$P(j|i) = \frac{e^{-\|X_S(i)-X_S(j)\|_2^2}}{Z(i)} \tag{4.5}$$

$$\text{where} \quad Z(i) = \sum_{k=1}^{|L|} e^{-\|X_S(i)-X_S(k)\|_2^2} \tag{4.6}$$

where, Z(i) is the normalization term, also called as the partition function. This is expensive to compute consider the number of locations in a LBSN is usually in the order of millions. The model obviates this computation by employing pair-wise ranking $>_{u,i}$ to compare two locations j and k based on the transition probabilities from a location i. Given an observed transition pair (i, j), we assume that the user prefers this location over any k in an unobserved (i,k) transition pair. Therefore for a user u, we can denote this preference order as, $j >_{u,i} k$. Therefore, we convert the prediction problem into a ranking task to generate a total order of all items through pairwise preferences.

We can also express this as follows: The location j is preferred over k for a user u, if

$$P(j >_{u,i} k) = P(j|u,i) - P(k|u,i) > 0 \tag{4.7}$$

As the normalizing constant is the same for both pairs (i, j) and (i,k) and we transformed the optimization to a ranking problem, it is sufficient to just use the Euclidean distances in the comparisons, instead of the complete exponent calculations. Given a user u and current POI i, the transition from i to a location j is modelled using the transition probability $P(j|u,i)$ as follows:

$$P(j|u,i) \propto -\left(\alpha\, D^P_{u,j} + (1-\alpha)\, D^S_{i,j}\right) \tag{4.8}$$

where

$$D^S_{a,b} = \|X_S(a) - X_S(b)\|_2^2 \tag{4.9}$$

$$D^P_{a,b} = \|X_P(a) - X_P(b)\|_2^2 \tag{4.10}$$

The model hyper-parameter α controls the contribution of the personal space towards making the prediction. Since l_2 norm is a distance metric, the model easily generalizes to unobserved pairs.

Finally, the authors use a time-gap threshold, τ, to consider pairs as sequential. The sequential space is considered only if the time difference between the two check-ins $\Delta_t(i, j)$ is less than or equal to the threshold. This is modeled as follows:

$$P(j|u,i) \propto \begin{cases} -\,D^P_{u,j}, & \text{for } \Delta_t(i,j) > \tau \\ -\Big(\alpha\, D^P_{u,j} + (1-\alpha)\, D^S_{i,j}\Big), & \text{otherwise} \end{cases} \tag{4.11}$$

The authors also introduce a variation of PRME incorporating geographical influence. This model, referred to as *PRME-G*, uses the distance between the two locations $d(i,j)$ as a weighting factor.

$$P(j|u,i) \propto \begin{cases} -\,D^P_{u,j}, & \text{for } \Delta_t(i,j) > \tau \\ -w_{ij}\Big(\alpha\, D^P_{u,j} + (1-\alpha)\, D^S_{i,j}\Big), & \text{otherwise} \end{cases} \tag{4.12}$$

$$w_{ij} = (1 + d(i,j))^{0.25} \tag{4.13}$$

4.3.2 Considering directionality

Inherent in user's sequential patterns, is a sense of directionality that is not considered by the metric embedding-based model. We can observe transitions like (Gym, Bar) or (Gym, Juice bar), but the reverse transitions are highly unlikely. However, we do observe bidirectional pairs like (home, office) and (station, platform). Inspired by the use of the dual-point model (LME) [7] in playlist prediction, we incorporate a dual point embedding space for sequential dynamics. The user preference space is inherently bidirectional as we learn to model user-location compatibility through the latent features.

Therefore, each location i in the sequential space has a pair of embeddings - an entry vector $X^S_e(i)$ and an exit vector $X^S_x(i)$ If a user moves from a location i to a location j, the model should learn to place the entry vector $X^S_e(j)$ of location j close to the exit vector $X^S_x(i)$ of location i. Thus, the transition probability is still modeled similar to PRME but differs in the calculation of distance in the sequential space. We refer to this model as *PRME-DP*:

$$P(j|u,i) \propto \begin{cases} -\,D^P_{u,j}, & \text{for } \Delta_t(i,j) > \tau \\ -\Big(\alpha\, D^P_{u,j} + (1-\alpha)\, D^S_{i,j}\Big), & \text{otherwise} \end{cases} \tag{4.14}$$

$$\text{where, } D^S_{i,j} = \|X^S_x(i) - X^S_e(j)\|^2_2 \tag{4.15}$$

4.3.3 Considering popularity of locations

We can boost the model by separately capturing the overall location popularity through an additional model parameter β_j for each location j. We refer to this model as *PRME-pop*. The transition probability is,

$$P(j|u,i) \propto \begin{cases} \beta_j -\, D^P_{u,j}, & \text{for } \Delta_t(i,j) > \tau \\ \beta_j - \Big(\alpha\, D^P_{u,j} + (1-\alpha)\, D^S_{i,j}\Big), & \text{otherwise} \end{cases} \tag{4.16}$$

4.4 Optimization and Learning

4.4.1 Optimization

FPMC optimizes the pair-wise tensor decomposition using Sequential Bayesian Personalized Ranking (SBPR) [34] optimization criterion. PRME uses the same optimization technique. In this section, we briefly describe the process.

The optimization process relies on the key assumption that if a user *u* transitions from a location *i* to a location *j*, we can rank the location *j* higher than all locations he has not visited.

$$P(j|u,i) > P(k|u,i)\ \forall\, k \in L - L_u$$

We optimize the PRME model parameters by using the following maximum a posterior

(MAP):

$$\hat{\Theta} = \underset{\Theta}{\text{argmax}} \prod_{u \in U} \prod_{i \in L_u} \prod_{j \in L_u} \prod_{k \notin L_u} P(j >_{u,i} k) P(\Theta) \tag{4.17}$$

For differentiability, we consider the sigmoid of $P(j >_{u,i} k)$ which is generally the Heaviside non-differentiable function. Therefore,

$$\hat{\Theta} = \underset{\Theta}{\text{argmin}} \sum_{u \in U} \sum_{i \in L_u} \sum_{j \in L_u} \sum_{k \notin L_u} \log \sigma(p_{u,i,j} - p_{u,i,k}) - \Omega(\Theta) \tag{4.18}$$

where $\Omega(\Theta)$ is the standard $\mathcal{L}_2$ regularizer, and $p_{u,i,j}$ is the simplified notation for $P(j|u,i)$

4.4.2 Learning Algorithm

We can use stochastic gradient descent to learn the parameters. We use bootstrap sampling to generate the quadruplet $< u, i, j, k >$. We first sample a user u and then an observed transition pair (i, j) for the user. We then sample the negative item k accordingly.

The gradient update at every iteration is given as:

$$\Theta = \Theta + \eta \left(\sigma(p_{u,i,k} - p_{u,i,j}) \frac{\partial(p_{u,i,j} - p_{u,i,k})}{\partial \theta} - 2\lambda\Theta \right) \tag{4.19}$$

where, η is the learning rate for the update step. The complexity of the algorithm is $O(K|N||H|)$ where N is the number of iterations required over K latent dimensions, given the number of observed listening data-points as H.

4.4.3 PRME-DP

We add an additional regularizer $\Phi(\Theta)$ to the optimization criterion in order to constrain the distance between the entry and exit vectors of locations. Ideally, the entry and exit vectors should be nearby, as they represent one item.

$$\Phi(\Theta) = \sum_{i \in L} D^{S}_{i,i} \tag{4.20}$$

Chapter 5

Incorporating Temporal Influences

Our activities through the day are influenced by time. On a weekday, we spend the nights sleeping at home; in the morning, we leave to work, but not before getting a coffee at a coffee shop or cafe close to work. At the end of the day, we hit the gym, and then maybe a pub or a restaurant, before calling it a night. This activity time-line varies with each person, the time of the day, and the day of the week. Some of our activities are also influenced by seasonal influences - ice skating during winter, or beach days during hot summers. Many of our activities are in fact also bursts of activities in short time gaps that together can be considered as a session. Therefore incorporating the influence of time in our models is important. In this section, we describe some of the temporal dynamics; we primarily concentrate on the Foursquare datasets while incorporating the temporal influences.

5.1 Related Work

One of the earliest works with incorporating temporal influences in collaborative filtering was by Koren [19]; he introduces the concept of local and global drifts of a user and proposes time-aware latent factor models which incorporate time evolving user and item biases, as well as drifting user preferences. The modeling choices involve piecewise (or bins) time periods or decay-based weighting. It also allows modeling sudden spikes or transient effects.

Most research on incorporating temporal influences divides time of the day into hour-based slots. Some models like CARS [2] use the number of check-ins in each time slot as a weighting feature. Other methods use the user-item-time cube to model interaction of a user with an item at a particular time period. Some of these models use smoothing using related or similar time slots [50] to solve the data sparsity problem. However, they are limited to using temporal patterns as weighting mechanism.

Two key properties inherent in user check-in patterns through the day are non-uniformness and consecutiveness. The number of check-ins varies through the day and check-in preferences at consecutive slots are more similar than across large time-gaps. The LRT model [11] incorporates these properties using time evolving latent user features (one per time slot) along with weighted regularization of consecutive slot user features. A similar incorporation of time-aware user preferences is adopted in [29] to extend the LME model [7] with long-term temporal dynamics in song listening preferences of users. The user preference metric space uses the distance between the song and specific user-time embedding.

Another key property with respect to locations is periodicity [8]. Users tend to visit locations of a particular category consistently at the same time. A user visits pubs at night, and gyms during the evening. The STELLAR model [55] uses time encoding to incorporate month, weekday/weekend and hour temporal factors. The model uses similar tensor factorization as FPMC, using an additional space for location-time interaction.

In [52], a probabilistic framework using kernel density estimate is employed to incorporate the temporal context. The model uses the correlation of user check-ins to similar check-ins in user's social groups as well as similar check-ins at the particular time to alleviate data sparsity caused by the use of user-time-POI cube.

