Efficient Point-of-Interest Recommendation Services with Heterogenous Hypergraph Embedding

Chen Wang, *Senior Member, IEEE*, Mengting Yuan, Rui Zhang, *Member, IEEE*, Kai Peng, and Ling Liu, *Fellow, IEEE*

Abstract—Point-of-interest (POI) recommendation service has drawn growing attention with the widespread popularity of location-based social networks (LBSNs). Recent research methods on POI recommendation based on graph embedding have mainly focused on explicit interactions of LBSN objects such as user's check-ins on POIs and social relationships, while neglecting implicit relationship that cannot be directly observed but may notably contribute to the POI recommendation. This paper presents VirHpoi, a heterogeneous hypergraph embedding method for POI recommendation in LBSNs with three original contributions. First, we model the LBSNs as a hypergraph to capture the complex interactions in LBSNs and learn the hypergraph by preserving homophily and interaction attribute affinity of the LBSNs. Second, we introduce the notion of "virtual hyperedges" to capture the intrinsic correlations of POIs. Virtual hyperedges incorporate implicit yet informative connections of the check-in patterns in LBSNs in terms of geographical and semantic characteristics. Third, we propose techniques to learn heterogenous hypergraph embedding on the complex LBSN graph with both homogenous edges and heterogenous hyperedges with dual objectives: we aim to preserve the homophily of objects intra domain by maximizing the co-occurrence probability of all homogenous edges, and we want to learn the interaction attribute affinity across domains by maximizing the probability of predicting the target object in the hyperedges. As a result, our approach can preserve both the intra domain homophily of objects and the interaction attribute affinity across domains by learning low-dimensional embeddings of LBSN objects and then make more effective recommendations based on the embeddings. Extensive experiments on four real-world datasets show the effectiveness and superiority of VirHpoi compared with the state-of-the-art methods.

Index Terms—POI recommendation, location-based social networks, graph embedding, heterogeneous hypergraph.

1 Introduction

Location-based social networks (LBSNs) [1]–[3] continue to receive growing attention with the popularity of smart mobile devices and the advancement of location acquisition technologies. Millions of users engage in LBSN services like Facebook, Foursquare, to name a few. In order to improve service experience for users, the point-of-interest (POI) recommendation service [4]–[6] which aims to recommend new and potentially attractive POIs to users has gained great research interest, as it can benefit not only users, but also advertising agencies for effective advertisements.

Studies on POI recommendation mainly consider four influencing factors: geographical influence, social relations, temporal dynamics and activity categories. Early works usually consider these factors separately, and combine their corresponding results together to perform recommendation [7]–[10]. Recent advances of representation learning in networks (a.k.a. graph embedding) provide an opportunity to exploit and integrate these influencing factors [11]–[13], by modeling the LBSNs as a graph and embedding each node of the network into a low-dimensional latent space while preserving the key topological information.

While most of existing graph embedding works [14]–[16] model the LBSNs as several relational bipartite graphs or regard the LBSNs as a graph with only pair-wise connections (c.f. Fig. 1(a)), LBSNs are inherently heterogenous which have various types of relations and objects including users, POIs, categories and time slots. Therefore, simplifying the essential tuple-wise relationship into pair-wise one cannot fully capture the information from the check-ins, and may degrade the joint interactions of all objects from four domains in a check-in record into the interactions of several pair-wise objects, which will inevitably lead to the loss of structure information [17], [18].

To rectify such drawbacks, recent works [19], [20] propose to adopt the hypergraph to better describe the tuple-wise relationship, where check-ins are modeled as hyperedges and friendships are modeled as classical edges. In their designs, however, they only concentrate on explicit check-ins and friendships, while neglecting implicit relationships that cannot be directly observed but may notably contribute to the recommendation. In particular, there are classical edges in social domain and hyperedges linking four nodes, one from each domain, but no direct connection exists in other domains. In practice, there often exists correlation between POIs and they are not completely independent.

- *This work was supported in part by the National Natural Science Foundation of China under Grants 61872416, 2171189, 52031009, and U20A20181; by the Fundamental Research Funds for the Central Universities of China under Grant 2019kfyXJJS017; and by the special fund for Wuhan Yellow Crane Talents (Excellent Young Scholar). Ling Liu's research is partially sponsored by the National Science Foundation CISE grants 2038029, 2026945, 1564097, an IBM faculty award, and a Cisco grant on Edge Computing. (Corresponding author: Kai Peng).*
- *C. Wang, M. Yuan and K. Peng are with the Hubei Key Laboratory of Smart Internet Technology, School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China. Email: {chenwang, mtyuan, pkhust}@hust.edu.cn.*
- *R. Zhang is with the Hubei Key Laboratory of Transportation Internet of Things, School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430070, China. Email: zhangrui@whut.edu.cn.*
- *L. Liu is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0765, USA. Email: lingliu@cc.gatech.edu.*

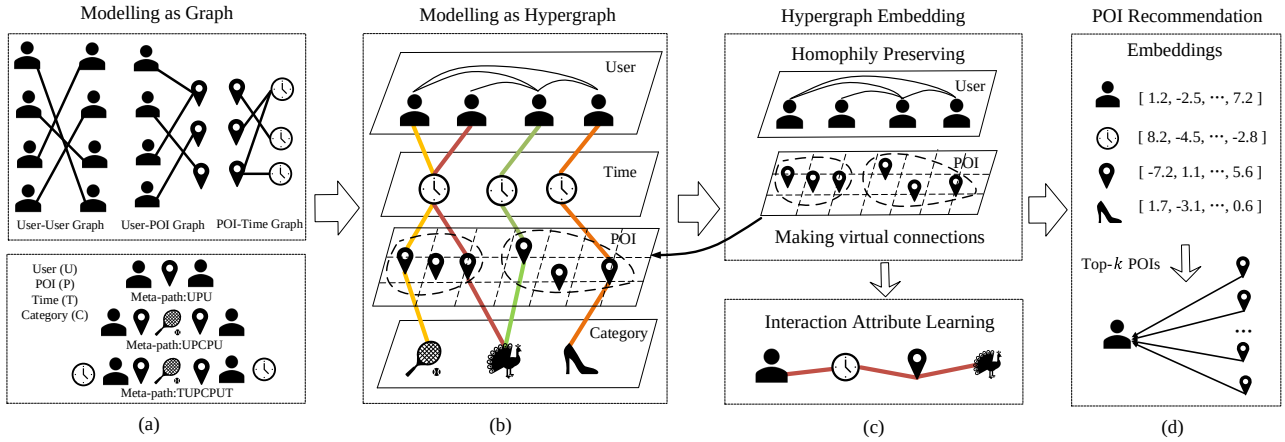


Fig. 1. (a) The LBSNs are modeled as bipartite graphs or heterogenous graphs with only pair-wise connections, where the joint interactions of all objects from four domains (user-time-POI-category) in a check-in record are degraded into the interactions of several pair-wise objects. (b) We model the LBSNs as a heterogenous hypergraph and introduce virtual hyperedges (clusters containing multiple objects) to incorporate implicit relationships in the LBSNs. Friendships are represented by classical edges (black lines) linking two users, while check-ins are modeled as hyperedges (colored thick line) linking four nodes, one from each domain. (c) We learn heterogenous hypergraph embedding by preserving both the homophily of objects intra domain and the interaction attribute affinity across domains. (d) Top-k unvisited POIs are recommended to each user based on the heterogenous hypergraph embeddings.

For example, a classroom is more related to a library than a gym. Some POIs may be of different categories, but they have correlations and similar appeals to users. Therefore, the correlations of POIs are much useful for recommendation tasks and should be taken into consideration. The existing works only concentrate on explicit interaction links. We aim to leverage these enriched correlations for improving new POI recommendation.

In this paper, we address the above problems by developing a novel heterogeneous hypergraph embedding method for POI recommendation in LBSNs, coined as VirHpoi. Our key idea is to model the LBSNs as a heterogenous hypergraph and introduce the notion of the "virtual hyperedges" to incorporate implicit yet informative connections of the LBSNs. Intuitively, users are more likely to check in on new POIs that have a higher degree of relationship with their previously visited POIs, and new POI recommendation would benefit from the embedding of POIs with high correlations. Moreover, the virtual connections are conducive to shorten the distance between users and potentially appealing but unvisited POIs and thus help make more precise recommendation. Besides, the graph based methods are potentially more vulnerable to data sparsity. The virtual connections will help deal with the data sparsity issue to an extent.

