Learning Hierarchical Spatial Tasks with Visiting Relations for Next POI Recommendation

NICHOLAS LIM, GrabTaxi Holdings, Singapore
BRYAN HOOI and SEE-KIONG NG, Institute of Data Science and School of Computing, National University of Singapore
YONG LIANG GOH, RENRONG WENG, and RUI TAN, GrabTaxi Holdings, Singapore

Sparsity is an established problem for the next Point-of-Interest (POI) recommendation task, where it hinders effective learning of user preferences from the User-POI matrix. However, learning multiple hierarchically related spatial tasks, and visiting relations between users and POIs, can help to alleviate this sparsity problem. In this article, we propose our Hierarchical Multi-Task Graph Recurrent Network (HMT-GRN) approach, which alleviates the sparsity problem by learning different User-Region matrices of lower sparsities in a multi-task setting. We then perform a Hierarchical Beam Search (HBS) on the different region and POI distributions to hierarchically reduce the search space with increasing spatial granularity and predict the next POI. Our HBS provides efficiency gains by reducing the search space, resulting in speedups of 5 to 7 times over an exhaustive approach. In addition, we propose a selectivity layer to predict if the next POI has been visited before by the user to balance between personalization and exploration. Further, we propose a novel Joint Triplet Loss Learning (JTLL) module to learn visited and unvisited relations between users and POIs for the recommendation task. Experimental results on two real-world Location-Based Social Network (LBSN) datasets show that our proposed approach significantly outperforms baseline and the state-of-the-art methods.

CCS Concepts: • **Information systems** → **Recommender systems;**

Additional Key Words and Phrases: Recommender system, Graph Recurrent Network, spatio-temporal, joint learning, triplet loss

ACM Reference format:
Nicholas Lim, Bryan Hooi, See-Kiong Ng, Yong Liang Goh, Renrong Weng, and Rui Tan. 2023. Learning Hierarchical Spatial Tasks with Visiting Relations for Next POI Recommendation. *ACM Trans. Recomm. Syst.* 1, 4, Article 20 (November 2023), 26 pages.
https://doi.org/10.1145/3610584

20

This work is an extension of "Hierarchical Multi-Task Graph Recurrent Network for Next POI Recommendation" [Nicholas Lim, Bryan Hooi, See-Kiong Ng, Yong Liang Goh, RenrongWeng, and Rui Tan. 2022. Hierarchical multi-task graph recurrent network for next POI recommendation. In *SIGIR*. 1133–1143.], published at SIGIR '22. In this work, we propose a novel Joint Triplet Loss Learning (JTLL) module to further alleviate the sparsity problem for improved performance.
This work was funded by the Grab-NUS AI Lab, a joint collaboration between GrabTaxi Holdings Pte. Ltd. and National University of Singapore, and the Industrial Postgraduate Program (Grant: S18-1198-IPP-II) funded by the Economic Development Board of Singapore.
Authors' addresses: N. Lim, Y. L. Goh, R. Weng, and R. Tan, GrabTaxi Holdings, 3 Media Cl, Singapore 138498, Singapore; emails: {nic.lim, yongliang.goh, renrong.weng, rui.tan}@grab.com; B. Hooi and S.-K. Ng, Institute of Data Science and School of Computing, National University of Singapore, 3 Research Link, Singapore 117602, Singapore; emails: {dcsbhk, seekiong}@nus.edu.sg.

2770-6699/2023/11-ART20 $15.00
https://doi.org/10.1145/3610584

1 INTRODUCTION

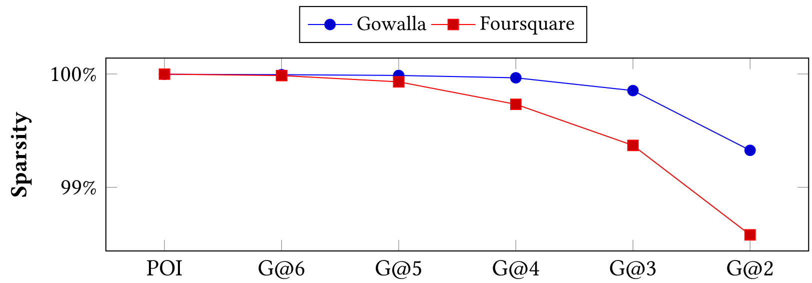
Recent years have seen rapid growth of sequential check-in data in social networks, where users share their checked-in locations or **Point-of-Interests (POIs)**. Personalized web recommender systems learn from these historical check-in sequences to recommend the next POIs to visit for a user, in order to improve user experience on their platforms.

Existing works have studied the next POI recommendation task primarily on **Location-Based Social Networks (LBSN)**, with simple baseline methods which make use of POI visit frequencies, followed by traditional methods of **Matrix Factorization (MF)** and **Markov Chains (MC)**. To better model the sequential dependencies or successive transitions of user POI sequences, **Recurrent Neural Networks (RNN)**, and its variants of **Long Short-Term Memory (LSTM)** and **Gated Recurrent Unit (GRU)** have been extended by several works, such as the ability to leverage spatio-temporal intervals between adjacent POIs [21, 36, 72], the inclusion of textual information of user activities [63], and the learning of long- and short-term user preferences [11, 20, 48, 55, 57, 58]. More recently, attention-based methods [19, 31, 40, 51, 71] have been proposed to instead learn from *both* successive and non-successive POI transitions of user sequences, such as via the representation of POIs into graphs for a global view [8, 26, 35], or by the aggregation of past hidden states with spatio-temporal weights [61]. However, a recurring underlying problem among these existing works is the high sparsity of the User-POI matrix that makes it difficult to learn and accurately predict the next POI a user would visit in the future. This sparsity problem is prominent as users would typically only visit a few preferred POIs, out of all POIs in the dataset as the search space.

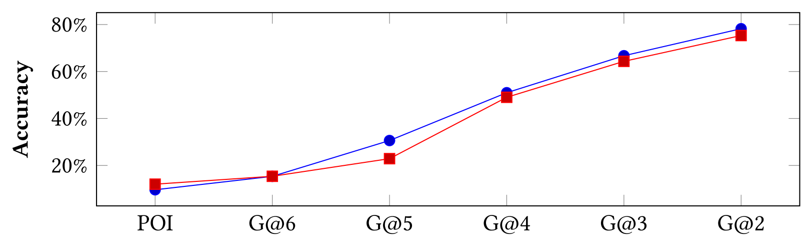
Shown in Figure 1, we apply the publicly available geocoding system Geohash[1] (G@P) on the popular LBSN datasets of Gowalla [5] and Foursquare [62]. Given each POI's location coordinates, G@P maps it to the respective grid cell, among all equally sized grid cells, and the precision or P determines the fixed cells' size, which decreases as P gets larger, where $P \in \{2, 3, 4, 5, 6\}$. We can observe in Figure 1(a) that for both datasets, the sparsity level (i.e., percentage of zeroes in the matrix) decreases from User-POI (99.99% for both datasets) towards User-G@2, where G@2 uses the largest grid cell size. Further, in Figure 1(b), given the same experimental setup with a traditional LSTM baseline, as commonly used in recent works for evaluation [21, 35, 48, 61, 72], we see that predicting the *next regions* or G@P (i.e., the region where the next POI resides in), always have better accuracy than predicting the sparse next POI *directly*, as done in existing works. However, although the significant performance gains of the next region-based tasks over the next POI task is evident, it is not clear how these different next region task distributions, when learned, could be utilized to predict the next POI more accurately, which is our main task of interest.

An often overlooked relation that can also help to alleviate sparsity between users and POIs, are the POIs which the users has *never* visited before, or the *unvisited* relation. For example, given the search space of POIs in a region $\{l_1, l_2, l_3, l_4, l_5, l_6\}$ to determine the next POI, where a user u_m has visited l_1 and l_2 before in her historical records, most existing works had focused on the learning of the *visited* relation (i.e., u_m visited $\{l_1, l_2\}$) for the personalized recommendation task, however, the unvisited POIs (i.e., $\{l_3, l_4, l_5, l_6\}$) are often not directly used to better learn the User-POI relations for this task. For instance, the *dissimilarity* between u_m and $\{l_3, l_4, l_5, l_6\}$ can also be used to enrich the learning of user and POI representations, while learning the similarity between u_m and $\{l_1, l_2\}$. Further, the additional learning of this unvisited relation can help to alleviate sparsity, given the

[1] http://geohash.org/, where G@P= 2 (1,251km × 625km), G@P= 3 (156km × 156km), G@P= 4 (39km × 19.5km), G@P= 5 (4.9km × 4.9km), G@P= 6 (1.2km × 0.61km).



(a) Decreasing sparsity levels as region size increases: Comparison of the User-POI and User-G@P matrices where $P \in \{2, 3, 4, 5, 6\}$. As P increases, the region or cell size decreases.



(b) Increasing predictive accuracy as sparsity levels decreases: Comparison of classification accuracy of the next POI and G@P region tasks with a LSTM baseline of the same experimental setup.

Fig. 1. Incorporating different region sizes alleviates the data sparsity problem for next POI recommendation.

severe sparsity problem of the User-POI matrix, for example, there now exists a dissimilarity relation between user u_m and $\{l_3, l_4, l_5, l_6\}$ that can be learned, but was not possible with only using the visited relation by most existing works. Although most existing works have learned POI-POI relations between visited POIs by the user (e.g., $\{l_1, l_2\}$ for user u_m) and the unvisited POIs (e.g., $\{l_3, l_4, l_5, l_6\}$), such as via spatial-temporal-preference factors [35], the unvisited relation, however, which is between the user u_m herself (not her visited POIs), and her own unvisited POIs $\{l_3, l_4, l_5, l_6\}$, has not been learned.

With this motivation, we first propose our novel **Hierarchical Multi-Task Graph Recurrent Network (HMT-GRN)** model to learn both User-POI and User-G@P matrices in the form of multi-task learning, to predict the next POI and G@P regions, then perform our **Hierarchical Beam Search (HBS)** on the learned task distributions to reduce the search space hierarchically and improve efficiency to predict the next POI. To balance between personalization and exploration, we also propose a selectivity layer that predicts if the next POI is a historically visited POI, or an unvisited POI by the user, thereby performing personalization and exploration respectively. Following, we propose the **Graph Recurrent Network (GRN)** module to learn both sequential dependencies within POI visit sequences, and global spatio-temporal POI-POI relationships simultaneously with spatial and temporal graphs that considers POI-Region and POI-Timeslot relationships to alleviate sparsity. Lastly, we propose a novel **Joint Triplet Loss Learning (JTLL)** module to learn both visited and unvisited relations to further support sparsity alleviation. Specifically, we first design a triplet-loss-based loss function to reduce the distance of user and POI embeddings (user visited POI before), and increase the distance of user and POI embeddings (user never visited POI before). Then, we jointly train our JTLL module with POI recommendation models to better learn the User-POI matrix, via a joint training framework that includes parameter sharing.

To summarize, the following are the contributions of this article:

- We propose a novel HMT-GRN[2] model, and a JTLL module, to alleviate the data sparsity problem that hinders effective learning of the User-POI matrix. To the best of our knowledge, this is the first work which proposes a triplet-loss-based loss function, included in our JTLL module, that uses User-POI visited and unvisited relations for this recommendation task.
- Our HMT-GRN model includes the multi-task learning of next POI and next regions or G@*P*, HBS as a search space reduction method, as well as a selectivity layer to balance between personalization and exploration. Further, our GRN module learns both sequential dependencies and global spatio-temporal POI-POI relationships simultaneously. Lastly, our JTLL module learns both visited and unvisited relations of users and POIs, to better learn the User-POI matrix.
- Experiments conducted on two popular real-world LBSN datasets show that our approach outperforms baseline and state-of-the-art methods significantly, as well as efficiency gains, with speedups of 5 to 7 times over an exhaustive approach for our HBS. Further, results show that our JTLL module was able to mostly improve the performances of extended baselines for the next new POI recommendation task.

2 RELATED WORK

Next POI Recommended Task. In this recommendation task, the objective is to predict a ranked set of POIs for each user, where the next POI visited is highly ranked. RNN-based approaches have been shown to be effective in modelling sequential dependencies for this recommendation task. ST-RNN [36] introduced the use of spatio-temporal intervals between adjacent POIs in a RNN, applying linear interpolation and learning time and distance transition matrices to mitigate the continuous nature of the intervals. This usage of intervals has also been applied in LSTMs [21, 72] through new gating mechanisms. Approaches which study the use of additional information of POI categories [17, 23, 27, 30, 32, 37, 57, 58] and textual information [1, 24, 25, 63] have also been proposed, however, such categorical and textual information are not always available among real-world LBSN datasets [68] (e.g., Gowalla [5]). DeepMove [11] learns sequential transitions with a GRU, as well as applying a historical attention module. LSTPM [48] uses several LSTM-based encoders to learn long and short term user preferences with a context-aware nonlocal network architecture. Luo et al. [40] proposed STAN, a spatio-temporal bi-attention model that focuses on learning regularities of non-contiguous visits and non-adjacent POIs. Flashback [61] utilizes spatio-temporal intervals among the current and historical visits to compute weights, with the goal of identifying past RNN hidden states of similar contexts, then aggregating them to be used for prediction. NEXT [70] incorporates multiple context factors of temporal, geographical influence, sequential relations and auxiliary metadata under a unified framework.

Multi-Task Approaches. Recently, multi-task approaches have also been proposed for next POI recommendation. In [74], the proposed multi-task approach predicts when and where the next POI event will occur, and computing weights from the time related task (when) to adjust the next POI distribution predicted (where). Similarly, in [54], they proposed to predict both next POI and check-in timeslot to better learn relationships between POI and timeslots, where the timeslot prediction task is an auxiliary task. For [15], they proposed TLR-M, which learns and predicts next POI and queuing time simultaneously, further showing the effectiveness of the queuing time prediction task in improving the performance of the next POI recommendation task. However, as they have highlighted, popular datasets such as Gowalla and Foursquare, do not contain queuing time

[2] https://github.com/poi-rec/HMT-GRN

information, and therefore, the proposed queuing time prediction task is not always suitable for all datasets.