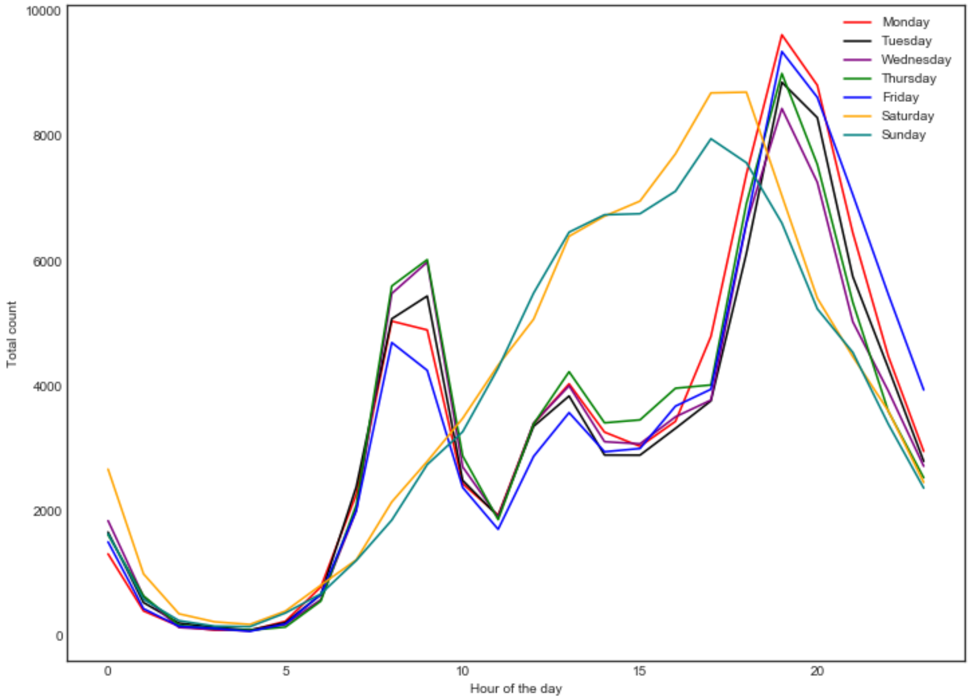


Figure 5.1: Distribution of check-ins over the time of day in FourSquare (TKY) dataset

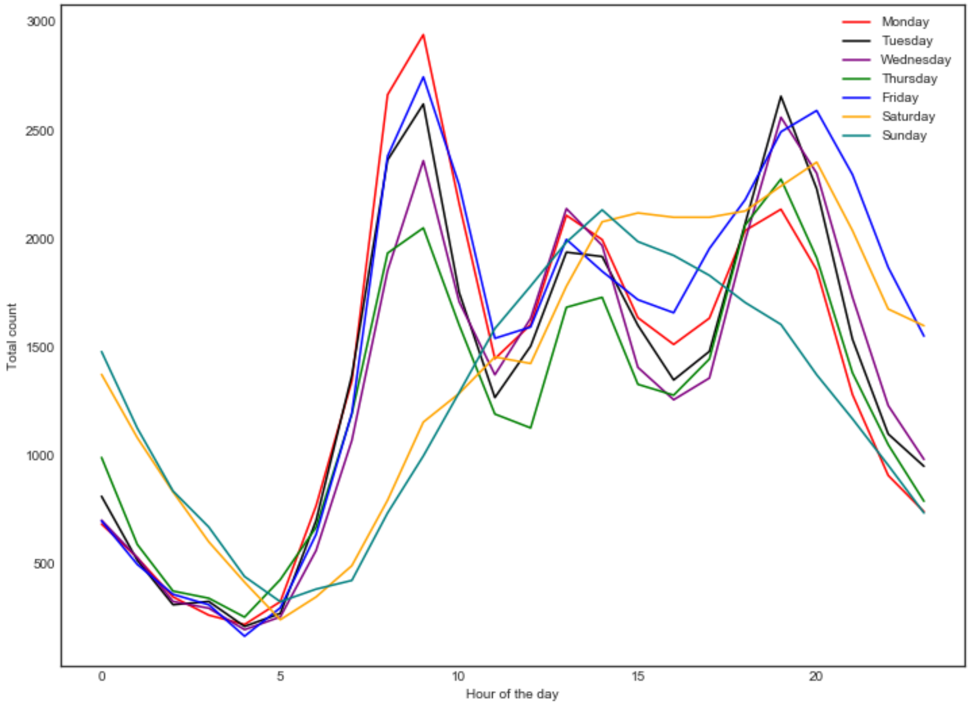


Figure 5.2: Distribution of check-ins over the time of day in FourSquare (NYC) dataset

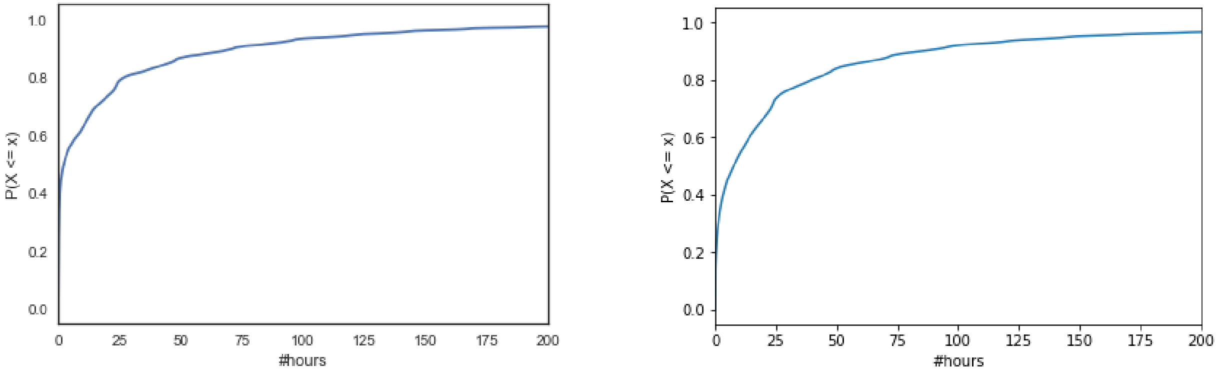
5.2 Temporal rhythms and dynamics

User's daily activities vary across the time of the day. We plot the distribution of check-ins across the time of the day in figures 5.1,5.2 for the Foursquare Tokyo (TKY) and New York City (NYC) datasets. In both datasets, we first observe largely three peaks at approximately 8-9 am, 12 pm and 6-8 pm covering the morning, evening and lunch hours. The large volume of check-ins at train stations as evidenced by table3.2 indicate significant commute during the morning and evening peaks. These periods of time are also peak times for coffee shops (mornings) and restaurants (evenings).

However, we see considerably different patterns on weekdays and weekends. On weekends, the curve is vastly different, indicating a late start in the day, but a gradual rise in user activity through the day. Tokyo users seem to have an early end to the day on weekends, in contrast to New York City users who have much more activities through late Friday and Saturday evenings.

However, the behavior on Sunday abruptly changes to early fall in activity, preparing for the week ahead. In both datasets, the highest peak of activities occurs on Mondays, although it is in the evening in Tokyo, while it is in the morning for New York. The NYC dataset predominantly has large activities during weekday morning revealing extensive commute check-ins at the rush hour. New York city users also have a much higher rise in activities at the start of the day; however, the patterns vary based on the day much more than Tokyo users who have considerably similar rise and falls in activity through both weekdays and weekends. This could be attributed however to the differences in the size of the two datasets, with the Tokyo dataset being denser.

We now look at the time-gaps between successive check-ins to estimate a suitable threshold for our base PRME model and to also further develop our analysis. We first look at the cumulative frequencies over the time differences between successive check-ins. From figure 5.3, we see that roughly 75% of the successive check-ins happen within a day. However, we also see a significant



(a) FourSquare (TKY)

(b) FourSquare (NYC)

Figure 5.3: CDF of time difference between successive check-ins in Foursquare datasets

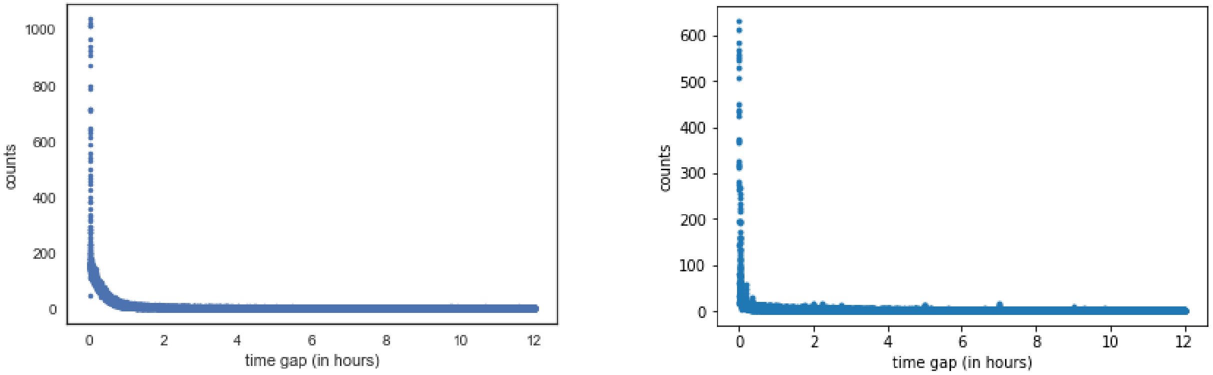
number of check-in pairs with large time gaps spanning more than 200 hours. Clearly, these check-ins should not be considered as sequential dynamics.

We now seek to obtain a reasonable threshold τ that retains as many check-in pairs as possible. From our analysis, we saw that 12 hours is a good threshold for Tokyo, and 16 hours is a good threshold for New York City, retaining approximately two-thirds of the data. The percentage of user pairs ignored at various thresholds is included in table 5.1.