Although the idea sounds simple, there are two technical challenges to be addressed. The first challenge is how to make effective virtual connections, which plays an important role in improving the performance of the POI recommendation. To tackle this issue, we analyze and reveal the check-in patterns in terms of geographical and semantic characters using real-world data, and further propose two kinds of similarity metrics to quantify the corresponding two factors, based on which we can perform valid virtual connections. The constructed virtual hyperedges contain multiple objects which have implicit relatedness. They are in fact potential neighbors that may not be directly reachable on graphs. The second challenge is how to learn the heterogenous hypergraph graph embedding on such complex graph with both homogenous edges and heterogenous hyperedges. To cope with this problem, we maximize the co-occurrence probability of all homogenous edges to preserve the homophily of objects intra domain, and learn the interaction attribute affinity across domains by maximizing the probability of predicting the target object in the hyperedges. As such, we can preserve interaction information of the LBSNs by learning low-dimensional embeddings of the objects and then conduct effective POI recommendations.

Our major contributions are summarized as follows:

- We propose a heterogenous hypergraph embedding based POI recommendation method, which models the LBSNs as a hypergraph to capture the complex interactions in the LBSNs and learns the hypergraph by preserving homophily and interaction attribute affinity of the LBSNs.
- We incorporate implicit connections of POIs by establishing virtual hyperedges in the LBSN heterogenous hypergraph, which enables more efficient embedding from latent geographical and semantic characters in the LBSN hypergraph.
- We conduct extensive experiments to evaluate the performance of VirHpoi on four real-world datasets. The results show the effectiveness and superiority of VirHpoi compared with the state-of-the-art methods.

We should emphasize that our virtual hyperedges are not limited to the POI domain. In other domains, such as user or category domain, we can also design appropriate metrics to establish virtual hyperedges, so as to incorporate more implicit correlations which introduce latent patterns of objects in the LBSNs to recommendation model and further improve the recommendation performance.

The remainder of this paper is organized as follows. In Section 2, we analyze the geographical and semantic characters of the LBSNs for better establishing virtual connections.

TABLE 1
Statistic of Datasets

Dataset	#Users	#POIs	#Check-ins	#Friendships
New York	12,062	11,422	443,284	14,346
Tokyo	14,441	16,265	1,311,614	40,252
Istanbul	16,925	12,780	650,451	2,466
Jakarta	11,407	11,184	502,540	14,998

Section 3 details the design of the heterogenous hypergraph embedding. We report our experiment results in Section 4. Section 5 reviews some related works, and finally Section 6 concludes the paper. The code of VirHpoi has been released for reproducibility purposes[1].

2 Empirical Data Analysis

In this section, we conduct an empirical data analysis to reveal check-in patterns of the LBSNs in terms of geographical and semantic factors for incorporating implicit relationships in the hypergraph embedding, using the global-scale check-in data collected in [21], [22] from Foursquare between Apr. 2012 and Jan. 2014. In our analysis, we take cities as basic units and select four different datasets with large numbers of check-ins from the raw check-ins: New York City (NYC), Tokyo (TKY), Istanbul (IST) and Jakarta (JK). Each check-in record contains a user, a POI, and a corresponding timestamp; descriptive information of POIs such as longitude, latitude and categories are also available. Each dataset also contains a snapshot of socially connected friendship. Following previous work [16], we select users who have check-ins at least 5 times and POIs with more than 10 visitors to avoid very inactive users and POIs. The detail statistics of the datasets are shown in Table 1.

2.1 Geographical Characters of the LBSNs

Geography is an important factor that distinguishes POI recommendation from traditional item recommendation, as the check-in behavior is closely related to locations geographical features. Intuitively, people tend to visit neighboring POIs subject to geographical constraints. We can infer that the influence of geographical adjacent on user check-ins follows a certain pattern, which can be utilized for POI recommendation. We perform a spatial analysis on the four datasets by measuring the likelihood that two of a user's check-ins are within a given distance. Specifically, to obtain the likelihood, we calculate the geographical distances between all pairs of check-ins by the same user and plot a histogram to show the statistics.

Fig. 2 shows the results of geographical influence on check-ins on the four datasets. From the results, we can observe an obvious geographical clustering phenomenon. As the distance between two POIs increases, the probability of a user checking in from one POI to another decreases. Comparing to distant places, users are more inclined to visit closer POIs or POIs near those already being visited.

1. https://www.dropbox.com/s/ey79crqa0xrdg8l/VirHpoi-Code.zip?dl=0

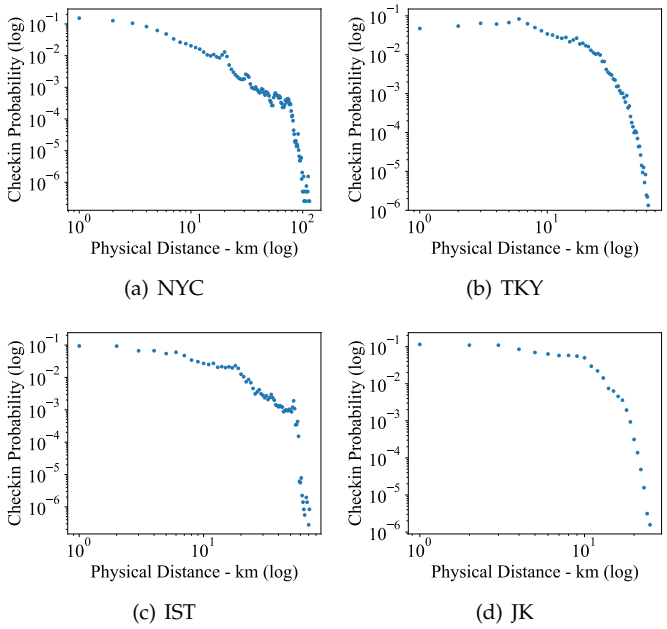


Fig. 2. Geographical influence probability distribution on four datasets. Our piece-wise function can fit the city-scale large datasets better.

Considering the data characteristics, we propose to employ the power-law distribution to model the geographical influence of the user's check-in behavior on POIs [23]. It can be observed from Fig. 2, however, that the check-in probability of POIs visited by the same user over distance is not a standard power-law distribution. That is because our datasets include POIs within the city range, and the geographical range is not very large. Therefore, unlike existing works [24]–[26] which usually directly use power-law distribution to fit all check-ins of a platform, we test it on city-scale large datasets and design a piece-wise function according to the tendency in our results, which is formulated as follows:

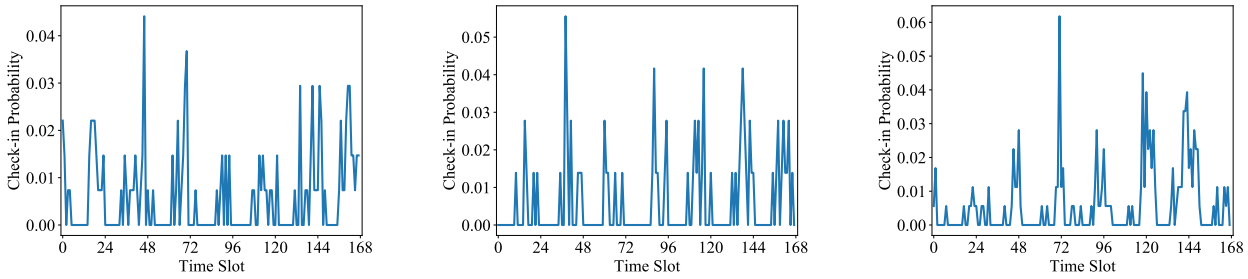
$$f_g(p_n, p_m) = \begin{cases} a_1 \cdot d(p_n, p_m)^{b_1} & 0 < d(p_n, p_m) < d_1 \\ a_2 \cdot d(p_n, p_m)^{b_2} & d_1 \leq d(p_n, p_m) < d_2 \\ a_3 \cdot d(p_n, p_m)^{b_3} & d_2 \leq d(p_n, p_m) \end{cases} \quad (1)$$

where a_i, b_i and d_i are parameters to be learned, and $d(p_n, p_m)$ refers to the physical distance between POI p_n and p_m. Physical distances that are less than 0.01 km are treated as 0.01 km. $f_g(p_n, p_m)$ represents the calculated probability of a user checking in from p_n to p_m. From the decreasing curve in Fig. 2, we can infer that the calculated parameter $b < 0$. So, when $d(p_n, p_m)$ is small enough, the calculated $f_g(p_n, p_m)$ may vary in a large scale.

