A Neural Network Approach to Jointly Modeling Social Networks and Mobile Trajectories

CHENG YANG and MAOSONG SUN, Tsinghua University
WAYNE XIN ZHAO, Renmin University of China
ZHIYUAN LIU, Tsinghua University
EDWARD Y. CHANG, HTC Research & Innovation

Two characteristics of location-based services are mobile trajectories and the ability to facilitate social networking. The recording of trajectory data contributes valuable resources towards understanding users' geographical movement behaviors. Social networking is possible when users are able to quickly connect to anyone nearby. A social network with location based services is known as location-based social network (LBSN). As shown in Cho et al. [2013], locations that are frequently visited by socially related persons tend to be correlated, which indicates the close association between social connections and trajectory behaviors of users in LBSNs. To better analyze and mine LBSN data, we need to have a comprehensive view of each of these two aspects, i.e., the mobile trajectory data and the social network.

Specifically, we present a novel neural network model that can jointly model both social networks and mobile trajectories. Our model consists of two components: the construction of social networks and the generation of mobile trajectories. First we adopt a network embedding method for the construction of social networks: a networking representation can be derived for a user. The key to our model lies in generating mobile trajectories. Second, we consider four factors that influence the generation process of mobile trajectories: user visit preference, influence of friends, short-term sequential contexts, and long-term sequential contexts. To characterize the last two contexts, we employ the RNN and GRU models to capture the sequential relatedness in mobile trajectories at the short or long term levels. Finally, the two components are tied by sharing the user network representations. Experimental results on two important applications demonstrate the effectiveness of our model. In particular, the improvement over baselines is more significant when either network structure or trajectory data is sparse.

CCS Concepts: • **Information systems** → **Location based services**; **Social recommendation**; • **Human-centered computing** → *Social networks*; • **Computing methodologies** → Neural networks;

Additional Key Words and Phrases: Link prediction, next-location recommendation, friend recommendation, recurrent neural network

This work was done during C. Yang's internship at HTC Beijing Research.
This work was supported by the 973 Program (2014CB340501), the Major Project of the National Social Science Foundation of China (13&ZD190), the National Natural Science Foundation of China (NSFC 61502502, 61572273, and 61532010), the Tsinghua University Initiative Scientific Research Program (20151080406), and the Beijing Natural Science Foundation (4162032).
Authors' addresses: C. Yang, M. Sun, and Z. Liu (corresponding author), Department of Computer Science and Technology, Tsinghua University, Beijing 100084; emails: cheng-ya14@mails.tsinghua.edu.cn, sms@mail.tsinghua.edu.cn, liuzy@tsinghua.edu.cn; W. X. Zhao (corresponding author), School of Information & Beijing Key Laboratory of Big Data Management and Analysis Methods, Renmin University of China, Beijing 100872; email: batmanfly@gmail.com; E. Y. Chang, HTC Research & Innovation, Palo Alto, CA 94306; email: eyuchang@gmail.com.

DOI: http://dx.doi.org/10.1145/3041658

ACM Reference Format:
Cheng Yang, Maosong Sun, Wayne Xin Zhao, Zhiyuan Liu, and Edward Y. Chang. 2017. A neural network approach to jointly modeling social networks and mobile trajectories. ACM Trans. Inf. Syst. 35, 4, Article 36 (August 2017), 28 pages.
DOI: http://dx.doi.org/10.1145/3041658

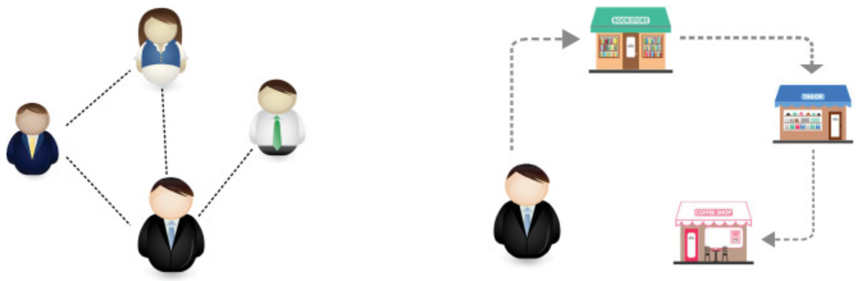
1. INTRODUCTION

In recent years, mobile devices (e.g., smartphones and tablets) have been widely used almost everywhere. With the innovation and development of Internet technology, mobile devices have become an essential connection to the broader world of online information for users. In daily life, a user can utilize her smartphone for conducting many life activities, including researching a travel plan, accessing online education, and looking for a job. The accelerated growth of mobile usage brings a unique opportunity to data mining research communities. Among these rich mobile data, an important kind of data resource is the huge amount of mobile trajectory data obtained from GPS sensors on mobile devices. These sensor footprints provide a valuable information resource to discover users' trajectory patterns and understand their moving behaviors. Several location-based sharing services have emerged and received much attention, such as Gowalla[1] and Brightkite.[2]

Apart from recording user trajectory data, another major feature of these location-based services is that they also allow the users to connect to whomever they like or are interested in. For example, with Brightkite, friends or any other Brightkite users nearby can be tracked using the phone's built-in GPS. A combination of social networking and location-based services has led to a specific style of social networks, termed *location-based social networks* (LBSNs) [Cho et al. 2011; Bao et al. 2012; Zheng 2015]. We present an illustrative example for LBSNs in Figure 1, where it can been seen that LBSNs usually include both the social network and mobile trajectory data. Recent literature has shown that social link information is useful to improve existing recommendation tasks [Machanavajjhala et al. 2011; Yuan et al. 2014a; Ma 2014]. Intuitively, users that often visit the same or similar locations are likely to be social friends,[3] and social friends are likely to visit the same or similar locations. In particular, several studies have found that there exists a close association between social connections and trajectory behaviors of users in LBSNs. On one hand, as shown in Cho et al. [2013], locations that are frequently visited by socially related persons tend to be correlated. On the other hand, trajectory similarity can be utilized to infer social strength between users [Pham et al. 2013; Zheng et al. 2010, 2011]. Therefore, we need to develop a comprehensive view to analyze and mine the information from the two aspects. In this article, our focus is to develop a joint approach to model LBSN data by characterizing both the social network and mobile trajectory data.

In the first aspect, social network analysis has attracted increasing attention during the past decade. It characterizes network structures in terms of nodes (individual actors, people, or things within the network) and the ties or edges (relationships or interactions) that connect them. A variety of applications have been developed on social networks, including network classification [Sen et al. 2008], link prediction [Liben-Nowell and Kleinberg 2007], anomaly detection [Chandola et al. 2009], and community detection [Fortunato 2010]. A fundamental issue is how to represent network nodes. Recently, networking embedding models [Perozzi et al. 2014] have been proposed to solve the data sparsity in networks. In the second aspect, location-based

[1]https://en.wikipedia.org/wiki/Gowalla.
[2]https://en.wikipedia.org/wiki/Brightkite.
[3]Note that an online social relationship does not necessarily indicate an offline friendship in real life.



(a) Friendship Network (b) User Trajectory

Fig. 1. Illustrative example for the data in LBSNs. (a) Link connections represent the friendship between users. (b) A trajectory generated by a user is a sequence of chronologically ordered check-in records.

services provide a convenient way for users to record their trajectory information, usually through what is called a *check-in*. Independent of social networking analysis, many studies have been constructed to improve the location-based services. A typical application task is the location recommendation, which aims to infer users' visit preference and make meaningful recommendations for users to visit. It can be divided into three different settings: general location recommendation [Zheng et al. 2009; Cheng et al. 2012; Ye et al. 2011], time-aware location recommendation [Yuan et al. 2013, 2014b; Liu et al. 2016], and next-location recommendation [Cheng et al. 2013; Ye et al. 2013; Zhang et al. 2014a]. General location recommendation will generate an overall recommendation list of locations for a users to visit, whereas time-aware or next location recommendation further imposes the temporal constraint on the recommendation task by either specifying the time period or producing sequential predictions.

These two aspects capture different data characteristics on LBSNs and tend to be correlated with each other [Cheng et al. 2012; Levandoski et al. 2012]. To conduct better and more effective data analysis and mining studies, there is a need to develop a joint model by capturing both network structure and trajectory behaviors on LBSNs. However, such a task is challenging. Social networks and mobile trajectories are heterogeneous data types. A social network is typically characterized by a graph, whereas a trajectory is usually modeled as a sequence of check-in records. A commonly used way to incorporate social connections into an application system (e.g., recommender systems) is to adopt the regularization techniques by assuming that the links convey user similarity. In this way, the social connections are exploited as the side information but not characterized by a joint data model, and the model performance highly relies on the "homophily principle" that like associates with like. In this article, we take the initiative to jointly model social networks and mobile trajectories using a neural network approach. Our approach is inspired by the recent progress in deep learning. Compared to other methods, neural network models can serve as an effective and general function approximation mechanism that is able to capture complicated data characteristics [Mittal 2016]. Specifically, recent studies have shown the superiority of neural network models on network and sequential data. First, several pioneering studies tried to embed vertices of a network into low-dimensional vector spaces [Tang and Liu 2011; Perozzi et al. 2014; Tang et al. 2015b], called *networking embedding*. With such a low-dimensional dense vector, it can alleviate the data sparsity from which a sparse network representation suffers. Second, neural network models are powerful computational data models that are able to capture and represent complex input/output relationships. In particular, several neural network models for processing sequential data have been proposed, such as recurrent neural networks (RNNs) [Mikolov et al. 2010]. RNN and its variants, including LSTM and GRU, have shown good performance in many applications.

By combining the merits from both network embedding and sequential modeling from deep learning, we present a novel neural network model that can jointly model both social networks and mobile trajectories. In specific, our model consists of two components: the construction of social networks and the generation of mobile trajectories. We first adopt a network embedding method for the construction of social networks: a networking representation can be derived for a user. The key to our model lies in the component generating mobile trajectories. We have considered four factors that influence the generation process of mobile trajectories: user visit preference, influence of friends, short-term sequential contexts, and long-term sequential contexts. The first two factors are mainly related to the users themselves, whereas the last two factors mainly reflect the sequential characteristics of historical trajectories. We set two different user representations to model the first two factors: a visit interest representation and a network representation. To characterize the last two contexts, we employ the RNN and GRU models to capture the sequential relatedness in mobile trajectories at different levels (e.g., short term or long term). Finally, the two components are tied by sharing the user network representations: the information from the network structure is encoded in the user networking representation, which is subsequently utilized in the generation process of mobile trajectories.