Table 5.1: Percentage of user pairs ignored at various thresholds in Foursquare data

τ	**Foursquare (TKY)** (%)	**Foursquare (NYC)** (%)
1	56.4	72.87
3	48.16	61.49
6	42.41	52.98
8	40.18	49.47
10	37.73	46.11
12	34.60	43.13
16	29.92	37.67
24	22.65	28.00

From figures 5.4a, 5.4b, we see that a high number of check-in pairs have an extremely small time difference of less than 15 minutes. A lot of these pairs can be attributed to the daily commute through train/subway in both cities. Since the dataset was filtered out of the suspicious burst of check-ins [46], these check-ins can be considered relevant.



(a) FourSquare (TKY)

(b) FourSquare (NYC)

Figure 5.4: Counts of check-ins pairs vs. time gap

Lastly, we note that the temporal behavioral patterns in Gowalla are harder to analyze given the sparsity of the datasets. For reference, we include figures 5.5,5.6.

5.3 Modeling temporal influences

In this section, our goal is to exploit these temporal patterns to improve POI recommendation. We primarily exploit the periodic nature of temporal dynamics by splitting time into 24 slots, one for each hour. We now have pentuples (u,i,j,k,t) including the negative sample k. Our objective is for the model to learn that user u prefer j over k at time t. For reasons similar to the FPMC and PRME embedding techniques, we perform canonical decomposition of this interaction into pair-wise interactions.

5.3.1 Incorporating temporal patterns of locations

We first consider the location-time (LT) interaction. Many POIs have certain peak hours in the day; for example, coffee shops tend to be busy in the mornings, pubs in the nights, and gyms during evenings. Of course, each location has its individual temporal characteristics. Therefore we create a new temporal space T where all time slots and locations are points embedded in this

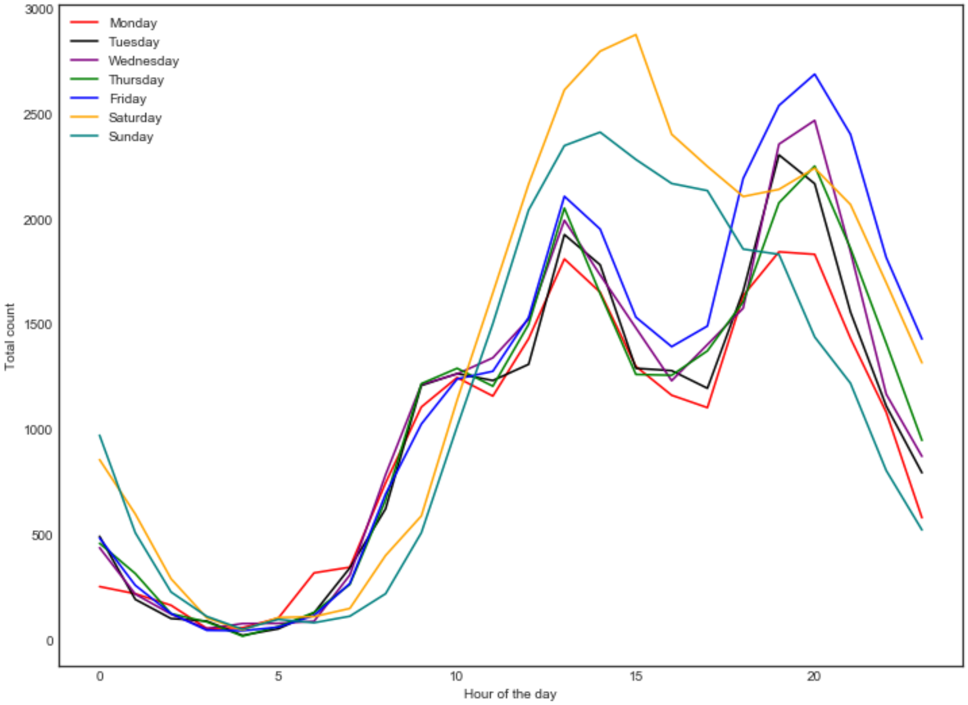


Figure 5.5: Distribution of check-ins over the time of day in Gowalla (SF) dataset

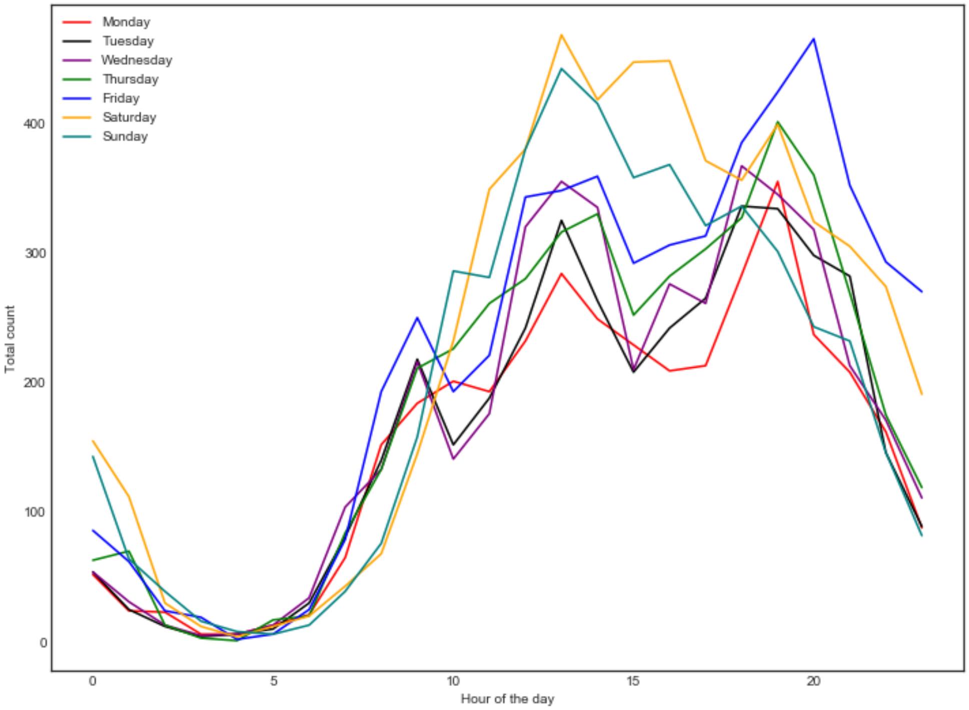


Figure 5.6: Distribution of check-ins over the time of day in Gowalla (SD) dataset

space. We incorporate the distance in this space, $D^T_{j,t}$ into our final transition probability.

$$p_{u,i,j,t} \propto -\left(\alpha\, D^P_{u,j} + \nu\, D^S_{i,j} + \gamma_{jt}\, D^T_{j,t} \right) \tag{5.1}$$

where, α, ν are the weights of user preference and sequential spaces.

- We first set $\gamma_{jt} = \gamma$, where $\gamma \in \{0.001, 0.01, 0.1, 1\}$ This is a global weight common to all location-time pairs. We refer to this model as *LocTime(Static)*
- We construct a Location-Time graph (LTG), W_{LT} where each edge between a location and time has a weight equal to the observed number of check-ins at the location j at the given time t. The weighting parameter γ_{jt} for each location pair is therefore the normalized number of check-ins (j,t).

$$\gamma_{jt} = \frac{w_{jt}}{\sum\limits_{(a,b)\in W_{LT}} w_{ab}} \tag{5.2}$$

  where W_{LT} is the set of all location-time pairs. We could also choose the normalize using the degree of each node, i.e. $Z = \sum\limits_{(j,b)\in G_{LT}} w_{jb}$ instead of the global count. Our results did not yield a significant increase in performance, and therefore we choose to retain the simpler normalizer. We refer to this model as *LocTime (LTG)*.

5.3.2 Incorporating type of day influences

From the temporal distributions in figures 5.1, 5.2, we see clear differences between weekday and weekend activities. Therefore we split the LocTime space T into weekday space A and weekend space B. The decision on the choice of space depends on whether the time of next check-in t falls on a weekday or weekend. Unlike the sequence pairs, this distance doesn't depend on the time threshold and therefore is retained irrespective of the sequential preference.

$$p_{u,i,j,t} \propto -\left(\alpha\, D^P_{u,j} + \nu\, D^S_{i,j} + \gamma_{jt}\, D^T_{j,t}\right) \tag{5.3}$$

where

$$D^T_{j,t} = \begin{cases} D^A_{j,t}, & \text{if weekday} \\ D^B_{j,j}, & \text{if weekend} \end{cases} \tag{5.4}$$

We refer to this model as *DayType*

5.3.3 Incorporating temporal influences in user preferences

Most locations show unimodal distributions with one high-volume time in a day (notable exceptions include gyms). Unlike locations, as observed in the temporal rhythms, users have multiple peaks in a day. Modeling user temporal preferences is therefore also relevant to improving recommendation. Thus, users need individual embeddings for each time slot.

One way to incorporate these influences is to add a space similar to the location temporal space C. We split (u,i,j,t) into $(u,j),(i,j)$ and (u,t) interactions. We create embeddings $X^C_t(u)$ for each user u and time slot t.

$$p_{u,i,j,t} \propto -\left(\ \alpha\, D^P_{u,j} + \nu\, D^S_{i,j} + \gamma\, D^U_{u,t}\right) \tag{5.5}$$

However, for a sample (u,i,j,k,t), the user-time influence component cancels out across the positive and negative samples, reducing this version to PRME. It would also be erroneous since the model tries to bring a user closer to multiple times in a day based on his preference, but these times are not necessarily similar/closer, considering all users and general time similarities. Having two times for positive and negative samples will fail for the same reason, aggravated by more data sparsity.

We, therefore, consider user preferences dependent on time and model user embeddings u_t for each hour t in a day. We emphasize that we do not create 24 new embedding spaces, rather 24 points per user. The hypothesis is that the user who prefers the same type of locations at multiple times will have the corresponding user time embeddings close to each other, in turn, close to the preferred location.

$$p_{u,i,j,t} \propto -\left(\alpha\, D^P_{u,j,t} + \beta\, D^S_{i,j} \right) \tag{5.6}$$

$$\text{where } D^P_{u,j,t} = \|X^P(u_t) - X^P(j)\|_2^2 \tag{5.7}$$

We refer to this model as *UserTime*

5.3.4 Complexity analysis

The new temporal spaces in *LocTime* and *DayType* increase space complexity in the order of $O(|L| + |T|)$, where $|T| = 24$. The number of parameters, therefore, continues to grow in the order of the number of items. In the *UserTime* model, each user has 24 embeddings, instead of one embedding in the preference space. As we shall see, this increase is problematic for sparse datasets.