Region-Aware Approaches. To better model the latent spatial relations between POIs for the next POI recommendation task, existing works have proposed region-aware approaches. For instance, Cheng et al. [4], Lu and Huang [39], and Zhang et al. [64] have proposed to incorporate the localized region constraint. In their analyses of the successive POI check-in datasets, they have observed that users have a movement constraint when deciding which POI to visit next, from their previous POI check-in. Therefore, to incorporate localized regional awareness in their approaches, Cheng et al. [4] used a 40-km threshold to filter the predicted POI recommendation set to reduce noise, and thereby performing better for the recommendation task. Similarly, in Lu and Huang [39] and Zhang et al. [64], the threshold used is instead 10 km, as analyzed by them individually, to only consider POIs within the nearby regions for the recommendation task. Apart from modeling regional awareness with a fixed distance threshold, recent works [21, 36, 72] have instead, used the pair-wise distance of the POI transition pairs, as spatial input to the model, where the model can then compute region-aware POI recommendations to the users. While these existing works have used POI distances to incorporate regional awareness, other existing works have instead used region embedding as input to their models [23, 55, 60]. For instance, Lai et al. [23] first constructs category-region-latent vectors from the category and region embedding sequences, which corresponds to their respective POI check-in sequences, then uses the computed category-region-latent vectors for improved performances at the next POI recommendation task.

Graph-Based Approaches. Xie et al. [60] proposed GE, an approach to learn POI, time slot, region, and word embeddings from various graphs, such as POI-POI graphs [52, 53, 69], then using these embeddings to score each POI for the recommendation task. Wang et al. [56] proposed DRAN, which uses graph convolutions on their distance-based and transition-based graphs to learn disentangled representations, while considering the complex correlation among POIs. Li et al. [29] have proposed to use a preference-based graph to perform augmentation for each POI sequence, thereby allowing collaborative signals to be propagated from other users' POI sequence that contains from correlated POIs. Apart from using a standard user trajectory or POI sequence, Wang et al. [55] proposed to also use their area transition patterns via a spatial graph in their ADQ-GNN model to better learn complex user transitions between different areas, where both **Graph Neural Networks (GNN)** and LSTMs are used. In [26], a heterogeneous multi-modal check-in graph is proposed to learn multiple aspects in a unified way. Specifically, the multi-modal graph include user and POI nodes, and edges of location-location (co-occurrence), location-location (spatial), location-location (textual comment), user-location, and user-user relations. Accordingly, the graph is used to learn multi-modal embeddings for their attentional recurrent network to perform recommendations. With a similar framework, Wang et al. [51] instead uses a POI check-in sequence graph with User-POI interaction and POI-POI transition edges as input to their GNN for feature representation learning, before using an attention network for prediction. STP-UDGAT [35] proposed the use of Graph Attention Networks (GAT) [50] to learn global POI-POI relationships from various graphs to model spatial, temporal, and preference factors.

Next New POI Recommendation Task. Extending from the next POI recommendation problem, a more challenging problem of Next New (N^2) POI recommendation have received recent research interest. As this N^2 POI recommendation task focuses on only recommending unvisited or new POIs where the user has never visited before, approaches proposed cannot merely rely on users' historical check-in sequence to perform well for this task. Early works explored conventional collaborative filtering and sequential approaches such as **Matrix Factorisation (MF)** and **Markov**

Chains (MC), respectively. For example, Cheng et al. [4] extended the **Factorizing Personalized Markov Chain (FPMC)** approach [43] that integrates both MF and MC, to include localised region constraints and recommend nearby POIs for the N^2 POI recommendation task. PRME-G [12], a metric embedding method, models both POIs and users in a sequential transition space and a user preference space respectively. As their method is not based on FPMC, it avoids the drawback of the independence assumption of FPMC to model the transitions [12]. Also using a metric embedding approach, Feng et al. [13] jointly learns the different relationships of **POI sequential transitions (POI-POI)**, **user preferences (POI-User)**, **regional (POI-Region),** and **categorical (POI-Category)** information in a unified way, by projecting them on a shared low-dimensional hyperbolic space. The learned hyperbolic embeddings are used with the Einstein midpoint aggregation [14, 49] to integrate the effect of user preferences and sequential transitions for prediction. Lu and Huang [39]proposed GLR_GT_LSTM, that uses an LSTM to model users' transition behaviors with latent vectors of POIs and regions, based on temporal user preference and temporal successive transition influence, as well as the spatial influence of POIs for the N^2 POI recommendation task. While these existing works [4, 12, 13, 39] have demonstrated effectiveness for the N^2 POI recommendation task, they have a limitation of only considering POI samples visited within the next 6-hour threshold of the preceding POI check-in, learning, and evaluating only short-term preferences of the proposed methods. Accordingly, in this article, we overcome this limitation by removing the threshold, to evaluate our proposed approaches for both short- and long-term preferences.

Overall, the existing works mainly focus on learning only the User-POI matrix for prediction, which entails a prominent sparsity problem and deters effective learning. Therefore, we propose our HMT-GRN model, and our JTLL module to alleviate this sparsity problem. Among the existing works which also seek to alleviate data sparsity [3, 7, 11, 45–47, 71], these includes a weighted loss function to accelerate learning with more informative samples [31], the leverage of additional information of POI categories [2, 9, 16, 29, 38, 64–68, 73], and the modeling of spatio-temporal relations in LSTM's existing multiplicative gates [21]. These existing works, along with the other multi-task inspired approaches [15, 54, 59, 74], can all be observed to differ from our HMT-GRN model significantly, such as the novel adoption of multi-task learning for the next POI and region tasks with different spatial granularity, our HBS to traverse the learned task distributions efficiently and spatially reduce the search space, a new selectivity layer to balance between personalization and exploration, and others.

For our HMT-GRN model, the related work of ADQ-GNN [55] is the closest to our region-aware approach, from the use of a similar hierarchical region graph. Specifically, ADQ-GNN applies a GNN on the POI check-in sequence and its hierarchical region graph for spatial preference modeling. Although our HMT-GRN model uses a Hierarchical Spatial Graph, which is similar, the purpose of the graph's usage fundamentally differs. Specifically, ADQ-GNN uses the hierarchical region graph as an input to the model, whereas our HMT-GRN model uses the graph after the model has predicted or computed the different task distributions with the multi-task learning framework. Then, we use the graph to perform our Hierarchical Beam Search to reduce the search space and noise, and proceed to compute the joint probabilities to derive the POI ranked set. Apart from the different usage of the graph, the definition of the graph differs, for instance, all vertices in our Hierarchical Spatial Graph are weighted with the probability score from their respective task distributions to perform the search algorithms, whereas the vertices in the hierarchical region graph used by ADQ-GNN are unweighted, as both works have different intentions on the usage of the graph. For all other region-aware approaches [4, 39, 64] that have incorporated the localized region constraint, our HMT-GRN model does not apply a distance threshold on the ranked set for the recommendation task. Further, our HMT-GRN model also does not use pair-wise distances of

POI transition pairs as input to the model, as done in Kang and Wu [21], Liu et al. [36], and Zhao et al. [72].

For our JTLL module, the closest related work is the **Personalized Ranking Metric Embedding (PRME)** [12] approach, where it jointly learns POI embeddings in the sequential transition latent space, and both User and POI embeddings in the user preference latent space. For the learning of POI embeddings, the approach uses both observed POI (i.e., visited relation), and unobserved POI (i.e., unvisited relation) to optimize the learning of POI embeddings, where the euclidean distance between the previous and next visited POI should be small, and the distance between the previous and unvisited POI should be large. Although both PRME and our JTLL approach uses both visited and unvisited relations of user visits to learn our embeddings, there are several key differences. First, our proposed loss function is triplet-loss-based, classically involving the three roles of an anchor, a positive, and a negative. In PRME, for both POI and user representation learning, it is instead achieved with a standard pairwise metric embedding approach (i.e., only pairs of latent vectors are used at each time for the respective latent space). Second, the visited and unvisited relations of user visits is used in their sequential transition latent space to only learn POI embeddings, whereas our loss function in Equation (21) uses the visited and unvisited relations to learn both user and POI embeddings, focusing more on the user representational learning. Third, our JTLL module is designed to overcome the limitation of existing works to additionally learn unvisited relations in a joint training framework and support sparsity alleviation, with evaluation results surpassing the state-of-the-art methods for the N^2 POI recommendation task. PRME is a standalone approach that has been shown by various works [21, 72] to perform significantly poorer than a classical LSTM model for all metrics and all datasets, whereas our best performing JTLL variant, surpassed the same LSTM baseline significantly, as shown in Table 2.

3 PRELIMINARIES

Problem Formulation. Let $L = \{l_1, l_2, \ldots, l_Q\}$ be a set of Q POIs and $U = \{u_1, u_2, \ldots, u_M\}$ be a set of M users. S is the set of visit sequences for all users where $S = \{s_{u_1}, s_{u_2}, \ldots, s_{u_M}\}$. Each user's sequence s_{u_m} consist of sequential POI visits $s_{u_m} = \{(l_{t_1}, loc_{t_1}, time_{t_1}), (l_{t_2}, loc_{t_2}, time_{t_2}), \ldots, (l_{t_i}, loc_{t_i}, time_{t_i})\}$, where l_{t_i} is the POI visited on time step t_i, with its corresponding location coordinates loc_{t_i}, and $time_{t_i}$ as the timestamp of the visit made. As each user's sequence s_{u_m} is partitioned into training and testing to predict future next POIs, we denote the superscript $train$ and $test$ respectively (e.g., $s_{u_m}^{train}$ and $s_{u_m}^{test}$).

Problem 1 (Next POI Recommendation). *Given user u_m, $s_{u_m}^{train} = \{(l_{t_1}, loc_{t_1}, time_{t_1}), (l_{t_2}, loc_{t_2}, time_{t_2}), \ldots, (l_{t_{i-1}}, loc_{t_{i-1}}, time_{t_{i-1}})\}$ from the sequential time steps of t_1 to t_{i-1} as her historical POI visit sequence, the next POI recommendation task is to consider a search space of POIs from L to compute a next POI ranked set y_{t_i} for the time step t_i, where the next POI visited l_{t_i}, should be highly ranked within y_{t_i}.*

3.1 LSTM

The LSTM [18] is a variant of RNN [10], capable of learning long-term sequential dependencies across a sequence by using gating mechanisms to control information flow to the cell state, and has been found to be effective in various sequential learning applications. For each time step t_i, the LSTM is defined as:

$$i_{t_i} = \sigma(\mathbf{W}_i\, x_{t_i} + \mathbf{U}_i\, h_{t_{i-1}} + \mathbf{b}_i) \tag{1}$$

$$f_{t_i} = \sigma(\mathbf{W}_f\, x_{t_i} + \mathbf{U}_f\, h_{t_{i-1}} + \mathbf{b}_f) \tag{2}$$

$$o_{t_i} = \sigma(\mathbf{W}_o\, x_{t_i} + \mathbf{U}_o\, h_{t_{i-1}} + \mathbf{b}_o) \tag{3}$$

$$\tilde{c}_{t_i} = tanh(\mathbf{W}_c\, x_{t_i} + \mathbf{U}_c\, h_{t_{i-1}} + \mathbf{b}_c) \tag{4}$$

$$c_{t_i} = f_{t_i} \odot c_{t_{i-1}} + i_{t_i} \odot \tilde{c}_{t_i} \tag{5}$$

$$\vec{h}_{t_i} = o_{t_i} \odot tanh(c_{t_i}) \tag{6}$$

where $i_{t_i}, f_{t_i}, o_{t_i} \in \mathbb{R}^{hdim}$ are the input, forget and output gates respectively of *hdim* dimension in the scale of 0 to 1 from the sigmoid activation function. The input gate seeks to learn "how much to input" based on the Hadamard product $\odot$ with the cell input $\tilde{c}_{t_i} \in \mathbb{R}^{hdim}$. The forget gate determines the information to be "forgotten" from the previous cell state $c_{t_{i-1}} \in \mathbb{R}^{hdim}$, and the output gate learns "how much to extract" from the current cell state c_{t_i} to compute the output hidden representation $\vec{h}_{t_i} \in \mathbb{R}^{hdim}$.

4 APPROACH

In this section, we first propose our **Hierarchical Multi-Task Recurrent Network (HMT-RN)** to learn the different next POI and region or G@P distributions in a multi-task setting, then performing HBS on the distributions to reduce search space hierarchically, and introduce the selectivity layer. Further, we propose the GRN module to replace the LSTM module, to model both sequential dependencies and global spatio-temporal POI-POI relationships simultaneously, resulting to our HMT-GRN model. Lastly, we propose our JTLL module to learn both visited and unvisited relations of users and POIs, to better learn the User-POI matrix, and to overcome the limitation of existing POI recommendation models, which includes our HMT-GRN model, to additionally learn unvisited relations for sparsity alleviation.