To demonstrate the effectiveness of the proposed model, we evaluate it using real-world datasets on two important LBSN applications: next-location recommendation and friend recommendation. For the first task, the trajectory data is the major information signal, whereas the network structure serves as auxiliary data. Our method consistently outperforms several competitive baselines. Interestingly, we have found that for users with little check-in data, the auxiliary data (i.e., network structure) becomes more important to consider. For the second task, the network data is the major information signal, whereas the trajectory data serves as auxiliary data. The finding is similar to that in the first task: our method still performs best, especially for those users with few friend links. Experimental results on the two important applications demonstrate the effectiveness of our model. In our approach, network structure and trajectory information complement each other. Hence, the improvement over baselines is more significant when either network structure or trajectory data is sparse.

Our contributions are threefold and summarized as follows:

—We propose a novel neural network model to jointly characterize social network structure and users' trajectory behaviors. In our approach, network structure and trajectory information complement each other. It provides a promising way to characterize heterogeneous data types in LBSNs.
—Our model considers four factors in the generation of mobile trajectories, including user visit preference, influence of friends, short-term sequential contexts, and long-term sequential contexts. The first two factors are modeled by two different embedding representations for users. The model further employs both RNN and GRU models to capture both short-term and long-term sequential contexts.
—Experimental results on two important applications demonstrate the effectiveness of our model. Interestingly, the improvement over baselines is more significant when either network structure or trajectory information is sparse.

The remainder of the article is organized as follows. Section 2 reviews the related work, and Section 3 presents the problem formulation. The proposed model, together with the learning algorithm, is provided in Section 4. We present the experimental evaluation in Section 5. Section 6 concludes the article and presents our future work.

2. RELATED WORK

Our work is mainly related to distributed representation learning, social link prediction, and location recommendation.

2.1. Distributed Representation Learning and Neural Network Models

Machine learning algorithms based on data representation learning have had great success in the past few years. Representations learning of the data can extract useful information for learning classifiers and other predictors. Distributed representation learning has been widely used in many machine learning tasks [Bengio et al. 2013], such as computer vision [Krizhevsky et al. 2012] and natural language processing [Mikolov et al. 2013].

During the past decade, many works were proposed for network embedding learning [Chen et al. 2007; Tang and Liu 2009, 2011; Perozzi et al. 2014]. Traditional network embedding learning algorithms learn vertex representations by computing eigenvectors of affinity matrices [Belkin and Niyogi 2001; Yan et al. 2007; Tang and Liu 2011]. For example, DGE [Chen et al. 2007] solves the generalized eigenvector computation problem on a combinational Laplacian matrix, and SocioDim [Tang and Liu 2011] computes the k smallest eigenvectors of a normalized graph Laplacian matrix as k-dimensional vertex representations.

DeepWalk [Perozzi et al. 2014] adapts Skip-Gram [Mikolov et al. 2013], a widely used language model in the natural language processing area, for NRL on truncated random walks. DeepWalk, which leverages a deep learning technique for network analysis, is much more efficient than traditional NRL algorithms and makes large-scale NRL possible. Following this line, LINE [Tang et al. 2015b] is a scalable network embedding algorithm that models the first-order and second-order proximities between vertices, and GraRep [Cao et al. 2015] characterizes local and global structural information for network embedding by computing SVD decomposition on a k-step transition probability matrix. MMDW [Tu et al. 2016] takes label information into account and learns semisupervised network embeddings.

TADW [Yang et al. 2015] and PTE [Tang et al. 2015a] extend DeepWalk and LINE by incorporating text information into NRL, respectively. TADW embeds text information into vertex representation by a matrix factorization framework, and PTE learns semisupervised embeddings from heterogeneous text networks. However, both TADW and PTE conduct experiments on document networks and fail to take sequential information between words into consideration.

Neural network models have achieved great success during the past decade. Two well-known neural network architectures are the convolutional neural network (CNN) and the RNN. The CNN is used for extracting fixed-length representation from various sizes of data [Krizhevsky et al. 2012]. RNN and its variant GRU, which aim at sequential modeling, have been successfully applied in sentence modeling [Mikolov et al. 2010], speech signal modeling [Chung et al. 2014], and sequential click prediction [Zhang et al. 2014b].

2.2. Social Link Prediction

Social link prediction has been widely studied in various social networks by mining graph structure patterns such as the triadic closure process [Romero and Kleinberg 2010] and user demographics [Huang et al. 2014]. In this article, we mainly focus on the applications on trajectory data.

Researchers used to measure user similarity by evaluating sequential patterns. For example, they used a sequence of stay points to represent a user trajectory and evaluated user similarity by a sequence matching algorithm [Li et al. 2008]. To improve these methods, people also took predefined tags and weights into consideration to better characterize stay points [Xiao et al. 2010]. As LBSNs become increasingly popular, trajectory similarity mining is attracting much more attention. Several factors have been considered to better characterize the similarity. As a result, physical distance [Cranshaw et al. 2010], location category [Lee and Chung 2011], spatial or temporal co-location rate [Wang et al. 2011], and co-occurrence with time and distance constraints [Pham et al. 2011] were proposed for social link prediction. The diversity of co-occurrence and popularity of locations [Pham et al. 2013] were proved to be important features among all of the factors. Using associated social ties to cluster locations, the social strength can be inferred in turn by extracted clusters shared by users [Cho et al. 2013; Zheng et al. 2011].

2.3. Location Recommendation

One of the most important tasks on trajectory modeling is location recommendation. For general location recommendation, several kinds of side information are considered, such as geographical [Cheng et al. 2012; Ye et al. 2011], temporal [Zhao et al. 2016], and social network information [Levandoski et al. 2012]. To address the data sparsity issue, content information, including location category labels, is also concerned [Yin et al. 2013; Zhou et al. 2016]. The location labels and tags can also be used in a probabilistic model, such as aggregate LDA [Gao et al. 2015]. Textual information that includes text descriptions [Gao et al. 2015; Li et al. 2010; Zhao et al. 2015] is applied for location recommendation as well. W^4 employs tensor factorization on multidimensional collaborative recommendation for Who (user), What (location category), When (time), and Where (location) [Zheng et al. 2010; Bhargava et al. 2015]. However, these methods, which are mainly based on collaborate filtering, matrix factorization, or LDA, do not model the sequential information in the trajectory.

For the time-aware location recommendation task, which recommends locations at a specific time, it is also worth modeling the temporal effect. The collaborate filtering–based method [Yuan et al. 2013] unifies temporal and geographical information with linear combinations. The geographical-temporal graph was proposed for time-aware location recommendation by doing preference propagation on the graph [Yuan et al. 2014b]. In addition, the temporal effect is also studied via nonnegative matrix factorization [Gao et al. 2013] and RNN [Liu et al. 2016].

Different from general location recommendation, next-location recommendation also needs to take the current state into account. Therefore, the sequential information is more important to consider in next-location recommendation. Most previous works model sequential behaviors (i.e., trajectories of check-in locations) based on Markov chain assumption, which assumes that the next location is determined only by THE current location and independent of previous ones [Rendle et al. 2010; Cheng et al. 2013; Ye et al. 2013; Zhang et al. 2014a]. For example, the factorized personalized Markov chain (FPMC) algorithm [Rendle et al. 2010] factorizes the tensor of the transition cube, which includes transition probability matrices of all users. Personalized ranking netric embedding (PRME) [Feng et al. 2015] further extends FPMC by modeling user-location distance and location-location distance in two different vector spaces. The hierarchical representation model (HRM) [Wang et al. 2015], which is originally designed for user purchase behavior modeling, can be easily adapted for modeling user trajectories. HRM builds a two-layer structure to predict items in the next transaction with user features and items in the last transaction. These methods are applied for next-location recommendation, which aims at predicting the next location that a user will visit, given the check-in history and current location of the user. Note that Markov chain property is a strong assumption that assumes the next location is determined only by the current location. In practice, the next location may also be influenced by the entire check-in history.

3. PROBLEM FORMALIZATION

We use L to denote the set of locations (also known as check-in points or POIs). When a user v checks in at a location l at the timestamp s, the information can be modeled as a triplet $\langle v, l, s\rangle$. Given a user v, her trajectory T_v is a sequence of triplets related to v: $\langle v, l_1, s_1\rangle, \ldots, \langle v, l_i, s_i\rangle, \ldots, \langle v, l_N, s_N\rangle$, where N is the sequence length and the triplets are ordered by timestamps ascendingly. For brevity, we rewrite the preceding formulation of T_v as a sequence of locations $T_v = \{l_1^{(v)}, l_2^{(v)}, \ldots, l_N^{(v)}\}$ in chronological order. Furthermore, we can split a trajectory into multiple consecutive subtrajectories: the trajectory T_v is split into m_v subtrajectories $T_v^1, \ldots, T_v^{m_v}$. Each subtrajectory is essentially a subsequence of the original trajectory sequence. To split the trajectory, we compute the time interval between two check-in points in the original trajectory sequence, and we follow Cheng et al. [2013] to make a splitting when the time interval is larger than 6 hours. To this end, each user corresponds to a trajectory sequence T_v consisting of several consecutive subtrajectories $T_v^1, \ldots, T_v^{m_v}$. Let T denote the set of trajectories for all users.

In addition to trajectory data, location-based services provide social connection links among users. Formally, we model the social network as a graph $G = (V, E)$, where each vertex $v \in V$ represents a user and each edge $e \in E$ represents the friendship between two users. In real applications, the edges can be either undirected or directed. As we will see, our model is flexible to deal with both types of social networks. Note that these links mainly reflect online friendship, which do not necessarily indicate that two users are friends in real life.

Given the social network information $G = (V, E)$ and the mobile trajectory information T, we aim to develop a joint model that can characterize and utilize both kinds of data resources. Such a joint model should be more effective than those built with a single data resource alone. To test the model performance, we set uptwo application tasks in LBSNs:

—***Task I*:** For the task of next-location recommendation, our goal is to recommend a ranked list of locations that a user v is likely to visit next at each step.
—***Task II*:** For the task of friend recommendation, our goal is to recommend a ranked list of users that are likely to be the friends of a user v.

We select these tasks because they are widely studied in LBSNs, respectively representing two aspects for mobile trajectory mining and social networking analysis. Other tasks related to LBSN data can be equally solved by our model, which are not our focus in this article.

4. THE PROPOSED MODEL

In this section, we present a novel neural network model for generating both social network and mobile trajectory data. In what follows, we first study how to characterize each individual component. Then we present the joint model, followed by the parameter learning algorithm. Before introducing the model details, we first summarize the used notations in this article in Table I.

