UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Modeling User Behavior Patterns in LBSNs: A Graph Embedding Approach

Permalink

https://escholarship.org/uc/item/8hw3d1t3

Author

Xu, Weiqi

Publication Date

2019

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA SAN DIEGO

Modeling User Behavior Patterns in LBSNs: A Graph Embedding Approach

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

in

Electrical and Computer Engineering (with a specialization in Intelligent Systems, Robotics, and Control)

by

Weiqi Xu

Committee in charge:

Professor Julian McAuley, Chair
Professor Truong Quang Nguyen, Co-Chair
Professor Nikolay Atanasov

2019

Copyright
Weiqi Xu, 2019
All rights reserved.

The Thesis of Weiqi Xu as it is listed on UC San Diego Academic is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____
Co-Chair

_____
Chair

University of California San Diego

2019

DEDICATION

To my parents, whom have supported and comforted me in this journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to acknowledge Professor Julian McAuley for his support as the chair of my committee. His guidance has helped me shape my work into what it is.

I would also like to acknowledge Professor Truong Quang Nguyen as the co-chair, and Professor Nikolay Atanasov as my committee. I am gratefully indebted to their valuable comments on this thesis.

Finally, I must express my very profound gratitude to my parents and to my boyfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

ABSTRACT OF THE THESIS

Modeling User Behavior Patterns in LBSNs: A Graph Embedding Approach

by

Weiqi Xu

Master of Science in Electrical and Computer Engineering (with a specialization in Intelligent Systems, Robotics, and Control)

University of California San Diego, 2019

Professor Julian McAuley, Chair
Professor Truong Quang Nguyen, Co-Chair

With the emerging of various location-based social networks (LBSNs), the study on user mobility patterns and many related tasks have become heated research topics, such as personalized location recommendation and friend recommendation. Many factors affect users' behavior patterns, such as geographical influence, temporal effect and semantic effect. However, most of the previous work on modeling user trajectories lacks consideration on treating these factors from a graph perspective, therefore fails to capture the potential correlations among the rich context. In this thesis, we demonstrate that using a heterogeneous graph-based model to jointly embed user

and POI attribute networks with a unified framework can well preserve the network properties. Multiple factors are embedded into a shared low-dimensional latent space where their joint effect and potential correlations can be well captured. We conduct extensive experiments on large real-world datasets to evaluate our model performance on several major tasks in LBSNs. The experimental results highlight the versatility of our method which shows higher recommendation effectiveness compared with the state-of-the-art baselines. In addition, a online updating strategy is proposed to incorporate new visiting records and dynamically track users' latest preference in linear time. We also show that this framework has the inherent ability to handle cold-start recommendations, which is a non-trivial task considering the network sparsity of LBSNs. The scalability and flexibility of our framework indicate that this method is promising to be put into practical use.

Chapter 1

Introduction

1.1 Motivation and Challenges

With the prevalence of mobile devices and the emerging of various location-based social networks (LBSNs) such as Yelp and Foursquare, massive user visiting records are collected based on users' voluntary reports. More and more users are getting used to check-in at point-of-interests (POIs), such as scenic spots, restaurants and museums, and share their experience. On the other hand, users rely on these networks to help them discover new places and activities, as well as new friends who share similar interests with them. In fact, the study on LBSNs has become a heated research area in recent years, which brings large profits not only to personal recommendation, but also to many high-level tasks, such as event prediction and regional traffic forecast.

The vast amount of check-in records makes it possible to study user mobility, and make personalized recommendations base on the learnt user preference and mobility pattern. However, the problem nature and the characteristics of LBSNs have posed many challenges:

- **Context awareness:** Different from traditional recommender systems such as e-commerce websites, the recommendations in LBSNs usually need to take spatiotemporal context into consideration when making predictions aside from users' personal preference. In

another word, users tend to make different choices in different time and places. In addition, users' mobility pattern exhibits sequential effect. For example, a user would first check-in at a shopping mall, then go to a restaurant, and watch a movie at the cinema later on a typical weekend; travelers might first check-in at the airport after landed, then check-in at a hotel straight after. Taking all these context information into consideration when making predictions is necessary.

- **Dynamic Tracking:** User preference changes over time. On the other hand, the sequential effect of POI transitions has a large impact on user mobility. Therefore, in order to provide satisfying predictions, the recommender system needs to respond to stimulus in real-time according to users' latest preference and spatiotemporal context. Besides, training the entire model takes time and can be costly. Thus, an ideal model should have the ability of online training, which can dynamically incorporate new records, and track users' latest preference without retraining the whole model.

- **Mutual effects of social and POI networks:** User relationships form the social network, and the POI attributes in LBSNs form the heterogeneous POI network. Both social network and POI network affect users' visiting behaviors, and the two networks show mutual effects towards each other. Friends tend to share similar preference and have co-visitation behaviors. Thus, incorporating social relationships can facilitate POI prediction. On the other hand, people who share similar mobility patterns are more likely to become friends, which uncovers the possibility of making social link recommendations based on POI network. Therefore, building a joint framework which captures the mutual effects between the two networks is expected to facilitate related prediction tasks while can be challenging.

- **Data Sparsity:** Visiting a place is more costly than rating movies and items online, and the check-in records in LBSNs is collected based on users' voluntary report. As a result, the data in LBSNs is much sparser than that of traditional recommender systems.

- **Cold Start:** In LBSNs, new places and activities emerges everyday. An ideal model should have the ability of incorporating newly emerged POIs and help users explore new places. In fact, cold-start recommendation is a non-trivial task in LBSNs which needs careful inspection.

Many work has been devoted to studying user mobility patterns in LBSNs while limitations exist. First of all, previous work incorporates a variety of influential factors to learn users' behavioral patterns mostly by combining all the partial factors. For example, probabilistic graphical models [1] generate different distributions for the considerable influential factors; collaborative filtering model simultaneously factorizes coupled tensors and matrices constructed from heterogeneous data sources [2] to perform multi-dimensional collaborative recommendations based-on user, activity, time and location. However, the objective functions of these models are not designed for the network structure of LBSNs, and the diverse nature of multiple factors makes it difficult to incorporate them with an integrated model. In fact, most of the work ends up building a complicated hybrid model, which fails to generalize to various scenarios. Second, among the many factors that affect user behaviors, sequential effect and geographical effect are mostly studied, while social influence is frequently ignored. Last but not least, traditional methods directly capture the interactions between user POI pairs, which fails to observe the global structure of the entire information network [3]. For example, Factorization Machine only optimizes the first-order proximity between user and POI by observing their direct interactions.

Methods which learn graph representations by embedding nodes into a lower-dimensional latent vector space have been attracting increasing attention in the recent years [4]. The performance has been proved to be promising in many tasks, such as text mining [5] and event detection [6].

1.2 Our Work and Contributions

In this work, we extend the state-of-the-art graph embedding method for next-POI prediction and social link prediction in LBSNs. The proposed method models user social network and POI attribute network with a joint framework where their mutual effect are captured. Users, POIs and POI attributes are all treated as network nodes and embedded into a shared lower-dimensional latent space. The observed links between network nodes include user-user, user-POI, POI-category, POI-time, POI-region, POI-rating, and POI-POI, which capture users' social connections, visiting history, categorical effect, temporal effect, geographical effect, semantic effect, and sequential effect in POI transitions respectively. By optimizing the first-order and second-order proximity between network nodes, the obtained embeddings not only preserve the local and global structure of the network, but can also leverage the sparsity problem in real-world datasets. In addition, the proposed framework is flexible to incorporate multiple factors and network attributes, and can scale to large real-world dataset. An online training strategy is introduced which can dynamically track users' latest preference in linear time without harming the prediction accuracy.

We conduct extensive experiments on two large real-world datasets, and compare our model with representative algorithms based on Probabilistic theory, Factorization Machine and Deep Learning. Experimental results demonstrate our model's superiority over other state-of-the-art comparison methods on several predictive tasks.

In context-aware next-POI recommendation, our method outperforms comparison models in terms of both prediction accuracy and efficiency. By incorporating the content information of the cold-start POIs, our model can directly learn the embeddings of cold-start POIs through the same framework without additional feature engineering or model modifications. In addition, experiments show that our model gives better performance in cold-start recommendations than strong baselines.

We also demonstrate that the user embeddings learnt from our framework can be directly

used to discover potential social links by measuring pairwise similarity. The intuition behind is that users' shared preference on POIs is captured through observing the second-order proximity of user-POI links. In another word, users with shared neighboring POIs are embedded close to each other in the embedding space. In fact, people who have similar visiting history are more likely to share similar interest and become friends. Model evaluation proves that our prediction framework achieves better results than heuristic methods, and the user embeddings obtained from our model are more representative compared with those of baselines.

By embedding all the network nodes into a shared latent space, our method not only optimizes on the observed links, but is also able to capture potential correlations among them. This explains for the more promising results in cold-start recommendation and social link prediction where side information is limited.

For each predictive task, we further investigate the impact of different influential factors. Methods to better capture these effects in our embedding framework are also discussed.

To summarize, in this thesis, we:

1. extend the state-of-the-art graph embedding method to model user mobility patterns in LBSNs. A joint framework is introduced which captures the mutual effect between user social network and POI attribute network.

2. introduce an online learning strategy which can dynamically update user preference without retraining the entire model, and can provide context-aware recommendations according to users' stimulus in real-time.

3. demonstrate the superiority of the proposed model in next-POI prediction, social link prediction, and gives promising results in cold-start recommendation.

4. investigate the impact of influential factors on the modeling of user mobility, and discuss the way to better capture these effects in our embedding framework.

1.3 Thesis Organization

The rest of work is organized as follows. In Chapter 2, we introduce the problem background, and provide a detailed explanation on the motivation behind this work. In Chapter 3, we explore the statistics and characteristics of the datasets we study on, and investigate the most significant effects the datasets exhibit. In Chapter 4, we explain the predictive tasks we study in this work, then introduce the model formulation, including embedding methodology, training process and prediction strategy. In Chapter 5, we report the experimental results of our method along with other representative comparison models on all predictive tasks. Thorough discussions on the experimental results are also included. In Chapter 6, we conclude our work, discuss the limits and shed light on some future directions.

Chapter 2

Background

2.1 Characteristics of Location-based Social Networks

The emerging of location-based social network is a natural product of the development of location-based service. The location network contains massive amount of POIs along with their multiple attributes, including location, opening hours, and content information such as categories and user reviews. The social network is usually formed based on friendship, common interests, and shared knowledge. The LBSN is not a simple combination of the location network and social network where users can share POI-related information, it is partially observed and contains potential interactions between network nodes. The implicit connections are derived from users' location-tagged check-in history, which includes timestamp, photos and reviews [7]. Figure. 2.1 [8] illustrates an example of a typical LBSN. As is illustrated in the diagram, the user network and location network are connected through visiting history to form the user-location graph, which contains location-tagged and user-generated content. The shared activities and mobility patterns among users indicate potential social closeness, while on the other hand, the mobility of similar users reveals content similarity among those locations.

The rich content information provided in LBSNs can help us get a better understanding of

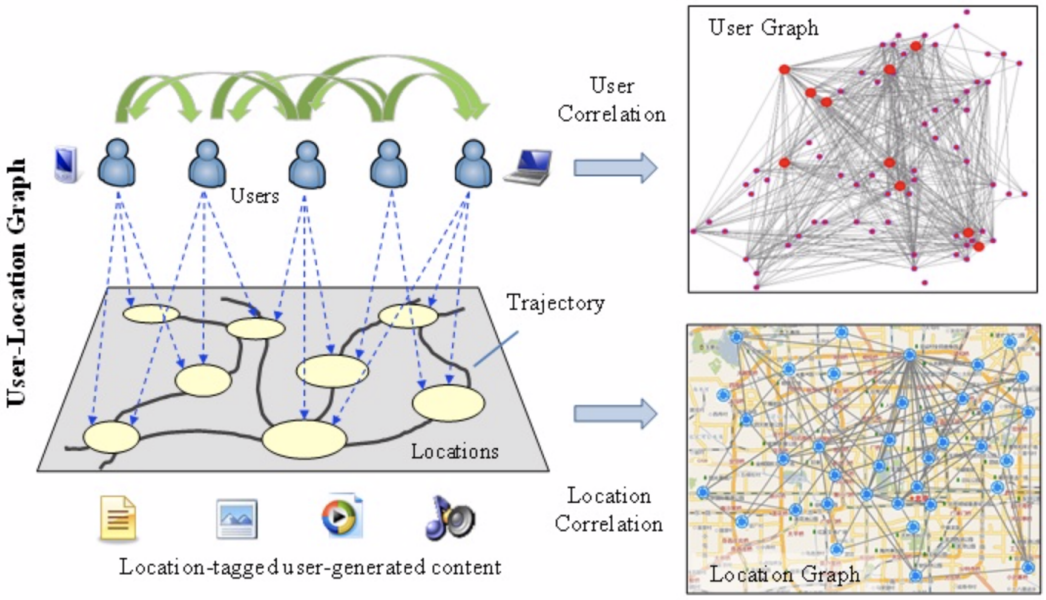


Figure 2.1: Example of a Typical LBSN

users, locations and their interactions, and bring profits to many related tasks, such as community detection, event detection, preference- and context-aware location recommendation, and friend recommendation.

The most popular and well studied LBSNs datasets are Foursquare [9] and Gowalla [10] check-in data. Each check-in record contains user information, check-in time and POI side information. Users' friend list is also included. Gowalla dataset a has larger size compared with Foursquare, but it lacks POI content information such as user reviews. These two datasets are collected based on users' voluntary report, therefore suffer from severe data sparsity. Various types of tasks are studied, including location recommendation, social link prediction and user mobility tracking. Tarasov et al. [11] combine Foursquare check-in history with Twitter friend list to test their radiation model for location prediction task. In the work by Yao et al. [12], a recurrent model for next location prediction is proposed and evaluated on Foursquare check-in data collected from New York City and Los Angeles. Users with fewer than 50 records are removed, and only semantic trajectories within a 10h time interval are kept to obtain a denser network.

Aside from check-in data collected through users' voluntary report, similar datasets that based on automatic collecting are also studied. For example, Huang et al. [13] worked on Wi-Fi access logs collected at Purdue University, which is much denser than Foursquare or Gowalla check-in data, and exhibits more significant temporal cyclic patterns.

2.2 Influential Factors

Among the many tasks in LBSNs, social link prediction and location recommendation have become popular research areas in the past few years. User mobility patterns is studied to learn personal preference, and rich POI attributes are frequently incorporated to provide valuable recommendation context. On the other hand, the side information helps to alleviate the sparsity problem of check-in data. For example, the insufficient observations of check-in activities can be compensated by tipping behaviors and rich text information such as reviews to infer user interests [14].

In social link prediction tasks, factors people usually consider to measure trajectory similarity [15] include location category [16], geographical distance [17], and co-occurrence with time and distance constraints, where the diversity of co-occurrence and popularity of locations were proved to show dominant effects and are most well studied. Pham et al. [18] introduce the notion of commitment and compatibility to measure similarity of users' mobile patterns. The similarity quantifies social distances between users, which is used to infer real-world co-occurrences. In another work by Pham et al. [19], they further propose an entropy-based model which estimates the strength of social connections by analyzing people's co-occurrences in space and time.