4.1 HMT-RN

Learning Next POI and Region Distributions. As shown in Figure 1, our motivation is to better perform the next POI recommendation task by learning not just the sparse User-POI matrix, but also the User-G@P matrices which have lower data sparsities. To this end, we propose to not only predict the next POI, but also the *next regions or G@P* where the next POI resides in, denoting $P \in \{2, 3, 4, 5, 6\}$ as all the precision levels to be considered in our model and $TK = \{G@2, G@3, G@4, G@5, G@6, POI\}$ as all the tasks to be learned. First, at time step t_i, given current user u_m, the previous POI $l_{t_{i-1}}$ and its mapped G@P grid cell $l_{t_{i-1}}^{G@P}$, we use a multi-modal embedding layer E_W to map to their trainable vector representations:

$$\vec{u}_m,\ \vec{l}_{t_{i-1}},\ \vec{l}_{t_{i-1}}^{G@P} = E_{\mathbf{W}}(u_m,\ l_{t_{i-1}},\ l_{t_{i-1}}^{G@P}); \tag{7}$$

$$\mathbf{W} \in \{\mathbf{W}_u, \mathbf{W}_l, \mathbf{W}_{G@2}, \mathbf{W}_{G@3}, \mathbf{W}_{G@4}, \mathbf{W}_{G@5}, \mathbf{W}_{G@6}\}$$

where $\mathbf{W}_u \in \mathbb{R}^{|U|\times\delta}, \mathbf{W}_l \in \mathbb{R}^{|L|\times\delta}, \mathbf{W}_{G@P} \in \mathbb{R}^{|L^{G@P}|\times\delta}$ are the user, POI and G@P weight matrices respectively for $P \in \{2, 3, 4, 5, 6\}$, $|L^{G@P}|$ as the total number of G@P grid cells after mapping all locations in L, and δ is the defined embedding dimension. Next, we use an LSTM layer Φ to learn the sequential dependencies among the POI sequences, with the previous POI embedding $\vec{l}_{t_{i-1}}$ as input:

$$\vec{h}_{t_i} = \Phi(\vec{l}_{t_{i-1}}) \tag{8}$$

$$tk^{POI} = softmax(\ DL_{\mathbf{W}_{L1}}(\ DO(\vec{h}_{t_i}\ \oplus\ \vec{u}_m)\ )\ ) \tag{9}$$

where $\vec{h}_{t_i} \in \mathbb{R}^{hdim}$ is the hidden representation computed from Equation (6), DO as the dropout layer, $\oplus$ as the concatenate operation, DL as a dense linear layer parameterized with $\mathbf{W}_{L1} \in \mathbb{R}^{hdim+\delta\times|L|}$ to project to $|L|$ POIs, then performing the softmax normalization to compute the conditional probability of the next POI task distribution $tk^{POI} = P(l_{t_i} \mid l_{t_{i-1}})$. Accordingly, the ranked set $y_{t_i} = \Psi(tk^{POI})$ can be computed by applying the sorting function $\Psi(.)$ to sort tk^{POI} in descending order for the next POI recommendation task. As per Equation (9), we use the user embedding $\vec{u}_m$ as the task-specific embedding for this task to include personalization.

Similarly, *in addition* to the next POI task, we perform multi-task learning to also predict the *next* $l_{t_i}^{G@P}$, where $l_{t_i}^{G@P}$ is the respective G@P grid cell of the next POI l_{t_i}. Here, instead of the user embedding $\vec{u}_m$, we use the task-specific embedding of the previous POI G@P representation $\vec{l}_{t_{i-1}}^{G@P}$ from Equation (7), to predict the next corresponding $l_{t_i}^{G@P}$, thereby computing the task distribution $tk^{G@P} = P(l_{t_i}^{G@P} \mid l_{t_{i-1}})$:

$$tk^{G@P} = softmax(\ DL_{\mathbf{W}_{L@P}}(\ DO(\vec{h}_{t_i}\ \oplus\ \vec{l}_{t_{i-1}}^{G@P}\ )\ )\ ); \tag{10}$$

$$P \in \{2, 3, 4, 5, 6\}$$

Intuitively, this can be interpreted as using the *previous region* to help predict the *next region* of the same P or grid cell size for each of the next G@P tasks. For all tasks TK, we use $\vec{h}_{t_i}$ as the common representation for shared feature learning, as per Equations (9) and (10).

Training. With each task $tk \in \{G@2, G@3, \ldots, G@6, POI\}$, the cross-entropy loss is $\mathcal{L}_{tk} = -\sum_{v=1}^{N_{train}} \log(pb_v)$, where pb_v is the predicted probability of the ground truth next POI or G@P, depending on the task tk, for the vth training sample, and N_{train} is the total number of training samples across all users. The overall supervised loss $\mathcal{L} = \frac{1}{|TK|} \sum \mathcal{L}_{tk}$ is computed with equal weights to not bias to any task.

Hierarchical Beam Search. Instead of using the sparse next POI task distribution $tk^{POI} = P(l_{t_i} \mid l_{t_{i-1}})$ alone for prediction, we propose to leverage the learned next POI and G@P task distributions $\{tk_1^{G@2}, tk_2^{G@3}, \ldots, tk_{i-1}^{G@6}, tk_i^{POI}\}$, that were computed from Equations (9) and (10), by computing the joint probability[3] of all tasks $P(G@2, G@3, \ldots, G@6, l_{t_i} \mid l_{t_{i-1}})$ to rank the search space of POIs from L, and predict the next POI. First, we propose a Hierarchical Spatial Graph G_{hs}:

Definition 4.1 (Hierarchical Spatial Graph). A directed graph denoted as $G_{hs} = (V_{hs}, E_{hs})$ where V_{hs} and E_{hs} are the sets of all tasks' vertices and edges, respectively. G_{hs} represents the multiple task distribution as a graph with a spatial hierarchy, in the increasing granularity order of $\{tk_1^{G@2}, tk_2^{G@3}, \ldots, tk_{i-1}^{G@6}, tk_i^{POI}\}$, where each vertex of G@2 is connected to the G@3 vertices within it, and similarly, each vertex of G@3 is connected to the G@4 vertices within it, and so on, until the last POI layer, in a hierarchical structure. Each task vertex $v_{hs} \in V_{hs}$ is weighted with the probability score from its respective task distribution, that were computed from Equations (9) and (10), which will be used in the search algorithms.

By representing the task distributions as a hierarchical spatial graph, we can perform an exhaustive search, or equivalently, **Breadth-first search (BFS)**, to compute the sums of log-probabilities for $P(G@2, G@3, \ldots, G@6, l_{t_i} \mid l_{t_{i-1}})$ by traversing all paths of the graph G_{hs} to rank all POIs in L, however, this would be highly inefficient. We instead propose a **Hierarchical Beam Search (HBS)** method to only expand the top-β promising vertices of each task distribution during traversals, where β is the *beam width*. Specifically, given *each* sequential pair of task distributions (tk_{i-1}, tk_i) from $\{tk_1^{G@2}, tk_2^{G@3}, \ldots, tk_{i-1}^{G@6}, tk_i^{POI}\}$ (e.g., $(tk_1^{G@2}, tk_2^{G@3})$ and $(tk_2^{G@3}, tk_3^{G@4})$), we compute:

[3]While the events predicted are not independent, we use the multi-task learning framework to alleviate the sparsity issue effectively by modelling the distributions independently.

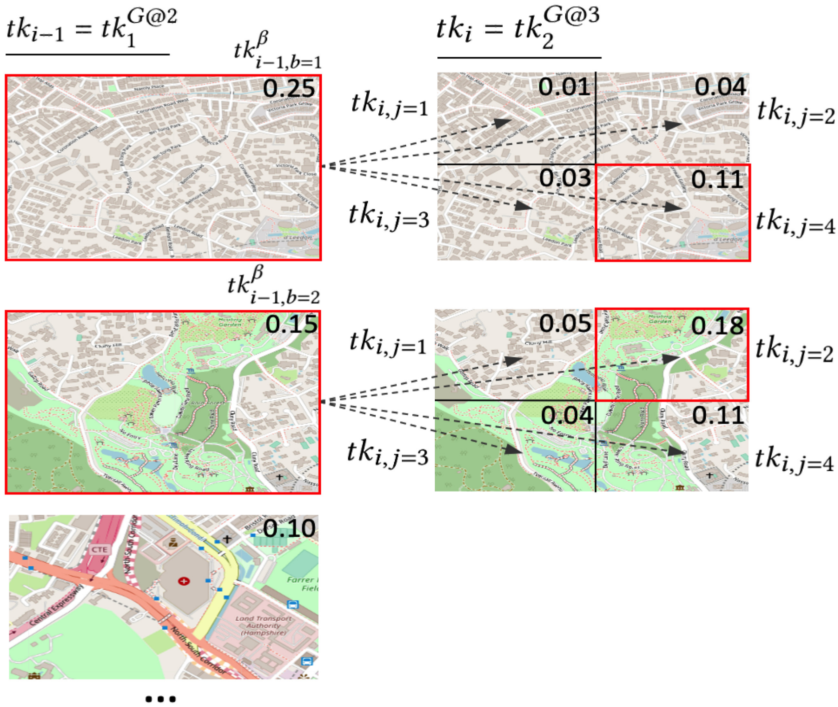


Fig. 2. Hierarchical Beam Search performed with top-$\beta = 2$ (red boxes) for the sequential pair of input tasks ($tk_{i-1} = tk_1^{G@2}, tk_i = tk_2^{G@3}$). Each vertex is weighted with its respective task probability score (top right of each vertex), for the computation of partial solutions, then ranking them to output tk_i^{β}. Maps © OpenStreetMap contributors, CC BY-SA.

$$tk_i^{\beta} = f\Bigg(\Bigg\{ \log\left(tk^{\beta}_{i-1,b}\right) + \log(tk_{i,j}) \mid tk_{i,j} \in \mathcal{N}\left(tk^{\beta}_{i-1,b}\right), \\ b \in \{1, 2, \dots, \beta\} \Bigg\} \Bigg) \tag{11}$$

where tk^{β}_{i-1} and tk^{β}_{i} are the top-β partial solutions for the input task distributions of tk_{i-1} and tk_i respectively, and a partial solution is the sum of log-probabilities of all vertices traversed in its path. For each top beam $b \in \{1, 2, \dots, \beta\}$ of the previous input task distribution $tk^{\beta}_{i-1,b}$, we identify its (directed) neighbourhood $tk_{i,j} \in \mathcal{N}(tk^{\beta}_{i-1,b})$ from the next input task distribution tk_i, and compute the sum of log-probabilities with *each* of its hierarchically connected j vertices (i.e., $\log(tk^{\beta}_{i-1,b}) + \log(tk_{i,j})$). After computing all the partial solutions for the current iteration, we use the function $f(.)$ to only consider the top-β partial solutions to compute tk^{β}_{i}, which retains the summed log-probabilities of its traversed vertices, and will be used for the next iteration. We illustrate an example of HBS with $\beta = 2$ in Figure 2, performed for only the input pair of $(tk_1^{G@2}, tk_2^{G@3})$ task distributions. Effectively, after performing the HBS on all task distributions from the graph G_{hs}, we can reduce the search space significantly to reduce noise. Specifically, with the input pair of $(tk^{G@6}_{i-1}, tk^{POI}_{i})$ for the last iteration, the search space is reduced to only consider the POIs within the top-β of the $tk^{G@6}_{i-1}$ regions or cells, instead of all $|L|$ POIs, as well as computing their respective joint probability $P(G@2, G@3, \dots, G@6, l_{t_i} \mid l_{t_{i-1}})$ to derive the ranked set.

Selectivity Layer. To balance between personalization and exploration, we propose a novel selectivity layer by predicting if the next POI has been visited before by the user, which would inform the model to *personalize* or *explore*. To predict if the next POI is a visited ($l_{t_i} \in s_{u_m}^{train}$) or unvisited ($l_{t_i} \notin s_{u_m}^{train}$) POI by the user as a binary classification task, and to reduce additional parameters, we simply retrieve the next predicted POI $\hat{l}_{t_i} = argmax_{l_{t_i}}(tk^{POI})$ from the next POI task distribution tk^{POI} in Equation (9) and compute:

$$y_{t_i} = \begin{cases} \Psi\,(\,P(l_{t_i} \mid l_{t_{i-1}})\,) & \hat{l}_{t_i} \in s_{u_m}^{train} \\ \Psi\,(\,P(G@2, G@3, \ldots, G@6, l_{t_i} \mid l_{t_{i-1}})\,) & \text{otherwise} \end{cases} \tag{12}$$

where y_{t_i} is the computed next POI ranked set and $\Psi(.)$ sorts a given distribution to rank POIs from their probability scores in descending order. If the predicted next POI has been visited before by the user ($\hat{l}_{t_i} \in s_{u_m}^{train}$), the selectivity layer *personalizes* by using the next POI task distribution $tk^{POI} = P(l_{t_i} \mid l_{t_{i-1}})$ to compute the ranked set, as it includes the use of user embedding to better capture user preferences from $s_{u_m}^{train}$, as per Equation (9). Otherwise, it *explores* by performing the HBS to rank POIs based on a regional context, using the joint probability of the multiple tasks $P(G@2, G@3, \ldots, G@6, l_{t_i} \mid l_{t_{i-1}})$.