4.1. Modeling the Construction of the Social Network

Networking representation learning is widely studied [Chen et al. 2007; Tang and Liu 2009, 2011; Perozzi et al. 2014], and it provides a way to explore the networking structure patterns using low-dimensional embedding vectors. Not limited to discover

Table I. Notations Used in This Article

Notation	Descriptions
V, E	Vertex and edge set
L	Location set
T_v, T_v^j	Trajectory and the j-th subtrajectory of user v
m_v	Number of subtrajectories in the trajectory T_v of user v
$m_{v,j}$	Number of locations in the j-th subtrajectory of trajectory T_v of user v
$l_i^{(v,j)}$	i-th location of the j-th subtrajectory of user v
U_{l_i}	Representation of location l_i used in representation modeling
U'_{l_i}	Representation of location l_i for prediction
P_v, F_v	Interest and friendship representation of user v
F'_v	Context friendship representation of user v
S_i	Short-term context representation after visiting location l_{i-1}
h_t	Long-term context representation after visiting location l_{t-1}

structure patterns, network representations have been shown to be effective to serve as important features in many network-independent tasks, such as demographic prediction [Huang et al. 2014] and text classification [Yang et al. 2015]. In our task, we characterize the networking representations based on two considerations. First, a user is likely to have similar visit behaviors with his friends, and user links can be leveraged to share common visit patterns. Second, the networking structure is utilized as auxiliary information to enhance trajectory modeling.

Formally, we use a d-dimensional embedding vector of use $F_v \in \mathbb{R}^d$ to denote the network representation of user v and matrix $F \in \mathbb{R}^{|V|\times d}$ to denote the network representations for all users. The network representation is learned with the user links on the social network and encodes the information for the structure patterns of a user.

The social network is constructed based on users' networking representations F. We first study how to model the generative probability for a edge of $v_i \rightarrow v_j$, formally as $\Pr[(v_i, v_j) \in E]$. The main intuition is that if two users v_i and v_j form a friendship link on the network, their networking representations should be similar. In other words, the inner product $F_{v_i}^\top \cdot F_{v_j}$ between the corresponding two networking representations will yield a large similarity value for two linked users. A potential problem will be that such a formulation can only deal with undirected networks. To characterize both undirected and directed networks, we propose to incorporate a context representation for a user v_j (i.e., F'_{v_j}). Given a directed link $v_i \rightarrow v_j$, we model the representation similarity as $F_{v_i}^\top \cdot F'_{v_j}$ instead of $F_{v_i}^\top \cdot F_{v_j}$. The context representations are only used in the network construction. We define the probability of a link $v_i \rightarrow v_j$ by using a sigmoid function as follows:

$$\Pr[(v_i, v_j) \in E] = \sigma\big(-F_{v_i}^\top \cdot F'_{v_j}\big) = \frac{1}{1+\exp\big(-F_{v_i}^\top \cdot F'_{v_j}\big)}. \tag{1}$$

When dealing with undirected networks, a friend pair (v_i, v_j) will be split into two directed links, namely $v_i \rightarrow v_j$ and $v_j \rightarrow v_i$. For edges not existing in E, we propose to use the following formulation:

$$\Pr[(v_i, v_j) \notin E] = 1 - \sigma\big(-F_{v_i}^\top \cdot F'_{v_j}\big) = \frac{\exp\big(-F_{v_i}^\top \cdot F'_{v_j}\big)}{1+\exp\big(-F_{v_i}^\top \cdot F'_{v_j}\big)}. \tag{2}$$

Combining Equations (1) and (2), we essentially adopt a Bernoulli distribution for modeling networking links. Following studies on networking representation learning [Perozzi et al. 2014], we assume that each user pair is independent in the generation process. In other words, the probabilities $\Pr[(v_i, v_j) \in E|F]$ are independent for different pairs of (v_i, v_j). With this assumption, we can factorize the generative probabilities by user pairs

$$\begin{aligned}\mathcal{L}(G) &= \sum_{(v_i,v_j)\in E} \log \Pr[(v_i, v_j) \in E] + \sum_{(v_i,v_j)\notin E} \log \Pr[(v_i, v_j) \notin E] \\ &= -\sum_{v_i,v_j} \log(1 + \exp(-F_{v_i}^{\top} \cdot F'_{v_j})) - \sum_{(v_i,v_j)\notin E} F_{v_i}^{\top} \cdot F'_{v_j}.\end{aligned} \tag{3}$$

4.2. Modeling the Generation of the Mobile Trajectories

In Section 3, a user trajectory is formatted as an ordered check-in sequence. Therefore, we model the trajectory generation process with a sequential neural network method. To generate a trajectory sequence, we generate the locations in it one by one, conditioned on four important factors. We summarize the four factors as follows:

—*General visit preference*: A user's preference or habits directly determine her own visit behaviors.
—*Influence of friends*: The visit behavior of a user is likely to be influenced by her friends. Previous studies [Cheng et al. 2012; Levandoski et al. 2012] indeed have shown that socially correlated users tend to visit common locations.
—*Short-term sequential contexts*: The next location is closely related to the last few locations visited by a user. The idea is intuitive in that the visit behaviors of a user are usually related to a single activity or a series of related activities in a short time window, making the visited locations have strong correlations.
—*Long-term sequential contexts*: It is likely that there exists long-term dependency for the visited locations by a user in a long time period. A specific case for long-term dependency will be periodic visit behaviors. For example, a user regularly has a travel in every summer vacation.

The first two factors are mainly related to the two-way interactions between users and locations, whereas the last two factors mainly reflect the sequential relatedness among the visited locations by a user.

4.2.1. Characterization of General Visit Preference. We first characterize the general visit preference by the interest representations. We use a d-dimensional embedding vector of $P_v \in \mathbb{R}^d$ to denote the visit interest representation of user v and matrix $P \in \mathbb{R}^{|V|\times d}$ to denote the visit preference representations for all users. The visit interest representation encodes the information for the general preference of a user over the set of locations in terms of visit behaviors.

We assume that one's general visit interests are relatively stable and do not vary too much in a given period. Such an assumption is reasonable in that a user typically has a fixed lifestyle (e.g., with a relatively fixed residence area) and her visiting behaviors are likely to show some overall patterns. The visit interest representation aims to capture and encode such visit patterns by using a d-dimensional embedding vector. For convenience, we call P_v the *interest representation* for user v.

4.2.2. Characterization of Influence of Friends. For characterizing the influence of friends, a straightforward approach is to model the correlation between interest representations from two linked users with some regularization terms. However, such a method usually has high computational complexity. In this work, we adopt a more flexible method: we incorporate the network representation in the trajectory generation process. Because the network representations are learned through the network links, the information

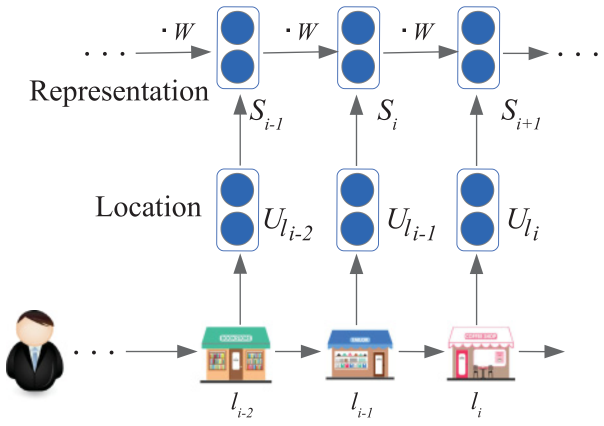


Fig. 2. An illustrative example of RNNs for modeling short-term sequential contexts.

from their friends is implicitly encoded and used. We still use the formulation of networking representation F_v introduced in Section 4.1.

4.2.3. Characterization of Short-Term Sequential Contexts. Usually the visited locations by a user in a short time window are closely correlated. A short sequence of the visited locations tend to be related to some activity. For example, a sequence "Home → Traffic → Office" refers to one's transportation activity from home to office. In addition, the geographical or traffic limits play an important role in the trajectory generation process. For example, a user is more likely to visit a nearby location. Therefore, when a user decides what location to visit next, the last few locations she visited should be of importance for next-location prediction.

Based on the preceding considerations, we treat the last few visited locations in a short time window as the sequential history and predict the next location based on them. To capture the short-term visit dependency, we use the RNN, a convenient way for modeling sequential data, to develop our model. Formally, given the j-th subsequence $T_v^j = \{l_1^{(v,j)}, l_2^{(v,j)} \dots l_{m_{v,j}}^{(v,j)}\}$ from the trajectory of user v, we recursively define the short-term sequential relatedness as follows:

$$S_i = \tanh(U_{l_{i-1}} + W \cdot S_{i-1}), \tag{4}$$

where $S_i \in \mathbb{R}^d$ is the embedding representation for the state after visiting location l_{i-1}, $U_{l_i} \in \mathbb{R}^d$ is the representation of location $l_i^{(v,j)}$ and $W \in \mathbb{R}^{d\times d}$ is a transition matrix. Here we call S_i *states*, which are similar to those in hidden Markov models. RNN resembles hidden Markov models in that the sequential relatedness is also reflected through the transitions between two consecutive states. A major difference is that in an RNN, each hidden state is characterized by a d-dimensional embedding vector. As shown in Figure 2, we derive the state representation S_i by forwarding S_{i-1} with a transformation matrix W and adding the embedding representation for the current location $U_{l_{i-1}}$. The initial representation S_0 is invariant among all users because short-term correlation is supposed to be irrelevant to user preference in our model. Our formulation in Equation (4) is essentially an RNN model without outputs. The embedding vector corresponding to each state can be understood as an information summary until the corresponding location in the sequence. In particular, the state corresponding to the last location can be considered the embedding representation for the entire sequence.

4.2.4. Characterization of Long-Term Sequential Contexts. We have discussed how short-term sequential contexts (five locations on average for our dataset) aim to capture the

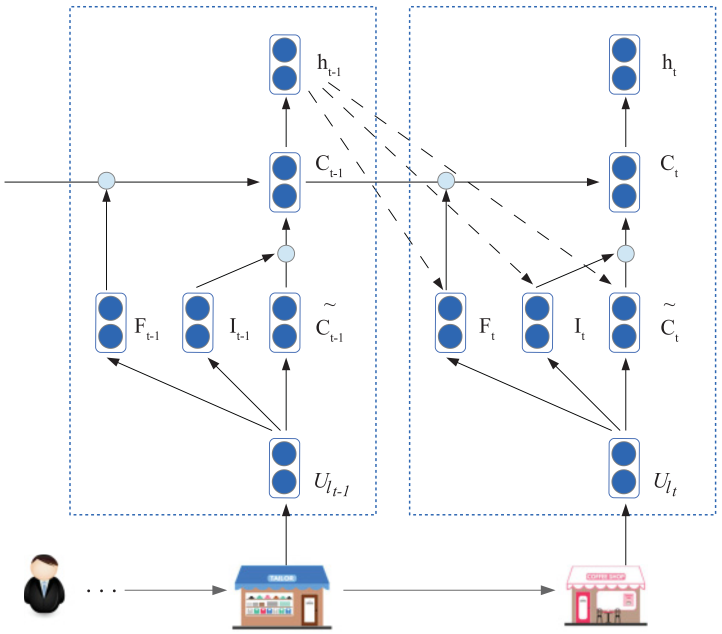


Fig. 3. Illustrative architecture of RNNs with GRUs. Let $\widetilde{C}_t$ denote a candidate state. The current state C_t is a mixture of the last state C_{t-1} and the current candidate state $\widetilde{C}_t$. I_t and F_t are input and forget gates, respectively, which can control this mixture.

sequential relatedness in a short time window. The long-term sequential contexts are also important to consider when modeling trajectory sequences. For example, a user is likely to show some periodical or long-range visit patterns. To capture the long-term dependency, a straightforward approach will be to use another RNN model for the entire trajectory sequence. However, the entire trajectory sequence generated by a user in a long time period tends to contain a large number of locations (e.g., several hundred locations or more). An RNN model over long sequences usually suffers from the problem of "vanishing gradient."