In location recommendation tasks, information used for context- and time- aware recommendation includes social ties, geographical information, temporal information, and text description such as categories and reviews. Temporal and geographical information are most

well studied to learn users' spatiotemporal preference or change in interest over time. Early work[20][21] argue that user preference is time-varying and has periodic patterns (e.g. day in a week). A direct approach is to add a dimension for time into the user-item adjacency matrix, then adopt Factorization-based methods [22], [2]. Yuan et al. linearly combine temporal cyclic patterns and geographical influence into a user-based collaborative filtering framework for time-aware POI recommendation. In another work done by Yuan et al. [23], preference propagation is applied on a geographical-temporal graph to realize time-aware location recommendation. Topic models are also widely adopted in time-aware recommendation. For example, [24] uses a topic model to learn users' temporal preference by creating unique time features for every topic.

In the next-POI prediction problem, the sequential transitions between locations is an addition factor to consider, which shows significant effect. Markov chain is widely used to capture the sequential effect in user mobility. Zhang et al.[25] introduce a probability-based additive Markov-chain model based on the assumption that the latest activities have the greatest impact on user's current preference. Another popular method is FPMC proposed by Rendle et al. [26]. It factorizes the tensor of transition cube consists of the transition probability matrices of all users. FPME [27] extends FPMC by modeling user-location distance and location-location distance in two different vector spaces. However, Markov chain methods are based on the assumption that the choice of next location is only affected by the previous location, which does not hold in the real-world scenarios. To track users' entire check-in history, recurrent models are introduced, such as RNN, LSTM and GRU. Liu et al. propose a Spatial Temporal Recurrent Neural Networks (ST-RNN) [28], which can model continuous time intervals and geographical distance by constructing time-specific and distance-specific transition matrices in each layer of the neural network.

Recently, a lot of research has been devoted to inferring next-POIs through analyzing human mobility. Wang et al.v[29] indicate that the social proximity is strongly bound up with one's mobility. The work in [30] distinguishes users' different periodic mobility between work and home. [13] works on Wi-Fi access logs collected at Purdue University, and demonstrates

that students' campus life exhibits more significant cyclic patterns. The work in [31] uses user mobility in LBSNs to measure the connectivity of various cities. More relevant work studying user mobility in LBSNs can be found in[32][33].

Some other work making use of other factors includes that of Liu et al. [34], which incorporates venue tags into a Matrix Factorization-based model for POI prediction.

The diverse nature of these influential factors poses challenges to incorporating them into a joint model. Many previous work comes up with hybrid models, where each individual component only captures one specific effect. In the work by Yang et. al [15], they first derive the network representation for each user with a probability-based generative model according to social links, then employ a RNN model and a GRU model to capture short-term and long-term effects of the sequential relatedness in users' mobile trajectories respectively. The final model is a linear combination of all the partial representations of user and sequential context. Wang et al. [5] incorporates visual contents by first extracting image features using CNN, then add them into a probabilistic Matrix Factorization model to learn user and POI latent features. Hybrid models may show good prediction performance in the specific scenarios they're designed for, while need careful feature engineering and lack the ability to generalize.

Joint frameworks do exist, such as aggregate LDA, Matrix Factorization-based and collaborative filtering-based methods. Matrix or Tensor Factorization is a popular methodology that has been proved to be efficient and effective in POI recommendation tasks. Zhang et al. propose a multi-dimensional collaborative recommendation framework which uses Tensor Factorization techniques to capture temporal, categorical and geographical effects. Gao et al. [14] build a low-rank Matrix Factorization model which employs a sentiment-enhanced weighting framework when consider user sentiment indications, user-interest content and POI-property content. Zhao et al. [35] propose a unified LDA-based probabilistic model to learn user preference based on reviews, categories and geolocations of POIs. The model can also capture the interaction of sentiment, categorical and spatial information. Pasricha et al. [36] modify the traditional Factor-

ization Machine by replacing the inner product with the squared Euclidean distance to measure the interaction strength between features. The proposed model can operate on arbitrary feature vectors and is flexible to incorporate additional content information. Although these methods incorporate all the factors into a general framework, however, they neither cannot scale to large networks, nor their objective functions are not designed for capturing network structures, therefore not necessarily preserve the global network structure. [3]

In the next section, we will provide a detailed literature of some widely used graph embedding approaches for network embedding tasks. Their characteristics and limitations will be discussed. Finally, we will introduce the approach we use in this work, and provide some insights of the intuition behind our choice.

2.3 Distributed Representation Learning and Graph Embedding Methods

In the recent years, graph embedding methods which represent networks along with their properties into a latent vector space have been attracting increasing attention. Specifically, these methods aim at learning graph representations of the networks by embedding nodes into a lower-dimensional vector space, where connected nodes are embedded close to each other in the embedding space [4]. In general, there are three main groups of approaches: Factorization-based, Random-Walk based and Deep Neural Network-based.

2.3.1 Factorization-based Approaches

The Factorization-based methods obtain the embeddings by factorizing the matrices representing the connections between network nodes. The most widely used matrices include adjacency matrix, transition probability matrix and Laplacian matrix. A representative work using

adjacency matrix is Locally Linear Embedding (LLE) [37], which is under the assumption that every node is a linear combination of its k-nearest neighbors in the embedding space. The weights of neighbors nodes forms the weight matrix W, and the reconstruction error is defined as:

$$\varepsilon(W) = \sum_i |\vec{X}_i - \sum_j W_{ij}\vec{X}_j|^2 \tag{2.1}$$

where:

$$\Sigma_j w_j = 1$$

Mapping every point X_i into a lower-dimensional representation Y_i, the objective turns into:

$$\min_Y \Phi(Y) = \sum_i |Y_i - \sum_j W_{ij}Y_j|^2 \tag{2.2}$$

where $\Phi(Y)$ is further turned into:

$$\Phi(Y) = \sum_{ij} M_{ij}(Y_i \cdot Y_j) \tag{2.3}$$

where:

$$M = (I - W)^T(I - W)$$

After decentralization, the problem comes down to an eigenvalue decomposition problem, where the embeddings are the eigenvectors corresponding to the last m non-zeros eigenvalues of the positive semi-definite matrix M.

The Laplacian matrix [38] is formed with Laplacian of the graph, which is more computationally efficient, and can preserve local structure well. The basic idea is to make the representations of two similar nodes close to each other in the low-dimensional embedding space. The objective

function is:

$$\min \sum_{ij} |y_i - y_j|^2 W_{ij} = \min trace(Y^T LY),\ s.t. Y^T DY = I \tag{2.4}$$

$L = D - W$ is the Laplacian matrix of the graph, where W is the adjacency matrix, and D is the degree matrix ($D_{ij} = \sum_{j=1}^{n} W_{ij}$).

Some related models include Graph Factorization (GF) [39], High-Order Proximity Preserved Embedding (HOPE) [40], and Structure Preserving Embedding (SPE) [41]. A further extension based on Factorization methods is Marginal Fisher Analysis proposed by Yan et al. [42]. Two types of graphs are introduced in this work: the intrinsic graph characterizes the intraclass compactness, while the penalty graph connects the marginal points between different classes and characterizes the interclass separability.

In general, Matrix Factorization methods obtain the embeddings by solving the leading eigenvectors of the affinity matrices, therefore suffer from high computational complexity and cannot scale to large real-world networks. Factorization Machine shows better scalability, while still, it only observe the direct interactions between network nodes, and fail to capture the global structure of the entire network. What's more, Factorization-based methods are not capable of learning arbitrary functions [4]. Their objective functions are not designed for capturing network structures, therefore cannot learn structural equivalence or preserve global structure.

2.3.2 Random Walk-based Approaches

By comparison, Random Walk-based methods show the ability to capture higher order proximities among nodes and can handle partially observed large-scale networks.

The random walk task is essentially a discrete Dirichlet problem applied on the graph. A random walk path usually starts from a selected node in the network, then moves to the random neighbors from the current node for a pre-defined number of steps. The relative probability

distributions of the child nodes are computed by sampling a set of random walk paths from the graph. To be specific, given a graph $G(V, E)$, if a random walk starts from node v_0 and reaches node v_t after t steps, the probability distribution can be represented as:

$$P_t(i) = Pr(v_t = i) \tag{2.5}$$

The transition probability matrix is $M = (p_{ij})_{i,j \in V}$, where

$$p_{ij} = \begin{cases} 1/d(i), & \text{if } ij \in E, \\ 0, & \text{otherwise} \end{cases} \tag{2.6}$$

Therefore, the walking rule from one node P_t to another node P_{t+1} can be described as:

$$P_{t+1} = M^T P_t \tag{2.7}$$

DeepWalk was first proposed for language modeling based on word2vec and random walk, but it has also shown good performance in other networks such as social networks. DeepWalk preserves higher-order proximities between nodes by maximizing the probability of observing the neighbors of a node within a certain window conditioned on its embedding using stochastic gradient descent[6]. Under this scheme, nodes with similar neighbors are closely embedded in the embedding space.

The objective function can be described as:

$$\mathcal{L}(s) = \frac{1}{s} \sum_{i=1}^{|s|} \sum_{i-t \leqslant j \leqslant i+t, j \neq i} \log Pr(v_j | v_i) \tag{2.8}$$

where

$$Pr(v_j | v_i) = \frac{exp(\vec{v_j}^T \cdot \vec{v_i})}{\sum_{v_k \in V} exp(\vec{v_k}^T \cdot \vec{v_i})} \tag{2.9}$$

where v_j is the neighboring nodes of node v_i within a $t - steps$ window, equation 2.9 is the conditional probability between node v_i and v_j, which captures their second-order proximity.

DeepWalk only relies on local information of the graph, therefore is suitable for sparse graph and can be implemented on distributed systems. In addition, adopting Deep Learning techniques accelerates the computation speed and improves scalability. However, since DeepWalk performs random walks randomly, the obtained embeddings do not preserve the local structure of each node very well.

Node2vec [43] fixes this problem by adopting biased-random walks, which provide a trade-off between breadth-first (BFS) and depth-first (DFS) graph searches. BFS focuses on exploring neighboring nodes, which captures the local structure; DFS on the other hand, captures higher-order similarity therefore preserves global structure.

For each step of a random walk c, the transition probability from a current node v to the next node x is defined as:

$$P(c_i = x | c_{i-1} = v) = \frac{\pi_{vx}}{Z} \tag{2.10}$$

where Z is used for normalization. The algorithm sets the bias between BFS and DFS by introducing two parameters p and q. Denote the previous node as t, the transition probability π_{vx} is defined as:

$$\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx} \tag{2.11}$$

where

$$\alpha_{pq}(t, x) = \begin{cases} 1/p, & \text{if } d_{tx} = 0 \\ 1, & \text{if } d_{tx} = 1 \\ 1/q, & \text{if } d_{tx} = 2 \end{cases} \tag{2.12}$$

w_{vx} is the weight of edge E_{vx}, d_{tx} is the shortest distance between t and x. It is set as 0 if the next node is the same as the previous one, 1 if the distance to the next node and the previous node are the same, otherwise it is set as 2. p is called "Return parameter", which stands for the probability of visiting a node again; q is called "In-out parameter", which controls the level of depth-first search.

Walklets [44] extends DeepWalk and node2vec by skipping over some nodes in the graph, and can explicitly captures multiple scales of relations in the network. Besides, the latent representations generated are more human-interpretable.

Despite the scalability and the ability to capture higher order proximities, one major drawback of Random Walk-based approaches is that they only apply to undirected networks, while in LBSNs, both directed and undirected edges exist.

2.3.3 Deep Neural Networks-based Approaches

Deep Neural Networks have been widely used for dimensionality reduction in representation learning, which show the advantage of capturing non-linearity of the network structure. Wang et al. [45] present a hybrid deep autoencoder called SDNE which can capture both first-order and second-order proximities. Specifically, they use a Laplacian Eigenmap-based model to measure the first-order proximity, and use another autoencoder to obtain the embedding of each node based on its observed neighborhoods. GCN [46] adopts an iterative approach, which aggregates the embeddings of the neighboring nodes and combine them with the those obtained from the previous iteration using a convolution operator to obtain the new embeddings. This model captures the global structure, and shows better scalability compared with the aforementioned methods. Recent work by Hamilton et al. propose a general inductive framework called GraphSAGE [47], which leverages node attributes to efficiently generate node embeddings for previously unseen data using forward propagation. The embedding of each node is generated by sampling and aggregating features from its local neighborhood. As this process iterates, nodes incrementally gain more and

more information from further reaches of the graph.

Deep Neural Network-based approaches can learn arbitrary functions due to their general optimization strategy. However, neural network is a black box and not human-interpretable, which poses challenge on model tuning and modifications. In addition, they rely on large amount of data for training, and will give poor performance when the amount of training data is not sufficient.

2.3.4 LINE-based Approaches

LINE captures both first-order and second-order proximities between nodes. To capture the first-order proximity, the joint probability between two nodes (equation 2.13) is optimized to approximate the empirical probability.

$$p_1(v_i, v_j) = \frac{1}{1 + exp(-\vec{v}_j^{\,T} \cdot \vec{v}_i)} \tag{2.13}$$

The empirical probability is defined as the relative weight among all edges:

$$\tilde{p}_1(i, j) = \frac{w_{ij}}{W} \tag{2.14}$$

where

$$W = \sum_{(i,j) \in E} w_{ij} \tag{2.15}$$

Measuring the second-order proximity aims at observing the shared neighborhoods. The probability of observing a context v_j generated by v_i is defined as:

$$p_2(v_j | v_i) = \frac{exp(\vec{u}_j'^{\,T} \cdot \vec{u}_i)}{\sum_{k=1 \in |V|} exp(\vec{u}_k'^{\,T} \cdot \vec{u}_i)} \tag{2.16}$$

where $|V|$ is the number of context vertices. u is the representation of v when it is treated as a vertex, a node v is represented with u' when it is treated as a context vertex.

The empirical distribution of the context probability among all neighboring nodes is defined as:

$$\tilde{p}_2(v_j|v_i) = \frac{w_{ij}}{d_i} \tag{2.17}$$

where $d_i = \sum_{k \in N(i)} w_{ik}$ is the out-degree of vertex i, and $N(i)$ is the set of out-neighbors of v_i.

KL-divergence is used to measure the proximity between the two distributions. Note that the first-order proximity is only applicable for undirected graph, while second-order proximity is applicable for both directed and undirected edges. The intuition is that an undirected edge can be taken as a combination of two directed edges with opposite directions and equal weights. Therefore, LINE is a more general method which can handle different types of network structures and capture arbitrary level of proximity.

In light of its universal nature, LINE-based approach is suitable for embedding various types of information networks, including language network, social network and citation network. It also shows good performance in many network based tasks, such as link prediction and node clustering. Experimental results on various real-world networks have proved the superiority of LINE over heuristic methods in terms of efficiency and effectiveness.

Previous work provides valuable insight into choosing suitable embedding techniques to better preserve network structure and properties, as well as adjusting to specific network characteristics.

2.4 Cold Start Problem

Cold-start is a non-trivial problem in LBSNs, where new places and events emerge everyday, and users rely on the recommender system to explore new places and activities, which makes cold-start recommendation a non-trivial task that needs careful inspection.

Massive previous work has focused on addressing location-side cold-start problem in

location recommendation tasks. For example, researchers [48] investigate the mutual influence between geographical distance and social networks, and show that leveraging the cold-start locations from a geo-social perspective might be promising. In the work by Gao et al. [49], they address the cold-start location recommendation problem by capturing the correlations between social networks and geographical distance with a geo-social correlation model. The intuition behind this implementation is that users in different geo-social circles tend to have various correlation strength. Four types of geo-social circles are considered that form a probability-based prediction model.

In conclusion, incorporating side information of the cold-start POIs is essential for alleviating the cold-start impact.