4.2 Graph Recurrent Network (GRN)

Next, we propose the GRN module to replace the LSTM in our HMT-RN model, to allow the additional learning of global POI-POI relationships. Among the existing works, recurrent models (e.g., LSTM) have been shown to be effective in learning sequential dependencies of each user's POI sequence, however, as highlighted in [35], it does not learn global POI-POI relationships directly as compared to graph neural networks (e.g., GAT). Similarly, a drawback of GAT [50], used in [35] can also be observed from being unable to learn the sequential dependencies in a sequence, unlike recurrent models. Therefore, a natural consideration is if both factors can be modelled by a single model, specifically, a GRN to learn both (a) sequential dependencies and (b) global POI-POI relationships. To this end, we propose a novel GRN module for the next POI recommendation task by extending the **Dimensional GAT (DGAT)** variant in [35] with the *addition* of (a) a recurrent structure and (b) the alleviation of data sparsity by using regions and time slots to connect POIs in the spatial and temporal graphs. First, the DGAT variant in [35] is defined as:

$$\vec{\alpha}_{l_{t_{i-1}},j} = \frac{exp(LeakyReLU(\mathbf{a}\,[\mathbf{W}_p \vec{l_{t_{i-1}}} \oplus \mathbf{W}_p \vec{j}]))}{\sum_{\vec{k} \in \hat{N}_G[l_{t_{i-1}}]} exp(LeakyReLU(\mathbf{a}\,[\mathbf{W}_p \vec{l_{t_{i-1}}} \oplus \mathbf{W}_p \vec{k}]))} \tag{13}$$

$$p_{t_i} = \sum_{\vec{j} \in \hat{N}_G[l_{t_{i-1}}]} \vec{\alpha}_{l_{t_{i-1}},j} \odot \mathbf{W}_p\, \vec{j} \tag{14}$$

where $\mathbf{W}_p \in \mathbb{R}^{\delta \times hdim}$ is an input projection, and $\mathbf{a}$ is a linear layer parameterized with $\mathbf{W}_a \in \mathbb{R}^{2 \cdot hdim \times hdim}$ to predict the attention weights $\vec{\alpha}_{l_{t_{i-1}},j} \in \mathbb{R}^{hdim}$ between the previous POI input $l_{t_{i-1}}$ and each POI of its closed neighbourhood (i.e., adjacent neighbours and itself) $\vec{j} \in \hat{N}_G[l_{t_{i-1}}]$ from a POI-POI graph G, and $\vec{l}_{t_{i-1}}, \vec{j} \in \mathbb{R}^{\delta}$ are POI embeddings from Equation (7). The predicted weights $\vec{\alpha}_{l_{t_{i-1}},j}$ are then applied in Equation (14) to compute a weighted sum of its respective neighbours, outputting the hidden representation $p_{t_i} \in \mathbb{R}^{hdim}$.

Next, different from [35] that uses sparse POI-POI relationships to connect POIs in their spatial and temporal POI-POI graphs, we instead propose to connect POIs based on their POI-Region and POI-Timeslot relationships respectively, to reduce the impact of data sparsity. Specifically:

Definition 4.2 (Spatial Graph). An undirected and unweighted POI-POI graph denoted as $G_s = (V_s, E_s)$ where $V_s = L$ and E_s are the sets of POIs and edges respectively. Each pair of POIs has adjacency if they are within the same G@4 grid cell.

Definition 4.3 (Temporal Graph). An undirected and unweighted POI-POI graph denoted as $G_t = (V_t, E_t)$ where $V_t = L^{train}$ and E_s are the sets of POIs and edges respectively. We first partition each day to 8 time slots of 3hrs each, with a total of 56 time slots, then map each visit in S^{train} to its corresponding time slot, where each POI vertex v_t will have a set of mapped time slots v_t^{slot}. Each pair of POIs (e.g., v_{t_i} and v_{t_j}) has adjacency if their time slot sets have a Jaccard similarity $\frac{|v_{t_i}^{slot} \cap v_{t_j}^{slot}|}{|v_{t_i}^{slot} \cup v_{t_j}^{slot}|}$ above 0.9.

To model global POI-POI relationships, here, we abbreviate the DGAT layer in Equation (14) to Γ and compute the hidden representations of $p_{t_i}^{G_s}$ and $p_{t_i}^{G_t}$ from the proposed G_s and G_t graphs respectively:

$$p_{t_i}^{G_s} = \Gamma_s(\vec{l}_{t_{i-1}}) \tag{15}$$

$$p_{t_i}^{G_t} = \Gamma_t(\vec{l}_{t_{i-1}}) \tag{16}$$

Further, to simultaneously learn the sequential dependencies, we include the computed representations with a recurrent structure by modifying Equations (1) to (4) to the below:

$$i_{t_i} = \sigma(\mathbf{W}_i\, x_{t_i} + \mathbf{U}_i\, h_{t_{i-1}} + \mathbf{b}_i + \mathbf{V}_i\, p_{t_i}^{G_s} + \mathbf{Z}_i\, p_{t_i}^{G_t}) \tag{17}$$

$$f_{t_i} = \sigma(\mathbf{W}_f\, x_{t_i} + \mathbf{U}_f\, h_{t_{i-1}} + \mathbf{b}_f + \mathbf{V}_f\, p_{t_i}^{G_s} + \mathbf{Z}_f\, p_{t_i}^{G_t}) \tag{18}$$

$$o_{t_i} = \sigma(\mathbf{W}_o\, x_{t_i} + \mathbf{U}_o\, h_{t_{i-1}} + \mathbf{b}_o + \mathbf{V}_o\, p_{t_i}^{G_s} + \mathbf{Z}_o\, p_{t_i}^{G_t}) \tag{19}$$

$$\tilde{c}_{t_i} = tanh(\mathbf{W}_c\, x_{t_i} + \mathbf{U}_c\, h_{t_{i-1}} + \mathbf{b}_c + \mathbf{V}_c\, p_{t_i}^{G_s} + \mathbf{Z}_c\, p_{t_i}^{G_t}) \tag{20}$$

where $\mathbf{V}_i, \mathbf{V}_f, \mathbf{V}_o, \mathbf{V}_c \in \mathbb{R}^{\delta \times hdim}$ and $\mathbf{Z}_i, \mathbf{Z}_f, \mathbf{Z}_o, \mathbf{Z}_c \in \mathbb{R}^{\delta \times hdim}$ are the weight matrices for $p_{t_i}^{G_s}$ and $p_{t_i}^{G_t}$ respectively, for all the gates and the cell input, to complete our proposed GRN module. Lastly, we illustrate our HMT-GRN model in Figure 3, after replacing the LSTM module in our HMT-RN model with the GRN module.

4.3 Joint Triplet Loss Learning (JTLL)

Our motivation of the JTLL module is to learn both visited and unvisited relations of users and POIs to better learn the User-POI matrix, while alleviating sparsity by learning the relations.

Loss Function. In Figure 4, we illustrate an example, where the search space of POIs in a region $\{l_1, l_2, l_3, l_4, l_5, l_6\}$ is considered, to determine the next POI, and user u_m having visited l_1 and l_2 before (i.e., visited relation), and user u_m not having visited $\{l_3, l_4, l_5, l_6\}$ before (i.e., unvisited relation). To best model these relations, we propose a designed triplet loss function, based on the visited and unvisited relations of users for the recommendation task. First, we compute an additional set of training data, based on users who visit a given POI. Formally, given the available users' historically visited POIs $s_{u_m}^{train}$, we compute the training data l^{train}, which is a set of tuples $(u_h, l_b) \in l^{train}$, and each tuple (u_h, l_b) denotes the check-in event relation of user u_h having visited POI l_b in $s_{u_m}^{train}$. Next, given each training tuple $(u_h, l_b) \in l^{train}$, and its embeddings of $\mathbf{u}_h$, $\mathbf{l}_b$ (in boldface letters) from the weight matrices of user $\mathbf{W}_u$ and POI $\mathbf{W}_l$ respectively, we design the following loss function, with inspirations from the triplet loss [44] and the negative sampling loss [42]:

$$J_{loss} = -\log\left(\sigma\left(\mathbf{u}_h\, \mathbf{l}_b^\top\right)\right) - \sum_{\mathbf{u}_n \in l_b^{Neg}} \log\left(\sigma\left(-\mathbf{u}_n\, \mathbf{l}_b^\top\right)\right) \tag{21}$$

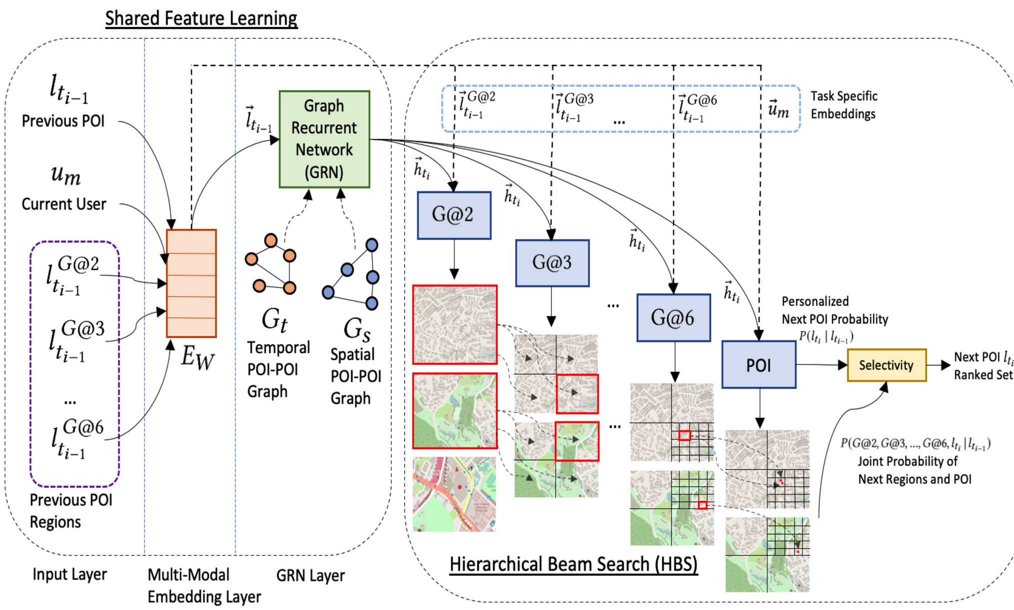


Fig. 3. Illustration of the HMT-GRN model that includes shared feature learning by our GRN module, followed by the multi-task learning of next regions and POI, then performing our HBS and selectivity layer. An example of top-β = 2 (red boxes) is used by HBS to traverse the multi-task distributions and reduce search space. Maps © OpenStreetMap contributors, CC BY-SA.

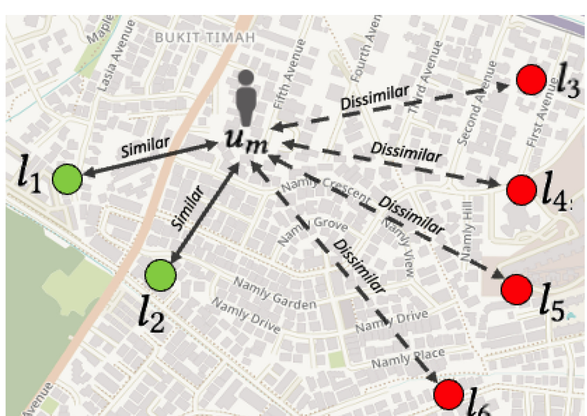


Fig. 4. Learning visited and unvisited relations of users and POIs with our designed triplet loss function. Maps © OpenStreetMap contributors, CC BY-SA.

where σ is the sigmoid activation function, POI $\mathbf{l}_b$ serves as an *anchor*, $\mathbf{u}_h$ as a *positive* user who have visited POI l_b before, $\mathbf{u}_n \in l_b^{Neg}$ as a *negative* user from the set of users l_b^{Neg}, who have *never* visited POI l_b before, and can be computed from $s_{u_m}^{train}$. Intuitively, optimizing the loss function reduces the distance of positive users and the anchor POI (user visited POI before), and increases the distance of negative users and the anchor POI (user never visited POI before).

Joint Training. As existing POI recommendation models, and our HMT-GRN model, do not learn the unvisited relation between users and POIs, which can be helpful for sparsity alleviation, we

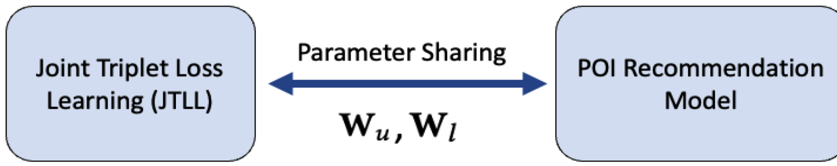


Fig. 5. Proposed joint training framework.

propose to use a joint training framework with parameter sharing, shown to be effective in other problems [41], to easily incorporate our JTLL module and a designated existing POI recommendation model (e.g., HMT-GRN), where the parameters $\{\mathbf{W}_u, \mathbf{W}_l\}$ are shared among the two models for their own updates. Specifically, for each epoch, our JTLL module first optimizes Equation (21) with the training data l^{train}, with gradient steps to the user $\mathbf{W}_u$ and POI $\mathbf{W}_l$ weight matrices. Then, the designated POI recommendation model will perform its own optimization and updates to these shared parameters, as well as other parameters unique to the model, concluding the end of a single epoch. We illustrate the proposed joint training framework in Figure 5, based on our JTLL module.

5 EXPERIMENTS

5.1 Datasets

We evaluate our HMT-GRN model, and our JTLL module, on two popular LBSN datasets of Gowalla [5] and Foursquare [62] for the next POI recommendation task, where each user has a sequence of POI visits that could be made in different days. For preprocessing, similar to [35, 72], we consider users with visit counts between 20 and 50 in the datasets, then removing POIs visited by less than 10 users, reporting the statistics in Table 1. Accordingly, there are 64 G@2, 231 G@3, 513 G@4, 990 G@5, 1,794 G@6 regions for the Gowalla dataset, and 111 G@2, 280 G@3, 547 G@4, 1,354 G@5, 2,757 G@6 regions for the Foursquare dataset, which are also the number of regions considered in our Hierarchical Spatial Graph. For training and testing, we similarly use the first 80% visits and the last 20% visits of each user's sequence respectively, after sorting the timestamps in chronological order. Same as the decreasing sparsity trend in Figure 1(a) based on the unprocessed datasets, we can observe in Table 1, after preprocessing, that the sparsity is the highest for the User-POI matrix (99.83% and 99.84%), and lowest for the User-G@2 matrix (97.39% and 98.27%).

