Hierarchical Multi-Task Graph Recurrent Network for Next POI Recommendation

Nicholas Lim
GrabTaxi Holdings, Singapore
nic.lim@grab.com

Bryan Hooi
Institute of Data Science and School of Computing, National University of Singapore, Singapore
dcsbhk@nus.edu.sg

See-Kiong Ng
Institute of Data Science and School of Computing, National University of Singapore, Singapore
seekiong@nus.edu.sg

Yong Liang Goh
GrabTaxi Holdings, Singapore
yongliang.goh@grab.com

Renrong Weng
GrabTaxi Holdings, Singapore
renrong.weng@grab.com

Rui Tan
GrabTaxi Holdings, Singapore
rui.tan@grab.com

ABSTRACT

Learning which Point-of-Interest (POI) a user will visit next is a challenging task for personalized recommender systems due to the large search space of possible POIs in the region. A recurring problem among existing works that makes it difficult to learn and perform well is the sparsity of the User-POI matrix. In this paper, we propose our Hierarchical Multi-Task Graph Recurrent Network (HMT-GRN) approach, which alleviates the data 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 also propose a novel selectivity layer to predict if the next POI has been visited before by the user to balance between personalization and exploration. Experimental results on two real-world Location-Based Social Network (LBSN) datasets show that our model significantly outperforms baseline and the state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

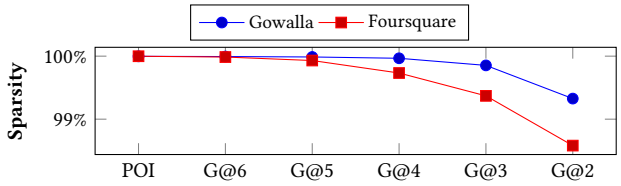
KEYWORDS

Recommender System; Graph Recurrent Network; Spatio-Temporal

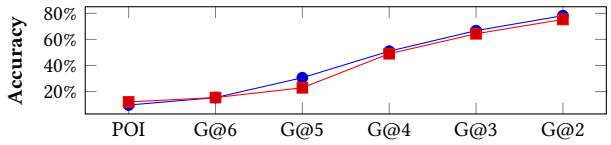
ACM Reference Format:
Nicholas Lim, Bryan Hooi, See-Kiong Ng, Yong Liang Goh, Renrong Weng, and Rui Tan. 2022. Hierarchical Multi-Task Graph Recurrent Network for Next POI Recommendation. In *Proceedings of the 45th Int'l ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3477495.3531989

SIGIR '22, July 11–15, 2022, Madrid, Spain.

ACM ISBN 978-1-4503-8732-3/22/07.
https://doi.org/10.1145/3477495.3531989



(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.

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

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 [20, 33, 60], the inclusion of textual information of user activities [52], and the learning of long and short term user preferences [11, 19, 42, 46–48]. More recently, attention-based methods [18, 30, 37, 44, 59] 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, 25, 32], or by the aggregation of past hidden states with spatio-temporal weights [50]. 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.

In this paper, we propose a Hierarchical Multi-Task Graph Recurrent Network (HMT-GRN) to learn the User-POI matrix, but also learning several User-Region matrices of different levels of granularity to better learn the sparse User-POI relationships. Shown in Fig. 1(a), we apply the publicly available geocoding system Geohash[1] (G@P) on the popular LBSN datasets of Gowalla [5] and Foursquare [51]. 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 Fig. 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 Fig. 1(b), given the same experimental setup with a traditional LSTM baseline, as commonly used in recent works for evaluation [20, 32, 42, 50, 60], 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.

With this motivation, we propose our novel 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. Lastly, 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.

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

- We propose a novel HMT-GRN[2] model to alleviate the data sparsity problem by learning both User-POI and different User-Region matrices for the next POI 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.
- 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.

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

2 RELATED WORK

Next POI Recommendation 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. By extending FPMC [38], FPMC-LR [4] was proposed to learn a personalized MC for each user. PRME-G [12] models both POIs and users in a sequential transition space and a user preference space respectively. NEXT [58] incorporates multiple context factors of temporal, geographical influence, sequential relations and auxiliary meta-data under a unified framework.