The probability $f_g(p_n, p_m)$ that a user checks in from one POI to another can be used to quantify the geographical similarity of POIs. For our POI recommendation task, if a user is more likely to check in from one POI to another within a certain distance, the geographical similarity between the two POIs is higher. So we describe the geographical similarity of POIs using:

$$sim_g(p_n, p_m) = f_g(p_n, p_m) \quad (2)$$

In this way, we convert the calculation of the geograph-



(a) French Restaurant (b) Sandwish Place (c) Sports Bar

Fig. 3. Weekly check-in traffic pattern of three typical POIs (168 hours in a week).

ical similarity of POIs into calculating the probability of the user's checking in from one POI to another within a certain distance.

2.2 Semantic Characters of the LBSNs

Different POIs have their own characteristics and the relationship of POIs cannot be well explained only by their physical distance, so we further explore the semantic characters of the LBSNs. The semantic character here refers to the check-in distribution followed by POIs, and we consider it in terms of check-in traffic distribution. On the one hand, the check-in traffic distribution is related to the property of POIs. POIs with similar property often have similar appeals for users. In this case, we are not confined to the categories of POIs, and POIs of different categories may be semantically similar. On the other hand, most of users' daily schedules often remain stable. POIs with high check-in traffic distribution similarity with user daily activity are more in line with users' check-in habits, so users are more likely to check in on POIs with similar check-in traffic distributions.

To measure the semantic similarity between POIs, a straightforward way is to compare all historical check-in traffic distribution of each POI. However, the time range for all check-ins is too large and the data is highly sparse, while both the user behavior and the check-in traffic pattern show periodicity. As such, we study the traffic distribution of users' check-in on a weekly basis. Note that different from existing works [27], [28] which focus on calculating the probability of user check-in on one POI at a specific time, we here pay more attention to the check-in time patterns from the perspective of POIs, instead of users, and compare POI similarity distribution by the patterns to illustrate the rationality of using check-in traffic distribution to compare POIs.

In particular, we model the check-in data by representing each POI as a 168-dimensional vectors, where 168 is the weekly time units (168 time slots for 168 hours of a week), and each element denotes the probability of check-ins on the POI. Formally, the check-in traffic information for each POI p_n is denoted as a vector $f_n = [f(1)...f(i)...f(168)]$, where $f(i)$ denotes the frequency of check-ins. The semantic similarity between two POIs p_n and p_m can be calculated by comparing check-in traffic distribution using the cosine

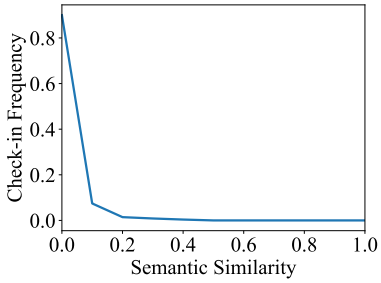


Fig. 4. Distribution of check-in semantic similarity of POIs.

similarity of the vectors:

$$sim_f(f_n, f_m) = \frac{f_n \cdot f_m}{\|f_n\|\|f_m\|} \quad (3)$$

Fig. 3 gives three typical examples of weekly traffic pattern for specific POIs. It can be observed that different POIs have different traffic patterns. We next measure the semantic similarity by Eq. (3) of these three POIs, and find that the semantic similarity of Sandwish Place and Sports Bar is 0.2489, while that of the Sandwish Place and French Restaurant is 0.4562. Intuitively, the correlation of Sandwish Place and French Restaurant is higher, and they should be embedded more closely. This is consistent with our life experience, as they are all about food and the semantic similarity of them should be higher, which demonstrates the practical rationality of using check-in traffic distribution to describe the semantic feature of LBSN check-ins.

Fig. 4 further shows the statistics of check-in semantic similarity of different POIs. It can be observed that almost all POI pairs' semantic similarity are between 0 and 0.15 and there is nearly no identical check-in traffic distribution of POIs. This indicates that there are distinct differences between different POIs in semantic similarity and we can thus distinguish different POIs by their semantic characters.

3 VirHPoi Design

3.1 Problem Formation

Before we formulate our problem, we first present some notations and definitions. Formally, let $\mathcal{U} = \{u_1, u_2, ...\}$ be the set of users, $\mathcal{P} = \{p_1, p_2, ...\}$ be the set of POIs, and $\mathcal{C} = \{c_1, c_2, ...\}$ be the set of categories where c_i denotes the category for p_i. Each POI p_i has a location $l_j = \{lon_j, lat_j\}$

representing its longitude and latitude. When a user initiates a check-in on a POI p_i, an interaction is formed. The user check-in time is discrete and the time slot set is denoted as $\mathcal{T} = \{t_1, t_2, ...\}$. Then our LBSN hypergraph with virtual hyperedges is defined as follows.

Definition 1 (Friendship Edge). A friendship edge is defined as $e_f =< u_1, u_2 >$ where users u_1 and u_2 are friends. $\mathcal{E}_f$ is the set of friendship edges.

Definition 2 (Check-in Hyperedge). A check-in record is defined as a four tuple $e_c =< u_1, t_1, p_1, c_1 >$ that depicts a user u_1 visiting POI p_1 at time t_1 and the category of p_1 is c_1. The tuple is regarded as a hyperedge which connects four different kinds of objects simultaneously. $\mathcal{E}_c$ is the set of check-in hyperedges.

Definition 3 (LBSN Hypergraph [19]). The friendship edges and check-in hyperedges actually form a heterogeneous hypergraph $G = (\mathcal{V}, \mathcal{E}_f, \mathcal{E}_c)$ where $\mathcal{V} = \mathcal{U} \cup \mathcal{T} \cup \mathcal{P} \cup \mathcal{C}$ consists of four data domains, i.e., user, temporal, POI and category.

Definition 4 (Virtual Hyperedge). A virtual hyperedge is defined as $e_v =< p_v^1, p_v^2, \dots p_v^L >$, where $p_v^1 \dots p_v^L$ are POIs with implicit correlation; L represents the number of POIs in the hyperedge e_v. $\mathcal{E}_v$ is the set of virtual hyperedges. A larger L means more new POIs accessible by hyperedges as well as more historical check-ins for each user.

Definition 5 (LBSN Hypergraph with Virtual Hyperedge). The constructed LBSN hypergraph with virtual hyperedges is defined as $G = (\mathcal{V}, \mathcal{E}_f, \mathcal{E}_c, \mathcal{E}_v)$ which includes both explicit relationship reflected by LBSN hypergraph and implicit relationship reflected by virtual hyperedges. The LBSN hypergraph includes both homogenous edges within user and POI domains and heterogeneous hyperedges across all the four domains.

We can then formulate our problem as follows: Given a user u and his historical check-in records $\mathcal{P}^u$ and socially connected friendship, VirHpoi aims to recommend top-k unvisited POIs $\mathcal{P}_{rec}^u = \{p^1, p^2..., p^k \in \mathcal{P} \backslash \mathcal{P}^u\}$ that u might be interested in, while maximizing the preservation of explicit and implicit relationships of the LBSNs by heterogenous hypergraph embedding. To this end, VirHpoi mainly involves two modules for POI recommendation: (1) Hypergraph construction, which incorporates implicit relationship of the LBSNs by establishing virtual hyperedges; and (2) Hypergraph embedding, which learns embeddings of the objects in the LBSNs based on the constructed heterogeneous hypergraph, by preserving homophily and interaction attribute affinity of the LBSNs.

3.2 Hypergraph Construction

We first make virtual connections synthetically considering both geographical and semantic characters of the LBSN and then construct the complete heterogenous hypergraph which is used for the hypergraph embedding.

3.2.1 Virtual Connections

We incorporate implicit relationships of the LBSNs by establishing virtual hyperedges based on the aforementioned geographical and semantic factors. A hyper-parameter α is used to balance the impacts of the geographical and semantic factors as follows:

$$sim(p_n, p_m) = \alpha \cdot sim_g(p_n, p_m) + (1-\alpha) \cdot sim_f(f_n, f_m) \quad (4)$$

where $sim_g(p_n, p_m)$ and $sim_f(f_n, f_m)$ denote the geographical similarity and semantic similarity, and can be obtained using Eqs. (2) and (3), respectively. Apparently, a small value of α gives less importance on the geographical features and more on semantic factors, and vice versa.

It is noticed that, there is a numerical range gap between the geographical similarity and the semantic similarity. Recall that the range of the semantic similarity is between 0 and 1, but the range of the geographical similarity is larger than that of the semantic similarity. The calculated final similarity will be dominated by the geographic similarity. Therefore, it is necessary to normalize the geographical similarity. The number of POI pairs to be selected for virtual connections is limited, so we consider more about the first β POI pairs for normalization. The selection of β is actually related to the general activity scope of users, and a larger β means more distant POIs from user current position will be considered, which is related to the boundary of the user possible daily activities. Although users may be more likely to check in on farther POIs, after more than a certain distance, users focus more on the semantic features of the POIs, and geography is no longer the primary factor.