Chapter 3

Data Exploration

In this work, we evaluate our method on two standard LBSN datasets: Foursquare [9] and Google Local [50]. In this chapter, we report detailed dataset statistics and investigate the characteristics of these two datasets. We will also explain how we choose the feature representations for the information provided in these datasets to facilitate our modeling.

3.1 Dataset Statistics

3.1.1 Foursquare

The Foursquare dataset is consists of check-in data collected based on users' voluntary report while using this app. The subset we use is from the time period Dec. 2009 to Dec. 2011. Every record contains six attributes: user ID, check-in time, venue ID, venue name, venue location and venue category. To be specific, check-in time is represented with GMT time. Venue location contains the geographical information, including coordinates (longitude and latitude) and administrative division such as city and state. Note that a venue may belong to more than one category. Foursquare also contains users' social links, which are organized in a pairwise manner. More data statistics and examples are shown in Table. 3.1.

Table 3.1: Foursquare Data Description and Examples

	Statistics	Description or Example
User	4,019	–
Social Link	16,256	pairwise, undirected
Check-in Time	2009.12 - 2011.12	EST. Mon Jul 25 02:03:30 +0000 2011
Venue Location	82,238	[32.7075, -117.1570], San Diego, CA
Venue Category	34	Shop & Service, Nightlife Spot etc.

3.1.2 Google Local

This dataset [36][**?**] contains a large collection of reviews about businesses from Google Local (Google Maps). Rich side information of the corresponding users and businesses is also included, such as user demographics, review time, geographic information, business opening hours and category. To be specific, geographical information contains coordinates and administrative division. Temporal information is provided in the form of both unix timestamp and GMT time. Note that this is the reviewing time rather than check-in time. The data statistics and examples are shown in Table. 3.2.

Table 3.2: Google Local Data Description and Examples

	Statistics	Description or Example
User	4,567,431	–
Rating	Integer ranges from 0 to 5	4, 5
Business Location	3,116,785	[29.517006, -98.436528], San Antonio, TX
Review Time	2005.02 - 2014.03	UTC. Mar 14, 2014 1394842359
Review	11,453,845	"Food is ok at best and the prices are out of this world."
Business Category	2700	Chinese Restaurant, Fireplace Store, etc.

Table 3.3: Data Statistics Before and After Filtering

Dataset	#users	#venues	#records	#usersFilter	#venueFilter	#recordFilter
Foursquare	4144	113473	483814	3244	111355	435289
California	123976	177783	214600	13462	26899	91086
Texas	96163	118306	147575	8155	16512	49343
Florida	85619	103322	116580	5754	12504	28482
Colorado	32519	37344	46977	2496	13705	4875
Washington	27129	43305	42823	2482	16512	15467
North Carolina	33337	42396	46748	2381	5381	12655

LBSNs are usually very sparse. In this work, we only use records collected from California in Foursquare dataset, and eliminate users with less than 10 check-in records in order to obtain a denser network. On Google Local dataset, we conduct experiments on data collected from six U.S. states of varying sizes and populations. Google Local dataset is even sparser than Foursquare dataset, with fewer records for each user and venue. To alleviate the effect of data sparsity, only users with more than 5 review records are kept.

The detailed dataset statistics before and after filtering are shown in Table. 3.3.

3.2 Data Characteristics and Properties

- **Geographical Distribution**

  In order to get a more intuitive understanding, we visualize the geographical distribution of all venues on the state map based on venue coordinates. Figure. 3.1 and Figure. 3.2 provide the example of the geographical distribution of California venues in Foursquare and Google Local.

  The visualization shows that the geographical distribution of venues from these two datasets are similar: venues are not evenly distributed across the state. The venue density of large cities such as LA county and the bay area is much higher compared with rural areas. This indicates that venue distribution is highly correlated with population distribution, and

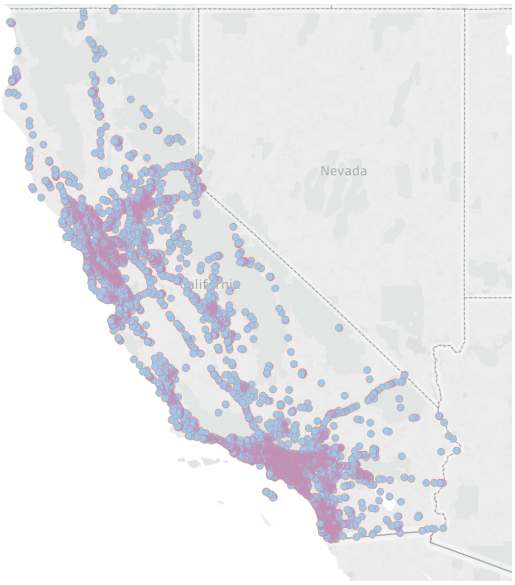


Figure 3.1: Geographical Distribution of Foursquare Venues

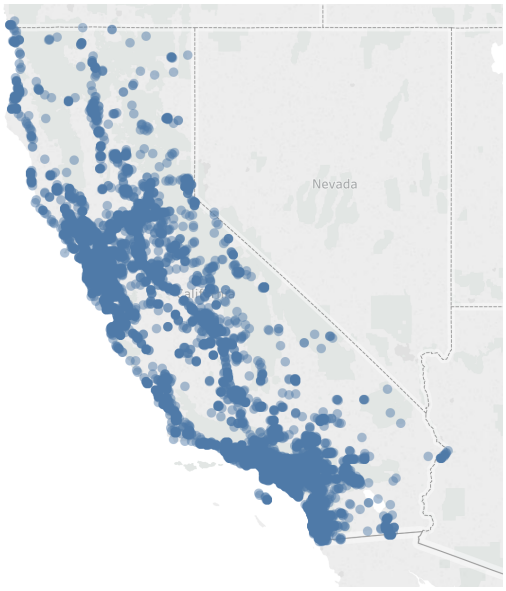


Figure 3.2: Geographical Distribution of Google Local Venues

geographical information can serve as a informative indicator of the place a user may visit.

- **Temporal Dynamics of User Mobility**

  Research [51][52] has shown that user mobility exhibits strong temporal cyclic patterns. Therefore, we investigate the temporal dynamics of user mobility shown in the datasets we use. Time-of-day effect and day-of-week effect are studied.

  Google Local dataset only contains the time a user wrote the reviews, which is usually different from the time of visit, therefore is not suitable for studying the time-of-day effect. As a result, we only investigate this effect on Foursquare dataset, and draw the heat maps showing the types of activities users do during 12 time slots in a day, where each time slot is a 2-hour interval. Since human mobility patterns might be different during workdays and weekends, we draw two separate heat maps for workdays and weekends respectively, which are illustrated in Figure. 3.3a and Figure. 3.3b.

  To study the day-of-week effect, we draw the heat map of different types of activities users do in each day of the week. We investigate this effect on both Foursquare and Google Local

dataset, which are shown in Figure. 3.4 and Figure. 3.5. For Google Local dataset, we assume that users tend to rate the business on the day of visit when memory is still fresh, and we only provide an illustration of 30 most popular business categories in California as an example.

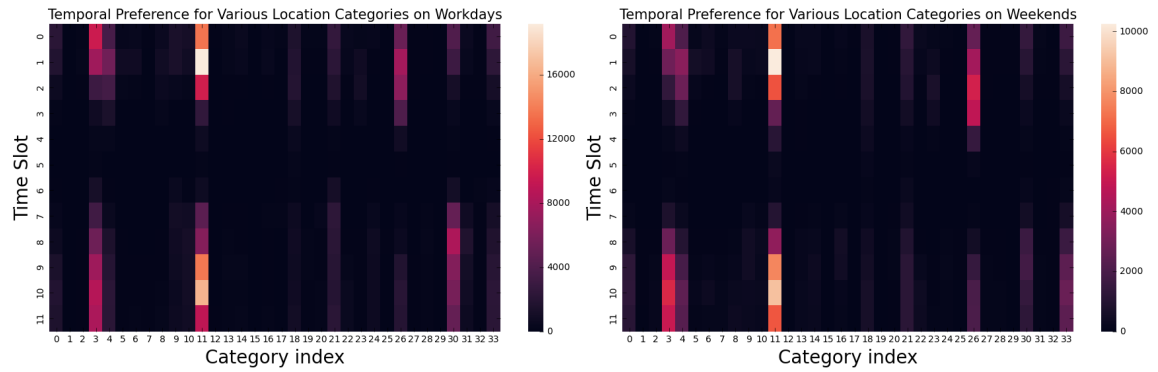
There are several observations we can get from the illustrations:

(i). User mobility exhibits significant time-of-day effect on both workdays and weekends. For example, 'Food' related venues are the most visited (denoted with index 11) during breakfast and dinner time, while there are only a few visits during other time of a day.

(ii). Time-of-day effect is more significant than day-of-week effect. User mobility does show day-of-week effect, for example, on Foursquare, people tend to have fun at "nighttime spots" (with index 3) and go for 'Shop & Service' (with index 26) more often during weekends, while during workdays, 'Professional & Other Places' are more frequently visited. Google Local shows similar effect: 'American Restaurant' (with index 26) and 'Seafood Restaurant' (with index 10) have the largest amount of visits on Sunday. However, day-of-week effect is not as obvious compared with time-of-day effect. In fact, user mobility generally shows similar daily patterns during workdays and weekends, for example, people grab food during breakfast and dinner time, and go shopping during the night.

(iii). Among all venue categories, only a few of them are popular, which receive most of the visits. For example, on Foursquare dataset, 'Food' and 'Shop & Service' have 153,238 and 83,976 visits respectively, which together takes up 59% of the total check-in records. On Google Local dataset, 'American Restaurant' takes up over 10% of the total visiting records. This effect is also clearly shown in the heat maps, as only a few activities carry bright color patch.

These observations prove that user trajectories exhibit strong temporal cyclic patterns, and suggest that (i). incorporating day-of-week effect can facilitate POI prediction task on both



(a) On Workdays (b) On Weekends

Figure 3.3: Users' Temporal Preference for Various Activities

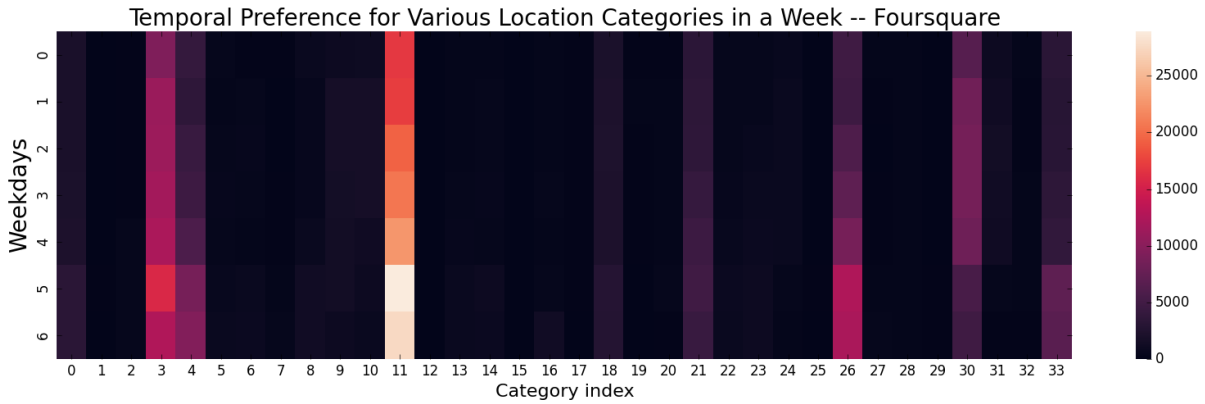


Figure 3.4: User Preference vs. Day-of-week on Foursquare

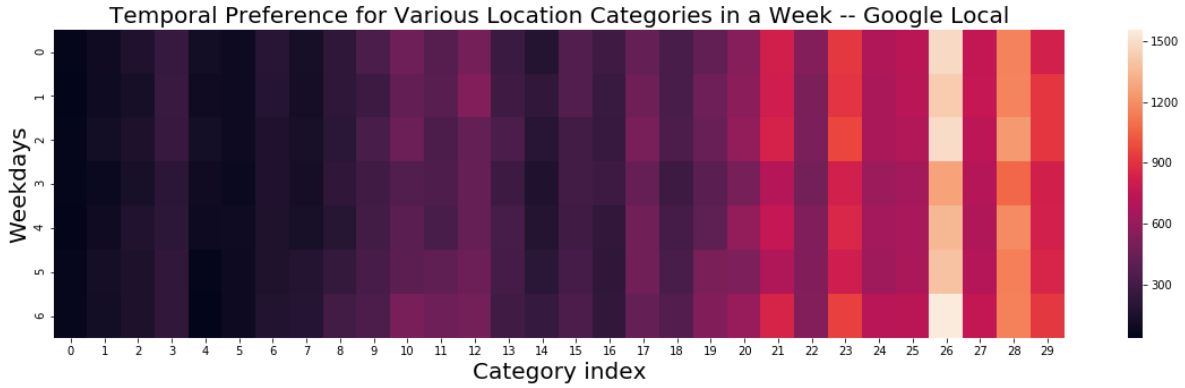


Figure 3.5: User Preference vs. Day-of-week on Google Local

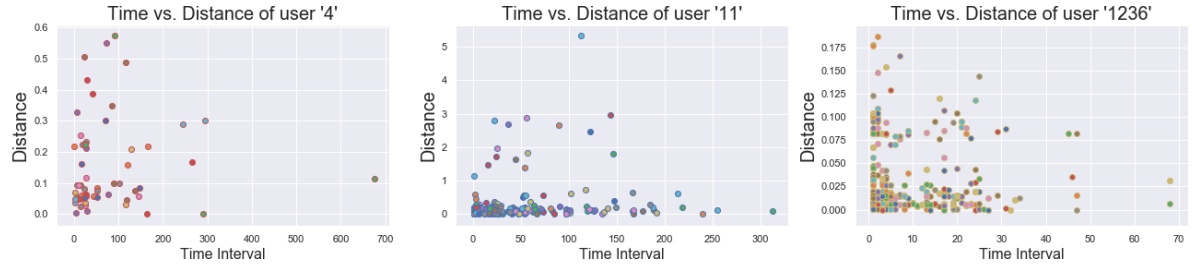


Figure 3.6: Sequential Patterns of User Behaviors in Foursquare

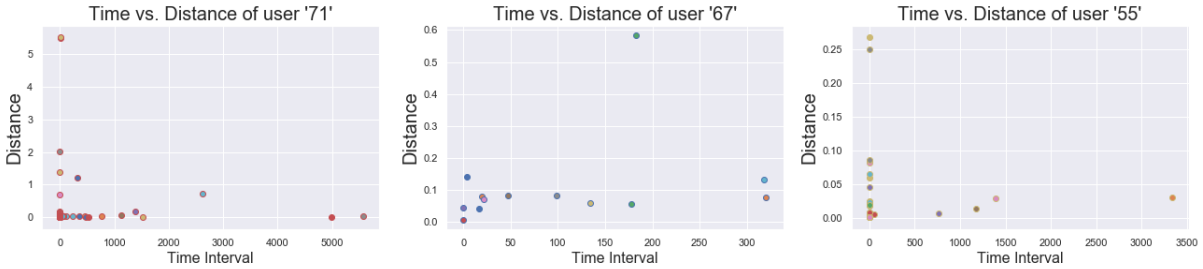


Figure 3.7: Sequential Patterns of User Behaviors in Google Local

datasets. (ii). time-of-day effect need to be considered in Foursquare dataset, while there is no need to incorporate it when studying Google Local dataset. (iii). Categorical tags are informative and can be used as a influential factor when predicting the next POI.