Recently, RNN-based approaches have been shown to be effective in modelling sequential dependencies for this recommendation task. ST-RNN [33] 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 [20, 60] through new gating mechanisms. Approaches which study the use of additional information of POI categories [16, 22, 26, 29, 34, 47, 48] and textual information [1, 23, 24, 52] have also been proposed, however, such categorical and textual information are not always available among real-world LBSN datasets [57] (e.g. Gowalla [5]). DeepMove [11] learns sequential transitions with a GRU, as well as applying a historical attention module. LSTPM [42] uses several LSTM-based encoders to learn long and short term user preferences with a context-aware nonlocal network architecture. [37] proposed STAN, a spatio-temporal bi-attention model that focuses on learning regularities of non-contiguous visits and non-adjacent POIs. STP-UDGAT [32] proposed the use of Graph Attention Networks (GAT) [43] to learn global POI-POI relationships from various graphs to model spatial, temporal and preference factors. Flashback [50] 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.

Overall, these 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 to alleviate this sparsity problem. Among the existing works which also seek to alleviate data sparsity [3, 7, 11, 39–41, 59], these includes a weighted loss function to accelerate learning with more informative samples [30], the leverage of additional information of POI categories [2, 9, 15, 28, 35, 53–57, 61], and the modeling of spatio-temporal relations in LSTM's existing multiplicative gates [20]. These existing works, along with the other multi-task [14, 45, 49, 62] inspired approaches, 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.

3 PRELIMINARIES

Problem Formulation. Let $L = \{l_1, l_2, ..., l_Q\}$ be a set of Q POIs and $U = \{u_1, u_2, ..., u_M\}$ be a set of M users. S is the set of visit sequences for all users where $S = \{s_{u_1}, s_{u_2}, ..., s_{u_M}\}$. Each user's sequence s_{u_m} consist of sequential POI visits $s_{u_m} = \{(l_{t_1}, loc_{t_1}, time_{t_1}), (l_{t_2}, loc_{t_2}, time_{t_2}), ..., (l_{t_i}, loc_{t_i}, time_{t_i})\}$, 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. $s^{train}_{u_m}$ and $s^{test}_{u_m}$).

Problem 1 (Next POI Recommendation). Given user u_m, $s^{train}_{u_m} = \{(l_{t_1}, loc_{t_1}, time_{t_1}), (l_{t_2}, loc_{t_2}, time_{t_2}), ..., (l_{t_{i-1}}, loc_{t_{i-1}}, time_{t_{i-1}})\}$ 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 [17] 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.

4.1 HMT-RN

Learning next POI and region distributions. As shown in Fig. 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 just 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 $l^{G@P}_{t_{i-1}}$, we use a multi-modal embedding layer E_W to map to their trainable vector representations:

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

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

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 a 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 Eq. (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 Eq. (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* $l^{G@P}_{t_i}$, where $l^{G@P}_{t_i}$ 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 $\vec{l}^{G@P}_{t_{i-1}}$ from Eq. (7), to predict the next corresponding $l^{G@P}_{t_i}$, thereby computing the task distribution $tk^{G@P} = P(l^{G@P}_{t_i} \mid l_{t_{i-1}})$:

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

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

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 Eq. (9) and (10).

Training. With each task $tk \in \{G@2, G@3, ..., G@6, POI\}$, 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 v-th training sample, and N_{train} is

the total number of training samples. 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 $\{tk_1^{G@2}, tk_2^{G@3}, ..., tk_{i-1}^{G@6}, tk_i^{POI}\}$, by computing the joint probability[3] of all tasks $P(G@2, G@3, ..., G@6, l_{t_i} \mid l_{t_{i-1}})$ 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 $\{tk_1^{G@2}, tk_2^{G@3}, ..., tk_{i-1}^{G@6}, tk_i^{POI}\}$, 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, 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 $P(G@2, G@3, ..., G@6, l_{t_i} \mid l_{t_{i-1}})$ 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 $\{tk_1^{G@2}, tk_2^{G@3}, ..., tk_{i-1}^{G@6}, tk_i^{POI}\}$ (e.g. $(tk_1^{G@2}, tk_2^{G@3})$ and $(tk_2^{G@3}, tk_3^{G@4})$), we compute:

$$tk_i^{\beta} = f\Big(\Big\{ \log(tk_{i-1,b}^{\beta}) + \log(tk_{i,j}) \mid tk_{i,j} \in \mathcal{N}(tk_{i-1,b}^{\beta}), \; b \in \{1, 2, ..., \beta\} \Big\} \Big) \tag{11}$$

where tk_{i-1}^{β} and tk_i^{β} 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 $b \in \{1, 2, ..., \beta\}$ of the previous input task distribution $tk_{i-1,b}^{\beta}$, we identify its (directed) neighbourhood $tk_{i,j} \in \mathcal{N}(tk_{i-1,b}^{\beta})$ 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. $\log(tk_{i-1,b}^{\beta}) + \log(tk_{i,j})$). 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_i^{β}, 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 Fig. 2, performed for only the input pair of $(tk_1^{G@2}, tk_2^{G@3})$ task distributions. Effectively, after performing the

[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.

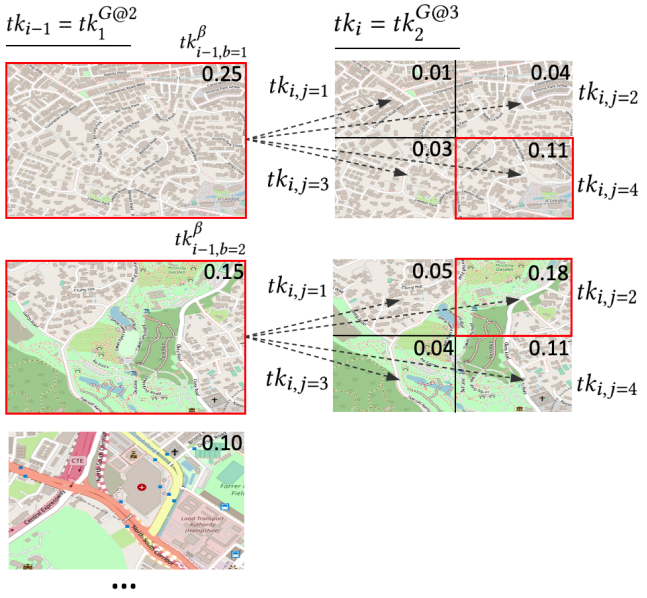


Figure 2: Hierarchical Beam Search performed with top-$\beta = 2$ (red boxes) for the sequential pair of input tasks $(tk_{i-1} = tk_1^{G@2}, tk_i = tk_2^{G@3})$. Each vertex is weighted with its respective task probability score (top right of each vertex), for the computation of partial solutions, then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

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 $(tk_{i-1}^{G@6}, tk_i^{POI})$ for the last iteration, the search space is reduced to only consider the POIs within the top-β of the $tk_{i-1}^{G@6}$ regions or cells, instead of all $|L|$ POIs, as well as computing their respective joint probability $P(G@2, G@3, ..., G@6, l_{t_i} \mid l_{t_{i-1}})$ 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 Eq. (9) and compute:

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

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 Eq. (9). Otherwise, it *explores* by performing the HBS to rank POIs based on a