On this basis, we set the midpoint of normalization as sim_g^β and we use $sim_g^{\beta/2} - sim_g^\beta$ to adjust the range of $sim_g(p_n, p_m)$, avoiding most of the data concentration in high-value. After adjusting the midpoint and scaling, we use $\frac{1}{1+e^{-x}}$ to normalize the data, where x describes the processed similarity. The final similarity formula can thus be calculated as follows:

$$\begin{aligned} sim(p_n, p_m) =& \alpha \cdot \frac{1}{1 + e^{-\frac{sim_g(p_n, p_m) - sim_g^\beta}{sim_g^{\beta/2} - sim_g^\beta}}} \\ &+ (1-\alpha) \cdot sim_f(f_n, f_m) \end{aligned} \quad (5)$$

where sim_g^β and $sim_g^{\beta/2}$ are the value of the top βth and the top $\beta/2$th geographical similarity, respectively.

With the similarities of all POI pairs based on Eq. (5), we can make virtual hyperedges for POIs with implicit correlations.

3.2.2 Hypergraph Construction

In order to obtain the virtual hyperedges, we use clustering to construct hyperedges with multiple POIs, and the clusters are regarded as hyperedges. We obtain POI clusters by borrowing the idea of hierarchical agglomerative clustering method [29]. The clustering starts from the partition of the data set into singleton object, each of which forms a cluster. Then, the clusters are merged step by step according to the distance between clusters (i.e., the distance between the nearest two POIs of the two clusters and can be calculated by Eq. (5)). Different from the original clustering, we merge the current closest POI into the set and stop until the set

reaches the specific numbers so as to ensure that each virtual hyperedge has a fixed number of POIs. Specially, when each virtual hyperedge contains only two POIs, we can also construct virtual hyperedges by linking most similar POI pairs (top 1% in our experiments considering the number of POIs and the density of virtual hyperedges).

Incorporating all explicit and implicit relationships, we can then construct the LBSN hypergraph $G = (\mathcal{V}, \mathcal{E}_f, \mathcal{E}_c, \mathcal{E}_v)$ consisting of users, time slots, POIs and categories. The constructed heterogeneous hypergraph contains objects from four domains and three kinds of edges. In the social domain, the friendships are about preserving user interactions. In the POI domain, the virtual hyperedges are about preserving implicit relationships in the LBSNs. Considering the efficiency issue, we only construct one type of virtual hyperedge. As the collected check-in time is continuous, we need to transform the continuous data into discrete ones. We define the time granularity as 168 hours for a week and each check-in is associated with a specific time slot. With users, time slots, POIs and categories of POIs, we model check-ins as hyperedges, which are about preserving interaction attribute relations of four kinds of objects across all domains. We next learn embeddings of objects in the LBSNs based on the constructed heterogeneous hypergraph.

3.3 Hypergraph Embedding

The constructed LBSN hypergraph is highly heterogeneous consisting of four types of objects and containing both heterogeneous hyperedges and homogenous edges. Graph embedding for such heterogeneous hypergraph is challenging. Here, we design a novel embedding model preserving both homophily and interaction attribute affinity of the LBSN hypergraph. To learn the hypergraph, we first randomly sample virtual hyperedges and friendship edges from POI domain and user domain to learn the homophily. After preserving the homophily of each domain, we sample check-in hyperedges for learning the interaction attribute affinity of objects from different domains. Note that the number of virtual hyperedges and friendship edges we select are proportional to the number of check-in hyperedges by the ratio γ.

3.3.1 Homophily Preserving

Homophily in networks refers to the property that interconnected nodes are tend to be similar [30]. Normally, some of the features of the connected nodes are very similar. For example, in our constructed heterogeneous hypergraph, interconnected user nodes may tend to have similar hobbies or have the same gender, etc., and interconnected POI nodes have highly implicit correlations. The distance in the embedding latent space indicates the proximity between two nodes, and the connected nodes should be embedded closely. It is therefore necessary to preserve homophily of the heterogeneous hypergraph while embedding.

We preserve homophily of the LBSNs by learning edges that connect LBSN objects of the same type. To be more concrete, given an LBSN heterogeneous hypergraph $G = (\mathcal{V}, \mathcal{E}_f, \mathcal{E}_c, \mathcal{E}_v)$, we preserve the proximity of POIs and users by learning these edges, where multiple objects in a virtual hyperedge are of the same type. So, virtual hyperedges are decomposed into pairwise homogenous edges for learning. We learn low-dimensional embeddings of LBSN objects and preserve the homophily by maximizing the co-occurrence probability of all homogenous edges. The objective function for calculating the log-probability is as follows:

$$\Theta_1 = \arg\max \sum_{(v_i, v_j) \in \mathcal{E}} \log Pr(v_j | v_i) \tag{6}$$

where $\mathcal{E}$ is the homogenous edge set including edge set $\mathcal{E}_f$ or $\mathcal{E}_v$. Nodes v_i and v_j connected by a homogenous edge are of the same type denoting users or POIs. $Pr(v_j|v_i)$ is the conditional probability that node v_i connects v_j.

For a given node v_i, the conditional probability $Pr(v_j|v_i)$ of connecting v_j is computed as follows:

$$Pr(v_j | v_i) = \frac{\exp(S_{v_j, v_i})}{\sum_{(v_i', v_j) \in \mathcal{E}} \exp(S_{v_j, v_i'})} \tag{7}$$

where S_{v_j, v_i} is a scoring function defined as $S_{v_j, v_i} = \vec{v_i}^{\mathrm{T}} \cdot \vec{v_j}$; it reflects the proximity between nodes v_i and v_j. $\vec{v_i} \in R^{1 \times D}$ and $\vec{v_j} \in R^{1 \times D}$ are respectively the embeddings of v_i and v_j. D is the dimensionality of the latent space.

3.3.2 Interaction Attribute Learning

The check-in hyperedges in the LBSNs reflect the interaction attribute relation of LBSN objects from different domains. Each object in the LBSNs is connected to many check-in hyperedges including four different types of objects, and a hyperedge can actually be regarded as a set of nodes (c.f. the hyperedge linking user, time slot, POI and category with the colored thick line in Fig. 1(c)). The interaction attribute of a specific object in a hyperedge can be reflected by its interaction with other objects of the same hyperedge. We call the specific object as the target object. That is to say, knowing all other nodes in the hyperedge, we can learn the interaction attribute of the target object by learning these hyperedges. For example, when a POI is regarded as the target object in a check-in hyperedge, the attribute of this POI object can be reflected by the interaction with specific user object, time slot object and category object in the hyperedge. We thus alternate the object in hyperedges as the target object to learn the hyperedge.

Specifically, the process of learning hyperedges is to maximize the likelihood of predicting the target object when knowing other interaction objects. The users' preferences are evolving from time to time, so we sample check-in records according to time evolution with the probability as follows:

$$Pr(t) = \exp \frac{-(t_0 - t)}{T} \tag{8}$$

where T is the time range for all selected data, t_0 and t are respectively the latest check-in time and the time of check-in record. According to Eq. (8), more recent check-in hyperedges have more chance to be sampled.

After sampling the check-in hyperedges, we can learn all interactions to preserve the interaction attribute affinity of the target node as follows:

$$\Theta_2 = \arg\max \prod_{e_c \in \mathcal{E}_c} [\prod_{v_j \in e_c} \prod_{v_i \in \mathcal{E}_c \setminus v_j} Pr(v_j | v_i)] \tag{9}$$

where v_j denotes the predicted target node, e_c is the hyperedge which preserves interaction relation of check-in record, and $\mathcal{E}_c$ is the check-in hyperedge set. Considering the heterogeneity of check-in hyperedges, the importance of different objects to a check-in record varies; therefore we set a hyper-parameter w_i to control the impact of objects from different domains on the prediction. We also apply log-likelihood transformation and then Θ_2 is turned into:

$$\Theta_2 = \arg\max \sum_{e_c \in \mathcal{E}_c} [\sum_{v_j \in e_c} \sum_{v_i \in \mathcal{E}_c \setminus v_j} w_i \log Pr(v_j|v_i)] \quad (10)$$

where w_i describes the weight of one type of object for a hyperedge. Users and POIs play an important role in check-in activities, so we set $w_i = 1$ for user, POI and category objects in a check-in hyperedge. Since the daily activities of users are regular, users have their preference for certain time slots to check-in and the time slots vary. We set $w_i = \frac{T_e}{\sum T_i}$ for time slot objects where T_e is the number of times that the user checks in at a specific time and $\sum T_i$ is the total number of user historical checked-ins. w_i is related to both the user and the time slot, and it is different when the selected hyperedge includes different users or time slots.