To address the problem, we employ the gated recurrent unit (GRU) for capturing long-term dependency in the trajectory sequence. Compared to traditional RNN, GRU incorporates several extra gates to control the input and output. Specifically, we use two gates in our model: input gates and forget gates. With the help of input and forget gates, the memory of GRU (i.e., the state C_t) can remember the "important stuff" even when the sequence is very long and forget less important information if necessary. We present an illustrative figure for the architecture for RNNs with GRUs in Figure 3.

Formally, consider the following location sequence $\{l_1, l_2, \ldots, l_m\}$. We denote the initial state by $C_0 \in \mathbb{R}^d$ and the initial representation by $h_0 = \tanh(C_0) \in \mathbb{R}^d$. At a timestep of t, the new candidate state is updated as follows:

$$\widetilde{C}_t = \tanh(W_{c_1} U_{l_t} + W_{c_2} h_{t-1} + b_c), \tag{5}$$

where $W_{c_1} \in \mathbb{R}^{d \times d}$ and $W_{c_2} \in \mathbb{R}^{d \times d}$ are the model parameters; U_{l_t} is the embedding representation of location l_t, which is the same representation used in short-term sequential relatedness; h_{t-1} is the embedding representation in the last step; and $b_c \in \mathbb{R}^d$ is the bias vector. Note that the computation of $\widetilde{C}_t$ remains the same as that in the RNN.

The GRU does not directly replace the state with $\widetilde{C}_t$ as the RNN does. Instead, the GRU tries to find a balance between the last state C_{t-1} and a new candidate state $\widetilde{C}_t$:

$$C_t = i_t * \widetilde{C}_t + f_t * C_{t-1}, \tag{6}$$

where $*$ is the entrywise product and $i_t, f_t \in \mathbb{R}^d$ are input and forget gates, respectively. The input and forget gates $i_t, f_t \in \mathbb{R}^d$ are defined as

$$i_t = \sigma(W_{i_1} U_{l_t} + W_{i_2} h_{t-1} + b_i) \tag{7}$$

and

$$f_t = \sigma(W_{f_1} U_{l_t} + W_{f_2} h_{t-1} + b_f), \tag{8}$$

where $\sigma(\cdot)$ is the sigmoid function, $W_{i_1}, W_{i_2} \in \mathbb{R}^{d\times d}$ and $W_{f_1}, W_{f_2} \in \mathbb{R}^{d\times d}$ are input and forget gate parameters, and $b_i, b_f \in \mathbb{R}^d$ are the bias vectors.

Finally, the representation of long-term interest variation at the timestep of t is derived as follows:

$$h_t = \tanh(C_t). \tag{9}$$

Similar to Equation (4), h_t provides a summary that encodes the information until the t-th location in a trajectory sequence. We can recursively learn the representations after each visit of a location.

4.2.5. The Final Objective Function for Generating Trajectory Data. Given the preceding information, we are now ready to present the objective function for generating trajectory data. Given the trajectory sequence $T_v = \{l_1^{(v)}, l_2^{(v)}, \ldots, l_m^{(v)}\}$ of user v, we factorize the log likelihood according to the chain rule as follows:

$$\begin{aligned}\mathcal{L}(T_v) &= \log \Pr\left[l_1^{(v)}, l_2^{(v)}, \ldots, l_m^{(v)} \middle| v, \Phi\right] \\ &= \sum_{i=1}^{m} \log \Pr\left[l_i^{(v)} \middle| l_1^{(v)}, \ldots, l_{i-1}^{(v)}, v, \Phi\right],\end{aligned} \tag{10}$$

where Φ denotes all related parameters. As we can see, $\mathcal{L}(T_v)$ is characterized as a sum of log probabilities conditioned on the user v and related parameters Φ. Recall that the trajectory T_v is split into m_v subtrajectories $T_v^1, \ldots, T_v^{m_v}$. Let $l_i^{(v,j)}$ denote the i-th location in the j-th subtrajectory. The contextual locations for $l_i^{(v,j)}$ contain the preceding $(i-1)$ locations (i.e., $l_1^{(v,j)} \ldots l_{i-1}^{(v,j)}$) in the same subtrajectory, denoted by $l_1^{(v,j)} : l_{i-1}^{(v,j)}$, and all locations in the previous $(j-1)$ subtrajectories (i.e., $T_v^1, \ldots, T_v^{j-1}$), denoted by $T_v^1 : T_v^{j-1}$. With these notions, we can rewrite Equation (10) as follows:

$$\mathcal{L}(T_v) = \sum_{i=1}^{m} \log \Pr\left[l_i^{(v,j)} \middle| \underbrace{l_1^{(v,j)} : l_{i-1}^{(v,j)}}_{\text{short-term contexts}}, \underbrace{T_v^1 : T_v^{j-1}}_{\text{long-term contexts}}, v, \Phi\right]. \tag{11}$$

Given the target location $l_i^{(v,j)}$, the term of $l_1^{(v,j)} : l_{i-1}^{(v,j)}$ corresponds to the short-term contexts, the term of $T_v^1 : T_v^{j-1}$ corresponds to the long-term contexts, and v corresponds to the user context. The key problem becomes how to model the conditional probability $\Pr[l_i^{(v,j)} | l_1^{(v,j)} : l_{i-1}^{(v,j)}, T_v^1 : T_v^{j-1}, v, \Phi]$.

For short-term contexts, we adopt the RNN model described in Equation (3) to characterize the the location sequence of $l_1^{(v,j)} : l_{i-1}^{(v,j)}$. We use S_i^j to denote the derived short-term representation after visiting the i-th location in the j-th subtrajectory. For

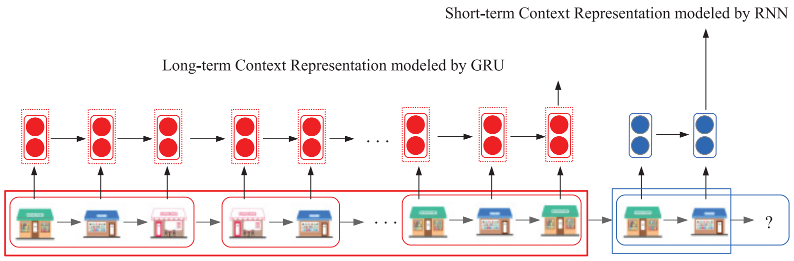

Short-term Context Representation modeled by RNN
Long-term Context Representation modeled by GRU
?

Fig. 4. Illustrative figure for modeling both short-term and long-term sequential contexts. The locations in a rounded rectangle indicate a subtrajectory. The locations in red and blue rectangles are used for long-term and short-term sequential contexts, respectively. A question mark (?) denotes the next location for prediction.

long-term contexts, the locations in the preceding subtrajectories $T_v^1 \dots T_v^{j-1}$ are characterized using the GRU model in Equations (5) through (9). We use h^j to denote the derived long-term representation after visiting the locations in the first j subtrajectories. We present an illustrative example for the combination of short-term and long-term contexts in Figure 4.

Thus far, given a target location $l_i^{(v,j)}$, we have obtained four representations corresponding to the four factors: networking representation (i.e., F_v), visit interest representation (i.e., P_v), short-term context representation S_{i-1}^j, and long-term context representation h^{j-1}. We concatenate them into a single context representation $R_v^{(i,j)} = [F_v; P_v; S_{i-1}^j; h^{j-1}] \in \mathbb{R}^{4d}$ and use it for next-location generation. Given the context representation $R_v^{(i,j)}$, we define the probability of $l_i^{(v,j)}$ as

$$\begin{aligned}
&\Pr\left[l_i^{(v,j)} \middle| l_1^{(v,j)} : l_{i-1}^{(v,j)}, T_v^1 : T_v^{j-1}, v, \Phi\right] \\
&= \Pr\left[l_i^{(v,j)} \middle| R_v^{(i,j)}\right] \\
&= \frac{\exp\left(R_v^{(i,j)} \cdot U'_{l_i^{(v,j)}}\right)}{\sum_{l \in L} \exp\left(R_v^{(i,j)} \cdot U'_l\right)},
\end{aligned} \tag{12}$$

where parameter $U'_l \in \mathbb{R}^{4d}$ is the location representation of location $l \in L$ used for prediction. Note that this location representation U'_l is totally different from the location representation $U_l \in \mathbb{R}^d$ used in short-term and long-term context modeling. The overall log likelihood of trajectory generation can be computed by adding up all of the locations.

4.3. The Joint Model

Our general model is a linear combination between the objective functions for the two parts. Given the friendship network of $G = (V, E)$ and user trajectory T, we have the following log-likelihood function:

$$\begin{aligned}
\mathcal{L}(G, T) &= \mathcal{L}_{\text{network}}(G) + \mathcal{L}_{\text{trajectory}}(T) \\
&= \mathcal{L}(G) + \sum_{v \in V} \mathcal{L}(T_v),
\end{aligned} \tag{13}$$

where $\mathcal{L}_{\text{network}}(G)$ is defined in Equation (3) and $\mathcal{L}_{\text{trajectory}}(T) = \sum_{v \in V} \mathcal{L}(T_v)$ is defined in Equation (11). We name our model the *Joint Network and Trajectory Model (JNTM)*.

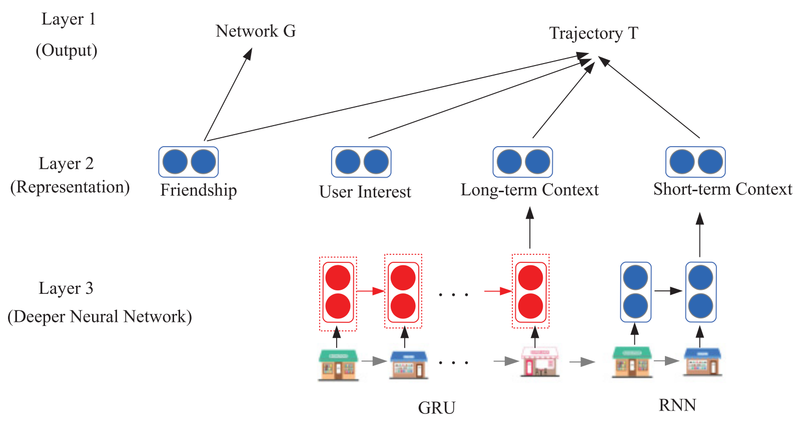


Fig. 5. Illustrative architecture of the proposed model JNTM.

