Activity-Aware Human Mobility Prediction With Hierarchical Graph Attention Recurrent Network

Yihong Tang*, Junlin He, and Zhan Zhao*

Abstract—Human mobility prediction is a fundamental task essential for various applications in urban planning, location-based services and intelligent transportation systems. Existing methods often ignore activity information crucial for reasoning human preferences and routines, or adopt a simplified representation of the dependencies between time, activities and locations. To address these issues, we present Hierarchical Graph Attention Recurrent Network (HGARN) for human mobility prediction. Specifically, we construct a hierarchical graph based on past mobility records and employ a Hierarchical Graph Attention Module to capture complex time-activity-location dependencies. This way, HGARN can learn representations with rich human travel semantics to model user preferences at the global level. We also propose a model-agnostic history-enhanced confidence (MAHEC) label to incorporate each user’s individual-level preferences. Finally, we introduce a Temporal Module, which employs recurrent structures to jointly predict users’ next activities and their associated locations, with the former used as an auxiliary task to enhance the latter prediction. For model evaluation, we test the performance of HGARN against existing state-of-the-art methods in both the recurring (i.e., returning to a previously visited location) and explorative (i.e., visiting a new location) settings. Overall, HGARN outperforms other baselines significantly in all settings based on two real-world human mobility data benchmarks. These findings confirm the important role that human activities play in determining mobility decisions, illustrating the need to develop activity-aware intelligent transportation systems. Source codes of this study are available at https://github.com/YihongT/HGARN

Index Terms—Human mobility, next location prediction, location-based services, graph neural networks, activity-based modeling.

I. INTRODUCTION

HUMAN mobility is critical for various downstream applications such as urban planning, location-based services and intelligent transportation systems. The ability to model and accurately predict future human mobility can inform important public policy decisions for managing traffic congestion, promoting social integration, and maximizing productivity [1]. Central to human mobility modeling is the problem of next location prediction, i.e., predicting where an individual is going next, which has received great attention in research and practices. On the one hand, the increasing prevalence of mobile devices and popularity of location-based social networks (LBSNs) provide unprecedented data sources for mining individual-level mobility traces and preferences [2]. On the other hand, the advancement of AI and machine learning offers a plethora of analytical tools for modeling human mobility. These innovations greatly enhance human mobility modeling in the past decade, especially for the next location prediction.

While traditional approaches to human mobility analysis typically used Markov Chains (MC) [3], [4], [5] to model transition patterns over location sequences, recurrent neural networks (RNN) [6] demonstrated superior predictive performance, including pioneering works that employed recurrent structures to model temporal periodicity [7] and spatial regularity [8]. Due to the great success of the TRANSFORMER architecture [9], the attention mechanism has also been adopted to model sequences and obtain competitive prediction results [10], [11]. In recent years, graph-based approaches leveraged graph representation learning [12], [13] and graph neural networks (GNNs) [14] to model user preferences [15] and spatial-temporal relationships [16] between locations, obtaining rich representations [17], [18] to improve the performance of next location prediction. However, most existing studies focus on predicting human mobility based on individual location sequences, overlooking the integral interplay between activity participation and location visitation behaviors. Classic travel behavior theories suggest that an individual’s travel decisions are determined by the need to participate in activities taking place at different locations and scheduled at different time of day [19]. Given that human activity data is becoming increasingly accessible and most location visits can be characterized by only a small number of activity categories, incorporating these activity dynamics into human mobility modeling offers a behaviorally insightful and computationally efficient approach.

Figure 1 shows several human mobility trajectories reflecting time-activity-location dependencies. For example, when the time is approaching noon, one user may dine at a nearby
restaurant, and another may go to the movie theater for a specific starting time, which illustrates that activities are usually scheduled according to the time of day. People typically make location decisions based on the intended activities, and thus considering activity information can lead to better predictability of human mobility. However, few studies have considered activity information (e.g., location categories) for next location prediction. Notably, CSLSL, proposed by Huang et al. [20], adopts an RNN-based structure [21] to model human travel decision logic, where the time, activity, and location are predicted sequentially. However, the design of CSLSL oversimplifies the time-activity-location dependencies. Given data sparsity and behavioral uncertainties, the time prediction tends to be more challenging [22], which may compromise the prediction of activities and locations.