Finally, we combine the homophily preserving and interaction attribute learning. The final objective function of our embedding model is:

$$\Theta_{emb} = \Theta_1 + \Theta_2 \quad (11)$$

We maximize the final objective function Θ_{emb} when learn the embeddings of the LBSNs.

3.3.3 Model Optimization

We notice that it is computationally expensive to optimize the objective functions Θ_1 and Θ_2, as calculating the conditional probability $Pr(v_j|v_i)$ needs to sum over the entire set of nodes. We thus adopt a speed up method which is similar to the negative sampling proposed in [31] to reduce the computation cost. Specifically, given a positive edge (v_i, v_j), several negative edges are sampled according to a specific noise distribution. We can rewrite the objective functions as:

$$\begin{aligned}\Theta_1 = \arg\max \sum_{(v_i,v_j)\in\mathcal{E}} & [\log \sigma(S_{v_j,v_i}) \\ & - \sum_{v_n \in \mathcal{E}_{neg}} \mathcal{E}_{v_n \sim P_n(v)} \log \sigma(-S_{v_j,v_n})]\end{aligned} \quad (12)$$

$$\begin{aligned}\Theta_2 = \arg\max \sum_{e\in\mathcal{E}_c} [\sum_{v_j \in e} \sum_{v_i \in \mathcal{E}_c \setminus v_j} & w_i [\log \sigma(S_{v_j,v_i}) \\ & - \sum_{v_n \in \mathcal{E}_{neg}} \mathcal{E}_{v_n \sim P_n(v)} \log \sigma(-S_{v_j,v_n})]]\end{aligned} \quad (13)$$

where $\mathcal{E}_{neg}$ is the negative sample edge set, $P_n(v) \propto d_v^{3/4}$ and d_v is the degree of node v. Note that we sample homogenous edges as negative samples for Θ_1 from the same domain as the learned edges and we sample hyperedges as negative samples for Θ_2. To learn the hypergraph embedding, we then use an alternate iterative update procedure and employ the widely used stochastic gradient descent (SGD) technique to optimize the objective functions.

3.4 POI Recommendation

After we train our model and learn the embeddings for users $\vec{u}$, time slots $\vec{t}$, POIs $\vec{p}$ and categories $\vec{c}$, we can perform new POI recommendation using simple operation on embeddings. The learnt embeddings preserve both explicit interactions and implicit relationships information and so we can capture the user preference to POIs from these embeddings. In particular, given a user u, we rank all the unvisited POIs by the score function $S(p|u)$ that reflects the user preference to POIs as follows:

$$S(p|u) = \vec{p}^{\mathrm{T}} \cdot \vec{u} \quad (14)$$

where $\vec{u}$ and $\vec{p}$ are embeddings for user u and POI p, respectively. Each unvisited POI is assigned a score for the given user u by Eq. (14). Finally, we select POIs with top-k highest scores to form the final recommendation list.

4 Performance Evaluation

4.1 Experiment Setup

4.1.1 Datasets

We utilize the global-scale check-in data collected in [21], [22] from Foursquare for evaluation, which is widely used in recent works [19], [20], [32], [33]. The preprocessing and statistics of the datasets are described in Section 2 and Table 1. To implement POI recommendation tasks, for each user, we chronologically split the check-in data into three parts, the first 70% for training, the remaining 20% for testing, and the last 10% as the tuning data.

4.1.2 Evaluation Metrics

We use two standard recommendation evaluation metrics including *Precision (Pre@k)* and *Recall (Rec@k)* to evaluate the performance:

$$Pre@k = \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} \frac{|\mathcal{P}^u_{rec} \cap \mathcal{P}^u|}{|\mathcal{P}^u_{rec}|} \quad (15)$$

$$Rec@k = \frac{1}{|\mathcal{U}|} \sum_{u\in\mathcal{U}} \frac{|\mathcal{P}^u_{rec} \cap \mathcal{P}^u|}{|\mathcal{P}^u|} \quad (16)$$

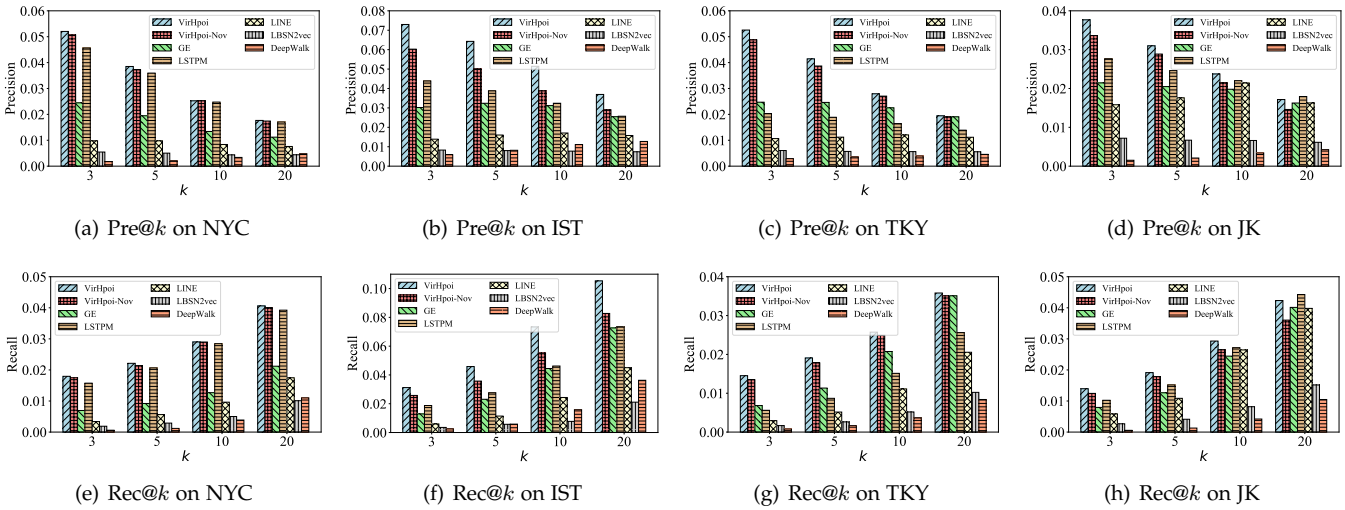
where $\mathcal{P}^u_{rec}$ is the set of top-k unvisited POIs in the recommendation list for user u, and $\mathcal{P}^u$ is the set of actually visited new POIs of u (i.e., the ground truth).

Here, Pre@k w.r.t. each user indicates how many POIs in the top-k recommended POIs correspond to the hold-off POIs in the testing data, while *Rec@k* w.r.t. each user indicates how many recommended POIs have been visited by the user. Considering the practical effect of recommendation tasks where the value of k is normally not quite huge, we thus experiment and report the performance for $k = \{3, 5, 10, 20\}$.

4.1.3 Baselines

We compare VirHpoi with the following state-of-the-art methods:

- ***LINE*** [34]. LINE adopts a large-scale network embedding model which preserves both the first-order and the second-order proximities, and thus can describe both the local pairwise proximity between



(a) Pre@k on NYC (b) Pre@k on IST (c) Pre@k on TKY (d) Pre@k on JK

(e) Rec@k on NYC (f) Rec@k on IST (g) Rec@k on TKY (h) Rec@k on JK

Fig. 5. Recommendation performance on the four datasets.

two nodes and the similarity between node pair neighborhood.

- ***DeepWalk*** [30]. DeepWalk is a classical network embedding method, which uses the skip-gram model to learn d-dimensional feature representations based on the node sequence sampled by truncated random walks.
- ***GE*** [13]. GE jointly captures multiple effects by embedding four relational bipartite graphs (i.e., POI-POI, POI-time, POI-region, and POI-activity) into a shared low dimensional space, before conducting real-time POI recommendation.
- ***LBSN2vec*** [19], [20]. LBSN2vec employs a random-walk-with-stay scheme to sample user check-in hyperedges and social relationships together, and preserve n-wise node proximity by maximizing the similarity between nodes and their best-fit-line under cosine similarity.
- ***LSTPM*** [35]. LSTPM models user check-ins as multiple trajectories. Based on the trajectories, LSTPM facilitates the modeling of users' long- and short-term preference. The long-term preference is usually stable, while the short-term preference tends to change frequently over time.