- **Sequential Transition Patterns in User Trajectories**

  As discussed in Chapter 2, user mobility may exhibit sequential pattern. For example, users usually sequentially check-in at a restaurant, then at a cinema at weekend nights. Such sequential pattern is even more noticeable at airport and hotels. This suggests that the time interval and distance between two sequentially visited places might be highly correlated. In another word, people would only go to nearby places during a short period of time. Figure. 3.6 and Figure. 3.7 show the spatiotemporal dynamics of thee randomly picked users from Foursquare and Google Local respectively. The x-axis denotes the time interval between two sequential check-ins measured with hour, and y-axis is the distance between the two check-in venues.

  According to the diagrams, Foursquare exhibits more significant sequential effect than

Google Local. In Foursquare dataset, large time interval generally corresponds to large distance from the previous check-in spot to the current location. However, in Google Local, such pattern is not as obvious. Further inspection reveals that, the time interval between two sequential reviews in Google Local dataset is usually very large. This may due to the larger data sparsity, and users tend to give ratings and write reviews afterwards. Therefore, the sequential transitions between venues are not well reflected in Google Local dataset.

The above investigation suggests that taking sequential effect into consideration can facilitate next-POI prediction on Foursquare, while might not has significant impact on Google Local.

- **Friendship Influence**

  Foursquare dataset contains users' friendship information. Friends tend to have similar preference and co-visitation behaviors. Previous work [18][19] has shown that social network can serve as auxiliary information to enhance user trajectory modeling, therefore facilitate location prediction and recommendation.

  To explore the friendship influence upon user behaviors, we randomly pick 50 users in Foursquare who has social links with other users, and compare their average trajectory similarity with their friends, and all other unconnected users. **Jaccard similarity** is used to measure the similarity between users, which is defined as below:

$$J(u_i, u_j) = \frac{|u_i \cap u_j|}{|u_i \cup u_j|} \tag{3.1}$$

  where u_i stands for the set of venues user i has visited before.

  As is shown in Figure 3.8, friends visit significantly more common places. This makes sense since friends usually share similar interest, and would recommend places they have visited to each other. Co-visiting behavior is also very common. In fact, the average Jaccard Similarity among friends (0.00825) is about 5 times higher than the overall similarity

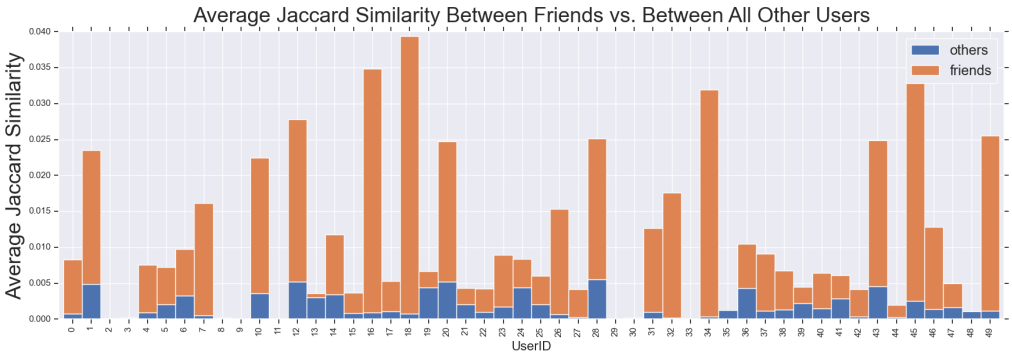


Figure 3.8: Average Jaccard Similarity Between Friends and All Users

(0.00176).

The statistical analysis further proves that friendship influence has a large impact on user behaviors, therefore can be used to facilitate location prediction and recommendation. On the other hand, similar mobility may indicate potential social links.

- **Semantic Effect**

  Google Local dataset contains ratings and reviews about POIs. The semantic feedback provides valuable information which helps us make inference of the POI popularity and user preference. For example, a user gave a '5-star' rating to a 'Chinese restaurant' wrote the review "Best Hot Sour soup anywhere." indicates that he has preference towards the food and is likely to visit again.

  We also observe that make inference solely based on 'rating' or 'review' can be biased. Some users would habitually give high ratings even though they are not satisfied with the place, and vice versa. For example, a user gave a restaurant a '3-star' rating wrote the comment 'Sad place. Won't be back.', while another user gave the same rating because 'OK BUT NOT GREAT!'. This phenomena indicates that evaluating rating and review sentiment jointly will lead to a more reliable and unbiased assessment.

- **Cold Start**

Table 3.4: Data Statistics of Cold-Start POIs

Dataset	Foursquare	CA	TX	FL	CO	WA	NC
# Total POIs	2887	9479	6003	4709	1860	1899	1929
# Cold-Start POIs	625	3948	2775	2314	814	813	968
Proportion	21.65%	41.65%	46.23%	43.76%	42.81%	49.14%	50.18%

As new places and events emerge everyday, LBSNs exhibit significant cold-start effect.

In this work, since we focused on studying user mobility for user side recommendations, we only care about the influence of cold-start POIs. As will be further explained in Chapter 4, during model evaluation, we use the latest visiting record of each user as the test example. If the first appearance of a POI is in the test set, then it is considered as a cold-start POI. We count the occurrence of cold-start POIs of both datasets, the result is reported in Table. 3.4.

Data statistics show that both datasets exhibit significant cold-start effect, and Google Local suffers from more severe cold-start problem compared with Foursquare dataset. In Foursquare, 21.648% of the POIs in the test set are cold-start POIs, while in Google Local, cold-start POIs take up about 45.62% of the total POIs on average.

This observation further proves that making cold-start recommendation is a non-trivial task which needs careful inspection.

Chapter 4

Methodology

4.1 Predictive Tasks

According to the analysis in Chapter 3, generally, user mobility in LBSNs exhibit six evident factors: geographical influence, temporal cyclic effect, sequential effect, semantic effect, categorical effect and friendship influence. These factors affect the spatiotemporal dynamics of user preference, therefore can be used for personalized location-based prediction. By measuring the similarity of user trajectories, we can discover users with shared behavior patterns and similar interest, thus have the potential to become friends. In addition, we have shown that enabling cold-start recommendation is a non-trivial task which needs careful inspection. In light of these observations, in this section, we define three predictive tasks we're going to study in the following chapters.

- **Context-aware Next-POI Prediction.**

  Next-POI prediction and recommendation is a most widely studied problem in LBSNs. User mobility patterns are learnt through investigating visiting history, and the rich side information about users and venues can help us understand users' personal preference. On the other hand, unlike traditional recommendation tasks such as movie and product

recommendation, next-POI recommendation involves a more structured and context-rich environment. To be specific, besides from personal preference and mobility patterns, a rich set of context information will jointly influence a user's choice, such as current time and location.

In this work, we study the context-aware next-POI prediction problem which takes both user preference and context information into consideration. To be specific, given a user and his or her visiting history, we make prediction on the next POI that user is likely to visit based on time and location context. We will also investigate the impact of each influential factor for a comprehensive understanding of this predictive task.

- **Social Link Prediction and Friend Recommendation**

  The social network is an important part of LBSNs. Friends share their visiting experience and help us explore new places and activities. On the other hand, users can discover new friends that have shared interests with them. By comparing the personal preference and mobility patterns, we are able to measure the social distances between users and discover users with the potential to become friends.

  In this work, we make effort to discover potential social links in LBSNs based on users' shared interest and similar visiting history. A personalized friend recommendation framework is therefore proposed.

- **Cold-start Recommendation**

  In Chapter 3, we have shown that LBSNs exhibit significant cold-start effect. In this work, we investigate the way to incorporate cold-start POIs into the recommendation framework. POI side information such as category and location can serve as auxiliary information that helps to find the potential correlation between cold-start POIs and observed POIs. In addition, the consideration of contextual features such as time and location also helps to build a robust recommender that has the ability to deal with the cold-start scenario.

4.2 Graph-based Embedding Model

Multiple factors jointly affect the predictive tasks in LBSNs. To model the interactions between these multiple factors, a graph structure turns out to be a natural choice since it is more amenable to representing and reasoning rich context compared to heuristic Factorization-based and RNN-based methods [13]. An expressive vector representation for each node in the graph needs to be found while this task is inherently difficult. A 'good' vector representation should preserve both global structure of the graph and the local connections between nodes. In addition, the network properties and characteristics also need to be well captured.

In this work, to investigate user mobility in LBSNs, we use a bipartite graph model to depict the relations between different types of network nodes, where each type of graph links captures a certain influential factor exhibits in the network. By optimizing the first-order and second-order proximity between graph nodes, local correlations and global structure of the entire graph can be preserved. A joint training framework is proposed which embeds all the relational graphs into a shared lower-dimensional latent space.

4.2.1 Bipartite Graph Construction

We form seven bipartite graphs to represent the relations between six types of graph nodes (POI, user, region, rating, time and category) based on the content information provided in the network. All nodes in the graphs need to be represented in a structured form. In this section, we explain how content information in the original dataset is depicted.

- **User-POI Graph.** . User-POI graph captures the interactions between users and POIs in the dataset. An edge is formed when a user visit a certain POI. It's worth mentioning that each edge corresponds to a specific visitation, meaning there can be multiple edges between a user and a POI if the user has visited a place for multiple times. The reason we do not use a weighted edge is because the high variance of edge weights may cause gradient explosion

during optimization. More details will be covered in Section 4.2.3.

- **POI-POI Graph.** POI-POI graph captures the sequential transitions between two POIs in a user's mobile trajectory. Note that a edge is formed only if the visiting time between two sequentially visited POIs is within a certain time interval. Like in the User-POI graph, one edge only corresponds to one transition record between two POIs.

- **POI-Time Graph.** POI-Time graph embeds the time of visit for each POI. The 'Time' attribute is originally represented in a detailed form, which cannot be directly used as node representation. According to dataset exploration, users' periodic mobility exhibits time-of-day and day-of-week effects. In order to encode the temporal cyclic patterns, we introduce a time-indexing strategy, which replaces the detailed temporal information with a timestamp representation. The details of the this indexing method is described as follows:

  (i). In Foursquare dataset, since both time-of-day and day-of-week effects exist, we denote every timestamp as a two-digit number: the first digit stands for weekdays. Monday through Sunday is represented with number 1 to 7. The second digit stands for 6 time-slots in a day, and each time-slot is a 4-hour time interval. For example, 12 am to 4 am is encoded as 0. Under this framework, a week is divided into 42 equal length time-slots.

  (ii). For Google Local dataset, only day-of-week effect shows significant influence. Therefore, the timestamp is simply represented using day of week, for example, 'Monday' and 'Friday'.

  Note that each edge only corresponds to one specific visitation.

- **POI-Region Graph.** The geographical information provided in the original dataset is a combination of coordinates and administrative division, which cannot be directly used for node representations. Location coordinates are too detailed and will add enormous nodes to the network. Use city as the region unit would work fine for small counties, while is not

discriminative for large cities. Therefore, we use postcode as the region unit which has suitable scale.

- **POI-Category Graph** POI-Category graph captures the categorical information of POIs. One POI may belong to multiple categories and there is only one edge formed between a POI and a certain category node.

- **POI-Rating Graph** Google Local dataset contains user ratings and reviews, which can help to make inference about POI properties and a user's preference towards certain POIs. In addition, evaluating rating and review sentiment jointly leads to a more reliable and unbiased assessment. Therefore, we construct the POI-Rating graph to capture the semantic effect of user ratings and reviews jointly. To be specific, we compute the sentiment score for each review through sentiment analysis, then scale it to 0 5. The average of rating and the scaled sentiment score is used as the final 'rating' value.

- **User-User Graph** User-User graph is constructed to capture the social ties between users. Note that each social link is undirected and unweighted, for example, if user u_i and u_j are 'friends', then the edge u_i-u_j is the same as u_j-u_i, and will be counted only once.

To provide a more intuitive illustration of this construction strategy, Figure. 4.1 gives an example of the set of bipartite graphs built from 9 sample records, which is shown in Table 4.1.

4.2.2 Heterogeneous Graph Embedding

A graph is considered as a heterogeneous graph if two end nodes of each edge is of different types. In another word, every edge in the graph is directed.

The location network is inherently a heterogeneous graph, since each POI has different types of attributes. Therefore, we introduce a heterogeneous graph embedding strategy to embed six bipartite graphs in the location network: User-POI graph, POI-Time graph, POI-Region graph,

Table 4.1: Sample Check-in Records

Record	User ID	Friends	Timestamp	Venue ID	Region ID	Category ID
1	u_1	u_2,u_3,u_5	t_1	v_1	r_1	c_1
2	u_1	u_2,u_3,u_5	t_2	v_3	r_3	c_1
3	u_1	u_2,u_3,u_5	t_3	v_4	r_1	c_2
4	u_2	u_3,u_4,u_5	t_1	v_4	r_2	c_1
5	u_3	u_1,u_2,u_4,u_5	t_1	v_2	r_3	c_1
6	u_3	u_1,u_2,u_4,u_5	t_2	v_3	r_2	c_2
7	u_4	u_2,u_3	t_3	v_2	r_3	c_1
8	u_5	u_1,u_2,u_3	t_1	v_4	r_2	c_1
9	u_5	u_1,u_2,u_3	t_2	v_5	r_2	c_2

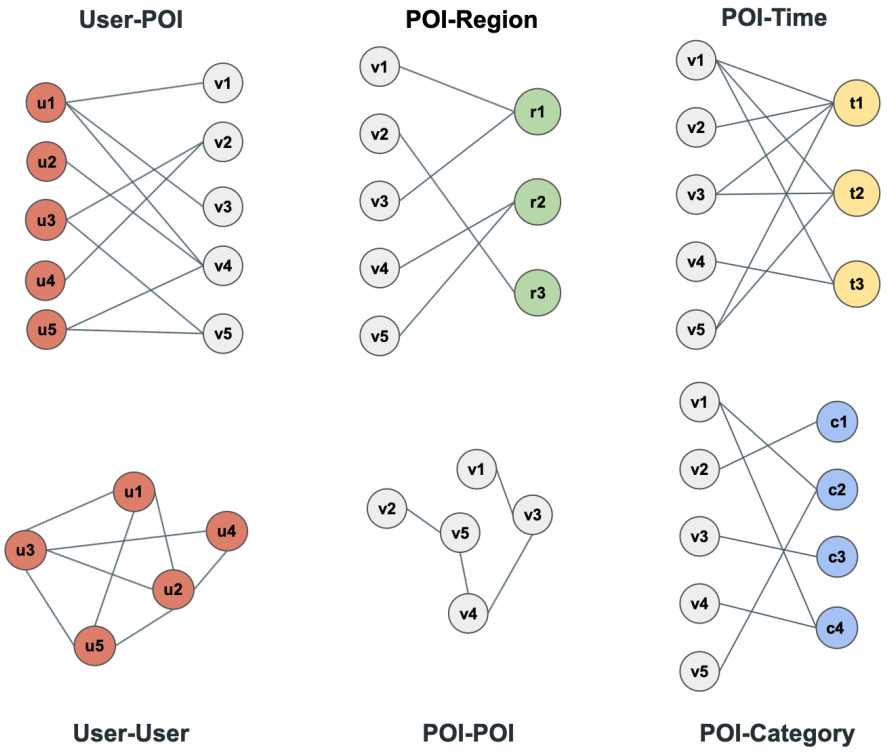


Figure 4.1: Bipartite Graphs Constructed From Sample Records

POI-Rating graph, POI-Category graph and POI-POI graph. Note that the POI-POI graph is heterogeneous, since each edge is directed, pointing from one POI to the sequentially visited POI.