We present an illustrative architecture of the proposed model JNTM in Figure 5. Our model is a three-layer neural network for generating both social network and user trajectory. In training, we require that both the social network and user trajectory should be provided as the objective output to the train the model. Based on such data signals, our model naturally consists of two objective functions. For generating the social network, a network-based user representation was incorporated; for generating the user trajectory, four factors were considered: network-based representation, general visiting preference, and short-term and long-term sequential contexts. These two parts were tied by sharing the network-based user representation.

4.4. Parameter Learning

Now we show how to train our model and learn the parameters—that is, user interest representation $P \in \mathbb{R}^{|V|\times d}$, user friendship representation $F, F' \in \mathbb{R}^{|V|\times d}$, location representations $U \in \mathbb{R}^{|L|\times d}, U' \in \mathbb{R}^{|L|\times 4d}$, initial short-term representation $S_0 \in \mathbb{R}^d$, transition matrix $W \in \mathbb{R}^{d\times d}$, initial GRU state $C_0 \in \mathbb{R}^d$, and GRU parameters $W_{i_1}, W_{i_2}, W_{f_1}, W_{f_2}, W_{c_1}, W_{c_2} \in \mathbb{R}^{d\times d}, b_i, b_f, b_c \in \mathbb{R}^d$.

Negative sampling. Recall that the log likelihood of network generation in Equation (3) includes $|V| \times |V|$ terms. Thus, it takes at least $O(|V|^2)$ time to compute, which is time consuming. Therefore, we employ a negative sampling technique that is commonly used in the NLP area [Mikolov et al. 2013] to accelerate our training process.

Note that real-world networks are usually sparse (i.e., $O(E) = O(V)$). The number of connected vertex pairs (positive examples) are much less than the number of unconnected vertex pairs (negative examples). The core idea of negative sampling is that most vertex pairs serve as negative examples, and thus we do not need to compute all of them. Instead, we compute all connected vertex pairs and n_1 random unconnected vertex pairs as an approximation where $n_1 \ll |V|^2$ is the number of negative samples. In our settings, we set $n_1 = 100|V|$. The log likelihood can be rewritten as

$$\mathcal{L}(G|F, F') = \sum_{(v_i,v_j)\in E} \log \Pr[(v_i, v_j) \in E] + \sum_{k=1,(v_{ik},v_{jk})\notin E}^{n_1} \log \Pr[(v_{ik}, v_{jk}) \notin E]. \quad (14)$$

Then the computation of likelihood of network generation only includes $O(E + n_1) = O(V)$ terms.

However, the computation of Equation (12) takes at least $O(|L|)$ time because the denominator contains $|L|$ terms. Note that the computation of this conditional probability

ALGORITHM 1: One Iteration of Network Generation

```
for each user $v \in V$ do
    Random pick 100 vertices $\{v_1, \dots, v_{n_2}\}$ that are not connected with $v$;
    Compute the log likelihood $\sum_{v':(v,v') \in E} \log \Pr[(v, v') \in E] + \sum_{k=1}^{100} \log \Pr[(v, v_k) \notin E]$;
    Compute the gradients of $F$ and $F'$ by backpropagation;
end
Update $F$ and $F'$ according to the gradients;
```

ALGORITHM 2: One Iteration of Trajectory Generation

```
for each user $v \in V$ do
    Compute the forward propagation of GRU by Equations (5) through (9) and get long-term
    context representations $h^1, \dots, h^{m_v}$;
    for each subtrajectory $T_v^j$ of user $v$ do
        for each location $l_i^{(v,j)}$ of $T_v^j$ do
            Update short-term location dependency representation by Equation (6);
            Concatenate four representations $[F_v; P_v; S_{i-1}^j; h^{j-1}]$;
            Compute log likelihood by Equation (15);
            Compute the gradients of $U'$, $F_v$, $P_v$, $S_{i-1}^j$ and $h^{j-1}$
        end
        for $i = m_{v,j}, \dots, 1$ do
            Compute the gradients of $S_0$, $U$, $W$ through backpropagation of the gradient of $S_{i-1}^j$
        end
    end
    for $j = m_v, \dots, 1$ do
        Compute the gradients of $C_0$, $W_{i_1}$, $W_{i_2}$, $W_{f_1}$, $W_{f_2}$, $W_{c_1}$, $W_{c_2}$, $b_i$, $b_f$, $b_c$ through
        backpropagation of the gradient of $h^j$
    end
end
Update all parameters according to their gradients
```

needs to be done for every location. Therefore, the computation of trajectory generation needs at least $O(|L|^2)$, which is not efficient. Similarly, we do not compute every term in the denominator. Instead, we only compute location $l_i^{(v,j)}$ and other n_2 random locations. In this article, we use $n_2 = 100$. Then we reformulate Equation (12) as

$$\Pr\left[l_i^{(v,j)} | R_v^{(i,j)}\right] = \frac{\exp\left(R_v^{(i,j)} \cdot U'_{l_i^{(v,j)}}\right)}{\exp\left(R_v^{(i,j)} \cdot U'_{l_i^{(v,j)}}\right) + \sum_{k=1, l_k \neq l_i^{(v,j)}}^{n_2} \exp\left(R_v^{(i,j)} \cdot U'_{l_k}\right)}. \tag{15}$$

Then the computation of the denominator only includes $O(n_2 + 1) = O(1)$ terms.

We compute the gradients of the parameters by backpropagation through time (BPTT) [Werbos 1990]. Then the parameters are updated with AdaGrad [Duchi et al. 2011], a variant of stochastic gradient descent (SGD), in mini-batches.

In more detail, we use pseudocodes in Algorithms 1 and 2 to illustrate the training process of our model. The network iteration and trajectory iteration are executed iteratively until the performance on the validation set becomes stable.

Complexity analysis. We are first given the complexity analysis on time cost. The network generation of user v takes $O(d)$ time to compute log likelihood and gradients of F_v and corresponding rows of F'. Thus, the complexity of network generation is $O(d|V|)$. In trajectory generation, we denote the total number of check-in data as $|D|$.

Then the forward and backward propagation of GRU take $O(d^2|D|)$ time to compute since the complexity of a single check-in is $O(d^2)$. Each step of RNN takes $O(d^2)$ time to update local dependency representation and compute the gradients of S_0, U, W. The computation of log likelihood and gradients of U', F_v, P_v, S_{i-1}^j and h^{j-1} takes $O(d^2)$ time. Hence, the overall complexity of our model is $O(d^2|D| + d|V|)$. Note that the representation dimension d and number of negative samples per user/location are much less than the data size $|V|$ and $|D|$. Hence, the time complexity of our algorithm JNTM is linear to the data size and scalable for large datasets. Although the training time complexity of our model is relatively high, the test time complexity is small. When making location recommendations to a user in the test stage, it takes $O(d)$ time to update the hidden states of RNN/LSTM and $O(d)$ time to evaluate a score for a single location. Usually, the hidden dimensionality d is a small number, which indicates that our algorithm is efficient to make online recommendations.

In terms of space complexity, the network representations F and location representations U take $O((|V| + |L|)d)$ space cost in total. The space cost of other parameters is at most $O(d^2)$, which can be neglected since d is much less than $|V|$ and $|L|$. Thus, the space complexity of our model is similar to that of previous models, such as FPMC [Rendle et al. 2010], PRME [Feng et al. 2015], and HRM [Wang et al. 2015].

5. EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our proposed model JNTM. We consider two application tasks: next-location recommendation and friend recommendation. In what follows, we discuss the data collection, baselines, parameter setting, and evaluation metrics. Then we will present the experimental results together with the related analysis.

5.1. Data Collection

We consider using two publicly available LBSN datasets[4] [Cho et al. 2011], Gowalla and Brightkite, for our evaluation. Gowalla and Brightkite have released the mobile apps for users. For example, with Brightkite you can track on your friends or any other BrightKite users nearby using a phone's built in GPS; Gowalla has a similar function: use GPS data to show where you are and what is near you.

These two datasets provide both connection links and users' check-in information. A connection link indicates a reciprocal friendship, and a check-in record contains the location ID and the corresponding check-in timestamp. We organized the check-in information as trajectory sequences. Following Cheng et al. [2013], we split a trajectory wherever the interval between two successive check-ins is larger than 6 hours. We performed some preprocessing steps on both datasets. For Gowalla, we removed all users who have fewer than 10 check-ins and locations that have fewer than 15 check-ins, and finally obtained 837,352 subtrajectories. For Brightkite, since this dataset is smaller, we only removed users who have fewer than 10 check-ins and locations that have fewer than 5 check-ins, and finally obtained 503,037 subtrajectories after preprocessing. Table II presents the statistics of the preprocessed datasets. Note that our datasets are larger than those in previous works [Cheng et al. 2013; Feng et al. 2015].

A major assumption we made is that there exists a close association between social links and mobile trajectory behaviors. To verify this assumption, we constructed an experiment to reveal basic correlation patterns between these two factors. For each user, we first generate a location set consisting of the locations that have been visited

[4]http://snap.stanford.edu/data/.

Table II. Statistics of Datasets

Dataset	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert D\rvert$	$\lvert L\rvert$
Gowalla	37,800	390,902	2,212,652	58,410
Brightkite	11,498	140,372	1,029,959	51,866

$|V|$, number of vertices; $|E|$, number of edges; $|D|$, number of check-ins; $|L|$, number of locations.

by the user. Then we can measure the similarity degree between the location sets from two users using the overlap coefficient.[5] The average overlap coefficients are 11.1% and 15.7% for a random friend pair (i.e., two users are social friends) on the Brightkite and Gowalla datasets, respectively. As a comparison, the overlap coefficient falls to 0.5% and 0.5% for a random nonfriend pair (i.e., two users are not social friends) on the Brightkite and Gowalla datasets, respectively. This finding indicates that users who are socially connected indeed have more similar visit characteristics. We next examined whether two users with similar trajectory behaviors are more likely to be socially connected. We found that the probabilities that two random users are social friends are 0.1% and 0.03% on the Brightkite and Gowalla datasets, respectively. However, if we select two users with more than three common locations in their location set, the probabilities that they are social friends increase to 9% and 2%, respectively. The preceding two findings show that social connections are closely correlated with mobile trajectory behaviors in LBSNs.

5.2. Evaluation Tasks and Baselines

Next-location recommendation. For the task of next-location recommendation, we consider the following baselines:

—*Paragraph vector (PV)* [Le and Mikolov 2014]: PV is a representation learning model for both sentence and documents using a simple neural network architecture. To model trajectory data, we treat each location as a word and each user as a paragraph of location words.