While LINE and DeepWalk are two well-known graph embedding methods, GE, LBSN2vec and LSTPM are especially designed for LBSNs.

We also compare VirHpoi with its variant ***VirHpoi-NoV***, in which we remove all the virtual connections, to further validate the benefits brought by the virtual hyperedges.

4.1.4 Experiment Settings

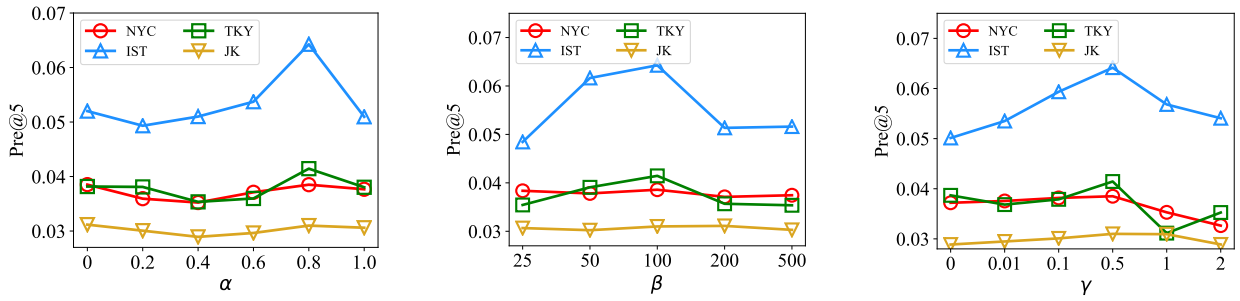
For all baselines, the parameter settings are initialized the same as reported in their original works. For LINE and DeepWalk, we break each of our hyperedges into multiple classic edges for fair comparisons, as they are designed for classical edges and cannot performed for learning hyperedges. In our experiments, the dimension of the graph embedding is set to 150 and the learning rate is set to 0.001.

4.2 Performance of POI Recommendation

We first evaluate the recommendation effectiveness and present the results of VirHpoi, which makes virtual hyperedges connecting two POIs by selecting top 1% POI pairs, along with the baselines with well tuned parameters. Fig. 5 shows the results using four datasets in terms of Pre@k and Rec@k. From the results, we can see that VirHpoi consistently outperforms other methods on all datasets.

Firstly, we compare our model with two graph embedding methods, DeepWalk and LINE. As excepted, VirHpoi outperforms DeepWalk and LINE greatly. These two universal graph embedding models perform worse since they only concentrate on explicit interactions and ignore particular features like the geographical character of the LBSNs. Furthermore, they regard all the objects in the LBSNs equally as one type and take little count to the heterogenity structure. In contrast, in VirHpoi we regard check-ins as hyperedges to capture complex spatial-temporal interaction and preserve more side information, thereby obtaining better performance.

Secondly, compared with other two garph embedding models especially designed for LBSNs, i.e., GE and LBSN2vec, VirHpoi also presents its advantage. For example, in terms of Pre@10, the improvements of VirHpoi are 45.74% and 38.48%, respectively, on IST and JK dataset, indicating the benefits we can obtain by incorporating virtual connections and preserving both homophily and interaction attribute affinity of the LBSNs. VirHpoi-NoV which removes the virtual hyperedges also verifies the effectiveness of preserving both homophily and interaction attribute affinity of LBSNs. LBSN2vec performs classical random walk on user domain based on their friendships, and stays on each encountered user node to sample a set of hyperedges. LBSN2vec only performs random walk on user domain without hyperedges, so the sample for check-in hyperedges is highly related to friendships. However, the friendships of most users are very limited and their check-ins are not adequately sampled, which is the main reason for less effectiveness than other baselines. Besides, LSBN2vec concentrates on friendships and check-in interactions and ignores side



(a) Impact of α (b) Impact of β (c) Impact of γ

Fig. 6. Impact of parameters of VirHpoi.

information like geographical information. GE embeds several influences of POI recommendation and exploits more side information. Therefore, GE performs well. However, GE dose not model user object in graph and computes the user embeddings based on more recently visited histories. Since most users have few check-in records, computing the user embedding in this way may fail to capture personal preferences. Besides, GE only learns user embeddings from POI embeddings and ignores the informative friendships of user objects, which makes it not that effective than our methods.

Thirdly, we compare VirHpoi with LSTPM, which considers both long- and short-term preference for POI recommendation. LSTPM shows better performance comparing with other baselines, mainly due to the consideration of long-term preference and adjusting their influence for new POIs recommendation by short-term preference. New POI recommendations are often related to users' stable preference, so LSTPM can achieve better results than other baselines. However, LSTPM only considers the explicit check-in sequence of LBSNs. The advantage of our model is that VirHpoi includes more interaction information than LSTPM. VirHpoi can preserve the implicit connections in LBSNs, which makes some new POIs reachable. The results of GE and VirHpoi also show the effectiveness of learning LBSN interactions by modeling LBSNs as a graph.

To explore the benefits of incorporating the virtual linkage, we compare the performance of VirHpoi with VirHpoi-NoV that removes the virtual hyperedges. The results show that the performance of VirHpoi significantly increases in all datasets and at most 9.63% on top-3 when virtual connections are taken into consideration. This indicates that the virtual hyperedges play an important role in incorporating the implicit relationship of the LBSNs. Virtual connections shorten the distance between users and potentially appealing yet unvisited POIs, and preserve the implicit connections of LBSNs which helps improve the new POI recommendation effectiveness.

4.3 Parameter Sensitivity

4.3.1 *Impact of* α

The parameter α controls the proportion of semantic and geographical factors in establishing virtual hyperedges. More geographical character is considered when α is closer to 1 and vice versa. We tune the value of α from 0 to 1 with a step of 0.2. Only semantic similarity is considered when $\alpha = 0$. Fig. 6(a) shows the performance under different α. Compared with $\alpha = 0$, when α begins to get larger, the performance first gets worse and then becomes better. It reaches the peak value when $\alpha = 0.8$ in IST and TKY datasets. The performance for $\alpha = 0$ is almost the same as $\alpha = 0.8$ in NYC and JK datasets, and the trend is the same for all datasets.

From the result we can also observe that the semantic and geographical factors have an asymmetric effect on POI recommendation. In particular, when α is close to 0, the virtual connections are dominated by the semantic similarity, while when α gets bigger, POI pairs that are geographically closer to each other but farther than the users general scope of activity are connected by the virtual connections, and their semantic similarity is not high enough. In this sense, the effect of the semantic similarity will be weaken and the geographical similarity has little or even negative effect. When α is close to 1, the geographical similarity is dominant, and the effect of physical distance on POI pairs will be adjusted. Geographically close POI pairs with high semantic similarity will be taken into consideration.

4.3.2 *Impact of* β

In this part, we discuss the effect of the hyper-parameter β on the performance. β decides the geographical similarity percentage for normalization and is actually related to the general activity scope of users. A larger β means that the farther distance POIs pairs are considered for virtual connections. From the results in Fig. 6(b), we can observe that β indeed influences the POI recommendation performance. The performance is sensitive to β on TKY and IST datasets which means the users activity range is generally within 100 POIs from the current position. However, the performance is not sensitive to β on the rest two datasets. This may be owing to the different density of POIs in different cities.

4.3.3 *Impact of* γ

Virtual connections contribute to appropriate new POI recommendation. Here, we change the parameter γ to examine the impact of virtual connections ratio in achieving optimal performance. As we can see in Fig. 6(c), with the increase of γ, more virtual connections are sampled. The introduction of virtual connections incorporates implicit relationship of the LBSN and brings users closer to POIs that are unvisited but are more likely to be checked-in. Furthermore, only limited check-in records are collected for each user, so the virtual

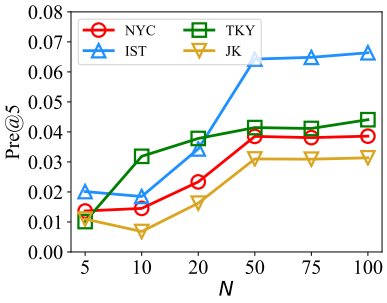


Fig. 7. Impact of N.

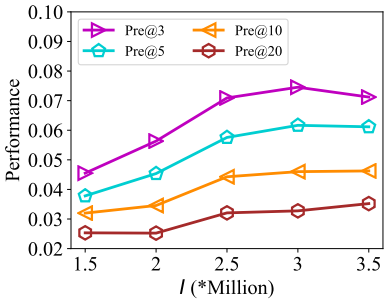


Fig. 8. Impact of I on IST dataset.