Chapter 6

Incorporating geographical influence

The average user spends most of his time in a few locations like home and work and visits locations in the vicinity of these locations. In the case where a user travels to a location far away from his usual check-in region, the mobility of the user after this transition is within the vicinity of this new location i.e. the travel itinerary. Therefore, modeling geographical influences is crucial to the task of POI recommendation. In this section, we look at the geographical dynamics in the cities across the datasets and propose metric embedding models to incorporate its influence.

6.1 Related Work

Human mobility exhibits periodic patterns of movement which are geographically limited to few location areas[8]. The role of geography plays an important role in user interaction with locations. Many of our daily activities are concentrated around a few representative locations like ”home” and ”work”, the latent states of which can be modeled using Gaussian distributions.

In [50], [47], the authors notice that the user successive check-ins pairs follow power-law distributions, the parameters of which can be obtained by linear regression. The studies in [21] incorporate the geographical influence of neighboring POIs of current POI and consider ranking based on Ordered Pairwise Classification (OWPC) criterion, instead of BPR.

In [26], the authors model the geographical neighborhood of POIs at an instance and region level and use weighted regularized matrix factorization to learn the geographical and user preferences together. In [14], the authors study the fusing of sequential behavior with latent behavioral patterns with the use of softmax function over the low-rank tensor factorization model (FPMC).

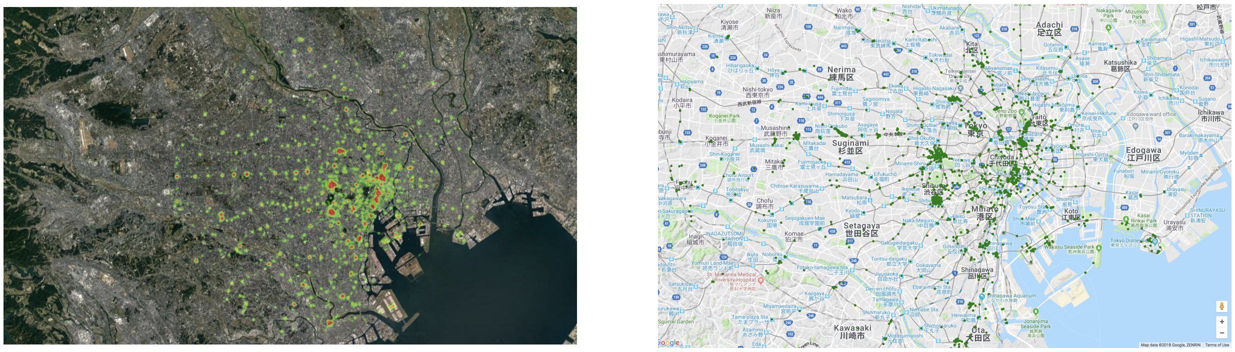
The geographical influence can also be modeled by a two-dimensional check-in distribution identified by the geographical coordinates of each location instead of merely looking at the distance between successive check-in pairs. In [52], personalized two-dimensional check-in probability density is estimated for each user based on kernel density estimation with a standard two-dimensional normal kernel.

Another approach to consider spatial influences is the use of generative models. In SPORE [40] model, the sequential and user preferences are fused in one latent space and the authors propose asymmetric Locality Sensity Hashing (ALSH) to speed up online top-k recommendations. In LCARS [48], a statistical mixture model is used to not only mine user preferences but to also learn the local preference distribution over the latent topics obtained through the generative process. On the other hand, LSARS [39] incorporates sentiment from reviews to generate crowd preferences to model region-wise personal preferences.

Geo-Teaser [54] makes an additional assumption in pairwise ranking that users prefer unvisited neighboring POIs over unvisited non-neighboring POIs. The model, therefore, incorporates the geographical influences through hierarchical pairwise ranking. Lastly, one of the more recent works [32] moves away from POI recommendation to region recommendation problem to predict a likely out-of-town region that a user is likely to visit. An approximate line-sweeping based search algorithm is used to find the next optimal region a user is likely to visit

6.2 Geographical dynamics

We first visualize our locations using their GPS (latitude, longitude) coordinates. We use gmaps python API for visualization [1]. From figures 6.1,6.2a, 6.2b, 6.2c, we observe that all of these cities have hubs with large user check-ins. We also see check-ins aligned with commute paths - road or rail routes. This is clearly seen in the FourSquare Tokyo dataset with a large number of train station check-ins.



(a) Heatmap of check-ins

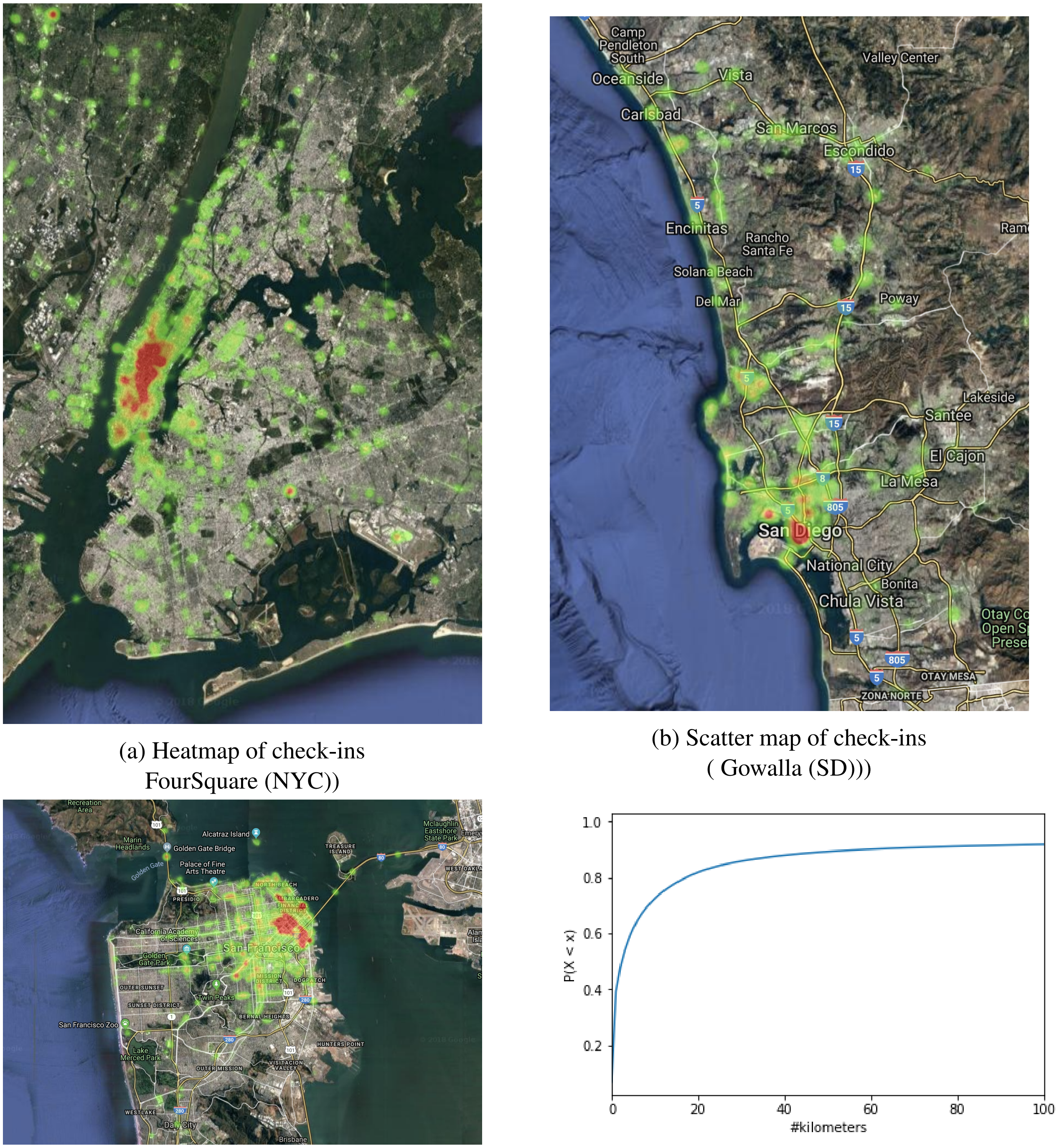
(b) Scatter map of check-ins

Figure 6.1: Check-ins of Foursquare Tokyo dataset)

From the San Francisco data, we see that while there is a major hub with peak activity count, there are several ”mini” hubs (shown in red in the heat map) far away from the major hub. This indicates that there are ”*regions*” with popular location(s) spread across the city, each with its own characteristics. For example, it is evident that Castro street and Mission district are hubs with popular locations. From the Tokyo dataset, the hubs are also well-connected by rail and road transport, a unique characteristic serving the purpose.

From figure 6.2d, we see that most of the check-ins are within short distances; Roughly 80% of the successive check-in pairs are within 20 kilometers of each other.

[1] `https://jupyter-gmaps.readthedocs.io/en/latest/`



(a) Heatmap of check-ins FourSquare (NYC))

(b) Scatter map of check-ins (Gowalla (SD)))

(c) Scatter map of check-ins Gowalla (SF)

(d) CDF of successive check-in distances (Gowalla_Snap)

Figure 6.2: Heatmaps of check-ins of various datasets in (a)-(c), and CDF of successive check-in distances in (d)

6.3 Incorporating geographical influences

We, therefore, want to incorporate these geographical influences to improve the accuracy of POI recommendation. In this section, we discuss our proposed models.