5.2 Baseline Methods and Evaluation Metrics

- **TOP:** We rank the POIs using their global frequencies in S^{train} for popular POIs. **U-TOP** instead ranks the POIs using each users' historical sequence $s^{train}_{u_m}$ based on the users' visiting frequencies.
- **MF** [22]**:** A popular collaborative filtering method for recommendation problems by factorizing the User-POI matrix.
- **RNN** [10]**:** A recurrent model that learns sequential dependencies of POI visit sequences but suffers from the vanishing gradient problem. The variants of **LSTM** [18] and **GRU** [6] uses various multiplicative gates to control information flow.
- **HST-LSTM** [21]**:** A LSTM-based model that leverages spatial and temporal intervals between sequential POIs into the LSTM existing gates. Like [35] and [72], we use their ST-LSTM variant here as the data does not have session information. **STGCN** [72] similarly models the intervals with new time and distance gates, and a new cell state for short term preference learning.

Table 1. Statistics of the LBSN Datasets (After Preprocessing)

Dataset	#Country	#User	#POI	#Visits	Sparsity					
					G@2	G@3	G@4	G@5	G@6	POI
Gowalla[4]	41	11,864	3,359	86,670	97.39%	99.07%	99.45%	99.62%	99.73%	99.83%
Foursquare[5]	63	16,636	4,455	170,573	98.27%	99.11%	99.39%	99.62%	99.77%	99.84%

- **LSTPM** [48]**:** A LSTM-based model that learns long term user preferences through a nonlocal network, and short-term user preferences with a geo-dilated network.
- **STAN** [40]**:** A bi-attention model that incorporates spatio-temporal correlations of non-adjacent POIs and non-contiguous visits.
- **Flashback** [61]**:** A RNN architecture that leverages spatial and temporal intervals to compute an aggregated hidden state from past hidden states for prediction. We use their best performing RNN variant here for evaluation.
- **STP-UDGAT** [35]**:** A GAT-based approach that models spatio-temporal-preference factors though various POI-POI graphs in an explore-exploit architecture.
- **HMT-RN:** Our HMT-RN model, as defined in Section 4.1, that includes (a) multi-task learning, (b) HBS, and (c) selectivity layer.
- **HMT-GRN:** We replace the LSTM layer in the HMT-RN model with a GRN layer.

For the recent baseline of STP-UDGAT, and our HMT-GRN, we use them each as the designated POI recommendation model in our joint learning framework (Figure 5), with our JTLL module, and denoting the extended variants as **STP-UDGAT-*JTLL*** and **HMT-GRN-*JTLL*** accordingly. Note that for both STP-UDGAT, and our HMT-GRN model, they do not learn the unvisited relations between users and POIs, and therefore, following our motivation, we extend these POI recommendation models with our JTLL module to additionally learn the unvisited relations, and further alleviating sparsity.

For both HST-LSTM and STGCN, following [34], [35], and [61] , given the previous POI $l_{t_{i-1}}$ to predict the next POI l_{t_i}, we use the spatial and temporal intervals between $l_{t_{i-2}}$ and $l_{t_{i-1}}$ instead of $l_{t_{i-1}}$ and l_{t_i} as this would require knowing the next POI visit l_{t_i} in advance, which is impractical in a real-world setting [11].

Metrics. Same as in [21], [35], and [72], and other existing works, we use the standard metric of Acc@$K = \frac{1}{N_{test}} \#hit@K$ where $\#hit@K$ is the number of samples with the correct predictions made within the top K of the ranked set for $K \in \{1, 5, 10, 20\}$, and N_{test} is the total number of test samples. We also evaluate for the metric of **Mean Reciprocal Rank (MRR)** where $MRR = \frac{1}{N_{test}} \cdot \sum_{v=1}^{N_{test}} \frac{1}{rank_v(l_{t_i})}$, and $rank_v(l_{t_i})$ is the position of the ground truth next POI l_{t_i} in the predicted ranked set for each vth test sample. Effectively, Acc@K helps to understand the performance of the recommender system for the top K recommendations, whereas MRR gives an overall performance of the ranked set predicted.

Next New Metrics. A limitation of recent N^2 POI recommendation works [4, 12, 13, 39] is that they only consider POI samples visited within the next $\tau = 6$ hours of the previous POI check-in for both training and testing to learn short-term preferences. As observed in [39], the $\tau = 6$ hours threshold filters the data to a small subset (around 20% to 30% for Foursquare and Gowalla) of all real-world cases, limiting the practicality of the recommender system where long-term preferences

[4] https://snap.stanford.edu/data/loc-gowalla.html
[5] https://sites.google.com/site/yangdingqi/home

are not learned. Thus, to overcome this limitation, in addition to the Acc@K and MRR metrics, we use the N^2 extensions of N^2-Acc@K and N^2-MRR to only evaluate next *unvisited or new* POI recommendations, and *without* the τ time constraint so that both short and long term user preferences can be learned and evaluated for all real-world cases. This set of N^2 metrics is necessary to ensure that the recommender system does not always just recommend historically visited POIs correctly, but also new unvisited POIs which the user will visit in the future, correctly, thereby improving the user experience and supporting them to explore new places of interest [28, 75].

Concretely, given the total number of test samples N_{test} that contains both visited and unvisited next POIs, used for evaluation in Acc@K and MRR, we replace N_{test} with N_{test}^{new} where the test samples should be new or unvisited $l_{t_i} \notin s_{u_m}^{train}$ for the N^2 metrics. Additionally, only for the N^2 metrics, given the predicted ranked set of y_{t_i} from each baseline and model, we remove visited POIs (i.e., $y_{t_i} \setminus s_{u_m}^{train}$) to correctly evaluate for the unvisited test samples. For our HMT-GRN model, we *deactivate* the selectivity layer in Equation (12) for only the N^2 metrics, and compute the ranked set $y_{t_i} = \Psi\left(P(G@2, G@3, \ldots, G@6, l_{t_i} \mid l_{t_{i-1}})\right)$ via joint probability directly as N_{test}^{new} only contains unvisited test samples, removing the need for the selectivity layer.

5.3 Experimental Settings

For our HMT-GRN model, we use the Adam optimizer, 20 epochs, a batch size of 32, a learning rate of 0.0001, $\beta = 100$ for HBS, and set the dropout rate to be 0.9, then set the POI, user, geohash embedding dimension δ and hidden dimension $hdim$ to be the same value of 1,024. For fair comparison, for MF, RNN, GRU, and LSTM, we use the same settings where applicable. For the other recent works of HST-LSTM, STGCN, LSTPM and STP-UDGAT, we follow their recommended settings as described. For baseline models that do not perform as well as expected (compared to the reported values in their original papers) when using their respective recommended hyperparameters (i.e., Flashback and STAN), we have performed grid search for them, including their embedding dimensions and number of epochs, and had reported the best performing models in our evaluation. For our JTLL module, we similarly use the Adam optimizer with a learning rate of 0.001, as well as a batch size of 64 tuples from the training data l^{train}. Further, we apply a dropout of 0.8 to the user and POI embeddings from the weight matrices $\mathbf{W}_u$ and $\mathbf{W}_l$ before optimizing the loss function in Equation (21). For all other hyperparameters (e.g., number of epochs and embedding dimension size), we set it to be the same as the respective designated POI recommendation model for simplicity and ease of implementation.

5.4 Results

We report the evaluation results of our proposed HMT-GRN model, and our JTLL module, as well as the baselines in Table 2. For all baselines and models, except TOP and U-TOP which are deterministic, we show the averaged results of 5 runs on different random seeds, as well as their respective standard deviations:

- For the N^2 POI recommendation task, evaluated by the set of N^2 metrics, which are more challenging [12], we observe that our final variant, HMT-GRN-*JTLL* have the best performance for all N^2 metrics and for all datasets. For the next POI recommendation task, or the Acc@K and MRR metrics, we see our HMT-GRN model outperforms all the baselines significantly, except for the Acc@1 metric of the Foursquare dataset by 1%.
- U-TOP and the recent works of Flashback, STP-UDGAT and LSTPM are the most competitive baselines. U-TOP was able to perform well as users would tend to visit their own frequently visited POIs [35]. Further, unlike the other machine learning baselines, as U-TOP does not rely on learning the sparse User-POI matrix, it was thus able to perform competitively.

Table 2. Performance in Acc@K and MRR for all Next POI Test Samples (i.e., Visited and Unvisited POIs), as Well as the Corresponding N^2 Metrics of N^2-Acc@K and N^2-MRR for Only Unvisited Next POI Test Samples

Gowalla										
	Acc@1	Acc@5	Acc@10	Acc@20	MRR	N^2-Acc@1	N^2-Acc@5	N^2-Acc@10	N^2-Acc@20	N^2-MRR
TOP	0.0084	0.0351	0.0678	0.1022	0.0270	0.0068	0.0281	0.0574	0.0874	0.0227
U-TOP	0.1423	0.2767	0.3035	0.3110	0.1986	0	0	0	0	0
MF	0.0644 ± 0.001	0.0785 ± 0.001	0.0825 ± 0.001	0.0879 ± 0.001	0.0736 ± 0.001	0.0015 ± 0.001	0.0032 ± 0.001	0.0046 ± 0.001	0.0073 ± 0.001	0.0040 ± 0.001
RNN	0.0844 ± 0.002	0.1873 ± 0.001	0.2440 ± 0.001	0.3050 ± 0.001	0.1381 ± 0.001	0.0356 ± 0.001	0.1045 ± 0.001	0.1496 ± 0.001	0.2034 ± 0.001	0.0746 ± 0.001
GRU	0.0865 ± 0.001	0.1869 ± 0.001	0.2489 ± 0.001	0.3161 ± 0.001	0.1406 ± 0.001	0.0367 ± 0.001	0.1064 ± 0.001	0.1563 ± 0.001	0.2150 ± 0.001	0.0773 ± 0.001
LSTM	0.0968 ± 0.001	0.1968 ± 0.001	0.2575 ± 0.001	0.3276 ± 0.002	0.1510 ± 0.001	0.0419 ± 0.001	0.1140 ± 0.001	0.1661 ± 0.001	0.2291 ± 0.001	0.0843 ± 0.001
HST-LSTM	0.0087 ± 0.001	0.0366 ± 0.001	0.0636 ± 0.002	0.1004 ± 0.001	0.0279 ± 0.001	0.0069 ± 0.001	0.0293 ± 0.001	0.0545 ± 0.002	0.0854 ± 0.001	0.0233 ± 0.001
STGCN	0.0313 ± 0.001	0.0909 ± 0.003	0.1351 ± 0.005	0.1955 ± 0.004	0.0684 ± 0.001	0.0126 ± 0.001	0.0460 ± 0.002	0.0777 ± 0.003	0.1269 ± 0.003	0.0374 ± 0.001
LSTPM	0.1297 ± 0.001	0.2282 ± 0.001	0.2720 ± 0.001	0.3200 ± 0.002	0.1803 ± 0.001	0.0353 ± 0.001	0.0869 ± 0.002	0.1199 ± 0.002	0.1613 ± 0.003	0.0653 ± 0.001
STAN	0.0939 ± 0.002	0.1928 ± 0.003	0.2440 ± 0.003	0.3039 ± 0.006	0.1460 ± 0.002	0.0143 ± 0.001	0.0513 ± 0.002	0.0843 ± 0.002	0.1323 ± 0.005	0.0398 ± 0.001
Flashback	0.1266 ± 0.001	0.2342 ± 0.001	0.2770 ± 0.002	0.3285 ± 0.001	0.1821 ± 0.001	0.0153 ± 0.001	0.0517 ± 0.002	0.0825 ± 0.003	0.1288 ± 0.001	0.0412 ± 0.001
STP-UDGAT	0.1194 ± 0.001	0.2374 ± 0.001	0.2783 ± 0.001	0.3202 ± 0.002	0.1770 ± 0.001	0.0251 ± 0.001	0.0712 ± 0.001	0.1014 ± 0.001	0.1393 ± 0.002	0.0517 ± 0.001
STP-UDGAT-*JTLL*	0.1198 ± 0.001	0.2371 ± 0.003	0.2812 ± 0.003	0.3288 ± 0.003	0.1780 ± 0.002	0.0338 ± 0.001	0.0926 ± 0.001	0.1290 ± 0.002	0.1746 ± 0.002	0.0671 ± 0.001
HMT-RN	0.1434 ± 0.001	0.2677 ± 0.001	0.3213 ± 0.001	0.3781 ± 0.001	0.2053 ± 0.001	0.0523 ± 0.001	0.1334 ± 0.001	0.1888 ± 0.001	0.2536 ± 0.001	0.0987 ± 0.001
HMT-GRN	**0.1455 ± 0.001**	**0.2783 ± 0.001**	**0.3394 ± 0.001**	**0.4033 ± 0.001**	**0.2120 ± 0.001**	0.0539 ± 0.001	0.1369 ± 0.001	0.1920 ± 0.001	0.2579 ± 0.001	0.1008 ± 0.001
HMT-GRN-*JTLL*	0.1379 ± 0.001	0.2560 ± 0.002	0.3173 ± 0.001	0.3842 ± 0.001	0.1994 ± 0.001	**0.0543 ± 0.001**	**0.1388 ± 0.001**	**0.1970 ± 0.001**	**0.2637 ± 0.001**	**0.1026 ± 0.001**
Foursquare										
	Acc@1	Acc@5	Acc@10	Acc@20	MRR	N^2-Acc@1	N^2-Acc@5	N^2-Acc@10	N^2-Acc@20	N^2-MRR
TOP	0.0082	0.0353	0.0546	0.0869	0.0263	0.0056	0.0247	0.0373	0.0604	0.0192
U-TOP	**0.1690**	0.3297	0.3796	0.3979	0.2382	0	0	0	0	0
MF	0.0687 ± 0.001	0.0859 ± 0.001	0.0905 ± 0.001	0.0954 ± 0.001	0.0789 ± 0.001	0.0009 ± 0.001	0.0028 ± 0.001	0.0043 ± 0.001	0.0061 ± 0.001	0.0031 ± 0.001
RNN	0.1078 ± 0.001	0.2246 ± 0.001	0.2973 ± 0.002	0.3752 ± 0.001	0.1700 ± 0.001	0.0444 ± 0.001	0.1272 ± 0.001	0.1858 ± 0.001	0.2536 ± 0.001	0.0914 ± 0.001
GRU	0.1103 ± 0.001	0.2300 ± 0.001	0.3027 ± 0.001	0.3852 ± 0.002	0.1740 ± 0.001	0.0459 ± 0.001	0.1306 ± 0.001	0.1908 ± 0.001	0.2644 ± 0.001	0.0945 ± 0.001
LSTM	0.1191 ± 0.001	0.2437 ± 0.001	0.3174 ± 0.001	0.4032 ± 0.002	0.1854 ± 0.001	0.0505 ± 0.001	0.1400 ± 0.001	0.2035 ± 0.001	0.2828 ± 0.001	0.1023 ± 0.001
HST-LSTM	0.0076 ± 0.001	0.0307 ± 0.001	0.0500 ± 0.001	0.0806 ± 0.001	0.0244 ± 0.001	0.0058 ± 0.001	0.0218 ± 0.001	0.0369 ± 0.001	0.0591 ± 0.001	0.0181 ± 0.001
STGCN	0.0276 ± 0.002	0.0948 ± 0.005	0.1531 ± 0.006	0.2323 ± 0.005	0.0703 ± 0.003	0.0114 ± 0.001	0.0497 ± 0.002	0.0896 ± 0.003	0.1503 ± 0.003	0.0400 ± 0.001
LSTPM	0.1478 ± 0.001	0.2671 ± 0.002	0.3214 ± 0.002	0.3778 ± 0.001	0.2078 ± 0.001	0.0426 ± 0.001	0.1052 ± 0.001	0.1466 ± 0.001	0.1951 ± 0.001	0.0782 ± 0.001
STAN	0.1066 ± 0.004	0.2382 ± 0.007	0.3136 ± 0.008	0.3987 ± 0.010	0.1759 ± 0.005	0.0249 ± 0.002	0.0870 ± 0.007	0.1384 ± 0.008	0.2070 ± 0.011	0.0643 ± 0.004
Flashback	0.1442 ± 0.001	0.2768 ± 0.002	0.3347 ± 0.002	0.4012 ± 0.001	0.2118 ± 0.002	0.0229 ± 0.001	0.0742 ± 0.001	0.1185 ± 0.001	0.1820 ± 0.001	0.0577 ± 0.001
STP-UDGAT	0.1397 ± 0.001	0.2926 ± 0.002	0.3556 ± 0.002	0.4187 ± 0.001	0.2136 ± 0.001	0.0382 ± 0.001	0.1156 ± 0.002	0.1699 ± 0.002	0.2324 ± 0.001	0.0811 ± 0.001
STP-UDGAT-*JTLL*	0.1381 ± 0.001	0.2798 ± 0.001	0.3346 ± 0.002	0.3900 ± 0.002	0.2063 ± 0.001	0.0412 ± 0.001	0.1153 ± 0.002	0.1612 ± 0.001	0.2154 ± 0.002	0.0811 ± 0.001
HMT-RN	0.1617 ± 0.001	0.3257 ± 0.001	0.3961 ± 0.001	0.4673 ± 0.001	0.2415 ± 0.001	0.0670 ± 0.001	0.1738 ± 0.001	0.2486 ± 0.001	0.3357 ± 0.001	0.1269 ± 0.001
HMT-GRN	0.1673 ± 0.001	**0.3357 ± 0.002**	**0.4148 ± 0.001**	**0.4983 ± 0.001**	**0.2510 ± 0.001**	0.0686 ± 0.001	0.1756 ± 0.002	0.2507 ± 0.001	0.3386 ± 0.001	0.1288 ± 0.001
HMT-GRN-*JTLL*	0.1619 ± 0.001	0.3085 ± 0.001	0.3870 ± 0.001	0.4733 ± 0.001	0.2374 ± 0.001	**0.0704 ± 0.001**	**0.1779 ± 0.001**	**0.2553 ± 0.001**	**0.3452 ± 0.001**	**0.1313 ± 0.001**