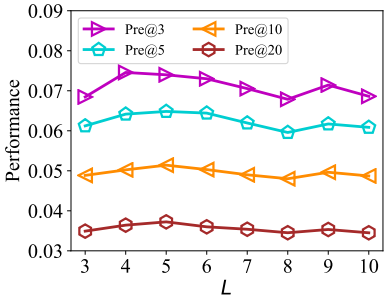


Fig. 9. Effectiveness of virtual hyperedges on IST dataset.

connections will help deal with the data sparsity issue to an extent. However, when virtual connections are excessively incorporated, the embeddings of POIs will be trained with too much time and the process of graph embedding will be greatly influenced by the extra virtual connections. In this case, the effect of check-ins on embedding is weakened and the performance will get worse. From Fig. 6(c), we can see that $\gamma = 0.5$ yields the best performance for all datasets.

4.4 Impact of the Negative Sample Size

In our hypergraph embedding, the negative sampling method is adopted to speed up the computation and N represents the number of negative samples for each edge from a specific noise distribution. Fig. 7 show the results with different numbers of negative sample per example. It can be observed that with more negative examples sampled, the performance improves rapidly, and converges when the negative sample size is around 50. At the same time, the need of computational resources increases when more negative examples are sampled. Therefore, in our case, 50 negative samples yield the best performance in all datasets.

4.5 Impact of the Number of Samples

For embedding based methods, the number of samples has a great impact on the computational speed and storage. Therefore, we analyze the convergence performance by varying the number of samples I. Fig. 8 shows the performance of VirHpoi on the IST dataset. We can see that the performance improves with the increase of number of samples and converges when more than 2.5 million samples on the IST dataset. To achieve a satisfying trade off between the effectiveness and the efficiency, $I = 3$ million is a good choice to ensure convergence while maintaining desirable efficacy. The convergence is related to the size of the dataset, so we set different I for different dataset proportional to the number of collected check-in records.

4.6 Effectiveness of Virtual Hyperedges

In this subsection, we aim to validate the effectiveness of virtual hyperedges containing different numbers of objects for new POI recommendation on IST dataset. Instead of linking the top 1% POI pairs as virtual hyperedges, we use clustering to construct virtual hyperedges with more than two objects. From the results in Fig. 5(b) and Fig. 9, we can see that the performance with virtual hyperedges is consistently better than that without them. When $L = 3$, the performance is worse than that using virtual hyperedges by linking the top 1% POI pairs. This is mainly because that a POI belongs to only one edge while clustering and the number of potential neighbors for each POI is small. When L increases, the relationships preserved by virtual hyperedges grow exponentially, so the performance becomes better than linking top 1% POI pairs as virtual hyperedges. When $L > 6$, the performance starts to decrease and then keeps stable, as some suboptimal pairs are involved to construct the hyperedges.

5 Related Work

5.1 POI Recommendation

POI recommendation is an inevitable product with the wide spread of LBSNs. It has been studied by plenty of researchers in the past several years. There are abundant contextual information in LBSNs and previous work make recommendations by modeling these contextual information. The methods can be divided into two categories: fused model and joint model. Fused models [36], [37] establish models for each factor, and then combine their results. Joint models depict the check-in behaviors as a synchronized decision influenced by factors together, which reflects the real scenario better. More recently, lots of joint models have been proposed for POI recommendation. For example, Xie et al. [13] jointly capture several effects in a unified way by embedding four correspond relation graphs into a shared latent space. Numbers of POI recommendation models have been proposed nowadays, but most of the existing models have difficulty dealing with data sparsity for the limitted check-ins of each user. Recent advance in graph embedding provides an effective and convenient way to address the issue of data sparsity in a unified way. By applying graph

embedding on POI recommendation, our proposed VirHpoi not only gains benefit brought by synthetically integrating different factors but also learns massive heterogeneous LBSN data effectively.

Note that there are different types of POIs recommending in different scenarios [33], [35], [38], and the most common one is for the next POI recommendation that is to recommend a list of possible POIs for a user to visit at next time point [4], [35]. In addition to a single POI recommendation, sequential POI recommendations [33] focus on recommending consecutive POIs to users, which are usually used for travel route suggestion. Considering that users often visit new POIs that they have not been visited before, some works [8], [24], [38], [39] focus on recommending new unvisited POIs to users, which is also the scenario we consider in this paper.

5.2 Graph Embedding

We make POI recommendations based on graph embedding [40], [41], which has attracted a great deal of attention in recent years. The main idea is to use a correlation algorithm to represent nodes in the network with a low-dimensional vector space, while the necessary structure and properties of the original network are reserved. Numbers of graph embedding models have been developed so far. DeepWalk [30] exploits truncated random walk and skip-gram model to learn the vector representations. Node2vec [42] improves the random walk scheme to better sample the network structure. LINE [34] models the first-order and second-order proximities between vertices. But the works discussed above concentrate on homogenous network. Therefore, researchers turn to more complex networks like heterogeneous networks and hypergraphs. For example, metapath2vec [15] uses meta-path-based random walks to construct heterogeneous neighborhood. DHNE [18] proposes a deep model to embed hyper-networks.

More recently, graph embedding techniques are applied to LBSNs. For example, LBSN2vec [19], [20] uses random-walk-with-stay scheme to sample edges and employs a hypergraph embedding model. LBSN2vec performs random walk on user objects based on their friendship, so its sampling relies heavily on social relations. Wang et al. [16] construct the heterogeneous neighborhood of a node by formalizing a meta-path based random walk. These works decompose the LBSNs as arbitrary graphs or directly perform embedding on heterogeneous networks. Unlike existing studies that concentrate on explicit connections, we regard check-ins as hyperedges and design a heterogeneous hypergraph embedding model to better preserve complex interactions in the LBSNs.

6 Conclusion

In this paper, we have presented VirHpoi, a novel heterogenous hypergraph embedding method for POI recommendation in LBSNs. To preserve the implicit relationship of the LBSN which synthetically reflects the relationship from the perspective of POIs, we make virtual connections by capturing semantic and geographical characters. We then preserve the complex structure of the LBSN hypergraph by learning homophily and interaction attribute relations of LBSN objects. Extensive experiment results on real-word datasets validate the benefits from the virtual hyperedges and show that VirHpoi can consistently outperform the stateof-the-art graph embedding methods.

References

[1] C. Xu, L. Zhu, Y. Liu, J. Guan, and S. Yu, "DP-LTOD: Differential privacy latent trajectory community discovering services over location-based social networks," *IEEE Transactions on Services Computing*, vol. 14, no. 4, pp. 1068–1083, 2021.

[2] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, "Location privacy-preserving mechanisms in location-based services: A comprehensive survey," *ACM Computing Surveys*, vol. 54, no. 1, pp. 1–36, 2021.

[3] X. Ma, F. Ding, K. Peng, Y. Yang, and C. Wang, "Cp-link: Exploiting continuous spatio-temporal check-in patterns for user identity linkage," *IEEE Transactions on Mobile Computing*, 2022. To appear. DOI: 10.1109/TMC.2022.3157292.

[4] L. Huang, Y. Ma, S. Wang, and Y. Liu, "An attention-based spatiotemporal LSTM network for next poi recommendation," *IEEE Transactions on Services Computing*, to appear. DOI: 10.1109/TSC.2019.2918310.

[5] Z. Zhang, M. Dong, K. Ota, Y. Zhang, and Y. Kudo, "Context-enhanced probabilistic diffusion for urban point-of-interest recommendation," *IEEE Transactions on Services Computing*, to appear. DOI: 10.1109/TSC.2021.3085675.

[6] T. Huang, Y. Dong, M. Ding, Z. Yang, W. Feng, X. Wang, and J. Tang, "MixGCF: An improved training method for graph neural network-based recommender systems," in *Proceedings of ACM KDD*, 2021.

[7] C. Cheng, H. Yang, I. King, and M. Lyu, "Fused matrix factorization with geographical and social influence in location-based social networks," in *Proceedings of AAAI*, 2012, pp. 17–23.

[8] H. Li, Y. Ge, R. Hong, and H. Zhu, "Point-of-interest recommendations: Learning potential check-ins from friends," in *Proceedings of ACM KDD*, 2016, pp. 975–984.

[9] J.-D. Zhang and C.-Y. Chow, "GeoSoCa: Exploiting geographical, social and categorical correlations for point-of-interest recommendations," in *Proceedings of ACM SIGIR*, 2015, pp. 443–452.