For heterogeneous graph embedding, we only investigate the second-order proximity between nodes. The reason is that there is no point in observing the similarity between two directly connected nodes since they're of different types. Instead, we aim at embedding similar nodes of the same type close to each other in the embedding space, and the closeness is measured by observing the shared neighborhoods. Here, we explain the detailed embedding method:

Given a bipartite graph $G_{AB} = (V_A \cup V_B, \varepsilon)$, where V_A and V_B are two disjoint sets of vertices. Vertex v_i is in V_A and vertex v_j is in V_B. Then the conditional probability of 'context' v_j been generated by v_i is defined as:

$$p_{hetero}(v_j|v_i) = \frac{exp(\vec{v_j}^T \cdot \vec{v_i})}{\sum_{v_k \in V_B} exp(\vec{v_k}^T \cdot \vec{v_i})} \tag{4.1}$$

The empirical distribution is computed as:

$$\tilde{p}_{hetero}(v_j|v_i) = \frac{w_{ij}}{deg_i} \tag{4.2}$$

where deg_i is the degree of vertex v_i, and $deg_i = \sum_{k \in N(i)} w_{ik}$ where $N(i)$ is the set of out-neighbors of v_i .

To preserve the second-order proximity, the conditional distribution needs to be closely approximated to the empirical distribution by minimizing the distance between the two distributions, which is represented by the following objective function:

$$O_{hetero} = \sum_{v_i \in V_A} \lambda_i d(\tilde{p}_{hetero}(\cdot|v_i), p_{hetero}(\cdot|v_i)) \tag{4.3}$$

where $d(\cdot,\cdot)$ is computed using KL-divergence. λ_i is the 'importance' of vertex v_i, which can be represented as the degree deg_i.

Omitting some constants, Equation. 4.3 can be calculated as:

$$O_{hetero} = -\sum_{e_{ij} \in \varepsilon} w_{ij} log p_{hetero}(v_j|v_i) \tag{4.4}$$

We can obtain the embedding vectors of $\vec{v}_i$ and $\vec{v}_j$ by minimizing Equation. 4.4

4.2.3 Homogeneous Graph Embedding

A graph is considered as a homogeneous graph if two end nodes of each edge are of the same type, and every edge in the graph is undirected.

The social network is a homogeneous graph, since User-User edge is undirected and unweighted. For homogeneous graph embedding, we only capture the first-order proximity between nodes. This makes sense for social graph embedding since users with the same neighbors make up only a small fraction of the population. The detailed embedding method is described as follows:

Given a bipartite graph $G_{AB} = (V_A \cup V_B, \varepsilon)$, where V_A and V_B are two disjoint sets of vertices, ε is the set of edges between them. Given a vertex v_i in V_A and a vertex v_j in V_B, the joint distribution between v_i and v_j is defined as:

$$p_{homo}(v_i, v_j) = \frac{1}{1 + exp(-\vec{v}_j^{\,T} \cdot \vec{v}_i)} \tag{4.5}$$

where $\vec{v}_i$, $\vec{v}_j$ are the vector representations of vertices v_i, v_j.

The empirical probability of v_i and v_j is:

$$\tilde{p}_{homo}(v_i, v_j) = \frac{w_{ij}}{W} \tag{4.6}$$

where w_{ij} is the weight between v_i and v_j, and $W = \sum_{e_{ij} \in \varepsilon} w_{ij}$.

To preserve the first-order proximity, the joint distribution needs to be closely approximated to the empirical distribution by minimize the distance between the two distributions, which is represented by the following objective function:

$$O_{homo} = d(\tilde{p}_{homo}(\cdot,\cdot), p_{homo}(\cdot,\cdot)) \tag{4.7}$$

where $d(\cdot,\cdot)$ is computed using KL-divergence. Omitting some constants, we have:

$$O_{homo} = -\sum_{e_{ij}\in\varepsilon} w_{ij} \log p_{homo}(v_j, v_i) \tag{4.8}$$

We can get the embeddings of all vertices in a homogeneous graph by minimizing Equation. 4.8.

4.2.4 Model Training and Optimization

- **Negative Sampling Approach**

  Optimizing objective function Equation. 4.3 is computationally expensive, as calculating the conditional probability $p_{hetero}(\cdot|v_i))$ requires to sum over the entire set of vertices in $\sum_{v_k\in V_B} exp(\vec{v_k}^T \cdot \vec{v_i})$. Facing this problem, negative sampling method is adopted. Given a positive edge (v_i, v_j), several negative edges are sampled according to a specific noise distribution. For each edge (v_i, v_j), the optimization objective is defined as:

$$log\sigma(\vec{v_j}^T \cdot \vec{v_i}) + \sum_{i=1}^{K} E_{v_n\sim p_n(v)}[\log\sigma(-\vec{v_n}^T \cdot \vec{v_i})] \tag{4.9}$$

  where $\sigma(x)$ is the sigmoid function, and K represents the number of negative samples for each edge. The first term is for the observed edge been optimized, and the second term is the set of negative edges sampled from a specific noise distribution. In this work, we set $K = 5$ according to empirical evidence, and choose the noise distribution to be $p_n(v) \propto d_v^{3/4}$

according to literature [53].

Table method [54] is adopted to draw a sample, which reduces the sampling complexity to $O(1)$. Since the number of negative samples for each edge is a constant, for a graph with N edges, the overall time complexity of the sampling procedure is $O(N)$. This ensures that our model shows good scalability.

During training, we adopt the Adaptive Gradient Descent (AdaGrad) for optimization. The gradient with respect to the embedded vector $\vec{v}_i$ in edge (v_i, v_j) can be calculated as:

$$\frac{\partial O}{\partial \vec{v}_i} = w_{ij} \cdot \frac{\partial \log p_{hetero}(v_j|v_i)}{\partial \vec{v}_i} \tag{4.10}$$

As mentioned in the previous section, we give all edges equal weight in order to avoid the gradient explosion problem. The probability of an edge been sampled is proportional to the number of occurrence of that edges in the graph.

- **Joint Training Framework**

The heterogeneous bipartite graphs use the same embedding strategy, therefore can be trained with a joint framework. A simple approach is to embed all the bipartite graphs collectively by minimizing the sum of their objective functions. Here, we use Foursquare dataset as an example. Given five heterogeneous bipartite graphs $G_{uv}, G_{vc}, G_{vt}, G_{vr}, G_{vv}$, the joint objective is defined as:

$$O = O_{uv} + O_{vv} + O_{vc} + O_{vt} + O_{vr} \tag{4.11}$$

where:

$$O_{uv} = -\sum_{e_{ij} \in \varepsilon_{uv}} w_{ij} \log p_{hetero}(v_j|v_i) \tag{4.12}$$

$$O_{vv} = -\sum_{e_{ij} \in \varepsilon_{vv}} w_{ij} \log p_{hetero}(v_j|v_i) \tag{4.13}$$

$$O_{vc} = -\sum_{e_{ij} \in \varepsilon_{vc}} w_{ij} \log p_{hetero}(v_j|v_i) \tag{4.14}$$

$$O_{vt} = -\sum_{e_{ij} \in \varepsilon_{vt}} w_{ij} \log p_{hetero}(v_j|v_i) \tag{4.15}$$

$$O_{vr} = -\sum_{e_{ij} \in \varepsilon_{vr}} w_{ij} \log p_{hetero}(v_j|v_i) \tag{4.16}$$

The objective function Equation. 4.11 is optimized by merging five types of edges ($\varepsilon_{uv}, \varepsilon_{vv}$, $\varepsilon_{vc}, \varepsilon_{vt}, \varepsilon_{vg}$) then train them together. During updating, a positive edge is sampled according to its 'weight' in its group, which is proportional to the number of occurrence. Note that the negative edge needs to be sampled from the same group where the positive sample belongs. In light of the constraints, we adopt alternative updating to train each graph alternatively.

Before we jointly train the heterogeneous location network, we first train the homogeneous social network (User-User graph), then use the obtained user embeddings as the initialization of user vectors $\vec{u}$ for the joint training of the other bipartite graphs. A detailed description of the training algorithm is illustrated in **Algorithm. 1**.

4.3 Context-aware Prediction Framework with Dynamic Tracking of User Preference

When predicting the next POI a user would visit, the context information need to be considered to facilitate the prediction, which includes time and user's current location.

Concretely, denote user, timestamp and region as u, t, r, given a query $q = (u, t, r)$, for each candidate POI v, its ranking score can be computed as:

$$S(q, v) = \vec{u_\tau}^T \cdot \vec{v} + \vec{r}^T \cdot \vec{v} + \vec{t}^T \cdot \vec{v} \tag{4.17}$$

Algorithm 1 Training Algorithm

Input: Bipartite graphs $G_{uu}, G_{uv}, G_{vv}, G_{vc}, G_{vt}, G_{vr}$, vector dimension d, number of samples N, number of negative samples K

Output: user embeddings $\vec{u}$, POI embeddings $\vec{v}$, category embeddings $\vec{c}$, timestamp embeddings $\vec{t}$, region embeddings $\vec{r}$.

procedure HOMOGENEOUS GRAPH EMBEDDING
    Initialize $\vec{u}$
    while $iter \leq N$ **do**
        sample an edge from ε_{uu}, and update user embeddings.
    return $\vec{u}$ ▷ embeddings of user
procedure HETEROGENEOUS GRAPH EMBEDDING
    Initialize $\vec{u}$ with outputs of the previous procedure, initialize $\vec{v}, \vec{c}, \vec{t}, \vec{r}$
    while $iter \leq N$ **do**
        sample an edge from ε_{uv}, update user and POI embeddings with K negative edges.
        sample an edge from ε_{vv}, update POI embeddings with K negative edges.
        sample an edge from ε_{vc}, update POI and category embeddings with K negative edges.
        sample an edge from ε_{vt}, update POI and time-slot embeddings with K negative edges.
        sample an edge from ε_{vr}, update POI and region embeddings with K negative edges.
    return $\vec{u}, \vec{v}, \vec{c}, \vec{t}, \vec{r}$ ▷ embeddings of user, POI, category, timestamp, and region

Instead of directly use the user embeddings obtained from joint training, we introduce a special 'embedding' strategy to get user representation $\vec{u_\tau}$ that can dynamically model user's latest preference without retrain the whole model when new history is added. Specifically, we consider each user as a collection of POIs from his/her mobile trajectory before time τ. Each POI in the visiting history is multiplied with a time-decaying factor in the exponential form. The intuition behind this method is that the latest activity is supposed to have the largest impact on user's current preference, while visitations from long time ago won't have too much influence.

Under this setting, $\vec{u_\tau}$ is computed as:

$$\vec{u_\tau}^T = \sum_{(u,v_i,t_i)\in D_u \cap (t_i<\tau)} exp^{-(\tau - t_i)} \cdot \vec{v_i} \tag{4.18}$$

where D_u is the collection of visiting records associated with user u, where each record contains POI v_i and visit time t_i. Note that t_i and τ are both represented with unix timestamp.

This user embedding method resembles the dynamics of user preference. More importantly, it allows us to update user preference online in linear time complexity without retraining the entire model.

Chapter 5

Experiments and Results

In this chapter, we first introduce the comparison models and basic experiment settings, then report the experimental results of three predictive tasks: next-POI prediction, cold-start recommendation and social link prediction. We conduct model comparisons and explore the impact of influential factors.

5.1 Datasets

We conduct extensive experiments on two real-world large-scale LBSN datasets, including Foursquare and Google Local. A detailed exploration on the dataset statistics and characteristics is presented in Chapter 3.

5.2 Comparison Models

We conduct experiments to compare our method with four baselines, which are representative models of Probability-based, Factorization-based and Deep Learning-based methods.

- **Naive Bayes Classifier.** NBC is a probability-based predictor, which makes predictions based on the assumption that all the features are independent. In next-POI prediction task,

given a query $q = (u, t, r)$, the probability that user u will visit v at given time t and region r is computed as:

$$p(v|u,t,r) \propto p(u,t,r|v) \cdot p(v) \tag{5.1}$$

NBC is a strong baseline in many problems. In addition, in next-POI prediction task, NBC can realize cold-start recommendation by incorporating content attributes of the cold-start POIs as auxiliary information, and using contextual features to facilitate prediction.

- **SVDFeature.** SVDFeature [55] is a machine learning toolkit designed to solve feature-based Matrix Factorization. SVD is a classic Factorization-based method which represents the interactions between nodes with a matrix. The node embeddings are obtained through Matrix Factorization. SVDFeature is an improved version of traditional SVD in that nodes can have both local features and global features. The model objective is defined as:

$$y = \mu + (\sum_j b_j^{(g)} \gamma_j + \sum_j b_j^{(u)} \alpha_j + \sum_j b_j^{(i)} \beta_j) + (\sum_j p_j \alpha_j)^T (\sum_j q_j \beta_j) \tag{5.2}$$

where α, β, γ are user feature, item feature and global feature respectively.

The final model function is:

$$\hat{r} = f(y) \tag{5.3}$$

$$Loss = L(\hat{r}, r) + regularization \tag{5.4}$$

where f is the activation function and L is the loss function.

In next-POI prediction task, temporal factor is considered as global feature, user social links are user local feature, POI category, POI region and POI rating are considered as POI local features.

In addition to POI prediction task, SVDFeature can also be used for friendship recommendation.

- **GRU.** GRU [56] is an improved version of standard Recurrent Neural Network which uses update gate and reset gate to address the vanishing gradient problem, and can capture long-term sequential context. Since it has a long-term dependency for locations such as periodical visiting behaviors, GRU is suitable for modeling user mobility in LBSNs and solving time-aware POI prediction problem based on visiting history.

  Given a visiting trajectory $v_1, v_2, ..., v_m$, denote a state after visiting the t-th location as $C_t \in \mathbb{R}^d$, denote the corresponding long-term representation as $h_t = tanh(C_t)$, the update rule is defined as:

$$\widetilde{C}_t = tanh(W_{c_1} U_{v_t} + W_{c_2} h_{t-1} + b_c) \tag{5.5}$$

  where $W_{c_1}, W_{c_2} \in \mathbb{R}^{d \times d}$ are model parameters, U_{v_t} is the embedding of v_t, b_c is the bias term. C_t is obtained through input and forget gates:

$$C_t = i_t * \widetilde{C}_t + f_t * \widetilde{C}_{t-1} \tag{5.6}$$

  where $i_t, f_t \in \mathbb{R}^d$ stand for the input and forget gate, and are computed by:

$$i_t = \sigma(W_{i_1} U_{v_t} + W_{i_2} h_{t-1} + b_i) \tag{5.7}$$

$$f_t = \sigma(W_{f_1} U_{v_t} + W_{f_2} h_{t-1} + b_f) \tag{5.8}$$

- **TransFM.** TransFM [36] adapts ideas from Factorization Machine into translation-based sequential recommenders. Specifically, it replace the inner product in the FM interaction terms with squared Euclidean distance to measure the interaction strength between features.

An embedding vector and a translation vector are learned for each feature dimension. The modification allows it to incorporate arbitrary real-valued features for the purpose of sequential recommendation. The model equation is given by:

$$\hat{y}(\vec{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} d^2(\vec{v}_i + \vec{v}'_i, \vec{v}_j) x_i x_j \tag{5.9}$$

where $\vec{v}_i, \vec{v}'_i$ are the embedding vector and translation vector for feature x_i respectively. w_0 is the global bias term, and w_i is the linear term for feature x_i. $d^2(\cdot, \cdot)$ stands for squared Euclidean distance.