However, as it is only able to rank visited POIs by the user, and not unvisited POIs, the set of N^2 metric scores will always be 0, as shown in Table 2, making it a less practical approach in a real-world setting.

- Unlike [35], which focuses on only learning from short sequences (i.e., $|s_{u_m}|$<30), we observe in Table 2 that when learning from instead longer sequences (i.e., $|s_{u_m}|$<50) in our experiments, STP-UDGAT does not always perform well as it is unable to learn the sequential dependencies of POI transitions, due to the design of GAT.
- Comparing the LSTM baseline with our LSTM-based HMT-RN variant, where the former only learns the User-POI matrix and the latter learns both User-POI and User-G@P matrices, we can observe that our HMT-RN variant always surpasses the LSTM baseline significantly, demonstrating the effectiveness of our proposed region-based tasks to alleviate sparsity.
- For the MF, RNN, and GRU baselines, like the LSTM baseline, as they similarly rely on the sparse User-POI matrix for learning, they do not perform as well. Although Flashback, STP-UDGAT and LSTPM also learn from the sparse User-POI matrix, each of them model additional factors to improve performances, such as spatio-temporal relationships among POIs in different ways.
- Our HMT-GRN variant always outperforms the HMT-RN variant, with the only difference being replacing the LSTM layer in HMT-RN with a GRN layer to perform shared feature

learning. This increase in performance indicates the importance of the GRN module to also learn global spatio-temporal POI-POI relationships, as the LSTM only learns sequential dependencies.

- For STGCN and HST-LSTM, similar to [34] and [35] , we believe that these models may not have performed well by learning from the spatial and temporal intervals between $l_{t_{i-2}}$ and $l_{t_{i-1}}$.
- For our JTLL module, while we observe that the final variant of HMT-GRN-$\mathcal{JTLL}$, extended from our HMT-GRN model, have the best performance for all N^2 metrics and for all datasets, its performance for the Acc@K and MRR metrics, however, are not better than the HMT-GRN model. This result is to be expected due to the explore-exploit tradeoffs for the user embedding. Traditionally, the user embedding is used by existing works [11, 12, 35, 61, 74], as well as our HMT-GRN model, to enable personalization, mostly by biasing the learning of the user embedding to the historical POIs visited by the respective user, and thus, performing well for the Acc@K and MRR metrics that include historical POI ground truths. However, such personalization, or exploitation, can lead to overfitting to the respective users, based on their historical records. Therefore, as our designed triplet loss function in Equation (21) considers the novel incorporation of the unvisited relation between users and unvisited POIs, this reduces the bias of the user embedding to the historically visited POIs by the users, and was thus able to achieve best performances across all baselines for all N^2 metrics and datasets, in predicting the next new POI correctly.
- For the N^2 metrics, apart from the observation that HMT-GRN-$\mathcal{JTLL}$ always have better performances than the HMT-GRN model, comparing STP-UDGAT-$\mathcal{JTLL}$ and STP-UDGAT, we similarly observe the unanimous improvements for the Gowalla dataset. For the Foursquare dataset, STP-UDGAT-$\mathcal{JTLL}$ was only better for the N^2-Acc@1 metric (i.e., predicting the next new POI at the first position of the ranked set correctly). We believe that this is due to the use of a 0.8 dropout rate in our experimental setting for the regularization of our JTLL module, however, the high dropout rate might have hindered the general learning capability of the model, and thus, not performing better for the remaining N^2 metrics. However, we see that the same high dropout rate did not adversely affect the learning ability of the model for the Gowalla dataset, as STP-UDGAT-$\mathcal{JTLL}$ always have better performances than STP-UDGAT.

5.5 Analysis

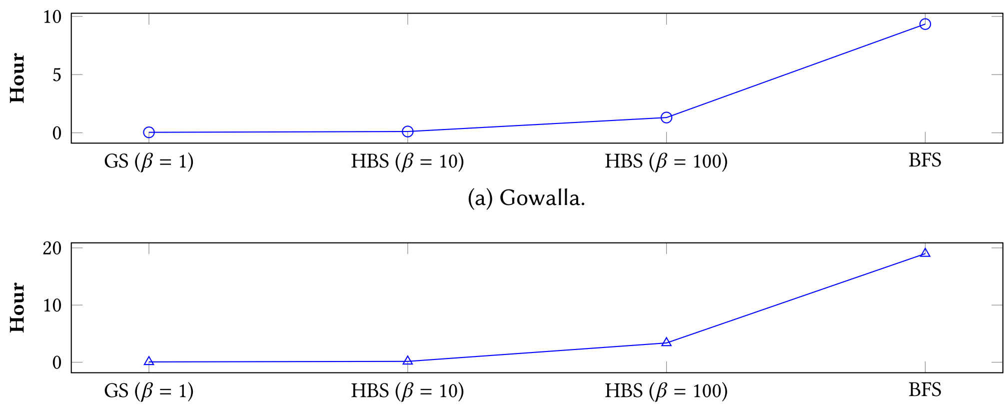
In this section, we conduct several analysis, mostly of our HMT-GRN model, as it has the best overall results (Acc@K and MRR). Following, we conduct an ablation study of our JTLL module, as well as sharing a case study.

Efficiency. In Table 3, for our HMT-GRN model, we compare the performance between Greedy Search (GS) (i.e., $\beta = 1$), an exhaustive method of BFS, and our HBS ($\beta = 10$ and $\beta = 100$) on the multiple task distributions in the hierarchical graph G_{hs} with the N^2 metrics, which are more challenging. Results show that our HBS with $\beta = 100$ has mostly better or comparable performances to BFS by only expanding the top-β promising vertices of each task distribution instead of all vertices (i.e., BFS). More importantly, in Figure 6, we can see that our HBS ($\beta = 100$) requires much less time to compute the joint probabilities, specifically 5 and 7 times faster than BFS for the Foursquare and Gowalla datasets, respectively. Therefore, our HBS provides both performance and efficiency gains significantly.

Importance of Proposed Tasks. In Table 4, we similarly evaluate the performance of the HBS, used in our HMT-GRN model, on the N^2 metrics, but by *deactivating* certain task distributions when performing the HBS. Specifically, *All Tasks* considers all task distributions in

Table 3. Performance Comparison of Search Methods

Dataset	Search Method	N^2-Acc@1	N^2-Acc@5	N^2-Acc@10	N^2-Acc@20	N^2-MRR
Gowalla	GS (β = 1)	0.0408	0.0762	0.0911	0.1122	0.0592
	BFS	0.0537	**0.1381**	0.1913	0.2566	**0.1008**
	HBS (β = 10)	0.0536	0.1366	0.1857	0.2354	0.0966
	HBS (β = 100)	**0.0539**	0.1369	**0.1920**	**0.2579**	**0.1008**
Foursquare	GS (β = 1)	0.0473	0.0745	0.0820	0.0872	0.0613
	BFS	0.0672	0.1750	0.2504	0.3385	0.1285
	HBS (β = 10)	0.0671	0.1742	0.2444	0.3017	0.1205
	HBS (β = 100)	**0.0686**	**0.1756**	**0.2507**	**0.3386**	**0.1288**



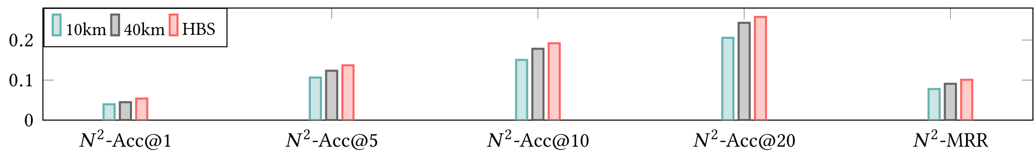
(a) Gowalla.

(b) Foursquare.

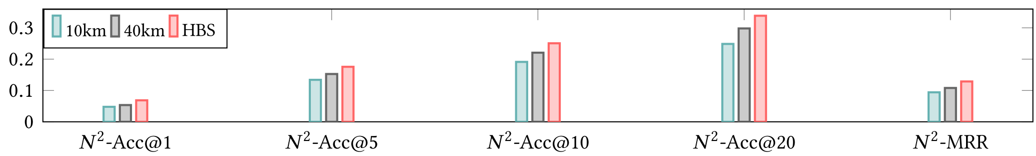
Fig. 6. Efficiency comparison of search methods.

Table 4. Effectiveness of Proposed Tasks for HBS

Dataset		N^2-Acc@1	N^2-Acc@5	N^2-Acc@10	N^2-Acc@20	N^2-MRR
Gowalla	All Tasks	**0.0539**	**0.1369**	0.1920	0.2579	**0.1008**
	G@2 + POI	0.0490	0.1335	0.1883	0.2611	0.0980
	G@3 + POI	0.0517	0.1364	**0.1943**	**0.2652**	0.1004
	G@4 + POI	0.0471	0.1299	0.1871	0.2588	0.0946
	G@5 + POI	0.0459	0.1294	0.1815	0.2479	0.0931
	G@6 + POI	0.0462	0.1255	0.1739	0.2401	0.0912
	POI	0.0461	0.1215	0.1711	0.2347	0.0898
Foursquare	All Tasks	**0.0686**	**0.1756**	**0.2507**	**0.3386**	**0.1288**
	G@2 + POI	0.0597	0.1621	0.2303	0.3221	0.1183
	G@3 + POI	0.0605	0.1637	0.2341	0.3229	0.1193
	G@4 + POI	0.0593	0.1605	0.2295	0.3183	0.1167
	G@5 + POI	0.0589	0.1586	0.2253	0.3088	0.1152
	G@6 + POI	0.0582	0.1566	0.2211	0.3046	0.1137
	POI	0.0563	0.1535	0.2178	0.3009	0.1113



(a) Gowalla.



(b) Foursquare.

Fig. 7. Comparison of search space reduction methods.