[10] J. He, X. Li, L. Liao, D. Song, and W. Cheung, "Inferring a personalized next point-of-interest recommendation model with latent behavior patterns," in *Proceedings of AAAI*, 2016, pp. 137–143.

[11] H. A. Rahmani, M. Aliannejadi, R. Mirzaei Zadeh, M. Baratchi, M. Afsharchi, and F. Crestani, "Category-aware location embedding for point-of-interest recommendation," in *Proceedings of ACM SIGIR*, 2019, pp. 173–176.

[12] M. Backes, M. Humbert, J. Pang, and Y. Zhang, "Walk2friends: Inferring social links from mobility profiles," in *Proceedings of ACM CCS*, 2017, pp. 1943–1957.

[13] M. Xie, H. Yin, H. Wang, F. Xu, W. Chen, and S. Wang, "Learning graph-based POI embedding for location-based recommendation," in *Proceedings of ACM CIKM*, 2016, pp. 15–24.

[14] H. Gui, J. Liu, F. Tao, M. Jiang, B. Norick, and J. Han, "Large-scale embedding learning in heterogeneous event data," in *Proceedings of IEEE ICDM*, 2016, pp. 907–912.

[15] Y. Dong, N. V. Chawla, and A. Swami, "Metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of ACM KDD*, 2017, pp. 135–144.

[16] Y. Wang, H. Sun, Y. Zhao, W. Zhou, and S. Zhu, "A heterogeneous graph embedding framework for location-based social network analysis in smart cities," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2747–2755, 2019.

[17] S. Huang, M. Elhoseiny, A. Elgammal, and D. Yang, "Learning hypergraph-regularized attribute predictors," in *Proceedings of IEEE CVPR*, 2015, pp. 409–417.

[18] K. Tu, P. Cui, X. Wang, F. Wang, and W. Zhu, "Structural deep embedding for hyper-networks," in *Proceedings of AAAI*, 2018.

[19] D. Yang, B. Qu, J. Yang, and P. Cudre-Mauroux, "Revisiting user mobility and social relationships in LBSNs: A hypergraph embedding approach," in *Proceedings of WWW*, 2019, pp. 2147–2157.

[20] ——, "LBSN2Vec++: Heterogeneous hypergraph embedding for location-based social networks," *IEEE Transactions on Knowledge and Data Engineering*, to appear. DOI: 10.1109/TKDE.2020.2997869.
[21] D. Yang, D. Zhang, Z. Yu, and Z. Yu, "Fine-grained preference-aware location search leveraging crowdsourced digital footprints from lbsns," in *Proceedings of ACM UbiComp*, 2013, pp. 479–488.
[22] D. Yang, D. Zhang, Z. Yu, Z. Yu, and D. Zeghlache, "SESAME: Mining user digital footprints for fine-grained preference-aware social media search," *ACM Transactions on Internet Technology*, vol. 14, no. 4, pp. 1–24, 2014.
[23] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *Proceedings of ACM SIGIR*, 2011, pp. 325–334.
[24] H. Wang, H. Shen, W. Ouyang, and X. Cheng, "Exploiting poi-specific geographical influence for point-of-interest recommendation," in *Proceedings of IJCAI*, 2018, pp. 3877–3883.
[25] B. Chang, G. Jang, S. Kim, and J. Kang, "Learning graph-based geographical latent representation for point-of-interest recommendation," in *Proceedings of CIKM*, 2020, pp. 135–144.
[26] X. Li, G. Cong, X.-L. Li, T.-A. N. Pham, and S. Krishnaswamy, "Rank-geofm: A ranking based geographical factorization method for point of interest recommendation," in *Proceedings of ACM SIGIR*, 2015, pp. 433–442.
[27] M. Aliannejadi, D. Rafailidis, and F. Crestani, "A joint two-phase time-sensitive regularized collaborative ranking model for point of interest recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 6, pp. 1050–1063, 2020.
[28] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann, "Time-aware point-of-interest recommendation," in *Proceedings of ACM SIGIR*, 2013, pp. 363–372.
[29] F. Murtagh and P. Legendre, "Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion?" *Journal of classification*, vol. 31, no. 3, pp. 274–295, 2014.
[30] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proceedings of ACM KDD*, 2014, pp. 701–710.
[31] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of NIPS*, 2013.
[32] P. Rosso, D. Yang, and P. Cudré-Mauroux, "Beyond triplets: hyper-relational knowledge graph embedding for link prediction," in *Proceedings of WWW*, 2020, pp. 1885–1896.
[33] D. Lian, Y. Wu, Y. Ge, X. Xie, and E. Chen, "Geography-aware sequential location recommendation," in *Proceedings of ACM KDD*, 2020, pp. 2009–2019.
[34] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proceedings of WWW*, 2015, pp. 1067–1077.
[35] K. Sun, T. Qian, T. Chen, Y. Liang, Q. V. H. Nguyen, and H. Yin, "Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation," in *Proceedings of the AAAI*, 2020, pp. 214–221.
[36] H. Yin, W. Wang, H. Wang, L. Chen, and X. Zhou, "Spatial-aware hierarchical collaborative deep learning for poi recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, pp. 2537–2551, 2017.
[37] Z. Zhang, Y. Liu, Z. Zhang, and B. Shen, "Fused matrix factorization with multi-tag, social and geographical influences for poi recommendation," *World Wide Web*, vol. 22, no. 3, pp. 1135–1150, 2019.
[38] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan, "Personalized ranking metric embedding for next new poi recommendation," in *Proceedings of AAAI*, 2015, pp. 2069–2075.
[39] H. Wang, M. Terrovitis, and N. Mamoulis, "Location recommendation in location-based social networks using user check-in data," in *Proceedings of ACM SIGSPATIAL*, 2013, pp. 374–383.
[40] S. Feng, L. V. Tran, G. Cong, L. Chen, J. Li, and F. Li, "Hme: A hyperbolic metric embedding approach for next-poi recommendation," in *Proceedings of ACM SIGIR*, 2020, pp. 1429–1438.
[41] G. Song, L. Zhang, Z. Li, and Y. Li, "Large scale network embedding: A separable approach," *IEEE Transactions on Knowledge and Data Engineering*, to appear. DOI: 10.1109/TKDE.2020.3002700.
[42] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of ACM KDD*, 2016, pp. 855–864.

Chen Wang (S'10-M'13-SM'19) received the B.S. and Ph.D. degrees from the Department of Automation, Wuhan University, China, in 2008 and 2013, respectively. From 2013 to 2017, he was a postdoctoral research fellow in the Networked and Communication Systems Research Lab, Huazhong University of Science and Technology, China. Thereafter, he joined the faculty of Huazhong University of Science and Technology where he is currently an associate professor. His research interests are in the broad areas of wireless networking, Internet of Things, and mobile computing, with a recent focus on privacy issues in wireless and mobile systems. He is a senior member of IEEE and ACM.

Mengting Yuan received the B.E. degree from Wuhan University of Technology, China, in 2020. She is currently pursuing the M.S. degree in Electronics and Information Engineering at Huazhong University of Science and Technology, China. Her research interests focus on big data analytics in the context of spatio-temporal data and social network data.

Rui Zhang (M'12) is an Associate Professor in School of Computer Science and Technology at Wuhan University of Technology, China. She received the M.S. degree and Ph.D. degree in Computer Science from Huazhong University of Science and Technology, China. From 2013 to 2014, she was a Visiting Scholar with the College of Computing, Georgia Institute of Technology, USA. Her research interests include machine learning, mobile computing and big data analytics.

Kai Peng received the B.S., M.S., and Ph.D. degrees from Huazhong University of Science and Technology, China, in 1999, 2002, and 2006, respectively. He is now the faculty of Huazhong University of Science and Technology as a full professor. His current research interests are in the areas of wireless networking and big data processing.

Ling Liu (F'15) is a Professor in the School of Computer Science at Georgia Institute of Technology, Atlanta, GA, USA. She directs the research programs in the Distributed Data Intensive Systems Lab (DiSL). Prof. Liu is an elected IEEE Fellow, a recipient of IEEE Computer Society Technical Achievement Award (2012), and a recipient of the best paper award from numerous top venues. Prof. Liu served on editorial board of over a dozen international journals, including the editor in chief of IEEE Transactions on Service Computing (2013-2016), and the editor in chief of ACM Transactions on Internet Computing (since 2019). Her current research is primarily supported by USA National Science Foundation under CISE programs, IBM and CISCO.