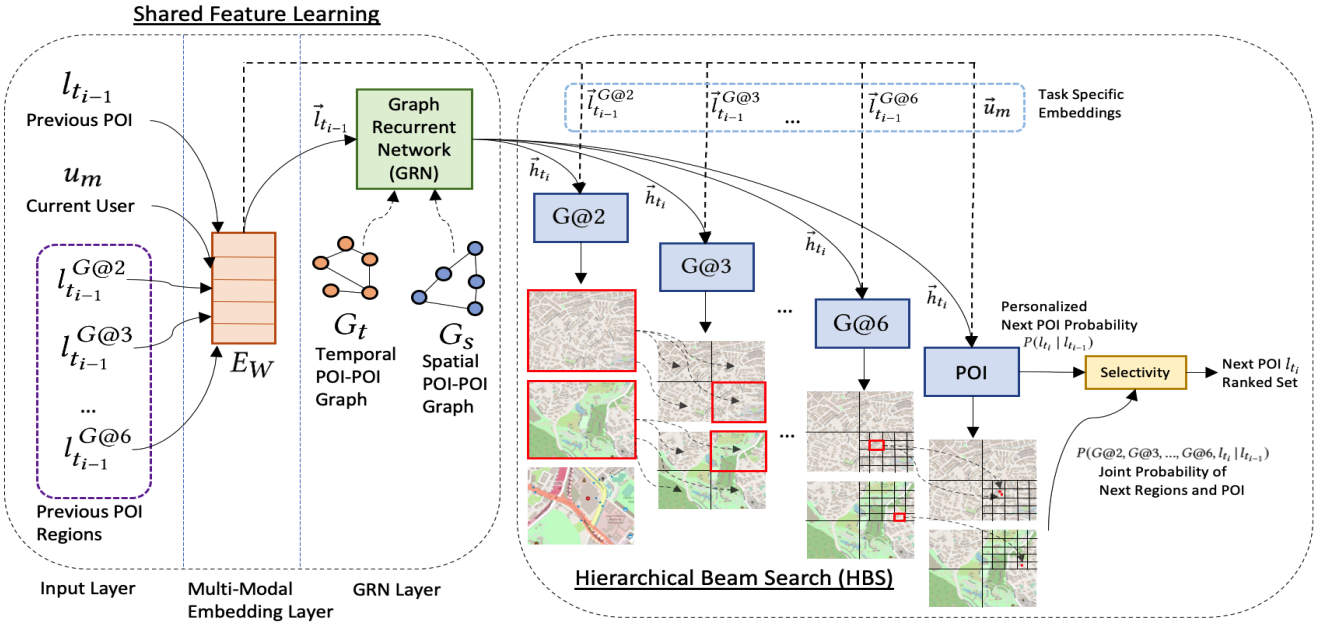


Figure 3: Illustration of the HMT-GRN model that includes shared feature learning by our GRN module, followed by the multi-task learning of next regions and POI, then performing our HBS and selectivity layer. An example of top-$\beta = 2$ (red boxes) is used by HBS to traverse the multi-task distributions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

regional context, using the joint probability of the multiple tasks $P(G@2, G@3, ..., G@6, l_{t_i} \mid l_{t_{i-1}})$.

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 [32], it does not learn global POI-POI relationships directly as compared to graph neural networks (e.g. GAT). Similarly, a drawback of GAT [43], used in [32] 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 [32] 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 [32] is defined as:

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

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

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 Eq. (7). The predicted weights $\vec{\alpha}_{l_{t_{i-1}},j}$ are then applied in Eq. (14) to compute a weighted sum of its respective neighbours, outputting the hidden representation $p_{t_i} \in \mathbb{R}^{hdim}$.

Next, different from [32] 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 Eq. (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 Eq. (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 final HMT-GRN model in Fig. 3, after replacing the LSTM module in our HMT-RN model with the GRN module.

5 EXPERIMENTS

5.1 Datasets

We evaluate our HMT-GRN model on two popular LBSN datasets of Gowalla [5] and Foursquare [51] for the next POI recommendation task, where they contain worldwide POIs in many countries. For preprocessing, similar to [32, 60], 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. 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 Fig. 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 $s_{u_m}^{train}$ based on the users' visiting frequencies.
- **MF** [21]: 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** [17] and **GRU** [6] uses various multiplicative gates to control information flow.
- **HST-LSTM** [20]: A LSTM-based model that leverages spatial and temporal intervals between sequential POIs into the LSTM existing gates. Same as [32, 60], we use their ST-LSTM variant here as the data does not have session information. **STGCN** [60] similarly models the intervals with new time and distance gates, and a new cell state for short term preference learning.
- **LSTPM** [42]: A LSTM-based model that learns long term user preferences through a nonlocal network, and short term user preferences with a geo-dilated network.

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%

- **STAN** [37]: A bi-attention model that incorporates spatio-temporal correlations of non-adjacent POIs and non-contiguous visits.
- **STP-UDGAT** [32]: A GAT-based approach that models spatio-temporal-preference factors though various POI-POI graphs in an explore-exploit architecture.
- **Flashback** [50]: 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.
- **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**: Our final variant by replacing the LSTM layer in the HMT-RN model with a GRN layer.