TransFM has been proved to outperform many FM-based models therefore can serve as a strong baseline.

- **JGEL.** Joint Graph Embedding for LBSNs. The model proposed in this work. Relational graphs are constructed to capture multiple effects in the user and POI attribute networks, then embedded into a shared latent vector space using a joint framework. An online embedding strategy is introduced to dynamically model user's latest preference without retrain the whole model when new history is added.

5.3 Evaluation Methodology

We sort the collection of visiting records associated with each user according to time. The latest record are used for testing, the second latest record contributes to the validation set for hyper-parameter tuning, and the rest of the records are used for model training.

To evaluate the prediction performance during model comparisons, we report the Area Under the ROC Curve (AUC) of each model, which is defined as the probability that a randomly sampled positive observation has a predicted probability greater than a randomly sampled negative observation. Under our problem settings, the AUC can be computed as:

$$AUC = \frac{1}{|U|} \sum_{u \in U} \frac{1}{|V \setminus S^u|} \sum_{v' \in V \setminus S^u} 1(R_{u,g_u} < R_{u,v'}) \tag{5.10}$$

where U is the entire user set, V is the entire POI set. g_u is the ground truth POI of user u in the test set, and $S^u = \{g_u\}$. $R_{u,v}$ stands for the rank of POI v in the prediction output list of user u. $1(\cdot)$ is the indicator function which returns 1 if the ground truth POI is ranked ahead of a unobserved POI v'.

However, a high AUC value only means that positive items are ranked relatively high. When the number of items is sufficiently large, an item doesn't necessarily need to be ranked among the very top to achieve a high AUC value. For example, suppose we have 100,000 items, a positive item ranked at the 100th place still achieve an AUC as high as 99.9%. This metric works fine for overall model evaluation, however, for real-world recommenders, only items ranked at the very top can be seen by users. In addition, the results of different models can be very close therefore hard to be used to differentiate model performance.

In light of these problems, we also adopt $Accuracy@k$ [57][52] when investigating Foursquare dataset since the number of POIs it contains is rather large. This metric is defined as:

$$Accuracy@k = \frac{\#hit@k}{|D|} \tag{5.11}$$

where D is the set for testing, and $|D|$ denotes the number of total test cases. For each test case, we compute the prediction probability of all candidate POIs and rank them in a descending order. If the ground truth POI appears in the top k ranked POIs, then it is considered as a $hit@k$. $Accuracy@k$ is defined as the number of $hit@k$ throughout the entire test set.

5.4 Sensitivity of Model Parameters

Hyper-parameter tuning is important in order for a model to achieve the optimal performance. The hyper-parameters in the proposed model include embedding dimensions, number of samples and granularity of temporal pattern. In this section, we implement sensitivity analysis on these model parameters to find the optimal parameter settings.

5.4.1 Embedding Dimension & Number of Samples

Table. 5.1 shows the model performance in terms of $Accuracy@15$ with different embedding dimensions and number of samples drawn during joint training. A more intuitive illustration is shown in Figure. 5.1a.

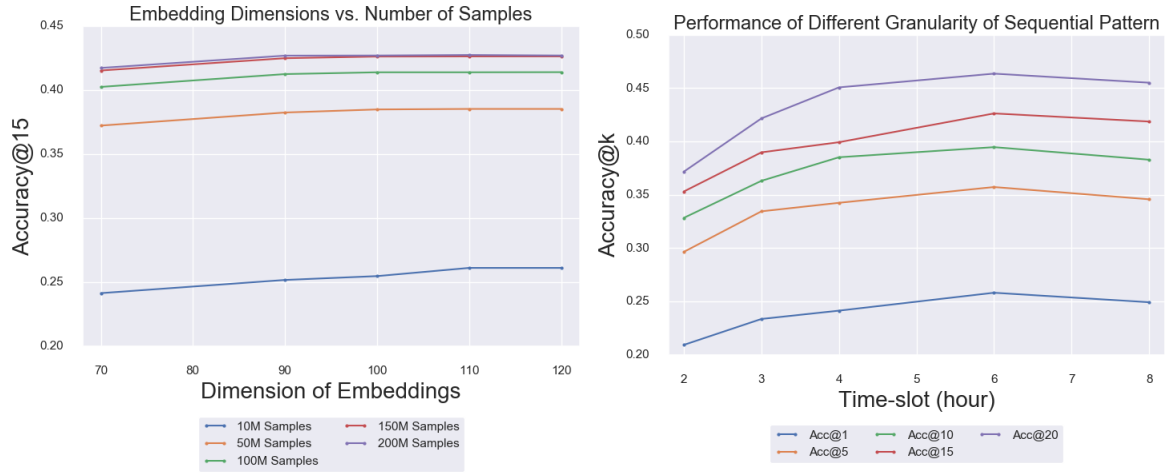
Table 5.1: Embedding Dimension VS. Number of Samples

	Embedding Dimensions d				
Number of Samples N (millions)	70	90	100	110	120
10M	0.2414	0.2517	0.2547	0.2611	0.2611
50M	0.3722	0.3824	0.3848	0.3852	0.3852
100M	0.4024	0.4124	0.4138	0.4138	0.4139
150M	0.4152	0.4248	**0.4261**	0.4262	0.4262
200M	0.4162	0.4254	0.4263	0.4263	0.4263

According to the results, the model is not very sensitive to embedding dimensions. The accuracy curves slightly go up as the number of dimensions increases and plateau at around 100 dimensions. As for the number of samples, the prediction accuracy increases dramatically as the amount of total samples goes from 10 millions to 100 millions, then converges at 150 millions.

In order to achieve the optimal performance and being computationally effective at the same time, we used dimensions $d = 100$ and the number of total samples $N = 150M$ for the following experiments.

On Google Local Dataset, we go through the same process, and find that setting dimensions



(a) Embedding Dimensions vs. Number of Samples

(b) Performance of Different Granularity of Sequential Pattern

Figure 5.1: Sensitivity Analysis of Model Parameters

$d = 100$ and the number of total samples $N = 100M$ gives the best performance.

5.4.2 Granularity of Sequential Pattern

On Foursquare Dataset, in order to find the suitable time interval that captures the sequential transition patterns between two sequentially visited POIs, we conduct experiments with different time intervals ΔT. A POI-POI edge is formed if only the transition time t between the two POIs satisfies $t < \Delta T$. The results are shown in Table. 5.2 and further illustrated in Figure. 5.1b.

Table 5.2: Prediction Accuracy with Different Time Intervals

Time Interval (hour)	Acc@1	Acc@5	Acc@10	Acc@15	Acc@20
2	0.2091	0.2963	0.3282	0.3529	0.3715
3	0.2334	0.3343	0.3629	0.3896	0.4214
4	0.2412	0.3423	0.3850	0.3901	0.4505
6	**0.2582**	**0.3571**	**0.3945**	**0.4261**	**0.4634**
8	0.2491	0.3456	0.3827	0.4185	0.4549

According to the result, as the time interval gets larger, the prediction accuracy first goes

up, then drops after surpassing the 6-hour time-slot. This makes sense since the time people stay at one place can be long, when the time interval is set to be very short, many valuable records will be dropped out therefore the model is unable to capture user's behavior patterns well. On the other hand, a large time interval will add noise since the two sequentially visited venues aren't essentially related. In conclusion, the 6-hour time-slot turns out to be an optimal temporal granularity to capture sequential effect in user mobility.

5.5 Performance of Context-aware Next-POI Prediction

In this section, we explore our model performance on context-aware next-POI prediction task. We first study the impact of influential factors, then conduct experiments to compare the performance of our method and four representative recommenders. Finally, we investigate the effectiveness of our prediction method with dynamic tracking of user preference.

5.5.1 Impact of Influential Factors

As mentioned in the previous chapter, we adopt bipartite graph modeling and construct six relational graphs $G_{uu}, G_{vv}, G_{vc}, G_{vt}, G_{vr}, G_{vs}$ to capture the friendship influence, sequential effect, categorical effect, temporal cyclic effect, geographical influence and semantic effect exhibit in user mobility. To examine their individual impact on our model performance of next-POI prediction, we design two experimental settings when studying one specific effect:

i. Remove the corresponding relational graph and investigate the prediction accuracy without considering this specific effect.

ii. Only keep the corresponding relational graph for evaluation to eliminate the joint effect resulting from the other graphs.

On Foursquare dataset, we study five influential factors which affect user mobility modeling: friendship influence (User-User graph), sequential effect (POI-POI graph), categorical effect

Table 5.3: Impact of Individual Graph on Next-POI Prediction in Foursquare

Type	Graph	Acc@1	Acc@5	Acc@10	Acc@15	Acc@20
Remove One Graph	User-User	0.2312	0.3271	0.3723	0.4092	0.4529
	POI-POI	**0.1963**	**0.2501**	**0.3015**	**0.3403**	**0.3801**
	POI-Category	0.2402	0.3150	0.3604	0.4034	0.4432
	POI-Time	0.2321	0.2871	0.3353	0.3802	0.4275
	POI-Region	0.2506	0.3277	0.3791	0.4184	0.4628
	All	0.2584	0.3356	0.3835	0.4345	0.4663
Keep One Graph	User-User	0.0091	0.0124	0.0131	0.0359	0.0391
	POI-POI	**0.0471**	**0.0935**	**0.1129**	**0.1485**	**0.1637**
	POI-Category	0.0082	0.0114	0.0115	0.0253	0.0293
	POI-Time	0.0194	0.0324	0.0471	0.0515	0.0598
	POI-Region	0.0273	0.0598	0.0799	0.1067	0.1211

(POI-category graph), temporal cyclic effect (POI-Time graph), and geographical influence (POI-Region graph). All models are trained with embedding dimensions $d = 100$, number of samples $N = 150M$ and number of negative samples for each sampled edge $K = 5$. The temporal granularity is set as $\Delta T = 6h$. The experimental results are shown in Table. 5.3.

For better illustration, we present the experimental results using a bar chart, which is shown in Figure. 5.2. The bars built from the bottom represent the prediction accuracy with only one effect considered (models trained with one individual graph). The bars built from the top represent the prediction accuracy trained with one relational graph eliminated, except for the first bar which is trained with all graphs as a reference. The results in $Accuracy@k, k \in \{1, 5, 10, 15, 20\}$ are represented with different shades of color. We also label the results in $Accuracy@20$ and fit the lines for easier comparison.

According to the diagram, the observations made from the results of two experimental settings are generally consistent.

i. First of all, all influential factors can facilitate next-POI prediction to some extent. Eliminating any relational graph will cause performance degradation.

ii. Sequential effect (POI-POI graph) has the greatest impact on the model performance.

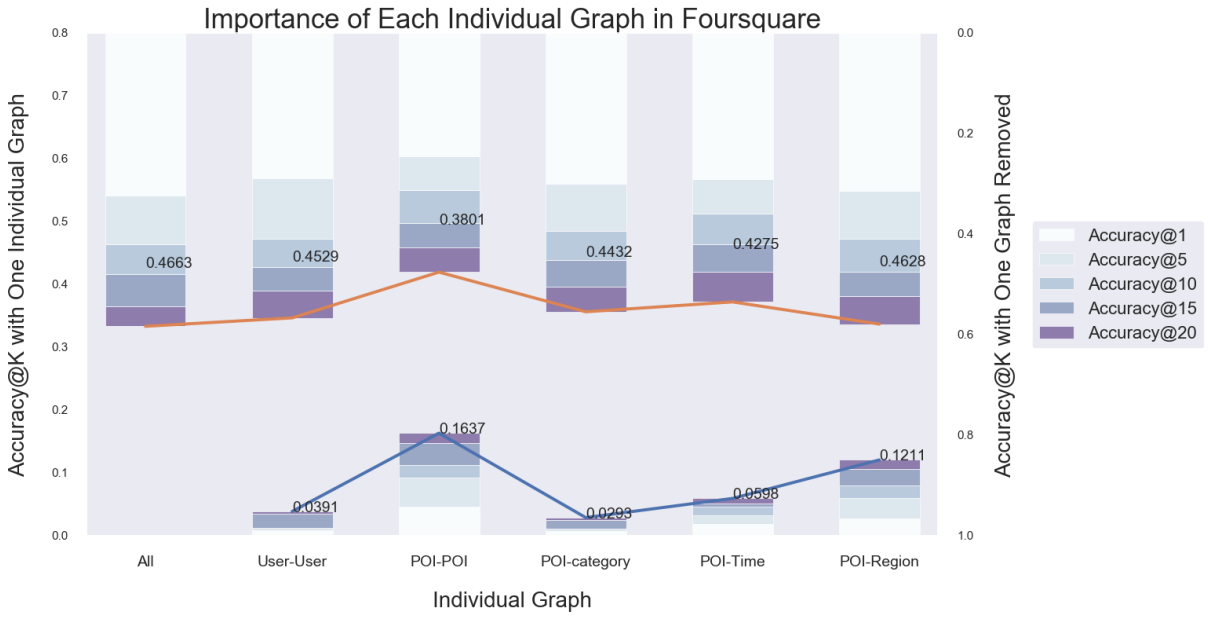


Figure 5.2: Analysis on the Impact of Individual Factors

It gives the best result when training with only one graph, and it causes the largest performance degradation if eliminated from the framework. Temporal cyclic effect is the second important factor, followed by categorical effect (POI-Category graph) and friendship influence (User-User graph).

iii. Finally, the results about geographical influence (POI-Region graph) are contradictory. Possible reason for the inconsistency is that geographical information can largely facilitate prediction with its individual impact, while the influence can be compensated by other factors.

On Google Local dataset, we conduct the same sets of experiments to investigate four influential factors exhibit in the dataset: categorical effect (POI-category graph), temporal cyclic effect (POI-Time graph), geographical influence (POI-Region graph) and semantic effect (POI-Rating graph). All models are trained with embedding dimensions $d = 100$, number of samples $N = 100M$ and number of negative samples for each sampled edge $K = 5$. The results are shown in Table. 5.4.

According to the results:

i. semantic effect turns out to have the most significant impact, temporal cyclic patterns

Table 5.4: Impact of Individual Graph on Next-POI Prediction in Google Local

Type	Graph	CA	TX	FL	CO	WA	NC
Remove One Graph	POI-Category	0.8521	0.8362	0.8360	0.8930	0.9065	0.8663
	POI-Time	0.6505	0.8253	0.6621	0.6578	0.8288	0.7988
	POI-Region	0.6589	0.8714	0.7829	0.8460	0.8430	0.8787
	POI-Rating	**0.5882**	**0.7453**	**0.6014**	**0.6238**	**0.7736**	**0.7534**
	All	0.7032	0.9091	0.8220	0.8771	0.8993	0.9088
Keep One Graph	POI-Category	0.5134	0.6553	0.5625	0.5358	0.5299	0.6526
	POI-Time	0.5603	0.8584	0.7898	0.6265	0.6158	0.7328
	POI-Region	0.5774	0.8111	0.8050	0.6295	0.6321	0.7364
	POI-Rating	**0.6059**	**0.8718**	**0.8132**	**0.6624**	**0.6480**	**0.7591**

and geographical influence have similar levels of impact.

ii. Categorical information exhibits an opposite effect on next-POI prediction. This might due to the large number of different categories and their sparse distributions. A further investigation on data statistics shows that, nearly 50% of categories have no more than three POIs.