Based on the above observations, a suitable human next location predictor should: (1) **take into account human activities when predicting next locations**, leveraging the predicted future activity information to enhance location prediction, and (2) **effectively manage intricate time-activity-location dependencies** while circumventing the difficulty in time prediction under data sparsity and uncertain human behaviors. In this study, we propose **Hierarchical Graph Attention Recurrent Network (HGARN)** for next location prediction. Specifically, we construct a hierarchical graph based on past mobility records and employ a **Hierarchical Graph Attention Module** to capture complex time-activity-location dependencies. This way, HGARN can learn representations with rich human travel semantics to model user preferences at the global level. We also propose a model-agnostic history-enhanced confidence (MAHEC) label to incorporate each user’s individual-level preferences. Finally, we introduce a **Temporal Module**, which employs recurrent structures to jointly predict users’ next activities and their associated locations, with the former used as an auxiliary task to enhance the latter prediction. Through such design, HGARN can leverage the learned time-activity-location dependencies to benefit both global- and individual-level human mobility modeling, and use predicted next activity distribution to facilitate next location prediction. In summary, this study makes the following contributions:

- We propose a **Hierarchical Graph** that incorporates human activity information to represent the activity-activity, activity-location, location-location dependencies. To our best knowledge, among the few methods considering activity information for next location prediction, this is the first work to model the dependencies of time, activities, and locations using a Hierarchical Graph.
- We design a **activity-aware Hierarchical Graph Attention Recurrent Network (HGARN)**, which contains a **hierarchical graph attention module** to model dependencies between time, activities, and locations, and a **temporal module** to incorporate the hierarchical graph representations into sequence modeling, leveraging next activity prediction to boost next location prediction.
- We introduce a simple yet effective model-agnostic history-enhanced confidence (MAHEC) label to guide the model learning of each user’s individual-level preferences, allowing the model to focus more on relevant locations in their history trajectories when predicting their next locations.
- Through extensive experiments, we evaluate the prediction performance of HGARN against existing SOTAs in both the **recurring**, and **explorative** settings, using two real-world LBSN check-in datasets. Our work is the first to separately evaluate next location prediction performance in these settings. The results show that HGARN can significantly outperform all baselines in all experimental settings.

## II. RELATED WORK

### A. Next Location Prediction

Next location prediction is essentially about sequence modeling since the next location visit is usually dependent on the previous one [23], [24]. Traditional MC-based methods often incorporate other techniques, such as matrix factorization [4] and activity-based modeling [5], for enhanced prediction performance. However, they are limited in capturing long-term dependencies or predicting explorative human mobility.

RNN-based models regard the next location prediction problem as a sequence-to-sequence task and have shown superior performance. STRNN [25] is a pioneering work that integrates spatial-temporal characteristics between consecutive human visits into RNNs, laying the groundwork for subsequent studies. Building on this, STGN [26] introduces spatial and temporal gates to Long Short-Term Memory (LSTM) networks to better capture users’ interests, while FLASHBACK [8] leverages spatial and temporal intervals to aggregate past RNN hidden states for improved predictions. Additionally, LSTPM [27] employs a non-local network and a geo-dilated LSTM to model both long- and short-term user preferences. Attention mechanisms are also utilized to enhance model performance. DEEPMOVE [7] combines attention mechanisms with RNN modules to effectively capture users’ long- and short-term preferences. Similarly, ARNN [11] uses a knowledge graph to identify related neighboring locations and employs attentional RNNS to model the sequential regularity of check-ins. Furthermore, STAN [10] extracts relative spatial-temporal information between both consecutive and non-consecutive locations through a spatio-temporal attention network. These
approaches collectively highlight the importance of integrating spatial-temporal dynamics and attention mechanisms to improve the accuracy of human mobility predictions. In addition, some efforts incorporate contextual information [28] such as geographical information [29], dynamic-static [30], text content about locations [31] into sequence modeling.