For both HST-LSTM and STGCN, following [31, 32, 50], 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 [20, 32, 60] 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 v-th 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. Recent works [4, 12, 13, 36] have proposed methods focused on recommending *Next New (N^2) or unvisited* POIs that were not historically visited by the user (i.e. $l_{t_i} \notin s_{u_m}^{train}$). A limitation of these works 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 [36], 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 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,

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

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
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
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
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**
Relative Improvement	2.2%	0.6%	11.8%	22.8%	6.7%	28.6%	20.1%	15.6%	12.6%	19.6%
	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
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
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
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**
Relative Improvement	-1.0%	1.8%	9.3%	19.0%	5.4%	35.8%	25.4%	23.2%	19.7%	25.9%

correctly, thereby improving the user experience and supporting them to explore new places of interest [27, 63].

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 Eq. (12) for only the N^2 metrics, and compute the ranked set $y_{t_i} = \Psi\left(P(G@2, G@3, ..., G@6, l_{t_i} \mid l_{t_{i-1}})\right)$ via joint probability directly as N_{test}^{new} only contains unvisited test samples, removing the need for the selectivity layer.

5.3 Experimental Settings

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 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 Flashback and STAN, as their recommended hyperparameters does not work as well in our experiments, we perform grid search and use the best performing models for evaluation.

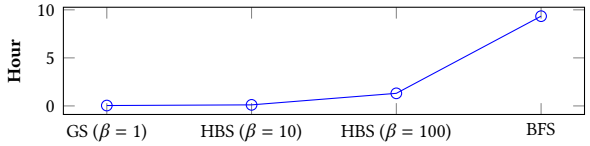
5.4 Results

We report the evaluation results of our proposed HMT-GRN model and the baselines in Table 2, where the relative improvement is computed between our HMT-GRN model and the best baseline. 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:

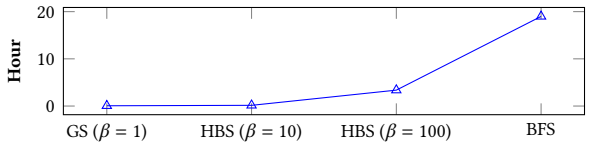
- Our HMT-GRN outperforms all the baselines significantly on all metrics for the next POI recommendation task, except for the Acc@1 metric of the Foursquare dataset by 1%.
- U-TOP and the recent works of Flashback, STP-UDGAT and LSTPM are the most competitive baselines. U-TOP was able to perform well as users would tend to visit their own frequently visited POIs [32]. 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. 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.

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 ($\beta = 1$)	0.0408	0.0762	0.0911	0.1122	0.0592
	BFS	0.0537	**0.1381**	0.1913	0.2566	**0.1008**
	HBS ($\beta = 10$)	0.0536	0.1366	0.1857	0.2354	0.0966
	HBS ($\beta = 100$)	**0.0539**	0.1369	**0.1920**	**0.2579**	**0.1008**
Foursquare	GS ($\beta = 1$)	0.0473	0.0745	0.0820	0.0872	0.0613
	BFS	0.0672	0.1750	0.2504	0.3385	0.1285
	HBS ($\beta = 10$)	0.0671	0.1742	0.2444	0.3017	0.1205
	HBS ($\beta = 100$)	**0.0686**	**0.1756**	**0.2507**	**0.3386**	**0.1288**



(a) Gowalla.



(b) Foursquare.

Figure 4: Efficiency comparison of search methods.

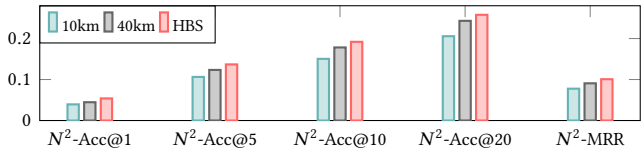
- Different from [32], which focuses on only learning from short sequences (i.e. $|s_{u_m}|$<30), we observe in Table 2 that when learning from instead longer sequences (i.e. $|s_{u_m}|$<50) in our experiments, STP-UDGAT does not always perform well as it is unable to learn the sequential dependencies 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 [31, 32], we believe that these models may not have performed well by learning from the spatial and temporal intervals between $l_{t_{i-2}}$ and $l_{t_{i-1}}$.