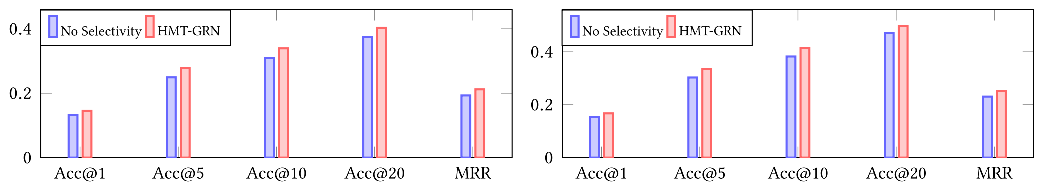
$TK = \{G@2, G@3..., G@6, POI\}$ for HBS from the graph G_{hs}, and is used in our HMT-GRN model, followed by the *G@P + POI* variants, where each of them uses only two task distributions for HBS, given $P \in \{2, 3, 4, 5, 6\}$. Further, we also include the *POI* variant which does not use HBS or any of the proposed next region task distributions, but only the sparse next POI task distribution tk^{POI} directly to compute the ranked set (i.e., $y_{t_i} = \Psi(tk^{POI})$). Notably, we observe that *All Tasks*, mostly achieves the best performance by involving all of the proposed tasks for HBS, demonstrating the necessity of multiple region-based tasks and our HBS to perform well.

Search Space Reduction. In Figure 7, we similarly compare the HBS from our HMT-GRN model with the existing search space reduction methods on the N^2 metrics:

- *HBS* is our proposed HMT-GRN model that uses HBS to traverse on all task distributions from G_{hs} to rank POIs. The search space is reduced significantly to only consider the POIs within the top-β *G@6* regions or cells in the last iteration of our HBS, instead of all POIs in L, as described in Section 4.1.
- *40 km* deactivates our HBS and computes the next POI ranked set $y_{t_i} = \Psi(tk^{POI}) \bigcap L^{40km}$ where $L^{40km} \subseteq L$ is the set of POIs within 40 km of the previous POI input $l_{t_{i-1}}$, as proposed in [4] to reduce the search space. Similarly, for the *10 km* variant, the distance threshold is instead 10 km, as separately proposed in [39] and [64].

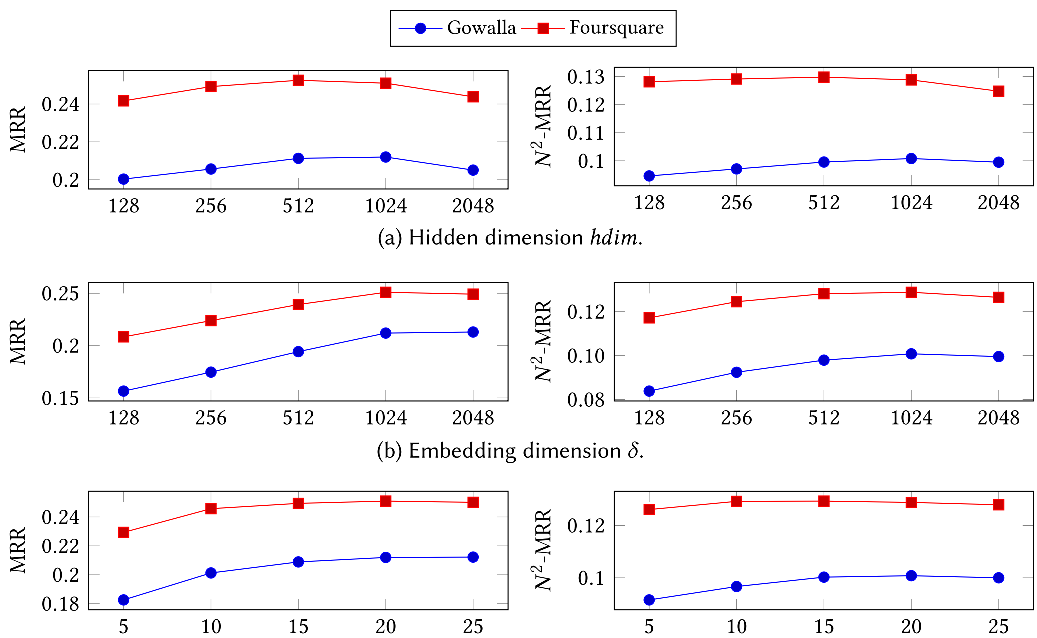
From Figure 7, we see that our *HBS* variant unanimously surpasses the *10 km* and *40 km* variants significantly, which uses simple distance thresholds. Notably, the threshold-based approaches only work for test samples if the next POI is indeed within the distance threshold and not above. This is partly why they did not perform better than our HBS which considers both near and far POIs. Also, as our HBS does not require a fixed distance threshold to reduce the search space and perform well, this eliminates the need for additional analyses of the datasets to determine the optimal thresholds, which was necessary and individually analyzed in [4], [39], and [64], to identify the 10-km and 40-km thresholds.

Selectivity Layer. In Figure 8, we evaluate the importance of our selectivity layer from Equation (12), used in our HMT-GRN model. As the layer is only used in the test samples for Acc@K and MRR that includes both visited and unvisited POIs, we omit the N^2 metrics as they would have the same results. Figure 8 shows a comparison of our HMT-GRN model and its variant where the selectivity layer is *deactivated*, with a prominent decrease of performance shown for all metrics and all datasets. As the layer is deactivated, the model does not know when to personalize or



(a) Gowalla. (b) Foursquare.

Fig. 8. Impact of the selectivity layer.



(a) Hidden dimension *hdim*.

(b) Embedding dimension δ.

(c) Number of training epochs.

Fig. 9. Sensitivity analysis of HMT-GRN.

explore, thereby always computing the joint probability for all test samples, resulting in poorer performance.

Sensitivity. In Figure 9, we study the sensitivity of our HMT-GRN model to different hyperparameters. For simplicity, we use the MRR and N^2-MRR metrics as they best describe the overall performance of the ranked set predicted as compared to the Acc@K and N^2-Acc@K metrics which focuses more on the top K for real-world applications. In Figure 9(a), we see that the model performs stably, with mostly the best performance at 1,024 *hdim* hidden size for our GRN module. In Figure 9(b), the embedding size δ used for our POI, user and geohash embeddings, similarly reaching best performance at 1,024. Lastly, in Figure 9(c), we observe that the model converges at epoch 20. Thus, we used these hyperparameters in our experiments.

JTLL Module. In Table 5, we conduct an ablation study of our designed triplet loss function in Equation (21), for the popular Gowalla dataset on the challenging N^2 metrics, where HMT-GRN is our proposed model, HMT-GRN-*JTLL-P deactivates* the second term in Equation (21), to only learn the visited relation (positive) between users and POI, with the HMT-GRN model as the

Table 5. Effectiveness of Proposed Visited and Unvisited Relations for the JTLL Module

		N^2-Acc@1	N^2-Acc@5	N^2-Acc@10	N^2-Acc@20	N^2-MRR
Gowalla	HMT-GRN-*JTLL*	**0.0543**	0.1388	**0.1970**	**0.2637**	**0.1026**
	HMT-GRN-*JTLL-N*	0.0542	**0.1392**	0.1965	0.2625	0.1022
	HMT-GRN-*JTLL-P*	0.0529	0.1362	0.1929	0.2589	0.1005
	HMT-GRN	0.0539	0.1369	0.1920	0.2579	0.1008

designated POI recommendation model for joint training, and similarly, HMT-GRN-*JTLL-N*, instead, *deactivates* the first term in Equation (21), to only learn the unvisited relation (negative) between users and POI, and the final variant, HMT-GRN-*JTLL*, learns both visited (positive) and unvisited (negative) relations, as per Equation (21). Table 5 shows that HMT-GRN-*JTLL* mostly have the best results, followed by HMT-GRN-*JTLL-N*, that only learns the unvisited relations (negative) between users and POIs. Following our motivation for sparsity alleviation with the proposed JTLL module, we believe that the learning of unvisited relation between users and their respective unvisited POIs, helps to alleviate sparsity, as there is now a relationship between a user and his unvisited POIs, and thus, HMT-GRN-*JTLL-N* performed better than the HMT-GRN model for all the N^2 metrics, as the HMT-GRN model does not learn the unvisited relations. However, overall, HMT-GRN-*JTLL* still have the most consistent results by learning both visited and unvisited relations.

5.6 Case Study

In Figure 10, we see a real-world test sample prediction made by our HMT-GRN model, which correctly predicted the next POI (airport) for a user who frequently visits airports in her historical sequence. Using our HBS to traverse the multiple different region and POI task distributions in the hierarchical spatial graph G_{hs}, the optimal search path with the highest log probability is illustrated with red boxes of increasing granularity (not drawn to scale), correctly predicting the airport in the southern region of the map. For comparison, we also use the sparse POI task distribution tk^{POI} directly for prediction (argmax), which also predicted an airport, but in the *wrong region*, specifically, the northern region of the map in Figure 10. As the user frequently visits the southern region from her past POI visits (airports and others), this regional preference was learned by our HMT-GRN model to consider only POIs in the southern region as the search space. Accordingly, the southern region's POI search space is hierarchically reduced by our HBS with decreasing region or grid cell sizes of sub-regions (e.g., cities and streets) to rank the search paths. Thus, our HMT-GRN model was able to correctly predict the next POI (airport), and from the correct region. Further, same as the variants used in this case study, the significant improvement of *All Tasks* from *POI* in Table 4 shows that there are numerous similar test samples which were predicted correctly by our model but not with the *POI* variant, indicating the necessity of regional preferences to be learned and utilized.

6 CONCLUSION

This work proposed a novel HMT-GRN model, as well as a JTLL module, to alleviate the data sparsity problem that hinders learning of the User-POI matrix for next POI recommendation. Our HMT-GRN model learns the next POI and region distributions in a multi-task setting, then performing HBS on the task distributions to reduce the search space of POIs, as well as a selectivity layer to determine personalization or exploration. Our GRN module also models both sequential dependencies and global spatio-temporal POI-POI relationships simultaneously. Lastly, our JTLL

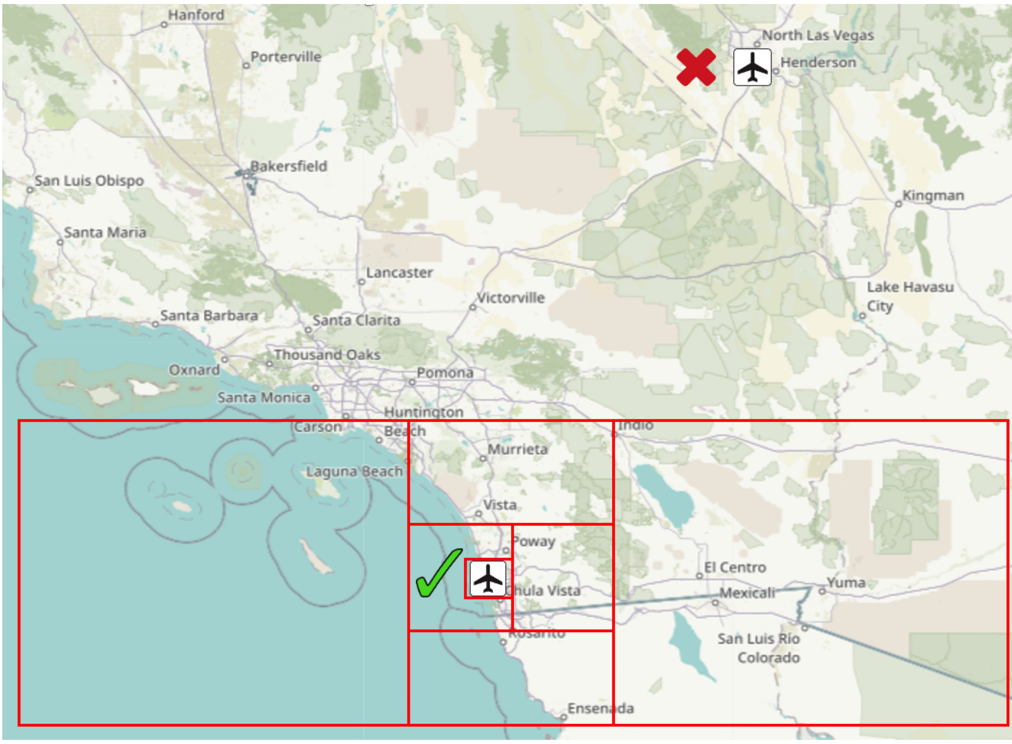


Fig. 10. Test sample prediction from the Gowalla dataset using our trained HMT-GRN model, predicts the next POI (airport) correctly, contrasting with the incorrect airport predicted by using the sparse POI task distribution tk^{POI} directly. Maps © OpenStreetMap contributors, CC BY-SA.

module learns both visited and unvisited relations of users and POIs with a designed triplet loss function. Experimental results on two popular real-world LBSN datasets with worldwide POIs demonstrate the effectiveness of the proposed approach with substantial improvements over existing works. For future work, we hope to explore temporally focused tasks to help further reduce data sparsity.

REFERENCES

[1] Buru Chang, Yonggyu Park, Donghyeon Park, Seongsoon Kim, and Jaewoo Kang. 2018. Content-aware hierarchical point-of-interest embedding model for successive POI recommendation. In *IJCAI*. 3301–3307.

[2] Ling Chen, Yuankai Ying, Dandan Lyu, Shanshan Yu, and Gencai Chen. 2021. A multi-task embedding based personalized POI recommendation method. *CCF TPCI* (2021), 1–17.

[3] Yudong Chen, Xin Wang, Miao Fan, Jizhou Huang, Shengwen Yang, and Wenwu Zhu. 2021. Curriculum meta-learning for next POI recommendation. In *KDD*. 2692–2702.

[4] Chen Cheng, Haiqin Yang, Michael R. Lyu, and Irwin King. 2013. Where you like to go next: Successive point-of-interest recommendation. In *IJCAI*. 2605–2611.

[5] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. 2011. Friendship and mobility: User movement in location-based social networks. In *KDD*. 1082–1090.