Graph-based models have become a cornerstone in the field of human mobility prediction due to their ability to effectively capture complex relationships and dependencies. For instance, LBSN2Vec [12] employs random walks on a hypergraph to learn embeddings, enhancing predictions for both locations and friendships. Similarly, STP-UDGAT [15] leverages graph attention networks (GAT) to discern location relationships from both local and global perspectives, utilizing spatial, temporal, and preference graphs. To address the data sparsity issue, HMT-GRN [32] constructs multiple user-region matrices at varying granularities to improve prediction accuracy. GCDAN [16] integrates graph convolutional networks (GCN) to capture high-order sequential dependencies in a dual attention framework to mitigate sparsity. GRAPH-FLASHBACK [18] innovatively combines knowledge graph embeddings with GCN to refine graph representations, further integrating with the FLASHBACK model for enhanced prediction capabilities. Moreover, to provide more activity awareness in human mobility modeling, generative adversarial imitation learning (GAIL) has been adapted to simulate activity trajectories [33]. Other significant contributions include the use of weighted category hierarchy in [34] to model activities, CATDM [35] which incorporates activities and spatial distances to reduce search space, and CSLSL [20] that introduces an RNN-based causal structure to capture the logic behind human travel decisions. However, most existing methods overlook the activity and cannot effectively model the time-activity-location dependencies, which are essential for predicting and understanding human mobility.

B. Hierarchical Graph Neural Network

The Hierarchical Graph Neural Network (HGNN) is a family of GNN models that gain significant attention in recent years due to their ability to capture complex dependencies in data using hierarchical structures. HGNN has been applied to various urban applications such as parking availability prediction [36], air quality forecasting [37], road network representation learning [38], real estate appraisal [39], and socioeconomic indicator estimation [40]. However, each model has its own structure design and graph construction mechanisms based on their specific application scenarios, resulting in fundamentally different architectures.

One relevant HGNN-based approach for next location prediction is HMT-GRN [32], which partitions the spatial map and performs a Hierarchical Beam Search (HBS) on different regions and POI distributions to hierarchically reduce the search space for accurate predictions. Unlike previous works, our proposed HGARN is an activity-based model designed for next location prediction. It constructs a hierarchical graph based on human activities and leverages graph attention mechanisms to capture complex time-activity-location dependencies. This activity-based design is unique and distinguishes HGARN from other HGNN-based models.

III. PRELIMINARIES

We use notations $U = \{u^1\}_{i=1}^{U}$, $L = \{l^1\}_{i=1}^{L}$, $C = \{c^1\}_{i=1}^{C}$, and $T = \{t^1\}_{i=1}^{T}$ to denote the sets of users, locations, activities and time series, respectively. For a specific user $u \in U$, we denote their sets of locations, activities and time series in a temporal order as $L_u = \{l^1_u\}_{i=1}^{L_u}$, $C_u = \{c^1_u\}_{i=1}^{C_u}$, and $T_u = \{t^1_u\}_{i=1}^{T_u}$, respectively.

Definition 1 (Mobility Record): Let us use $r$ to denote a single human mobility record. Each mobility record comprises a user $u \in U$, an activity $c^i_u \in C_u$, a location $l^i_u \in L_u$ and the visit time $t^i_u \in T_u$. The $i$th record of user $u$ is thus represented by a tuple $r^i_u = (u, c^i_u, l^i_u, t^i_u)$.

Definition 2 (Trajectory): A trajectory is a sequence of mobility records for a user $u$, denoted by $R_u = \{r^1_u\}_{i=1}^{R_u}$. Each trajectory $R_u$ can be divided into a activity trajectory $R^C_u = \{c^1_u\}_{i=1}^{R_u}$, location trajectory $R^L_u = \{l^1_u\}_{i=1}^{R_u}$, and time trajectory $R^T_u = \{t^1_u\}_{i=1}^{R_u}$.