—*Feature-based classification (FBC)*: FBC solves the next-location recommendation task by casting it as a multiclass classification problem. The user features are learned using the DeepWalk algorithm [Perozzi et al. 2014], and the location features are learned using the word2vec [Mikolov et al. 2013] algorithm (similar to the training method of PV). These features are subsequently incorporated into a softmax classifier (i.e., a multiclass generalization of logistic regression).

—*FPMC* [Rendle et al. 2010]: FPMC is a state-of-the-art recommendation algorithm that factorizes the tensor of transition matrices of all users and predicts the next location by computing the transition probability based on Markov chain assumption. It was originally proposed for product recommendation; however, it is easy to adapt FPMC to deal with next-location recommendation.

—*PRME* [Feng et al. 2015]: PRME extends FPMC by modeling user-location and location-location pairs in different vector spaces. PRME achieves state-of-the-art performance on the next-location recommendation task.

—*HRM* [Wang et al. 2015]: HRM is the latest algorithm for next-basket recommendation. By taking each subtrajectory as a transaction basket, we can easily adapt HRM for next-location recommendation. It is the first study in which distributed representation learning has been applied to the recommendation problem.

We selected these five baselines because they represent different recommendation algorithms. PV is based on simple neural networks, FBC is a traditional classification

[5]https://en.wikipedia.org/wiki/Overlap_coefficient.

model using embedding features, FPMC is mainly developed in the matrix factorization framework, PRME makes specific extensions based on FPMC to adapt to the task of next-location recommendation, and HRM adopts the distributed representation learning method for next-basket modeling.

Next, we split the data collection into the training set and test set. The first 90% of check-in subtrajectories of each user are used as the training data and the remaining 10% as test data. To tune the parameters, we use the last 10% of check-ins of training data as the validation set.

Given a user, we predict the locations in the test set in a sequential way: for each location slot, we recommend 5 or 10 locations to the user. For JNTM, we naturally rank the locations by the log likelihood, as shown in Equation (12). Note that negative sampling is not used in the evaluation. For the baselines, we rank the locations by the transition probability for FPMC and HRM, and transition distance for PRME. The predictions of PV and FBC can be obtained from the output of the softmax layer of their algorithms. Then we report Recall@5 and Recall@10 as the evaluation metrics, where Recall@K is defined as

$$Recall@K = \frac{\text{\# ground truth locations in the } K \text{ recommended locations}}{\text{\# ground truth locations in test data}}.$$

Note that another common metric, Precision@K, can be used here as well. In our experiments, we found that it is positively correlated with Recall@K. For instance, if method A has a higher Recall@K score than method B, then method A also has a higher Precision@K score than method B. We omit the results of Precision@K for ease of presentation.

Friend recommendation. For the task of friend recommendation, we consider three kinds of baselines based on the used data resources, including the method with only the networking data (DeepWalk), the method with only the trajectory data (PMF), and the methods with both networking and trajectory data (PTE and TADW).

—*DeepWalk* [Perozzi et al. 2014]: DeepWalk is a state-of-the-art NRL method that learns vertex embeddings from random walk sequences. It first employs the random walk algorithm to generate length-truncated random paths and applies the word embedding technique to learn the representations for network vertices.
—*PMF* [Mnih and Salakhutdinov 2007]: PMF is a general collaborative filtering method based on user-item matrix factorization. In our experiments, we build the user-location matrix using the trajectory data, and then we utilize the user latent representations for friend recommendation.
—*PTE* [Tang et al. 2015a]: PTE develops a semisupervised text embedding algorithm for unsupervised embedding learning by removing the supervised part and optimizing over the adjacency matrix and user-location co-occurrence matrix. PTE models a conditional probability $p(v_j|v_i)$ that indicates the probability that a given neighbor of v_i is v_j. We compute the conditional probabilities for friend recommendation.
—*TADW* [Yang et al. 2015]: TADW further extends DeepWalk to take advantage of the text information of a network. We can replace the text feature matrix in TADW with the user-location co-occurrence matrix by disregarding the sequential information of locations. TADW defines an affinity matrix where each entry of the matrix characterizes the strength of the relationship between corresponding users. We use the corresponding entries of the affinity matrix to rank candidate users for recommendation.

To construct the evaluation collection, we randomly select 20 to 50 of the existing connection links as the training set and leave the rest for the test set. We recommend

Table III. Results of Different Methods on Next-Location Recommendation

Dataset	Brightkite			Gowalla		
Metric (%)	R@1	R@5	R@10	R@1	R@5	R@10
PV	18.5	44.3	53.2	9.9	27.8	36.3
FBC	16.7	44.1	54.2	13.3	34.4	42.3
FPMC	20.6	45.6	53.8	10.1	24.9	31.6
PRME	15.4	44.6	53.0	12.2	31.9	38.2
HRM	17.4	46.2	56.4	7.4	26.2	37.0
JNTM	**22.1**	**51.1**	**60.3**	**15.4**	**38.8**	**48.1**

5 or 10 friends for each user and report Recall@5 and Recall@10. The final results are compared by varying the training ratio from 20% to 50%. Specifically, for each user v, we take all other users who are not her friends in the training set as the candidate users. Then we rank the candidate users and recommend the top 5 or 10 users with the highest-ranking scores. To obtain the ranking score of user v_j when we recommend friends for user v_i, DeepWalk and PMF adopt the cosine similarity between their user representations. For PTE, we use the conditional probability $p(v_j|v_i)$, which indicates the probability that a given neighbor of v_i is v_j as ranking scores. For TADW, we compute the affinity matrix A and use the corresponding entry A_{ij} as ranking scores. For our model, we rank users with the highest log likelihood according to Equation (1).

The baselines methods and our model involve an important parameter—that is, the number of latent (or embedding) dimensions. We use a grid search from 25 to 100 and set the optimal value using the validation set. Other parameters in baselines or our model can be tuned in a similar way. For our model, the learning rate and number of negative samples are empirically set to 0.1 and 100, respectively. We randomly initialize parameters according to uniform distribution $U(-0.02, 0.02)$.

All experiments are executed on a 12-core CPU server, and the CPU type is an Intel Xeon E5-2620 @ 2.0GHz.

5.3. Experimental Results on Next-Location Recommendation

Table III shows the results of different methods on next-location recommendation. Compared to FPMC and PRME, HRM models the sequential relatedness between consecutive subtrajectories, whereas the sequential relatedness in a subtrajectory is ignored. In the Brightkite dataset, the average number of locations in a subtrajectory is much less than that in the Gowalla dataset. Therefore, short-term sequential contexts are more important in the Gowalla dataset and less useful in the Brightkite dataset. Experimental results in Table III demonstrate this intuition: HRM outperforms FPMC and PRME on Brightkite, whereas PRME works best on Gowalla.

As shown in Table III, our model JNTM consistently outperforms the other baseline methods. JNTM yields 4.9% and 4.4% improvement on Recall@5 compared to the best baseline HRM on the Brightkite dataset and FBC on the Gowalla dataset. Recall that our model JNTM has considered four factors, including user preference, influence of friends, and short-term and long-term sequential contexts. All baseline methods only characterize user preference (or friend influence for FBC) and a single kind of sequential contexts. Thus, JNTM achieves the best performance on both datasets.

The preceding results are reported by averaging over all users. In recommender systems, an important issue is how a method performs in the cold-start setting (i.e., new users or new items). To examine the effectiveness on new users generating very few check-ins, we present the results of Recall@5 for users with no more than five subtrajectories in Table IV. In a cold-start scenario, a commonly used way to leverage the side information (e.g., user links [Cheng et al. 2012] and text information [Gao

Table IV. Results of Next-Location Recommendation Results for Users with No More Than Five Subtrajectories

Dataset	Brightkite			Gowalla		
Metric (%)	R@1	R@5	R@10	R@1	R@5	R@10
PV	13.2	22.0	26.1	4.6	7.8	9.2
FBC	9.0	29.6	39.5	4.9	12.0	16.3
FPMC	17.1	30.0	33.9	5.5	13.5	18.5
PRME	22.4	36.3	40.0	7.2	12.2	15.1
HRM	12.9	31.2	39.7	5.2	15.2	21.5
JNTM	**28.4**	**53.7**	**59.3**	**10.2**	**24.8**	**32.0**

Table V. Results of Different Methods on Next New Location Recommendation

Dataset	Brightkite			Gowalla		
Metric (%)	R@1	R@5	R@10	R@1	R@5	R@10
PV	0.5	1.5	2.3	1.0	3.3	5.3
FBC	0.5	1.9	3.0	1.0	3.1	5.1
FPMC	0.8	2.7	4.3	2.0	6.2	9.9
PRME	0.3	1.1	1.9	0.6	2.0	3.3
HRM	1.2	3.5	5.2	1.7	5.3	8.2
JNTM	**1.3**	**3.7**	**5.5**	**2.7**	**8.1**	**12.1**

et al. 2015; Li et al. 2010; Zhao et al. 2015]) to alleviate data sparsity. For our model, we characterize two kinds of user representations, either using network data or trajectory data. The user representations learned using network data can be exploited to improve the recommendation performance for new users to some extent. Indeed, networking representations have been applied to multiple network-independent tasks, including profession prediction [Tu et al. 2015] or text classification [Yang et al. 2015]. By utilizing the networking representations, the results indicate that our model JNTM is very promising to deal with next-location recommendation in a cold-start setting.

Note that the preceding experiments are based on general next-location recommendation, where we do not examine whether a recommended location has been previously visited or not by a user. To further test the effectiveness of our algorithm, we conduct experiments on the next new location recommendation task proposed by previous studies [Feng et al. 2015]. In this setting, we only recommend new locations when the user decide to visit a place. Specifically, we rank all locations that a user has never visited before for recommendation [Feng et al. 2015]. We present the experimental results in Table V. Our method consistently outperforms all baselines on the next new location recommendation in both datasets. By combining results in Tables III and IV, we can see that our model JNTM is more effective in the next-location recommendation task compared to these baselines.

In the preceding, we have shown the effectiveness of the proposed model JNTM on the task of next-location recommendation. Since trajectory data itself is sequential data, our model has leveraged the flexibility of RNNs for modeling sequential data, including both short-term and long-term sequential contexts. Now we study the effect of sequential modeling on the current task.

We prepare three variants for our model JNTM:

—JNTM$_{base}$, which removes both short-term and long-term contexts. It only employs the user interest representation and network representation to generate the trajectory data.
—JNTM$_{base+long}$, which incorporates the modeling for long-term contexts to JNTM$_{base}$.