5.5 Efficiency

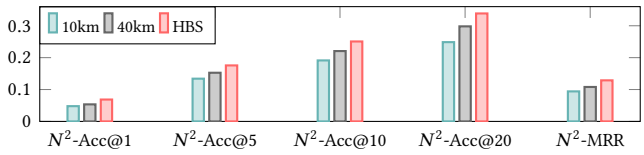
In Table 3, we compare the performance between Greedy Search (GS) (i.e. $\beta = 1$), an exhaustive method of BFS, and our HBS ($\beta = 10$ and $\beta = 100$) on the multiple task distributions in the hierarchical graph G_{hs} with the N^2 metrics, which are more challenging. Results show that our HBS 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 Fig. 4, 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.

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

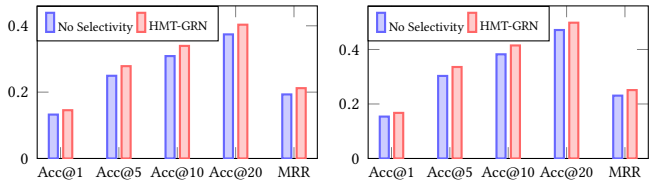


(a) Gowalla.



(b) Foursquare.

Figure 5: Comparison of search space reduction methods.

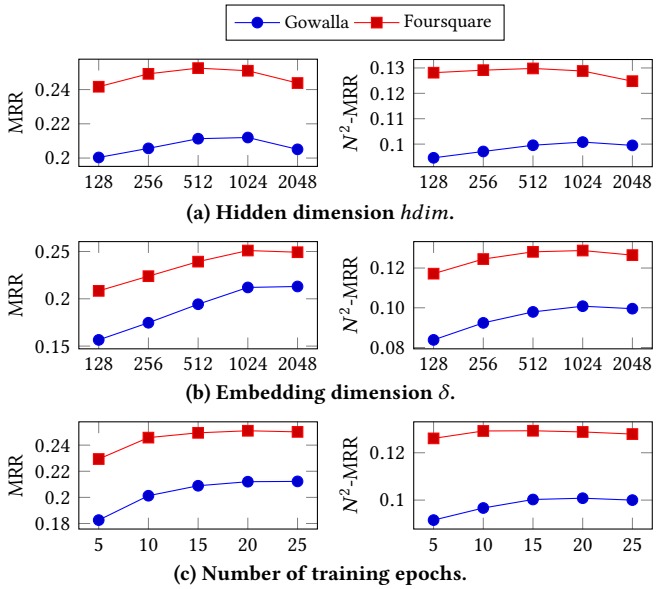


(a) Gowalla. **(b) Foursquare.**

Figure 6: Impact of the selectivity layer.

5.6 Analysis

Importance of proposed tasks. In Table 4, we similarly evaluate the performance of our HBS on the N^2 metrics, but by *deactivating* certain task distributions when performing the HBS. Specifically, *All Tasks* considers all task distributions in $TK = \{G@2, G@3..., G@6, POI\}$ for HBS from the graph G_{hs}, and is used in our HMT-GRN model, followed by the *G@P + POI* variants,



(a) Hidden dimension *hdim*.

(b) Embedding dimension δ.

(c) Number of training epochs.

Figure 7: Sensitivity analysis of HMT-GRN.

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 Fig. 5, we similarly compare our HBS 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.
- *40km* 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 40km of the previous POI input $l_{t_{i-1}}$, as proposed in [4] to reduce the search space. Similarly, for the *10km* variant, the distance threshold is instead 10km, as separately proposed in [36, 53].

From Fig. 5, we see that our *HBS* variant unanimously surpasses the *10km* and *40km* 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, 36, 53] to identify the 10km and 40km thresholds.