Through model evaluations on two datasets, we draw several **conclusions** about the impact of influential factors:

i. Sequential effect is a distinctive yet dominant factor in next-POI prediction problem in LBSNs. People's choice on the next place to visit is highly correlated with the latest check-in spot.

ii. Semantic effect from user feedback is an important factor in inferring the next-POI a user will visit. User feedback such as ratings and reviews directly reveal user preference and their potential of visiting a venue again. In addition, it reflects POI popularity as well.

iii. The model performance may degenerate due to the sparsity of network nodes. Constructing feature hierarchy to obtain suitable representations with the right scale is a non-trivial task.

5.5.2 Comparative Results

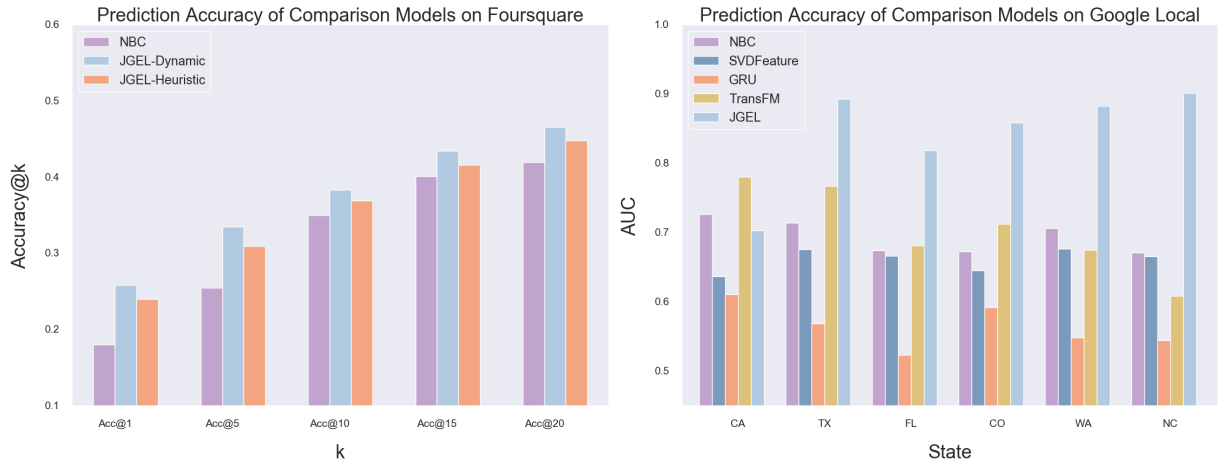
We compare our method with four representative models from Probability-based, Factorization-based and Deep Learning-based methods. Our model parameters when studying Foursquare dataset are set as: embedding dimensions $d = 100$, number of samples $N = 150M$ and number of negative samples for each sampled edge $K = 5$. The temporal granularity is set as $\Delta T = 6h$. The model parameters for Google Local Dataset are set as: embedding dimensions $d = 100$, number of samples $N = 100M$ and number of negative samples for each sampled edge $K = 5$. The comparison of results are shown in Table. 5.5

Table 5.5: Prediction Accuracy of Comparison Models

Model	Foursquare	CA	TX	FL	CO	WA	NC
NBC	0.9906	0.7269	0.7145	0.6747	0.6733	0.7068	0.6715
SVDFeature	0.8673	0.6371	0.6764	0.6668	0.6456	0.6771	0.6661
GRU	0.8320	0.6115	0.5691	0.5236	0.5921	0.5486	0.5446
TransFM	0.9172	**0.7808**	0.7673	0.6814	0.7131	0.6750	0.6086
JGEL-Dynamic	**0.9989**	0.7032	**0.8931**	**0.8193**	**0.8589**	**0.8834**	**0.9017**
JGEL-Heuristic	0.9947	0.7751	0.8832	0.7893	0.8127	0.8548	0.8425

According to the results, our method (JGEL-Dynamic) generally outperforms the other comparison models in terms of prediction AUC. Since the AUC of NBC and our method on Foursquare are pretty close, we further investigate their performance using $Accuracy@K$ as the evaluation metric. The comparative results are illustrated with bar plots, which are shown in Figure. 5.3a and Figure. 5.3b for Foursquare and Google Local respectively. A detailed comparison of the models is as follows:

- **JGEL v.s. TransFM.** TransFM shows comparative results with our method on Foursquare, and even surpasses our method on Google Local California. The result proves the advantage of metric learning and incorporating multiple factors to facilitate recommendation. However, our method outperforms TransFM for most cases, which indicates that our embedding



(a) Prediction Accuracy of Comparison Models on Foursquare

(b) Prediction Accuracy of Comparison Models on Google Local

Figure 5.3: Prediction Accuracy of Comparison Models

method is better at exploiting the side information with a network structure. Possible reason is that our method embeds all factors (relational graphs) into a shared latent space, which is able to capture the mutual effects between them, while TransFM uses two spaces to embed features and sequential transitions between visitations separately. In our method, not only the potential correlations among all the factors can be captured, the global structure is also preserved. However, our method may be more sensitive to network sparsity, which explains for the degradation of performance on Google Local.

- **JGEL v.s. GRU.** Our method significantly outperforms GRU on both datasets. The reason might be that GRU is good at capturing users' long-term periodic mobility, while fails to observe the potential correlations among other useful side information. In addition, in next-POI prediction problem, besides from visiting history, the prediction context also plays an important role, which GRU fails to consider.

- **JGEL v.s. SVDFeature.** Our method surpasses SVDFeature by a large amount on both datasets. The results prove the superiority of: (1). using a graph structure over the adjacency matrix to capture user-POI interactions and the correlations among different factors (2).

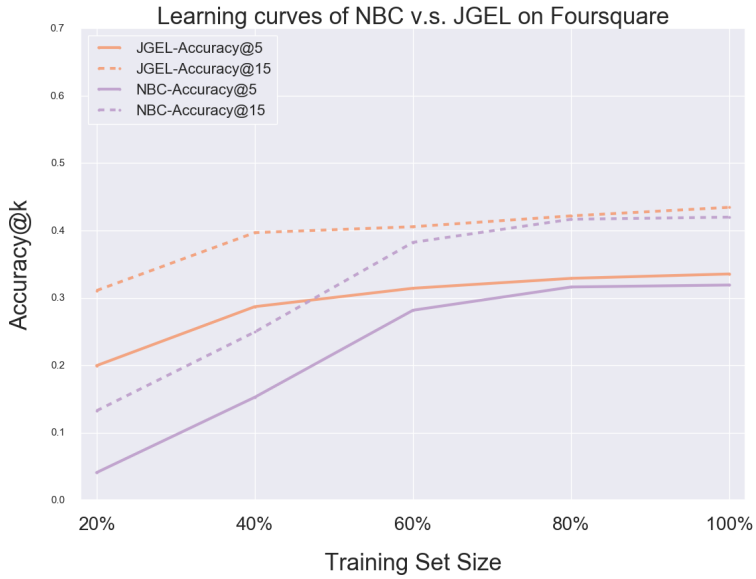


Figure 5.4: Learning curves of NBC v.s. JGEL on Foursquare

embedding users and POIs into a shared latent space with a unified model.

- **JGEL v.s. NBC.** NBC achieves comparable performance with our method on Foursquare, but our method outperforms NBC significantly on Google Local. Further investigation on Foursquare dataset using $Accuracy@k$ as the evaluation metric proves the superiority of our model. Although NBC receives comparable result at $Accuracy@20$, our model outperforms NBC by a significant amount in terms of top-ranking accuracy.

  In addition to prediction accuracy, our model shows better prediction effectiveness in that it takes fewer training samples to converge. According to the learning curve shown in Figure. 5.4, although NBC achieves comparable results eventually, the classifier relies on a large amount of observed samples in order to approximate the real-world feature distribution. This indicates that NBC is more sensitive to data sparsity, and also explains for its poor performance on Google Local dataset, which is much sparser compared with Foursquare.

5.5.3 Performance Analysis on Dynamic Preference Tracking

In Chapter 4, we introduce a context-aware prediction framework which can dynamically track user preference. Aside from its superiority in dynamic preference tracking, we also want to test its prediction accuracy compared with heuristic prediction framework.

We define the heuristic prediction method of computing the ranking score as:

$$S(q,v) = \vec{u}^T \cdot \vec{v} + \vec{r}^T \cdot \vec{v} + \vec{t}^T \cdot \vec{v} \tag{5.12}$$

where $\vec{u}$ is directly obtained from the outputs of joint training (Algorithm. 1). We compare the prediction accuracy using the two frameworks, which is reported in Table. 5.5. Results of further investigation on Foursquare using $Accuracy@k$ are illustrated in Figure. 5.3a.

According to the result, computing the prediction ranking score using user embeddings constructed with dynamic tracking strategy achieves higher accuracy, compared with using embeddings obtained from joint training. This observation indicates that user embeddings constructed with mobile trajectories is more representative for prediction purpose. Possible reason is that since each user embedding is a time-aware combination of history visits, the first term in the prediction Equation. 4.17 is actually measuring the similarity between visited POIs and the predicted candidates. Therefore the dynamic representation can better assess the compatibility between user and the candidate POI.

In conclusion, using the dynamic tracking method for user vector embedding not only shows superiority in online training, but also gives better accuracy in next-POI prediction task. Moreover, since user embedding is computed with linear time complexity, the prediction framework has the nice property of making fast updates when new visiting records are added, which makes it scalable to large datasets.

Table 5.6: Prediction Performance on Cold-Start POIs

Model	Foursquare	CA	TX	FL	CO	WA	NC
NBC	0.6632	0.6062	0.6846	0.6208	0.6198	0.6267	0.6996
SVDFeature-r	0.6603	0.5091	0.5236	0.6013	0.5166	0.5129	0.4993
SVDFeature-c	0.6485	0.5328	0.5605	0.6094	0.5689	0.5165	0.6041
SVDFeature-both	0.6787	0.6544	0.7518	0.6233	0.6324	0.7267	0.7892
JGEL-r	0.6412	0.5332	0.5624	0.6152	0.5253	0.5281	0.5145
JGEL-c	0.5527	0.6031	0.6625	0.6157	0.5875	0.5353	0.6327
JGEL-both	**0.6884**	**0.6836**	**0.8481**	**0.6253**	**0.6574**	**0.7838**	**0.8491**

5.6 Performance of Cold-Start Recommendation

In this section, we evaluate our model effectiveness in recommending cold-start POIs. By incorporating content information of the cold-start POIs in joint training, our model can learn the embeddings of cold-start POIs without additional feature engineering or model modifications. To be specific, we add the categorical and geographical information of the cold-start POIs to POI-category and POI-region relational graphs. In this way, the cold-start POIs are embedded into the same latent vector space along with other network nodes, therefore can capture the potential correlations with other network nodes.

TransFM and GRU won't work for cold-start scenario in next-POI recommendation since Factorization-based and Neural Network-based models rely on observing user-POI interactions, while cold-start POIs do not have records of history interactions. NBC handles the cold-start problem by incorporating geographical information about the cold-start POIs. Categorical information has no impact since it is not provided in the prediction context while NBC makes predictions based on the posterior probability of observed features. SVDFeature can make use of both geographical and categorical auxiliary information of the cold-start POIs by taking them as 'item features'.

Therefore, we compare our model performance on cold-start recommendations with NBC and SVDFeature. For SVDFeature and our model, we further investigate the impact of two auxiliary information: category and region. '$-c$' represents models incorporating categorical

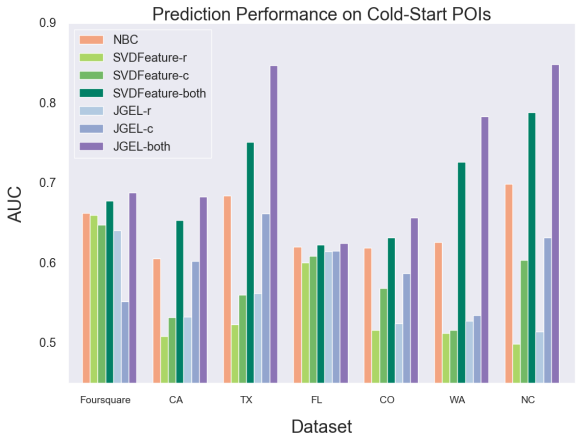


Figure 5.5: Performance on Cold-Start POIs

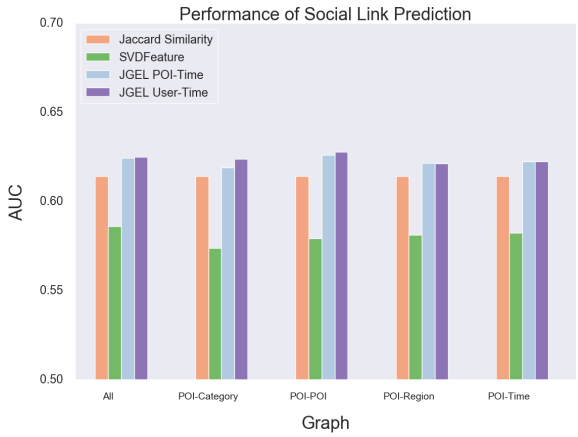


Figure 5.6: Performance of Social Link Prediction

information of cold-start POIs, and '$-r$' represents models incorporating geographical information. The experimental results are shown in Table. 5.6 and illustrated in Figure. 5.5.

There are several observations obtained from the results:

i. All models achieve better than random results, which proves their ability of handling cold-start situations.

ii. Our method outperforms other comparison models in cold-start recommendations on both datasets.

iii. Both geographical and categorical information can facilitate cold-start recommendations. Geographical information plays a more important role in cold-start recommendations on Foursquare. Incorporating only category information doesn't improve the performance too much from a random recommender. However, categorical information turns out to have a much larger impact on Google Local dataset for most of the states. Possible reason is that the category distribution in Google Local is more dispersed, so that more specific correlations can be captured which leads to more accurate predictions.

iv. NBC makes better use of geographical information and achieves the best predictions performance when it is the only side information being used.

Table 5.7: Performance of Social Link Prediction

Model	All features	w/o category	w/o sequential	w/o region	w/o time
JGEL POI-Time	0.6244	0.6192	0.6262	**0.6218**	**0.6225**
JGEL User-Time	**0.6252**	**0.6241**	**0.6280**	0.6213	**0.6225**
SVDFeature	0.5862	0.5739	0.5794	0.5813	0.5825
Jaccard Similarity			0.6143		

5.7 Performance of Social Link Prediction and Friend Recommendation

In this section, we test our model performance in discovering potential friends in the social network on Foursquare dataset. For each user, we rank his or her social closeness with the other users, the top-ranking users are considered to have higher probability of becoming friends.

Specifically, we train our model without incorporating User-User relational graph, then compute users' pairwise similarity using embeddings obtained from joint training. Prediction AUC is reported, where the similarity measurement works as the ranking score over all users in the dataset.

In the original settings, we build POI-Time graph which captures the temporal cyclic patterns of user mobility on the POI side. However, when evaluating the proximity between user embeddings, observing the temporal correlations on the user side may lead to better performance. Therefore, we propose a modified version of JGEL, which uses User-Time graph instead of POI-Time graph to capture temporal effect.

We compare our models with SVDFeature and a heuristic method based on Jaccard similarity (defined in Equation. 3.1). The results are reported in Table. 5.7. A illustration of the comparison results is shown in Figure. 5.6.