Our incorporation of geographical influence either revolves around modeling these geographical influence through a metric space, where the distance corresponds to the similarity between items or uses the explicit geographical distance between two locations $d(i,j)$ as weighting factors. One measure to calculate this distance is Haversine distance. It measures the great-circle distance between two points on earth given their longitudes and latitudes, assuming a spherical approximation of earth.

6.3.1 Incorporating inter check-in geographical distance

Let the Haversine distance between two successive check-ins (i,j) be denoted by $d(i,j)$. We add a component for this distance directly weighted per location.

$$p_{u,i,j} \propto -\left(\alpha\, D^P_{u,j} + \nu\, D^S_{i,j}\right) + \gamma_{ij}\,\frac{1}{d(i,j)^{0.25}} \tag{6.1}$$

The best exponent obtained for the inverse distance measure was 0.25.

- We first set $\gamma_{ij} = \gamma$, where $\gamma \in \{0.001, 0.01, 0.1, 1\}$ This is a global weight common to all item pairs. We refer to this model as *Geo-inv*

- We construct the Location-location transition graph (L2TG), W_{LL} where each directed edge between two locations has a weight equal to the observed number of transitions between the two locations. The weighting parameter γ_{ij} for each location pair is therefore the normalized number of check-in transitions (i,j).

$$\gamma_{ij} = \frac{w_{ij}}{\sum\limits_{(a,b)\in W_{LL}} w_{ab}} \tag{6.2}$$

where W_{LL} is the set of all location-location transitions. We refer to this model as *Geo-inv(L2TG)*.

6.3.2 Incorporating regional characteristics

Users who visit a location tend to also visit nearby locations. Therefore, we can assume that locations in a neighborhood share some intrinsic characteristics particular to the neighborhood. For example, the city of San Francisco Bay Area has multiple sub-regions each vastly different from the other. We, therefore, cluster all points into M regions using Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The two parameters, ε - radius of the cluster, and *minP* - the minimum number of points in the cluster need to be fine-tuned suitably. We visualized a subsample of the clustered data to arrive at the parameter values $\varepsilon = 0.007, minP = 10$, resulting in 38 clusters of the San Francisco dataset. In the case of FourSquare (Tokyo), we set $\varepsilon = 0.005, minP = 10$ and obtained 54 clusters. The outliers that aren't clustered to any region do not receive any regional influence.

We add new geographical space G for geographical influence of the region on the location. Given the check-in pair (i, j), where $i \in r_a$ and $j \in r_b$, and $r_a, r_b \in \hat{R}, |\hat{R}| = M$, we consider the distance between the region's embedding and the location's embedding in the geographical space.

$$p_{u,i,j} \propto -\Big(\alpha\, D^{P}_{u,j} + \nu D^{S}_{i,j} + \gamma_{ij}\, D^{G}_{r_b,j} \Big) \tag{6.3}$$

We refer to this model as *Geo-DBSCAN*

6.3.3 Complexity analysis

Explicit use of geographical distances does not add additional space complexity unless the location-location transition graph is used. The space complexity is in the order of the number of distinct check-in pairs (or the edges of the graph) The new geographical space in *Geo-DBSCAN* increases space complexity in the order of $O(|L|+M)$ where M is the number of clusters.

6.3.4 A spatial-temporal fused model

We finally combine the best performing spatial and temporal models from above:

$$p_{u,i,j} \propto -\left(\alpha\, D^{P}_{u,j} + \nu D^{S}_{i,j} + \gamma_{jt}\, D^{T}_{j,t} \gamma_{ij}\, D^{G}_{r_b,j} \right) \tag{6.4}$$

We refer to this model as *GeoTemp*. We analyze the performance of these models in the next chapter.

Chapter 7

Experiments and Results

7.1 Experimental setup

7.1.1 Dataset preparation

For all our experiments, we used Foursquare and Gowalla datasets, a detailed description of which is given in chapter 3. In this section, we use the following city abbreviations - TKY for Tokyo, NYC for New York City, SF for San Francisco, and SD for San Diego.

For each user, we consider their last check-in or successive check-in pair as test data, depending on the model. Correspondingly, the penultimate check-in or successive check-in pair is considered as the validation data. The negative samples are chosen uniformly from the unobserved items or item pairs.

7.1.2 Baselines methods

We use the following baselines to analyze the performance of our metric embedding models.

Most Popular: This is a naive baseline but should perform better than a trivial random predictor.

The frequencies of user interactions are tabulated for every location, and the model always predicts the most popular location for all users. There is no personalization involved.

BPR-MF: Bayesian Personalized Ranking (BPR-MF)[33] is a state-of-art pairwise ranking based recommendation model using matrix factorization. However, it does not consider the sequential interactions of the user.

FPMC: Factorized Personalized Markov Chains (FPMC)[34] considers the sequential dynamics of user interaction, and decomposes these interactions into a combination of pairwise interactions between the user and item, and item and item.

PRME: Personalized Ranking Metric Embedding (PRME)[10] differs from FPMC in the use of metric spaces and distance metrics over inner products.

STELLAR (hour): We will consider a variant of STELLAR - Spatial-Temporal Latent Ranking for Successive Point-of-Interest Recommendation [55]. We do not encode various levels of time, but simply use the hour of the day.

PRME-G: the geographical variant of PRME will serve as a baseline for incorporating geographical influence.

7.1.3 Evaluation metrics

We consider three widely used evaluation metrics:

Area Under the ROC Curve (AUC): AUC measures how much we correctly rank a uniformly drawn positive sample higher than a uniformly drawn negative sample.

$$AUC = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|u|} \sum_{j \in \mathcal{T}_u} \frac{1}{|\mathcal{L} \setminus \mathcal{T}_u|} \sum_{k \in \mathcal{L} \setminus \mathcal{T}_u} \mathbf{1}(r_{u,j} > r_{u,k})$$

where $\mathcal{T}_u$ is the hold-out testing set for each user u, $r_{u,i}$ is the rank of the item i for the user u. $\mathbf{1}(x)$ is an indicator function that returns 1 if the expression x evaluates to true. The AUC score does not depend on the position of ordering; all incorrect orderings at any position in the list affect equally.

Hit Rate@50:

Hit Rate considers whether the item was ranked within the top 50 positions across all items.

$$HR@50 = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbf{1}(r_{u,j} \leq 50) \tag{7.1}$$

Mean Reciprocal Rank (MRR):

$$HR@50 = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbf{1}(r_{u,j} \leq 50) \frac{1}{r_{u,j}} \tag{7.2}$$

An item is considered relevant if it is ranked within the top 50 items. MRR considers the position of the ranking. If the item does not appear in the top 50 ranks, we consider its contribution as 0.

7.1.4 Implementation Details

We used Stochastic Gradient Descent (SGD), Adam and Adagrad Optimizers within the TensorFlow [1] framework with Python. Adagrad optimizer performed well both in terms of speed and the minimum found. The regularization parameters were chosen from the set {1e-1,1e-2, 1e-3,1e-4,5e-1,1e-5,5e-5} and learning rate of 1 and 1e-4 for Adagrad and Adam respectively. The regularization parameter for negative samples was set to 1/10th of the best value used from the above set. The best values of regularization were chosen by grid search on validation data. We tried various values of α for PRME and chose the values that gave the best results. The best value of α obtained in PRME was used for subsequent models.

The embeddings were initialized using a random normal initializer with mean=0, standard deviation=0.1. The number of latent dimensions, K was chosen to 10 for simplicity.

For models with temporal and geographical components, the best mixture coefficients were usually either the mixture coefficients obtained in PRME or the value 1. We performed a

[1] https://www.tensorflow.org/

grid search on a combination of values of hyper-parameters. The cluster hyper-parameters were chosen by visualization and spread of locations across all clusters. With DBSCAN, we find an optimal configuration of the cluster radius, minimum number of points and number of outliers it produces. The code link is given below. [2]

7.2 Experimental Results

7.2.1 Quantitative analysis

Tables 7.1, 7.2 present the performance of various models, without including temporal and geographical influences.

Table 7.1: Pair-wise ranking results in various models for FourSquare datasets (improvements with respect to PRME)

Model	*FourSquare (TKY)*			*FourSquare (NYC)*		
	AUC	Hit Rate	MRR	AUC	Hit Rate	MRR
Most Popular	0.7784	0.1893	0.0747	0.6552	0.0757	0.0259
BPR-MF	0.8633	0.2817	0.1201	0.9013	0.4423	0.2522
FPMC	0.8752	0.2957	0.1007	0.9039	0.4377	0.2600
PRME	0.9422	0.4924	0.2224	0.9146	0.5383	0.3768
PRME-pop	0.9415	0.5233	0.2416	0.9219	0.5503	0.3837
Improvement	-0.07%	6.27%	8.63%	0.79%	2.23%	1.83%
PRME-DP	0.9453	0.5177	0.2397	0.9205	0.5410	0.3754
Improvement	0.32%	5.13%	7.79%	0.64%	0.50%	-0.37%

The results are compared for the number of latent dimensions, K = 10. *Most Popular* performs better than the trivial predictor by obtaining an AUC higher than 0.5. BPR-MF performs better than this simple baseline, indicating the strong necessity to model user preferences. However, across all datasets, we see that modeling just the user preferences through matrix factorization is clearly not enough. Modeling sequential dynamics provides a significant boost to all three metrics as evidenced by the performance of FPMC and PRME models.