[6] Kyunghyun Cho, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*. 1724–1734.

[7] Yue Cui, Hao Sun, Yan Zhao, Hongzhi Yin, and Kai Zheng. 2021. Sequential-knowledge-aware Next POI Recommendation: A Meta-learning Approach. *TOIS* (2021).

[8] Shaojie Dai, Yanwei Yu, Hao Fan, and Junyu Dong. 2021. Personalized POI recommendation: Spatio-temporal representation learning with social tie. In *DASFAA*. Springer, 558–574.
[9] Zheng Dong, Xiangwu Meng, and Yujie Zhang. 2021. Exploiting category-level multiple characteristics for POI recommendation. *TKDE* (2021).
[10] Jeffrey L. Elman. 1990. Finding structure in time. In *Cognitive Science 14*. 179–211.
[11] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. DeepMove: Predicting human mobility with attentional recurrent networks. In *WWW*. 1459–1468.
[12] 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 *IJCAI*. 2069–2075.
[13] Shanshan Feng, Lucas Vinh Tran, Gao Cong, Lisi Chen, Jing Li, and Fan Li. 2020. HME: A hyperbolic metric embedding approach for next-POI recommendation. In *SIGIR*. 1429–1438.
[14] Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. 2019. Hyperbolic attention networks. In *ICLR*.
[15] Sajal Halder, Kwan Hui Lim, Jeffrey Chan, and Xiuzhen Zhang. 2021. Transformer-based multi-task learning for queuing time aware next POI recommendation. In *PAKDD*. Springer, 510–523.
[16] Jing He, Xin Li, and Lejian Liao. 2017. Category-aware next point-of-interest recommendation via listwise Bayesian personalized ranking. In *IJCAI*, Vol. 17. 1837–1843.
[17] Jing He, Xin Li, Lejian Liao, Dandan Song, and William K. Cheung. 2016. Inferring a personalized next point-of-interest recommendation model with latent behavior patterns. In *AAAI*. 137–143.
[18] Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short term memory. In *Neural Computation* 9(8). 1735–1780.
[19] Liwei Huang, Yutao Ma, Shibo Wang, and Yanbo Liu. 2019. An attention-based spatiotemporal LSTM network for next poi recommendation. *IEEE Transactions on Services Computing* (2019).
[20] Jinsung Jeon, Soyoung Kang, Minju Jo, Seunghyeon Cho, Noseong Park, Seonghoon Kim, and Chiyoung Song. 2021. LightMove: A lightweight next-POI recommendation for Taxicab rooftop advertising. In *CIKM*. 3857–3866.
[21] Dejiang Kong and Fei Wu. 2018. HST-LSTM: A hierarchical spatial-temporal long-short term memory network for location prediction. In *IJCAI*. 2341–2347.
[22] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[23] Hsu-Chao Lai, Yi-Shu Lu, Mu-Fan Wang, Yi-Cheng Chen, Wen-Yueh Shih, and Jiun-Long Huang. 2021. SPENT+: A category-and region-aware successive POI recommendation model. In *APNOMS*. IEEE, 230–233.
[24] Changheng Li, Yongjing Hao, Pengpeng Zhao, Fuzhen Zhuang, Yanchi Liu, and Victor S. Sheng. 2021. Tell me where to go next: Improving POI recommendation via conversation. In *DASFAA*. Springer, 211–227.
[25] Lishan Li. 2021. Hierarchical POI attention model for successive POI recommendation. In *Advances in Data Science and Information Engineering*. Springer, 169–185.
[26] Lishan Li, Ying Liu, Jianping Wu, Lin He, and Gang Ren. 2019. Multi-modal representation learning for successive POI recommendation. In *ACML*. PMLR, 441–456.
[27] Miao Li, Wenguang Zheng, Yingyuan Xiao, Ke Zhu, and Wei Huang. 2021. Exploring temporal and spatial features for next POI recommendation in LBSNs. *IEEE Access* 9 (2021), 35997–36007.
[28] Xin Li, Dongcheng Han, Jing He, Lejian Liao, and Mingzhong Wang. 2019. Next and next new POI recommendation via latent behavior pattern inference. *TOIS* 37, 4 (2019), 1–28.
[29] Yang Li, Tong Chen, Yadan Luo, Hongzhi Yin, and Zi Huang. 2021. Discovering collaborative signals for next POI recommendation with iterative Seq2Graph augmentation. In *IJCAI*. 1491–1497.
[30] Yepeng Li, Xuefeng Xian, Pengpeng Zhao, Yanchi Liu, and Victor S. Sheng. 2021. MGSAN: A multi-granularity self-attention network for next POI recommendation. In *WISE*. Springer, 193–208.
[31] Defu Lian, Yongji Wu, Yong Ge, Xing Xie, and Enhong Chen. 2020. Geography-aware sequential location recommendation. In *KDD*. 2009–2019.
[32] Dongliang Liao, Weiqing Liu, Yuan Zhong, Jing Li, and Guowei Wang. 2018. Predicting activity and location with multi-task context aware recurrent neural network. In *IJCAI*. 3435–3441.
[33] Nicholas Lim, Bryan Hooi, See-Kiong Ng, Yong Liang Goh, Renrong Weng, and Rui Tan. 2022. Hierarchical multi-task graph recurrent network for next POI recommendation. In *SIGIR*. 1133–1143.
[34] Nicholas Lim, Bryan Hooi, See-Kiong Ng, Xueou Wang, Yong Liang Goh, Renrong Weng, and Rui Tan. 2021. Origin-aware next destination recommendation with personalized preference attention. In *WSDM*. 382–390.
[35] Nicholas Lim, Bryan Hooi, See-Kiong Ng, Xueou Wang, Yong Liang Goh, Renrong Weng, and Jagannadan Varadarajan. 2020. STP-UDGAT: Spatial-temporal-preference user dimensional graph attention network for next POI recommendation. In *CIKM*. 845–854.
[36] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *AAAI*. 194–200.

[37] Xin Liu, Yongjian Yang, Yuanbo Xu, Funing Yang, Qiuyang Huang, and Hong Wang. 2022. Real-time POI recommendation via modeling long- and short-term user preferences. *Neurocomputing* 467 (2022), 454–464.
[38] Yuwen Liu, Aixiang Pei, Fan Wang, Yihong Yang, Xuyun Zhang, Hao Wang, Hongning Dai, Lianyong Qi, and Rui Ma. 2021. An attention-based category-aware GRU model for the next POI recommendation. *International Journal of Intelligent Systems* (2021).
[39] Yi-Shu Lu and Jiun-Long Huang. 2020. GLR: A graph-based latent representation model for successive POI recommendation. *Future Generation Computer Systems* 102 (2020), 230–244.
[40] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. STAN: Spatio-temporal attention network for next point-of-interest recommendation. In *WWW*. 2177–2185.
[41] Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-MRC framework for aspect based sentiment analysis. In *AAAI*, vol. 35. 13543–13551.
[42] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *NIPS* 26 (2013).
[43] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW*. 811–820.
[44] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *CVPR*. 815–823.
[45] Meihui Shi, Derong Shen, Yue Kou, Tiezheng Nie, and Ge Yu. 2021. Next point-of-interest recommendation by sequential feature mining and public preference awareness. *Journal of Intelligent & Fuzzy Systems* Preprint (2021), 1–16.
[46] Huimin Sun, Jiajie Xu, Kai Zheng, Pengpeng Zhao, Pingfu Chao, and Xiaofang Zhou. 2021. MFNP: A meta-optimized model for few-shot next POI recommendation. In *IJCAI*.
[47] Huimin Sun, Jiajie Xu, Rui Zhou, Wei Chen, Lei Zhao, and Chengfei Liu. 2021. HOPE: A hybrid deep neural model for out-of-town next POI recommendation. *WWW* (2021), 1–20.
[48] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2020. Where to go next: Modeling long- and short-term user preferences for point-of-interest recommendation. In *AAAI*. 214–221.
[49] Abraham A. Ungar. 2005. *Analytic Hyperbolic Geometry: Mathematical Foundations and Applications*. World Scientific.
[50] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengjio. 2018. Graph attention networks. In *ICLR*.
[51] Dongjing Wang, Xingliang Wang, Zhengzhe Xiang, Dongjin Yu, Shuiguang Deng, and Guandong Xu. 2021. Attentive sequential model based on graph neural network for next POI recommendation. *WWW* 24, 6 (2021), 2161–2184.
[52] Pengyang Wang, Yanjie Fu, Hui Xiong, and Xiaolin Li. 2019. Adversarial substructured representation learning for mobile user profiling. In *KDD*. 130–138.
[53] Pengyang Wang, Jiawei Zhang, Guannan Liu, Yanjie Fu, and Charu Aggarwal. 2018. Ensemble-spotting: Ranking urban vibrancy via POI embedding with multi-view spatial graphs. In *SIAM*. 351–359.
[54] Xin Wang, Xiao Liu, Li Li, Xiao Chen, Jin Liu, and Hao Wu. 2021. Time-aware user modeling with check-in time prediction for next POI recommendation. In *ICWS*. IEEE, 125–134.
[55] Yu Wang, An Liu, Junhua Fang, Jianfeng Qu, and Lei Zhao. 2021. ADQ-GNN: Next POI recommendation by fusing GNN and area division with quadtree. In *WISE*. Springer, 177–192.
[56] Zhaobo Wang, Yanmin Zhu, Haobing Liu, and Chunyang Wang. 2022. Learning graph-based disentangled representations for next POI recommendation. In *SIGIR*. 1154–1163.
[57] Yuxia Wu, Ke Li, Guoshuai Zhao, and Xueming Qian. 2019. Long- and short-term preference learning for next POI recommendation. In *CIKM*. 2301–2304.
[58] Yuxia Wu, Ke Li, Guoshuai Zhao, and Xueming Qian. 2020. Personalized long- and short-term preference learning for next POI recommendation. *TKDE* (2020).
[59] Bin Xia, Yuxuan Bai, Junjie Yin, Qi Li, and Lijie Xu. 2020. MTPR: A multi-task learning based POI recommendation considering temporal check-ins and geographical locations. *Applied Sciences* 10, 19 (2020), 6664.
[60] Min Xie, Hongzhi Yin, Hao Wang, Fanjiang Xu, Weitong Chen, and Sen Wang. 2016. Learning Graph-based POI embedding for location-based recommendation. In *CIKM*. 15–24.
[61] Dingqi Yang, Benjamin Fankhauser, Paolo Rosso, and Philippe Cudre-Mauroux. 2020. Location prediction over sparse user mobility traces using RNNs: Flashback in hidden states. In *IJCAI*. 2184–2190.
[62] Dingqi Yang, Daqing Zhang, Longbiao Chen, and Bingqing Qu. 2015. Nationtelescope: Monitoring and visualizing large-scale collective behavior in LBSNs. *Journal of Network and Computer Applications* 55, (2015), 170–180.
[63] Di Yao, Chao Zhang, Jianhui Huang, and Jingping Bi. 2017. SERM: A recurrent model for next location prediction in semantic trajectories. In *CIKM*. 2411–2414.
[64] Fuqiang Yu, Lizhen Cui, Wei Guo, Xudong Lu, Qingzhong Li, and Hua Lu. 2020. A category-aware deep model for successive POI recommendation on sparse check-in data. In *WWW*. 1264–1274.

[65] Hongyu Zang, Dongcheng Han, Xin Li, Zhifeng Wan, and Mingzhong Wang. 2021. CHA: Categorical hierarchy-based attention for next POI recommendation. *TOIS* 40, 1 (2021), 1–22.
[66] Lu Zhang, Zhu Sun, Jie Zhang, Horst Kloeden, and Felix Klanner. 2020. Modeling hierarchical category transition for next POI recommendation with uncertain check-ins. *Information Sciences* 515 (2020), 169–190.
[67] Lu Zhang, Zhu Sun, Jie Zhang, Yu Lei, Chen Li, Ziqing Wu, Horst Kloeden, and Felix Klanner. 2021. An interactive multi-task learning framework for next POI recommendation with uncertain check-ins. In *IJCAI*. 3551–3557.
[68] Mingwei Zhang, Yang Yang, Rizwan Abbas, Ke Deng, Jianxin Li, and Bin Zhang. 2021. SNPR: A serendipity-oriented next POI recommendation model. In *CIKM*. 2568–2577.
[69] Yunchao Zhang, Yanjie Fu, Pengyang Wang, Xiaolin Li, and Yu Zheng. 2019. Unifying inter-region autocorrelation and intra-region structures for spatial embedding via collective adversarial learning. In *KDD*. 1700–1708.
[70] Zhiqian Zhang, Chenliang Li, Zhiyong Wu, Aixin Sun, Dengpan Ye, and Xiangyang Luo. 2020. Next: A neural network framework for next POI recommendation. *Frontiers of Computer Science* 14, 2 (2020), 314–333.
[71] Kangzhi Zhao, Yong Zhang, Hongzhi Yin, Jin Wang, Kai Zheng, Xiaofang Zhou, and Chunxiao Xing. 2020. Discovering subsequence patterns for next POI recommendation. In *IJCAI*. 3216–3222.
[72] Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S. Sheng, and Xiaofang Zhou. 2019. Where to go next: A spatio-temporal gated network for next POI recommendation. In *AAAI*. 5877–5884.
[73] Yanyan Zhao, Jingyi Liu, Daren Zha, and Kai Liu. 2021. Hierarchical and multi-resolution preference modeling for next POI recommendation. In *IJCNN*. IEEE, 1–8.
[74] Jinwen Zhong, Can Ma, Jiang Zhou, and Weiping Wang. 2020. From when to where: A multi-task learning approach for next point-of-interest recommendation. In *WASA*. Springer, 781–793.
[75] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *KDD*. 1079–1088.

Received 1 October 2022; revised 15 June 2023; accepted 19 July 2023