Table VI. Performance Comparison for Three Variants of JNTM on Next-Location Recommendation

Dataset	Brightkite			Gowalla		
Metric (%)	R@1	R@5	R@10	R@1	R@5	R@10
JNTM$_{base}$	20.2	49.3	59.2	12.6	36.6	45.5
JNTM$_{base+long}$	20.4 (+2%)	50.2 (+2%)	59.8 (+1%)	13.9 (+10%)	36.7 (+0%)	45.6 (+0%)
JNTM$_{base+long+short}$	**22.1**(+9%)	**51.1**(+4%)	**60.3**(+**2**%)	**15.4**(+**18**%)	**38.8**(+6%)	**48.1**(+6%)

Table VII. Performance Comparison for Three Variants of JNTM on Next New Location Recommendation

Dataset	Brightkite			Gowalla		
Metric (%)	R@1	R@5	R@10	R@1	R@5	R@10
JNTM$_{base}$	0.8	2.5	3.9	0.9	3.3	5.5
JNTM$_{base+long}$	1.0 (+20%)	3.3 (+32%)	4.8 (+23%)	1.0 (+11%)	3.5 (+6%)	5.8 (+5%)
JNTM$_{base+long+short}$	**1.3**(+63%)	**3.7**(+48%)	**5.5**(+41%)	**2.7**(+200%)	**8.1**(+145%)	**12.1**(+120%)

Table VIII. Friend Recommendation Results on the Brightkite Dataset

Training Ratio	20%		30%		40%		50%	
Metric (%)	R@5	R@10	R@5	R@10	R@5	R@10	R@5	R@10
DeepWalk	2.3	3.8	3.9	6.7	5.5	9.2	7.4	12.3
PMF	2.1	3.6	2.1	3.7	2.3	3.4	2.3	3.8
PTE	1.5	2.5	3.8	4.7	4.0	6.6	5.1	8.3
TADW	2.2	3.4	3.6	3.9	2.9	4.3	3.2	4.5
JNTM	**3.7**	**6.0**	**5.4**	**8.7**	**6.7**	**11.1**	**8.4**	**13.9**

—JNTM$_{base+long+short}$, which incorporates the modeling for both short-term and long-term contexts to JNTM$_{base}$.

Tables VI and VII show the experimental results of three JNTM variants on the Brightkite and Gowalla datasets. The numbers in the brackets indicate the relative improvement against JNTM$_{base}$. We can observe a performance ranking: JNTM$_{base}$ < JNTM$_{base+long}$ < JNTM$_{base+long+short}$. The observations indicate that both kinds of sequential contexts are useful to improve the performance for next-location recommendation. In general, for next-location recommendation (i.e., both old and new locations are considered for recommendation), we can see that the improvement from the short- and long-term contexts is not significant. The explanation is that a user is likely to show repeated visit behaviors (e.g., visiting the locations that have been visited before), and thus user preference is more important than sequential context to improve the recommendation performance. For next new location recommendation, the sequential context, especially the short-term context, yields a large improvement margin over the baseline. These results indicate that the sequential influence is more important than user preference for new location recommendation. Our finding is also consistent with previous work [Feng et al. 2015]—that is, sequential context is important to consider for next new location recommendation.

5.4. Experimental Results on Friend Recommendation

We continue to present and analyze the experimental results on the task of friend recommendation. Tables IX and VIII show the results when the training ratio varies from 20% to 50%.

Among the baselines, DeepWalk performs best and even better than the baselines using both networking data and trajectory data (PTE and TADW). A major reason is that DeepWalk is tailored to the reconstruction of network connections and adopts a distributed representation method to capture the topology structure. As indicated in other following studies [Perozzi et al. 2014; Tang et al. 2015b], distributed representation

Table IX. Friend Recommendation Results on the Gowalla Dataset

Training Ratio	20%		30%		40%		50%	
Metric (%)	R@5	R@10	R@5	R@10	R@5	R@10	R@5	R@10
DeepWalk	2.6	3.9	5.1	8.1	**7.9**	**12.1**	**10.5**	**15.8**
PMF	1.7	2.4	1.8	2.5	1.9	2.7	1.9	3.1
PTE	1.1	1.8	2.3	3.6	3.6	5.6	4.9	7.6
TADW	2.1	3.1	2.6	3.9	3.2	4.7	3.6	5.4
JNTM	**3.8**	**5.5**	**5.9**	**8.9**	**7.9**	11.9	10.0	15.1

Table X. Friend Recommendation Results for Users with Fewer than Five Friends When the Training Ratio Is 50%

Dataset	Brightkite		Gowalla	
Metric (%)	R@5	R@10	R@5	R@10
DeepWalk	14.0	18.6	19.8	23.5
JNTM	**16.1**	**20.4**	**21.3**	**25.5**

learning is particularly effective in network embedding. Although PTE and TADW utilize both network and trajectory data, their performance is still low. These two methods cannot capture the sequential relatedness in trajectory sequences. Another observation is that PMF (i.e., factorizing the user-location matrix) is better than PTE at a ratio of 20% but becomes the worst baseline, because PMF learns user representations using the trajectory data, and the labeled data (i.e., links) is mainly used for training a classifier.

Our algorithm is competitive with the state-of-the-art network embedding method DeepWalk and outperforms DeepWalk when the network structure is sparse. The explanation is that trajectory information is more useful when network information is insufficient. As the network becomes dense, the trajectory information is not as useful as the connection links. To demonstrate this explanation, we further report the results for users with fewer than five friends when the training ratio is 50%. As shown in Table X, our methods have yielded 2.1% and 1.5% improvement over DeepWalk for these inactive users on the Brightkite and Gowalla datasets, respectively. The results indicate that trajectory information is useful to improve the performance of friend recommendation for users with very few friends.

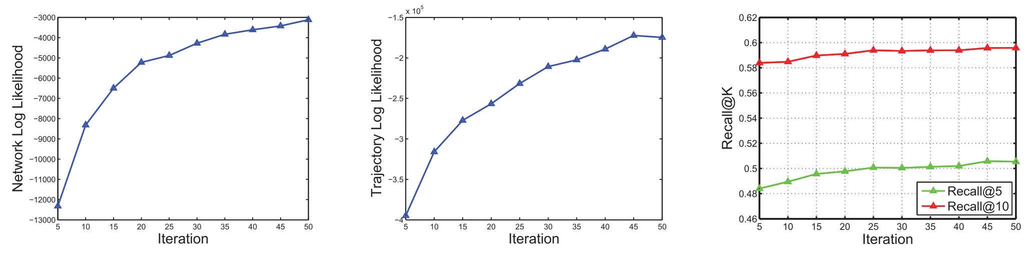
In summary, our method significantly outperforms existing state-of-the-art methods on both next-location prediction and friend recommendation. Experimental results on both tasks demonstrate the effectiveness of our proposed model.

5.5. Parameter Tuning

In this section, we study how different parameters affect the performance of our model. We mainly select two important parameters: the number of iterations and and the number of embedding dimensions.

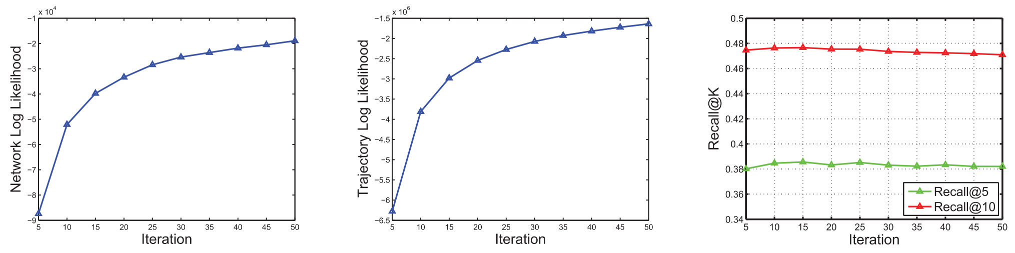
We conduct the tuning experiments on the training sets by varying the number of iterations from 5 to 50. We report the log likelihood for the network and trajectory data on the training sets and Recall@5 and Recall@10 of next-location recommendation on the validation sets.

Figures 6 and 7 show the tuning results of the iteration number on both datasets. From the results, we can see that our algorithm can converge within 50 iterations on both datasets, and the growth of log likelihood slows down after 30 iterations. However, the performance of next-location recommendation on the validation sets is relatively stable: JNTM can yield a relatively good performance after 5 iterations. The recall values increase slowly and reach the highest score at 45 iterations on Brightkite



(a) Network log likelihood (b) Trajectory log likelihood. (c) Recall@K

Fig. 6. Performance of the iteration number on the Brightkite dataset.



(a) Network log likelihood (b) Trajectory log likelihood (c) Recall@K

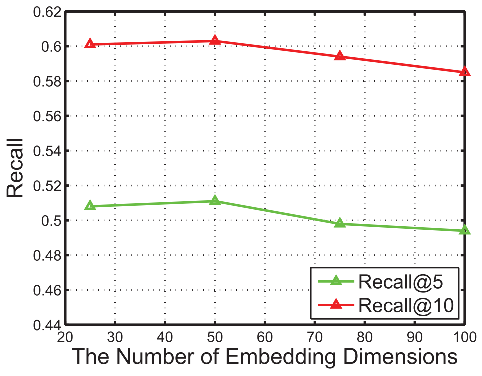
Fig. 7. Performance of the iteration number on the Gowalla dataset.

and 15 iterations on Gowalla. Here, the Gowalla dataset converges more quickly and smoothly than Brightkite, mainly because the Gowalla dataset contains three times more check-in data than that of Brightkite and has more than enough training data. However, the model may overfit before it gets the highest recall for next-location recommendation because the recall scores are not always monotonically increasing. In addition, the performance on new location prediction begins to drop after about 10 iterations. To avoid the overfitting problem, we reach a compromise and find that iteration numbers of 15 and 10 are a reasonably good choice to give good performance on Brightkite and Gowalla, respectively.

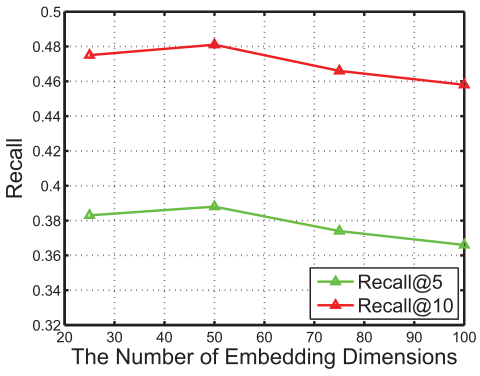
The number of embedding dimensions is also vital to the performance of our model. A large dimension number will have a strong expressive ability but will also probably lead to overfitting. We conduct experiments with different embedding dimension numbers on next-location recommendation and measure their performance on validation sets. In Figure 8, we can see that the performance of our algorithm is relatively stable when we vary the dimension number from 25 to 100. The recall values start to decrease when the dimension number exceeds 50. We finally set the dimension number to 50.