[2] `https://github.com/KannarKK/LocRec.git`

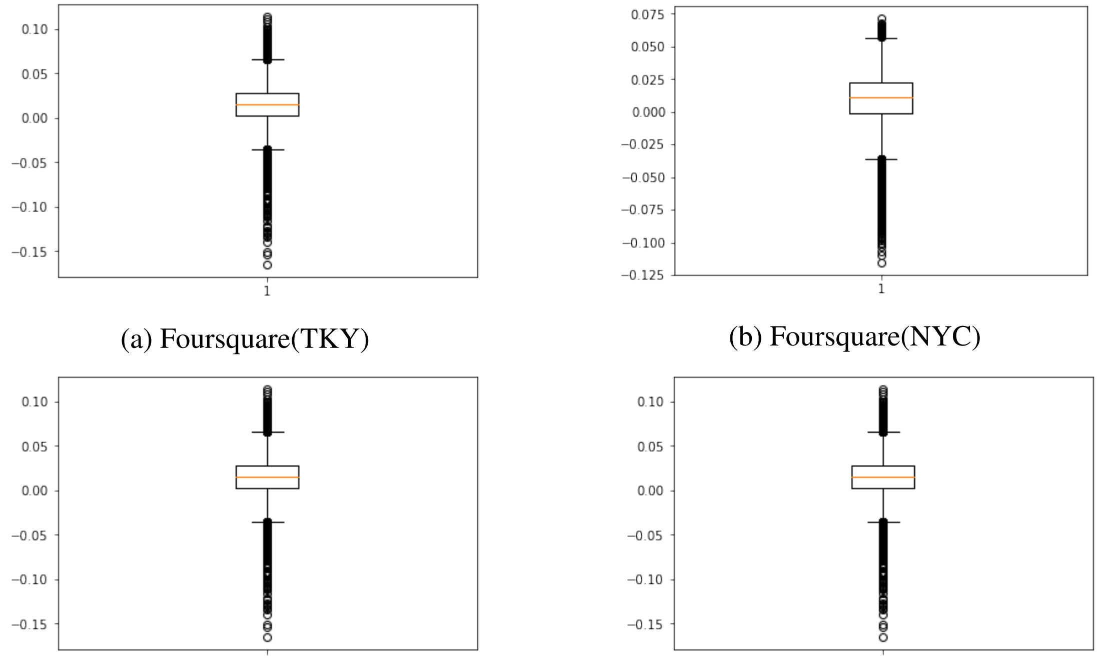
Table 7.2: Pair-wise ranking results in various models for Gowalla datasets (improvements with respect to PRME)

Model	*Gowalla (SF))*			*Gowalla (SD)*		
	AUC	Hit Rate	MRR	AUC	Hit Rate	MRR
Most Popular	0.7763	0.2704	0.1724	0.6694	0.2945	0.1806
BPR-MF	0.8619	0.3194	0.1034	0.8632	0.4650	0.1873
FPMC	0.8709	0.3066	0.1113	0.8600	0.4255	0.1797
PRME	0.8890	0.3260	0.1020	0.8788	0.5138	0.2120
PRME-pop	0.8919	0.4059	0.187	0.876	0.5223	0.2515
Improvement	0.32%	24.51%	83.33%	-0.31%	1.65%	18%
PRME-DP	0.8949	0.3273	0.1007	0.8702	0.5128	0.2276
Improvement	0.66%	1.55%	0.60%	-0.97%	-0.19%	7.36%

We note that the performance of PRME is higher than that of FPMC. The use of a distance metric over inner product allows us to exploit the triangular inequality which provides generalizability and boosts the performance. The performance of PRME on dense Foursquare datasets is much stronger than that on sparse Gowalla datasets, indicating that PRME works better on dense datasets. The best value of α in the Foursquare and Gowalla datasets was 0.4 and 0.5 respectively.

The improvement in the performance of metric embedding model with dual point vectors in the sequential space supports our initial hypothesis that location mobility is directional to an extent, and modeling this direction can boost prediction. The most sparse dataset, Gowalla (SD) however does not indicate improvements in hit rate with the dual point model, strongly necessitating the need for more data with the increased number of parameters. The number of parameters in the sequential space is doubled with two vectors per point in the space. The use of explicit popularity model parameter also improves model performance to an extent, at the cost of adding $|L|$ additional model parameters. Figure 7.1 presents the distributions of these additional parameters.

From tables 7.3, 7.4, we see that incorporating temporal influences can significantly boost prediction. While the incorporation of time with weighing all location-time pairs equally



(a) Foursquare(TKY)

(b) Foursquare(NYC)

(c) Gowalla (SF)

(d) (Gowalla (SD)

Figure 7.1: Box plot diagrams depicting distributions of popularity parameter of all items across various datasets

Table 7.3: Pair-wise ranking results in temporal models for FourSquare datasets (improvements with respect to PRME)

Model	*FourSquare (TKY)*			*FourSquare (NYC)*		
	AUC	Hit Rate	MRR	AUC	Hit Rate	MRR
STELLAR	0.9446	0.4841	0.2065	0.9208	0.5485	0.2970
PRME	0.9422	0.4924	0.2224	0.9146	0.5383	0.3768
LocTime(static)	0.9456	0.5138	0.2312	0.9317	0.5651	0.3603
Improvement	0.55%	2.22%	2.99%	1.87%	4.98%	-4.38%
LocTime(LTG)	0.9504	0.5364	0.2343	0.9241	0.5621	0.3778
Improvement	0.55%	2.22%	2.99%	1.04%	4.42%	0.26%
DayType	0.9480	0.5268	0.2414	0.9233	0.5568	0.3953
Improvement	0.55%	2.22%	2.99%	0.9512%	3.44%	4.91%
UserTime	0.9324	0.4779	0.2150	0.9092	0.5411	0.3646
Improvement	-1%	-1.45%	-3.33%	-0.59%	0.53%	-3.24%

improved the performance, we see that the adaptive weights according to the normalized number of check-ins is more useful. The split of the temporal space to weekday and weekend also

Table 7.4: Pair-wise ranking results in temporal models for Gowalla datasets (improvements with respect to PRME)

Model	*Gowalla (SF)*			*Gowalla (SD)*		
	AUC	Hit Rate	MRR	AUC	Hit Rate	MRR
STELLAR	0.8963	0.3787	0.1230	0.8549	0.4617	0.1983
PRME	0.889	0.3260	0.1020	0.8788	0.5138	0.2120
LocTime(static)	0.9063	0.4022	0.1146	0.8786	0.5329	0.2346
Improvement	1.95%	23.37%	12.35%	-0.02%	3.72%	10.66%
LocTime(LTG)	0.8874	0.3344	0.1001	0.8732	0.5213	0.2231
Improvement	-0.18%	2.57%	1.86%	-0.64%	1.46%	5.23%
DayType	0.8878	0.3331	0.1024	0.8723	0.5043	0.2275
Improvement	-0.13%	2.18%	0.39%	-0.74%	-1.85%	7.31%
UserTime	0.8417	0.3359	0.1111	0.8350	0.4362	0.2007
Improvement	-5.32%	3.004%	8.93%	-4.98%	-15.1%	-5.33%

Table 7.5: Pair-wise ranking results in geographical models for FourSquare datasets (improvement with respect to PRME-G)

Model	*FourSquare (TKY)*			*FourSquare (NYC)**		
	AUC	Hit Rate	MRR	AUC	Hit Rate	MRR
PRME-G	0.9380	0.5006	0.2335	0.9147	0.5512	0.392
Geo-inv	0.9432	0.5177	0.2405	0.9186	0.5393	0.3689
Improvement	0.55%	2.22%	2.99%	0.43%	-2.16%	-5.89%
Geo-L2TG	0.9446	0.5273	0.2095	0.9206	0.5448	0.3923
Improvement	0.71%	5.33%	-10%	0.64%	-1.16%	0.07 %
Geo-DBSCAN	0.9502	0.5482	0.2275	0.9175	0.5393	0.3217
Improvement	6.08%	9.51%	-2.57%	0.31%	-2.17%	-17.9%

Table 7.6: Pair-wise ranking results in geographical models for Gowalla datasets (improvement with respect to PRME-G)

Model	*Gowalla (SF))*			*Gowalla (SD)*		
	AUC	Hit Rate	MRR	AUC	Hit Rate	MRR
PRME-G	0.8712	0.3014	0.0820	0.8544	0.4766	0.2183
Geo-inv	0.8892	0.3223	0.1045	0.8630	0.4808	0.2231
Improvement	2.07%	6.93%	27.44%	1%	0.88%	2.20%
Geo-L2TG	0.8918	0.3211	0.1027	0.8700	0.4984	0.2290
Improvement	2.37%	6.54%	25.24%	1.83%	4.57%	4.9%
Geo-DBSCAN	0.8898	0.3446	0.1008	0.8714	0.5075	0.2227
Improvement	2.13%	14.33%	1.88%	1.99%	6.48%	2%

Table 7.7: Pair-wise ranking results in geographical models for Foursquare (NYC) dataset with time-gap threshold $\tau = 14$ (with improvements over PRME-G)

Model	*Foursquare (NYC))*		
	AUC	Hit Rate	MRR
PRME-G	0.9074	0.53	0.3819
Geo-inv	0.9175	0.5291	0.3738
Improvement	1.11%	-0.17%	-2.12%
Geo-L2TG	0.9171	0.5411	0.3834
Improvement	1.07%	2.09%	0.39%
Geo-DBSCAN	0.9176	0.5475	0.3759
Improvement	1.12%	3.30%	-1.57%

improved the performance to an extent; this confirms our hypothesis that the temporal influences are different on weekdays and weekends and modeling this signal can be highly useful. However, there is a slight deterioration in performance in comparison to LocTime(LTG) model, as the temporal information is now split across two spaces; some locations have similar temporal patterns regardless of the type of day, and check-ins pairs which are across the type of day are split across either of the spaces. The temporal evolution of user preferences offered little improvement in performance.

We also note that the performance degradation is the highest in the most sparse dataset - Gowalla (SD). In fact, in both of the sparse Gowalla datasets, the adaptive location-time weighting does not outperform other methods. The static weighted model *LocTime(static)* is also the only model that outperformed STELLAR in these datasets.

From table 7.5, 7.6, we see that while PRME-G incorporates geographical influence, its fused multiplicative model does not yield the best performance. Rather, the simple additive inverse of the power of distance provides much better results. Similar to the use of the normalized number of check-ins through the location-time graph, the use of the normalized number of transitions in location-location graph further improves the performance, indicating the strong need for adaptive weighting of the geographic space. Finally, the use of regions through DBSCAN provides the best performance across all of these models, indicating the need to model neighborhood influences.

We note that this method is dependent on the number and nature of clusters formed by DBSCAN.