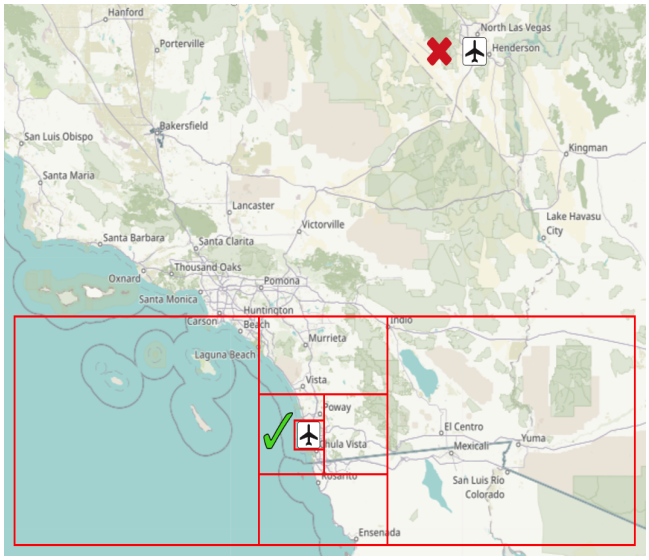


Figure 8: Test sample prediction from the Gowalla dataset using our trained HMT-GRN model, predicts the next POI (airport) correctly, contrasting with the incorrect airport predicted by using the sparse POI task distribution tk^{POI} directly. Maps © OpenStreetMap contributors, CC BY-SA.

Selectivity layer. In Fig. 6, we evaluate the importance of our selectivity layer from Eq. (12). 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. Fig. 6 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 explore, thereby always computing the joint probability for all test samples, resulting in poorer performance.

Sensitivity. In Fig. 7, 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 Fig. 7(a), we see that the model performs stably, with mostly the best performance at 1,024 *hdim* hidden size for our GRN module. In Fig. 7(b), the embedding size δ used for our POI, user and geohash embeddings, similarly reaching best performance at 1,024. Lastly, in Fig. 7(c), we observe that the model converges at epoch 20. Thus, we used these hyperparameters in our experiments.

5.7 Case Study

In Fig. 8, 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 Fig. 8. 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 to alleviate the data sparsity problem by learning 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. 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.

ACKNOWLEDGMENT

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.

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] 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.
[15] 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.
[16] 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.
[17] Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long Short Term Memory. In *Neural Computation 9(8)*. 1735–1780.
[18] 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).
[19] Jinsung Jeon, Soyoung Kang, Minju Jo, Seunghyeon Cho, Noseong Park, Seonghoon Kim, and Chiyoung Song. 2021. LightMove: A Lightweight Next-POI Recommendation forTaxicab Rooftop Advertising. In *CIKM*. 3857–3866.
[20] Dejiang Kong and Fei Wu. 2018. HST-LSTM: A Hierarchical Spatial-Temporal Long-Short Term Memory Network for Location Prediction. In *IJCAI*. 2341–2347.
[21] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.
[22] 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.
[23] 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.
[24] Lishan Li. 2021. Hierarchical POI Attention Model for Successive POI Recommendation. In *Advances in Data Science and Information Engineering*. Springer, 169–185.
[25] 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.
[26] 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.
[27] 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.
[28] 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.
[29] 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.
[30] Defu Lian, Yongji Wu, Yong Ge, Xing Xie, and Enhong Chen. 2020. Geography-aware sequential location recommendation. In *KDD*. 2009–2019.
[31] 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.
[32] 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.
[33] 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.
[34] 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.
[35] 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).
[36] 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.

[37] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. STAN: Spatio-Temporal Attention Network for next Point-of-Interest Recommendation. In *WWW*. 2177–2185.

[38] Steffen Rendle, Christoph Freudenthaler, , and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-Basket Recommendation. In *WWW*. 811–820.

[39] 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.

[40] 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*.

[41] 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.

[42] 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.

[43] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengjio. 2018. Graph Attention Networks. In *ICLR*.

[44] 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.

[45] 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.

[46] 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.

[47] Yuxia Wu, Ke Li, Guoshuai Zhao, and Xueming Qian. 2019. Long-and short-term preference learning for next POI recommendation. In *CIKM*. 2301–2304.

[48] Yuxia Wu, Ke Li, Guoshuai Zhao, and QIAN Xueming. 2020. Personalized long-and short-term preference learning for next POI recommendation. *TKDE* (2020).

[49] 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.

[50] 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.

[51] Dingqi Yang, Daqing Zhang, Longbiao Chen, and Bingqing Quc. 2015. NationTelescope: Monitoring and Visualizing Large-Scale Collective Behavior in LBSNs. *Journal of Network and Computer Applications* 0, 0 (2015), 1–16.

[52] 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.

[53] 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.

[54] 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.

[55] 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.

[56] 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.

[57] 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.

[58] 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.

[59] 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.

[60] 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.

[61] 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.

[62] 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.

[63] 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.