Results show that:

i. Use embeddings obtained from our method for social link prediction gives better

performance compared with SVDFeature and the heuristic method based on Jaccard Similarity.

ii. Use User-Time graph instead of POI-Time graph to capture temporal cyclic effect facilitates social link prediction. This further proves that using a user-focused representation instead of the POI-focused representation to capture temporal preference is more effective when inferring social closeness based on mobility similarity.

iii. All factors facilitate friends recommendation except for the POI sequential effect (POI-POI graph), which exhibits an opposite impact. The reason might due to that this graph captures POI transition patterns, which has no impact on user preference modeling but only adds noise to the user-focused social link prediction task. Among the other factors, categorical and geographical influence turn out to have relatively significant impact, while temporal cyclic effect shows the least influence. The result makes sense since friends tend to enjoy similar types of activities, and people who live close to each other are more likely to become friends. What's more, these two factors are discriminating in terms of forming social circles, while users' mobility generally exhibit similar temporal cyclic patterns therefore has less contribution to personal preference modeling.

5.8 Visualization of Embeddings

In this section, we present the visualization of POI embeddings learned on Foursquare. We especially want to examine whether POIs belong to the same group are embedded close to each other in the embedding space.

Specifically, we examine how categorical and temporal factors are captured in the embeddings. We use t-SNE [58] to project the 100-dimensional vector representations into 2D. Different colors are assigned to the POI embeddings according to their attributes. For temporal factor, we examine the 'day and night' effect, the visualization is shown in Figure. 5.7. For categorical effect, we illustrate seven most popular activity categories which takes up above 70% of the records. The

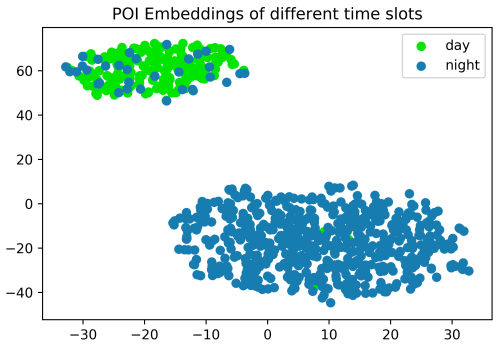


Figure 5.7: Visualization of POI Embeddings of Different Time Slots

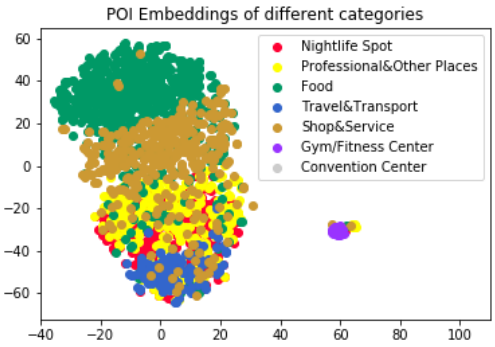


Figure 5.8: Visualization of POI Embeddings of Different Categories

visualization is shown in Figure. 5.8.

According to the illustrations, we can clearly recognize the 'day' and 'night' groups and different category groups, which proves the model's effectiveness. In addition, we can see that the POI embeddings are better clustered according to 'day' and 'night' time slots, compared with categories. This observation is consistent with the experimental results that temporal effect shows a larger influence than categorical effect.

Chapter 6

Conclusion

6.1 Discussion

In this work, we investigate the effectiveness of a graph embedding-based recommender used in LBSNs.

Through studying the impact of each individual factor, we get some interesting observations of how different contextual information facilitates the modeling of user mobility. The results also provide us with some insights on how to build an effective recommender system.

In dense LBSNs where sequential effect of POI transitions is evident, considering the sequential visiting patterns turns out to be the key to inferring user's next visiting place. The sequential effect directly captures transitions between POIs which forms the mobile trajectory. In addition, it also encodes the spatiotemporal context. Intuitively, user's previous location and check-in time put geographical and temporal constraints on the next place to visit. Analytically, geographical and temporal information are all POI attributes in the location network, and are encoded into the POI embeddings through our unified framework. This also explains why geographical influence shows the most significant individual impact, while eliminating it from the whole graph won't cause evident performance degradation. Another observation is that user

feedback is informative and deserves careful inspection. It not only works as a semantic indicator of user's personal preference, but also affects other users' choice by presenting the visiting experience. This effect is also very evident in traditional recommenders about products and movies, and is where POI recommendation connects with the traditional recommendation tasks. The ill performance when incorporating categorical information in Google Local dataset reminds us of the opposite effect may caused by feature sparsity. This finding stress the importance of constructing feature hierarchy to obtain suitable representations with the right scale.

We further show that in POI recommendations, incorporating auxiliary information helps to capture connections between unobserved POIs and the observed ones, therefore can alleviate the cold-start influence. By embedding all the factors into the shared latent space, our method is also able to capture the potential correlations between users, POIs and attributes, therefore leads to a better performance. This also explains for our model superiority over other baselines in social link prediction task.

Besides, our method highlight on its flexibility to incorporate multiple effects into the unified framework. The generic characteristic makes it adjustable to various information networks with diverse properties and attributes. However, in order to for the model to achieve the optimal performance, careful feature engineering on feature hierarchy and representation is necessary.

Our graph embedding-based recommender shows advantages in capturing and reasoning rich context and side information in the heterogeneous information network. However, comparing with the naive recommenders, it has a much higher model complexity. The online updating strategy we introduced to dynamically add new records and capture users' latest preference sheds light on realizing incremental updates in the recommender systems. Considering the massive scale of LBSNs, this investigation is quite meaningful, which makes the model more feasible to be put into practice use.

In fact, the generality-pertinency and accuracy-scalability trade-offs have always been dilemmas in the research of recommender system. Our work is an exploration of finding a

well-balanced solution.

6.2 Summary

In this thesis, we present a joint framework to combine user and POI attribute networks using a heterogeneous graph embedding-based model. We demonstrate that this method achieves good performance in several major tasks in local-based social networks, including next-POI prediction, and social link prediction. By capturing the first-order proximity between homogeneous graph nodes, local correlations are observed. Using second-order proximity to observe the shared neighborhood of heterogeneous graph nodes helps to preserve the global structure and alleviate network sparsity.

We conduct extensive experiments to evaluate our model and compare it with four representative baseline models on two real-world large-scale datasets. Experimental results show the versatility of our method in both context-aware next-POI prediction task and social link prediction task. The study on cold-start recommendations further proves the superiority of our method in embedding all network attributes into a shared latent space in a unified way. Potential correlations between the network nodes can be well captured which compensates for information loss. Another highlight of our work is an online update strategy which tracks users' latest preference without retraining the entire model.

Besides from model evaluations, multiple influential factors in LBSNs are investigated, which sheds light on future directions in incorporating valuable side information and constructing expressive representations that well reflect the primary effect.

We are able to demonstrate that the main goal of this thesis is to show that a unified graph embedding method works well in depicting heterogeneous information graph, and is suitable for building a reliable recommender which models user behavior patterns in LBSNs.

6.3 Future Work

Current work on modeling user behavior patterns is mainly focused on user's next check-in POI and observing potential social link. Although location and users' social links are the most important information encoded in users mobile trajectories that interests us, other predictive tasks concerning side information such as user emotions or content feedback deserve further study and more exploration.

In addition, the method presented in this work shows good performance on constructing the static portrait of users, but in the real world, users' personal preference as well as the surrounding environment is updating constantly. An immediate area of interest for future work is to build a more comprehensive incremental updating strategy to obtain more accurate prediction result and make the model more suitable for practical use.

Another area for improvement is that, our method proposed in this thesis uses a graph embedding-based algorithm to embed the influence of different factors on users' behavior patterns. Although extensive dataset exploration is conducted seeking for representative expressions of relevant network attributes, more expressive representations for these attributes may still exist. Future work focusing on designing personalized characterization methods for representing each network attribute is suggested.

It's also worth mentioning that, on Google Local dataset, the categorical information about POIs shows a negative impact on next-POI prediction task. Possible reason is the ill design of attribute representations. A closer look into this phenomenon is necessary in order to make this side information useful.

6.4 Acknowledgement

I would like to acknowledge Professor Julian McAuley for his support as the chair of my committee. His guidance has helped me shape my work into what it is.

I would also like to acknowledge Professor Truong Quang Nguyen as the co-chair, and Professor Nikolay Atanasov as my committee. I am gratefully indebted to their valuable comments on this thesis.

Finally, I must express my very profound gratitude to my parents and to my boyfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

Appendix A

JGEL Source Code

Reference source code for the model JGEL proposed in this work can be found at:

https://github.com/WeiqiXu.

Bibliography

[1] B. Liu, H. Xiong, S. Papadimitriou, Y. Fu, and Z. Yao, "A general geographical probabilistic factor model for point of interest recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 5, pp. 1167–1179, 2015.

[2] P. Bhargava, T. Phan, J. Zhou, and J. Lee, "Who, what, when, and where: Multi-dimensional collaborative recommendations using tensor factorization on sparse user-generated data," in *Proceedings of the 24th international conference on world wide web*, pp. 130–140, International World Wide Web Conferences Steering Committee, 2015.

[3] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," *WWW*, pp. 1067–1077, 2015.

[4] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowl.-Based Syst.*, vol. 151, pp. 78–94, 2018.

[5] S. Wang, Y. Wang, J. Tang, K. Shu, S. Ranganath, and H. Liu, "What your images reveal: Exploiting visual contents for point-of-interest recommendation," in *WWW*, 2017.

[6] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, pp. 1616–1637, 2018.

[7] Y. Zheng and E. Zhou, X, "Location-based social networks: Users," 2011.

[8] "Location-based social networks." https://www.microsoft.com/en-us/research/project/location-based-social-networks/.

[9] "Foursquare dataset." https://www3.foursquare.com/places/?utm_source=google&utm_medium=cpc&utm_campaign=brand.

[10] "Gowalla." blog.gowalla.com/.

[11] A. Tarasov, F. Kling, and A. Pozdnoukhov, "Prediction of user location using the radiation model and social check-ins," No. 8, ACM, Aug 2013.

[12] D. Yao, C. Zhang, J. Huang, and J. Bi, "Serm: A recurrent model for next location prediction in semantic trajectories," pp. 2411–2414, ACM, Nov 2017.

[13] M. Hang, I. Pytlarz, and J. Neville, "Exploring student check-in behavior for improved point-of-interest prediction," pp. 321–330, ACM, Aug 2018.

[14] H. Gao, J. Tang, X. Hu, and H. Liu, "Content-aware point of interest recommendation on location-based social networks," in *AAAI*, 2015.

[15] C. Yang, M. Sun, W. X. Zhao, Z. Liu, and E. Y. Chang, "A neural network approach to jointly modeling social networks and mobile trajectories," *ACM Trans. Inf. Syst.*, vol. 35, pp. 36:1–36:28, 2017.

[16] M.-J. Lee and C.-W. Chung, "A user similarity calculation based on the location for social network services," in *International Conference on Database Systems for Advanced Applications*, pp. 38–52, Springer, 2011.

[17] J. H. A. K. Justin Cranshaw, Eran Toch and N. Sadeh, "Bridging the gap between physical location and online social networks," pp. 119–128, ACM, 2010.

[18] L. H. Huy Pham and C. Shahabi, "Towards integrating real-world spatiotemporal data with social networks," in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 453–457, ACM, 2011.

[19] C. S. Huy Pham and Y. Liu, "Ebm: an entropy-based model to infer social strength from spatiotemporal data," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pp. 265–276, ACM, 2013.

[20] Y. Koren, "Collaborative filtering with temporal dynamics," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 447–456, ACM, 2009.

[21] Y. Ding and X. Li, "Time weight collaborative filtering," in *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 485–492, ACM, 2005.

[22] L. Xiong, X. Chen, T.-K. Huang, J. Schneider, and J. G. Carbonell, "Temporal collaborative filtering with bayesian probabilistic tensor factorization," in *Proceedings of the 2010 SIAM international conference on data mining*, pp. 211–222, SIAM, 2010.

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

[24] J. McInerney, J. Zheng, A. Rogers, and N. R. Jennings, "Modelling heterogeneous location habits in human populations for location prediction under data sparsity," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 469–478, ACM, 2013.

[25] J.-D. Zhang, C.-Y. Chow, and Y. L. Lore, "Exploiting sequential influence for location recommendations," *SIGSPATIAL*, pp. 103–112, 2014.

[26] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *WWW*, 2010.

[27] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan, "Personalized ranking metric embedding for next new poi recommendation," in *IJCAI*, 2015.

[28] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *AAAI*, 2016.

[29] D. Wang, D. Pedreschi, C. Song, F. Giannotti, and A.-L. Barabasi, "Human mobility, social ties, and link prediction," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1100–1108, Acm, 2011.

[30] E. Cho, S. A. Myers, and J. 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*, pp. 1082–1090, ACM, 2011.

[31] Y. Fu, G. Liu, Y. Ge, P. Wang, H. Zhu, C. Li, and H. Xiong, "Representing urban forms: A collective learning model with heterogeneous human mobility data," *IEEE transactions on knowledge and data engineering*, vol. 31, no. 3, pp. 535–548, 2019.

[32] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.

[33] Y. Wang, N. J. Yuan, D. Lian, L. Xu, X. Xie, E. Chen, and Y. Rui, "Regularity and conformity: Location prediction using heterogeneous mobility data," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1275–1284, ACM, 2015.

[34] B. Liu and H. Xiong, "Point-of-interest recommendation in location based social networks with topic and location awareness," *SDM*, pp. 396–404, 2013.

[35] K. Zhao, G. Cong, Q. Yuan, and K. Q. Zhu, "Sar: A sentiment-aspect-region model for user preference analysis in geo-tagged reviews," *2015 IEEE 31st International Conference on Data Engineering*, pp. 675–686, 2015.

[36] R. Pasricha and J. McAuley, "Translation-based factorization machines for sequential recommendation," in *RecSys*, 2018.

[37] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding.," *Science*, vol. 290 5500, pp. 2323–6, 2000.

[38] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *NIPS*, 2001.

[39] A. Ahmed, N. Shervashidze, S. M. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *WWW*, 2013.

[40] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *KDD*, 2016.

[41] B. Shaw and T. Jebara, "Structure preserving embedding," in *ICML*, 2009.

[42] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 40–51, 2007.

[43] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," *KDD : proceedings. International Conference on Knowledge Discovery Data Mining*, vol. 2016, pp. 855–864, 2016.

[44] B. Perozzi, V. Kulkarni, and S. Skiena, "Walklets: Multiscale graph embeddings for interpretable network classification," *CoRR*, vol. abs/1605.02115, 2016.

[45] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *KDD*, 2016.

[46] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *CoRR*, vol. abs/1609.02907, 2017.

[47] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017.

[48] S. Scellato, A. Noulas, R. Lambiotte, and C. Mascolo, "Socio-spatial properties of online location-based social networks," in *ICWSM*, 2011.

[49] H. Gao, J. Tang, and H. Liu, "Addressing the cold-start problem in location recommendation using geo-social correlations," *Data Mining and Knowledge Discovery*, vol. 29, pp. 299–323, 2014.

[50] R. He, W.-C. Kang, and J. McAuley, "Translation-based recommendation," in *RecSys*, 2017.

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

[52] H. Yin, X. Zhou, B. Cui, H. Wang, K. Zheng, and N. Q. V. Hung, "Adapting to user interest drift for poi recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, pp. 2566–2581, 2016.

[53] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[54] A. Q. Li, A. Ahmed, S. Ravi, and A. J. Smola, "Reducing the sampling complexity of topic models," *KDD*, pp. 891–900, 2014.

[55] T. Chen, W. Zhang, Q. Lu, K. Chen, Z. Zheng, and Y. Yu, "Svdfeature: a toolkit for feature-based collaborative filtering," *Journal of Machine Learning Research*, vol. 13, no. Dec, pp. 3619–3622, 2012.

[56] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

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

[58] L. van der Maaten and G. E. Hinton, "Visualizing data using t-sne," 2008.