A notable exception is the Foursquare(NYC) dataset that shows a reverse trend in geographical extensions. We note that all results in tables 7.5,7.6 were with time-gap threshold of 12 hours for simplicity. We increase the time-gap threshold to 14 hours, to accommodate more pairs and repeat our experiments, the results of which are outlined in 7.7. We see that the geographical models perform much better than the baseline and the deterioration of MRR with each model has also decreased. As observed in figure 5.3b, the two-thirds of the data can be explained in approximately 16 hours time gaps.

Finally, we fuse the temporal and geographical spaces to obtain the results in table 7.8. We do not include the user temporal preference component as we did not obtain a significant boost in performance. We see that the model performance does improve in terms of hit rate for all datasets. The MRR, however, is impacted significantly for sparse datasets, which could be due to the disproportionate increase in the number of parameters with respect to the data size.

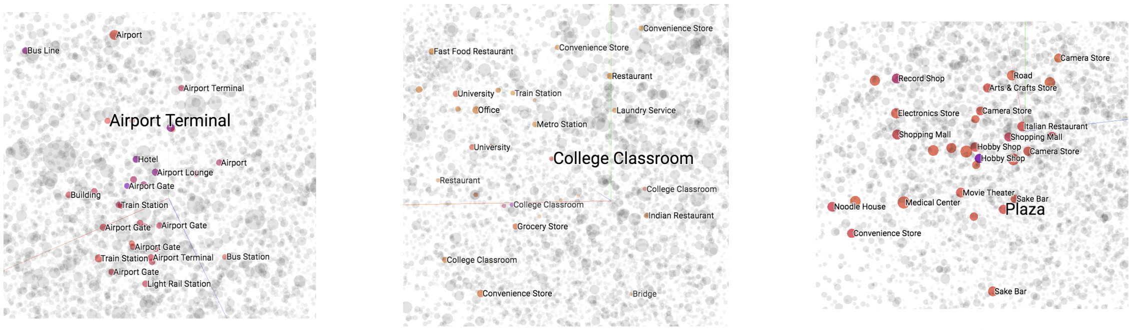
Table 7.8: Pair-wise ranking results in combined model *GeoTemp* for all datasets

Model	*FourSquare (TKY)*		
	AUC	Hit Rate	MRR
Foursquare (TKY)	0.949	0.5125	0.1880
Foursquare (NYC)	0.9259	0.5392	0.3067
Gowalla (SF)	0.9012	0.3299	0.1008
Gowalla (SD)	0.8777	0.5191	0.2286

7.2.2 Qualitative analysis

We visualize the sequential space of location-time embedding model as an example representation of points in the sequential space. From figure 7.2, we see that activities that are performed in succession are closer in the sequential space - various classrooms and buildings of a university, airport gates and terminal, and various stores of a shopping mall.

We also visualize the locations in the temporal space (see figure 7.4). For example, we see that many coffee shops are placed close to each other, indicating their peak times are similar. We can extrapolate this similarity through the triangular inequality assumption of the distance metric - two locations are close to an hour of the day implies they are closer to each other in the temporal space. In the temporal models, we employ an additional regularizer to smoothen embeddings across consecutive time slots. From the TSNE embedding (perplexity=5, learning rate =10) as shown in figure 7.3, we see that the temporal space does mirror this similarity of successive times of the day.



(a) Airport terminal

(b) College classroom

(c) Plaza

Figure 7.2: Nearest neighbours of various locations in a sample sequential space

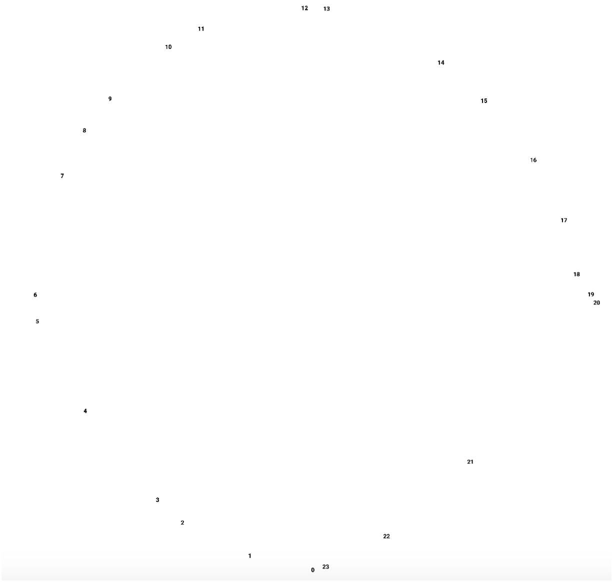
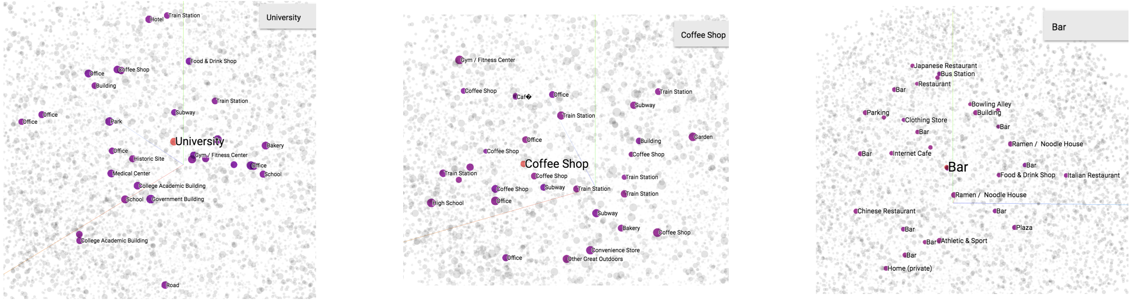


Figure 7.3: TSNE visualization of all hours of the day in a sample temporal space



(a) University

(b) Coffee Shop

(c) Bar

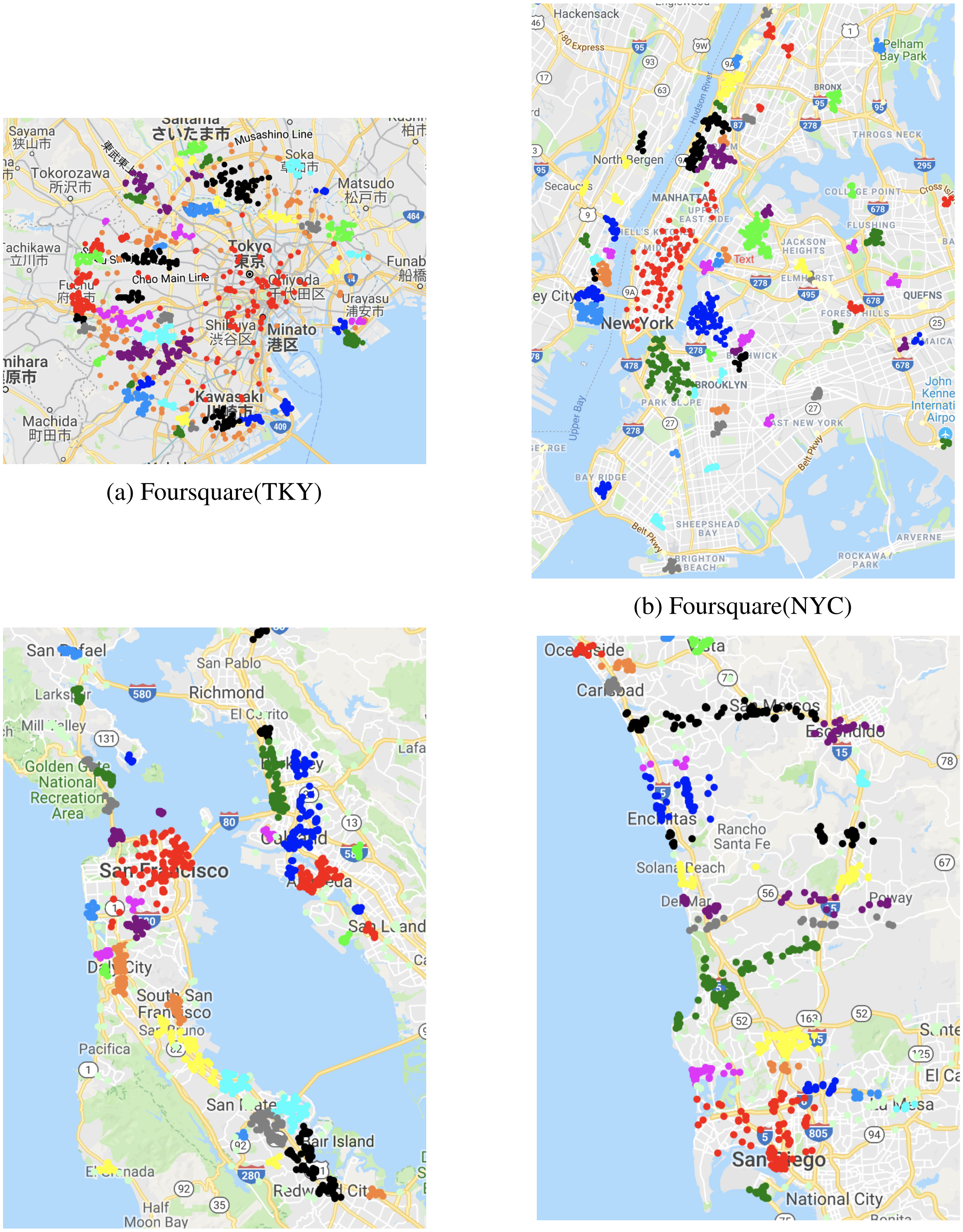
Figure 7.4: Nearest neighbours of various locations in a sample temporal space

We decided on the number of clusters based on knowledge of the cities and visualizing a subsample of the clustered data points. The subsampled cluster maps can be found in figure 7.5. We note that the color is indicative of the nearby points sampled as one cluster, and does not represent a unique cluster itself. We found DBSCAN to be susceptible to forming one large cluster especially in cities with large number of establishments in smaller regions. For example, the downtown area of SF gets modeled as one large cluster even with extremely small cluster radius, due to the sheer number of establishments present in this area.