5.6. Scalability

In this section, we conduct experiments on scalability and examine the time and space costs of our model. We perform the experiments on the Gowalla dataset and select the baseline method PRME [Feng et al. 2015] for comparison. We report the memory usage of both methods for space complexity analysis and running time on a single CPU for time complexity analysis. Since both methods have the same iteration number for convergence, we report the average running time per iteration for each algorithm. The running time for training and testing is presented separately. A major merit of neural network models is that they can be largely accelerated by supported hardware (e.g., the GPU). Therefore, we also report the running time of a variation of our model using



(a) Brightkite



(b) Gowalla

Fig. 8. Performance tuning with different dimension numbers.

Table XI. Experimental Results on Memory Usage (MBs) and Running Time (Minutes)

Model	*Memory*	*Training*	*Testing*	*Training (GPU)*
PRME	550	2	52	—
JNTM	1,125	107	82	9

GPU acceleration. Specifically, we use a single Tesla K40 GPU for training our model in the TensorFlow[6] software library. The experimental results are shown in Table XI.

From Table XI, we can see that our memory usage is almost twice as much as PRME. This is mainly because we set two representations for each user (i.e., friendship and preference representations), whereas PRME only has a preference representation. The time complexity of JNTM is $O(d^2|D| + d|V|)$, whereas the time complexity of PRME is $O(d|D|)$, where d is the embedding dimensionality ($d = 50$ in our experiments). Hence, the running time of JNTM is about d times as much as that of PRME. Although our model has a longer training time than PRME, the time cost of JNTM for testing is almost equivalent to that of PRME. On average, JNTM takes less than 30ms for a single location prediction, which is efficient to provide an online recommendation service after the training stage. Moreover, the GPU acceleration offers 12× speedup for the training process, which demonstrates that our model JNTM can be efficiently learned with supported hardware.

6. CONCLUSION AND FUTURE WORK

In this article, we presented a novel neural network model by jointly modeling both social networks and mobile trajectories. Specifically, our model consisted of two components: the construction of social networks and the generation of mobile trajectories. We first adopted a network embedding method for the construction of social networks. We considered four factors that influence the generation process of mobile trajectories: user visit preference, influence of friends, short-term sequential contexts, and long-term sequential contexts. To characterize the last two contexts, we employed the RNN and GRU models to capture the sequential relatedness in mobile trajectories at different levels (i.e., short term or long term). Finally, the two components were tied by sharing the user network representations. On two important application tasks, our model was consistently better than several competitive baseline methods. In our approach,

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

network structure and trajectory information complemented each other. Hence, the improvement over baselines was more significant when either network structure or trajectory data is sparse.

Currently, our model does not consider the GPS information—that is, a check-in record is usually attached with a pair of longitude and latitude values. Our current focus mainly lies in how to jointly model social networks and mobile trajectories. As future work, we will study how to incorporate the GPS information into the neural network models. In addition, the check-in location can be also attached with categorical labels. We will also investigate how to leverage such semantic information to improve performance. This semantic information can be utilized for the explanation of the generated recommendation results. Our current model provides a flexible neural network framework to characterize LBSN data. We believe that it will inspire more follow-up studies along this direction.

ACKNOWLEDGMENTS

The authors thank the anonymous reviewers for their valuable and constructive comments.

REFERENCES

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

Mikhail Belkin and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic (NIPS'01)*. 585–591.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 8, 1798–1828.

Preeti Bhargava, Thomas Phan, Jiayu Zhou, and Juhan Lee. 2015. 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 (WWW'15)*. ACM, New York, NY, 130–140.

Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM'15)*. 891–900.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM Computing Surveys* 41, 3, Article No. 15.

Mo Chen, Qiong Yang, and Xiaoou Tang. 2007. Directed graph embedding. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*. 2707–2712.

Chen Cheng, Haiqin Yang, Irwin King, and Michael R. Lyu. 2012. Fused matrix factorization with geographical and social influence in location-based social networks. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12)*, Vol. 12. 17–23.

Chen Cheng, Haiqin Yang, Michael R. Lyu, and Irwin King. 2013. Where you like to go next: Successive point-of-interest recommendation. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*. 2605–2611.

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

Yoon-Sik Cho, Greg Ver Steeg, and Aram Galstyan. 2013. Socially relevant venue clustering from check-in data. In *Proceedings of the KDD Workshop on Mining and Learning with Graphs*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555.

Justin Cranshaw, Eran Toch, Jason Hong, Aniket Kittur, and Norman Sadeh. 2010. Bridging the gap between physical location and online social networks. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*. ACM, New York, NY, 119–128.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, 2121–2159.

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

Santo Fortunato. 2010. Community detection in graphs. *Physics Reports* 486, 3-5, 75–174.

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

Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. 2015. Content-aware point of interest recommendation on location-based social networks. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*. 1721–1727.

Hong Huang, Jie Tang, Sen Wu, Lu Liu, and Xiaoming Fu. 2014. Mining triadic closure patterns in social networks. In *Proceedings of the 23rd International Conference on World Wide Web (WWW'14)*. ACM, New York, NY, 499–504.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS'12)*. 1097–1105.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *Computer Science* 4, 1188–1196.

Min-Joong Lee and Chin-Wan Chung. 2011. A user similarity calculation based on the location for social network services. In *Proceedings of the International Conference on Database Systems for Advanced Applications*. 38–52.

Justin J. Levandoski, Mohamed Sarwat, Ahmed Eldawy, and Mohamed F. Mokbel. 2012. LARS: A location-aware recommender system. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*. IEEE, Los Alamitos, CA, 450–461.

Quannan Li, Yu Zheng, Xing Xie, Yukun Chen, Wenyu Liu, and Wei-Ying Ma. 2008. Mining user similarity based on location history. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, New York, NY, 34.

Yize Li, Jiazhong Nie, Yi Zhang, Bingqing Wang, Baoshi Yan, and Fuliang Weng. 2010. Contextual recommendation based on text mining. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. 692–700.

David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology* 58, 7, 1019–1031.

Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*.

Hao Ma. 2014. On measuring social friend interest similarities in recommender systems. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 465–474.

Ashwin Machanavajjhala, Aleksandra Korolova, and Atish Das Sarma. 2011. Personalized social recommendations: Accurate or private. *Proceedings of the VLDB Endowment* 4, 7, 440–450.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of the International Conference on Spoken Language Processing (INTERSPEECH'10)*, Vol. 2. 3.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*. 3111–3119.

Sparsh Mittal. 2016. A survey of techniques for approximate computing. *ACM Computing Surveys* 48, 4, 62.

Andriy Mnih and Ruslan Salakhutdinov. 2007. Probabilistic matrix factorization. In *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS'07)*. 1257–1264.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14)*. 701–710.

Huy Pham, Ling Hu, and Cyrus Shahabi. 2011. Towards integrating real-world spatiotemporal data with social networks. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, New York, NY, 453–457.

Huy Pham, Cyrus Shahabi, and Yan Liu. 2013. 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*. ACM, New York, NY, 265–276.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*. 811–820.

Daniel Mauricio Romero and Jon M. Kleinberg. 2010. The directed closure process in hybrid social-information networks, with an analysis of link formation on Twitter. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media (ICWSM'10)*. 138–145.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Magazine* 29, Article No. 3.

Jian Tang, Meng Qu, and Qiaozhu Mei. 2015a. PTE: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'15)*. 1165–1174.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015b. LINE: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web (WWW'15)*. 1067–1077.

Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09)*. 817–826.

Lei Tang and Huan Liu. 2011. Leveraging social media networks for classification. *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*.

Cunchao Tu, Zhiyuan Liu, and Maosong Sun. 2015. PRISM: Profession identification in social media with personal information and community structure. In *Proceedings of the Chinese National Conference on Social Media Processing*. 15–27.

Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. 2016. Max-margin DeepWalk: Discriminative learning of network representation. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. 3889–3895.

Dashun Wang, Dino Pedreschi, Chaoming Song, Fosca Giannotti, and Albert-Laszlo Barabasi. 2011. Human mobility, social ties, and link prediction. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*. ACM, New York, NY, 1100–1108.

Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for NextBasket recommendation. In *Proceedings of of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'15)*. 403–412.

Paul J. Werbos. 1990. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE* 78, 10, 1550–1560.

Xiangye Xiao, Yu Zheng, Qiong Luo, and Xing Xie. 2010. Finding similar users using category-based location history. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, New York, NY, 442–445.

Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. 2007. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, 1, 40–51.

Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. 2015. Network representation learning with rich text information. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*. 2111–2117.

Jihang Ye, Zhe Zhu, and Hong Cheng. 2013. What's your next move: User activity prediction in location-based social networks. In *Proceedings of the SIAM International Conference on Data Mining.*

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

Hongzhi Yin, Yizhou Sun, Bin Cui, Zhiting Hu, and Ling Chen. 2013. LCARS: A location-content-aware recommender system. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, 221–229.

Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014a. COM: A generative model for group recommendation. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, 163–172.

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

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

Jia-Dong Zhang, Chi-Yin Chow, and Yanhua Li. 2014a. LORE: Exploiting sequential influence for location recommendations. In *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, New York, NY, 103–112.

Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014b. Sequential click prediction for sponsored search with recurrent neural networks. arXiv:1404.5772.

Kaiqi Zhao, Gao Cong, Quan Yuan, and Kenny Q. Zhu. 2015. SAR: A sentiment-aspect-region model for user preference analysis in geo-tagged reviews. In *Proceedings of the 2015 IEEE 31st International Conference on Data Engineering*. IEEE, Los Alamitos, CA, 675–686.

Wayne Xin Zhao, Ningnan Zhou, Wenhui Zhang, Ji-Rong Wen, Shan Wang, and Edward Y. Chang. 2016. A probabilistic lifestyle-based trajectory model for social strength inference from human trajectory data. *ACM Transactions on Information Systems* 35, 1, 8.

Vincent W. Zheng, Yu Zheng, Xing Xie, and Qiang Yang. 2010. Collaborative location and activity recommendations with GPS history data. In *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*. ACM, New York, NY, 1029–1038.

Yu Zheng. 2015. Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology* 6, 3, 29.

Yu Zheng, Xing Xie, and Wei-Ying Ma. 2010. GeoLife: A collaborative social networking service among user, location and trajectory. *IEEE Data Engineering Bulletin* 33, 2, 32–39.

Yu Zheng, Lizhu Zhang, Zhengxin Ma, Xing Xie, and Wei-Ying Ma. 2011. Recommending friends and locations based on individual location history. *ACM Transactions on the Web* 5, 1, 5.

Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. 2009. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th International Conference on World Wide Web (WWW'09)*. ACM, New York, NY, 791–800.

Ningnan Zhou, Wayne Xin Zhao, Xiao Zhang, Ji-Rong Wen, and Shan Wang. 2016. A general multi-context embedding model for mining human trajectory data. *IEEE Transactions on Knowledge and Data Engineering* 28, 8, 1945–1958.

Received June 2016; revised December 2016; accepted January 2017