We ideally want a reasonably good clustering, both in terms of geographical neighborhood characteristics as well successive check-in closeness. For example, locations in New Jersey close to Manhattan area, well connected through the bridges, can still be clustered together, despite not officially in the same region. Table 7.9 outlines the configurations that gave the ”best” clusters for all datasets. The outliers do not have a cluster associated with them and therefore do not receive any regional influence in the *Geo-DBSCAN* model.

Table 7.9: DBSCAN configurations and results for various datasets.

Dataset	**minP**	**ε**	**#clusters**	**# outliers**
Foursquare (TKY)	10	0.005	0.1880	518
Foursquare (NYC)	10	0.0035	0.3067	3187
Gowalla (SF)	10	0.007	0.1008	566
Gowalla (SD)	10	0.005	0.2286	335



(a) Foursquare(TKY)

(b) Foursquare(NYC)

(c) Gowalla (SF)

(d) (Gowalla (SD)

Figure 7.5: Maps of various cities depicting clusters formed by DBSCAN

Chapter 8

Conclusion

In this thesis, we first investigate various temporal and geographical patterns and rhythms in user mobility data available in location-based social networks like Foursquare and Gowalla. We then propose several ways to incorporate various geographical and temporal patterns to improve the quality of prediction in the task of next POI recommendation. Our experiments show the superior performance when these influences are accounted for. The superior performance of combined spatial-temporal metric embedding model across datasets of varying datasets indicates not only strong correlations between time and location but also that the model can capture these patterns effectively. We also demonstrate the use of modeling directionality of user mobility patterns; the improvement is most significant in dense datasets, highlighting the dependence of the increased number of parameters in PRME to the amount of data available.

One interesting avenue for future work is in improving spatial clustering. Currently, the validity of clusters from this step is done based on domain knowledge of the cities. DBSCAN does not have the option to allow maximum points in clusters, which therefore does not preclude the possibility of large clusters especially in heavy activity areas like city downtown areas. Another avenue for research is learning distance metrics for incorporating temporal and geographic influences, which currently is the Euclidean space.

Bibliography

[1] Ashton Anderson, Ravi Kumar, Andrew Tomkins, and Sergei Vassilvitskii. The dynamics of repeat consumption. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 419–430, New York, NY, USA, 2014. ACM.

[2] Linas Baltrunas and Xavier Amatriain. Towards time-dependant recommendation based on implicit feedback. In *In Workshop on context-aware recommender systems (CARS Z09*, 2009.

[3] Jie Bao, Yu Zheng, and Mohamed F. Mokbel. Location-based and preference-aware recommendation using sparse geo-social networking data. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, pages 199–208, New York, NY, USA, 2012. ACM.

[4] Richard Bellman. A markovian decision process. *Indiana Univ. Math. J.*, 6:679–684, 1957.

[5] Austin R. Benson, Ravi Kumar, and Andrew Tomkins. Modeling user consumption sequences. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 519–529, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[6] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 2787–2795, USA, 2013. Curran Associates Inc.

[7] Shuo Chen, Josh L. Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 714–722, New York, NY, USA, 2012. ACM.

[8] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. Friendship and mobility: User movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1082–1090, New York, NY, USA, 2011. ACM.

[9] Yi Ding and Xue Li. Time weight collaborative filtering. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 485–492, New York, NY, USA, 2005. ACM.

[10] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. Personalized ranking metric embedding for next new poi recommendation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 2069–2075. AAAI Press, 2015.

[11] Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. Exploring temporal effects for location recommendation on location-based social networks. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 93–100, New York, NY, USA, 2013. ACM.

[12] Aaron Halfaker, Oliver Keyes, Daniel Kluver, Jacob Thebault-Spieker, Tien Nguyen, Kenneth Shores, Anuradha Uduwage, and Morten Warncke-Wang. User session identification based on strong regularities in inter-activity time. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 410–418, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee.

[13] Jing He, Xin Li, and Lejian Liao. Category-aware next point-of-interest recommendation via listwise bayesian personalized ranking. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pages 1837–1843. AAAI Press, 2017.

[14] Jing He, Xin Li, Lejian Liao, Dandan Song, and William Cheung. Inferring a personalized next point-of-interest recommendation model with latent behavior patterns. 2016.

[15] Ruining He, Wang-Cheng Kang, and Julian McAuley. Translation-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, pages 161–169, New York, NY, USA, 2017. ACM.

[16] Ruining He and Julian McAuley. VBPR: visual bayesian personalized ranking from implicit feedback. *CoRR*, abs/1510.01784, 2015.

[17] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. Collaborative metric learning. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 193–201, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee.

[18] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.

[19] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 447–456, New York, NY, USA, 2009. ACM.

[20] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[21] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 433–442, New York, NY, USA, 2015. ACM.

[22] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2181–2187. AAAI Press, 2015.

[23] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, Jan 2003.

[24] Xin Liu, Yong Liu, Karl Aberer, and Chunyan Miao. Personalized point-of-interest recommendation by mining users' preference transition. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 733–738, New York, NY, USA, 2013. ACM.

[25] Yiding Liu, Tuan-Anh Nguyen Pham, Gao Cong, and Quan Yuan. An experimental evaluation of point-of-interest recommendation in location-based social networks. *Proc. VLDB Endow.*, 10(10):1010–1021, June 2017.

[26] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. Exploiting geographical neighborhood characteristics for location recommendation. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 739–748, New York, NY, USA, 2014. ACM.

[27] Charles H Martin. Foundations of Partition Function. `https://calculatedcontent.com/2013/11/14/foundations-the-parition-function/`, 2013. [Online; accessed 20-May-2018].

[28] Brian McFee and Gert Lanckriet. Metric learning to rank. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 775–782, USA, 2010. Omnipress.

[29] J. Moore, Shuo Chen, D. Turnbull, and T. Joachims. Taste over time: The temporal dynamics of user preferences. In *Conference of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 401–406, 2013.

[30] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 502–511, Washington, DC, USA, 2008. IEEE Computer Society.

[31] Weike Pan and Li Chen. Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 2691–2697. AAAI Press, 2013.

[32] Tuan-Anh Nguyen Pham, Xutao Li, and Gao Cong. A general model for out-of-town region recommendation. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 401–410, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee.

[33] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. *CoRR*, abs/1205.2618, 2012.

[34] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 811–820, New York, NY, USA, 2010. ACM.

[35] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. *Recommender Systems Handbook*. Springer-Verlag, 1st edition, 2010.

[36] R. Serfozo. *Basics of Applied Stochastic Processes*. Probability and Its Applications. Springer Berlin Heidelberg, 2009.

[37] Guy Shani, David Heckerman, and Ronen I. Brafman. An mdp-based recommender system. *J. Mach. Learn. Res.*, 6:1265–1295, December 2005.

[38] Ledyard R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, Sep 1966.

[39] Hao Wang, Yanmei Fu, Qinyong Wang, Hongzhi Yin, Changying Du, and Hui Xiong. A location-sentiment-aware recommender system for both home-town and out-of-town users. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1135–1143, New York, NY, USA, 2017. ACM.

[40] W. Wang, H. Yin, S. Sadiq, L. Chen, M. Xie, and X. Zhou. Spore: A sequential personalized spatial item recommender system. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 954–965, May 2016.

[41] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1112–1119. AAAI Press, 2014.

[42] Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, pages 2764–2770. AAAI Press, 2011.

[43] Han Xiao, Minlie Huang, and Xiaoyan Zhu. From one point to a manifold: Knowledge graph embedding for precise link prediction. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 1315–1321. AAAI Press, 2016.

[44] Min Xie, Hongzhi Yin, Fanjiang Xu, Hao Wang, and Xiaofang Zhou. Graph-based metric embedding for next poi recommendation. In *Proceedings of the 17th International Conference on Web Information Systems Engineering - Volume 10042*, WISE 2016, pages 207–222, New York, NY, USA, 2016. Springer-Verlag New York, Inc.

[45] Dingqi Yang, Daqing Zhang, Zhiyong Yu, and Zhu Wang. A sentiment-enhanced personalized location recommendation system. pages 119–128, 2013.

[46] Dingqi Yang, Daqing Zhang, Vincent. W. Zheng, and Zhiyong Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, 2015.

[47] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 325–334, New York, NY, USA, 2011. ACM.

[48] Hongzhi Yin, Bin Cui, Yizhou Sun, Zhiting Hu, and Ling Chen. Lcars: A spatial item recommender system. *ACM Trans. Inf. Syst.*, 32(3):11:1–11:37, July 2014.

[49] Junliang Yu, Min Gao, Wenge Rong, Yuqi Song, Qianqi Fang, and Qingyu Xiong. Make users and preferred items closer: Recommendation via distance metric learning. In Derong Liu, Shengli Xie, Yuanqing Li, Dongbin Zhao, and El-Sayed M. El-Alfy, editors, *Neural Information Processing*, pages 297–305, Cham, 2017. Springer International Publishing.

[50] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. Time-aware point-of-interest recommendation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 363–372, New York, NY, USA, 2013. ACM.

[51] Quan Yuan, Gao Cong, and Aixin Sun. Graph-based point-of-interest recommendation with geographical and temporal influences. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 659–668, New York, NY, USA, 2014. ACM.

[52] Jia-Dong Zhang and Chi-Yin Chow. Point-of-interest recommendations in location-based social networks. *SIGSPATIAL Special*, 7(3):26–33, January 2016.

[53] Shenglin Zhao, Irwin King, and Michael R. Lyu. A survey of point-of-interest recommendation in location-based social networks. *CoRR*, abs/1607.00647, 2016.

[54] Shenglin Zhao, Tong Zhao, Irwin King, and Michael R. Lyu. Geo-teaser: Geo-temporal sequential embedding rank for point-of-interest recommendation. In *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, pages 153–162, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee.

[55] Shenglin Zhao, Tong Zhao, Haiqin Yang, Michael R. Lyu, and Irwin King. Stellar: Spatial-temporal latent ranking for successive point-of-interest recommendation. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 315–321. AAAI Press, 2016.