Problem 1 (Next Location Prediction): Given a user $u$’s observed trajectory $R_u$ as input, we consider $u$’s next record as its future state. The human mobility prediction task $T$ maps $u$’s past trajectory $R_u$ to $u$’s next location $l^{|R_u|+1}_u$ in the future. The problem can be expressed as follows:

$$R_u \xrightarrow{T(\cdot;\theta)} l^{|R_u|+1}_u,$$

where $\theta$ is the parameters of mapping $T$.

IV. METHODOLOGY

HGARN’s workflow is demonstrated in Figure 2. The raw data is first encoded in the embedding module and then input to the hierarchical graph attention module to model multi-dependencies. Finally, the user’s personalized embeddings are fused with the learned hierarchical graph representations and input to the temporal module to make predictions. We will elaborate on the details of HGARN in the following sections.

A. Embedding Module

The embedding module aims to learn low-dimensional embedding vectors to represent each user, activity, location and time interval. It is worth noting that the first three elements are all naturally discrete, and the continuous time can be discretized into time intervals as well, making it easier to learn embedding vectors. In this work, the time is represented by two discrete variables, one for the hour of day $h \in H^h$ and the other for the day of week $w \in W^w$. Note that all $t \in T$ can be written in the form of $t = (h, w)$.

To illustrate how we generate the trainable embedding vectors for next location prediction, we first represent users, activities, locations, and time intervals as one-hot encoded vectors. Specifically, we define the one-hot vectors as follows: $v_u \in \mathbb{R}^{1 \times |U|}$ for users, $v_l \in \mathbb{R}^{1 \times |L|}$ for locations, $v_c \in \mathbb{R}^{1 \times |C|}$ for activities, and $v_t \in \mathbb{R}^{1 \times |T|}$ for time intervals. In each one-hot vector, only one element is set to 1, with all other elements being 0. This single 1 uniquely identifies
the corresponding entity (e.g., a specific user or location). To convert these high-dimensional discrete one-hot vectors into low-dimensional continuous trainable embeddings for actual use, we apply the following transformations:

\[ e_u = v_u W_u; \quad e_l = v_l W_l; \quad e_c = v_c W_c; \quad e_t = v_t W_t, \]  

where \( e_u \in \mathbb{R}^{d_u \times d'} \), \( e_l \in \mathbb{R}^{d_l \times d'} \), \( e_c \in \mathbb{R}^{d_c \times d'} \), and \( e_t \in \mathbb{R}^{d_t \times d'} \) represent the resulting embedding vectors for users, locations, activities, and time intervals, respectively. These embeddings are trainable and allow us to effectively capture latent information about each entity. \( W_u \in \mathbb{R}^{[u] \times d_u} \), \( W_l \in \mathbb{R}^{[l] \times d_l} \), \( W_c \in \mathbb{R}^{[c] \times d_c} \), and \( W_t \in \mathbb{R}^{[t] \times d_t} \) are the corresponding transformation matrices, which can be learned jointly with other model parameters through back propagation. \( d_u \), \( d_l \), and \( d_t \) are hyperparameters that denote the dimensions of the embedding vectors for users, activities/locations, and time intervals, respectively. After the transformation, the resulting embedding vectors \( e_u \), \( e_l \), \( e_c \), and \( e_t \) are typically “squeezed” to remove the extra dimension, resulting in vectors \( e_u \in \mathbb{R}^{d'} \), \( e_l \in \mathbb{R}^{d'} \), \( e_c \in \mathbb{R}^{d'} \), and \( e_t \in \mathbb{R}^{d'} \), respectively. To illustrate the embedding process, we provide an example to learn user embeddings in Appendix C.

### B. Hierarchical Graph Attention Module

To model the complex dependencies between activities and locations, the hierarchical graph attention module is designed with two parts: hierarchical graph construction and hierarchical graph attention networks for multi-dependencies modeling.

1) **Hierarchical Graph Construction:** The urban spatial network can be represented as a graph. GNNs provide an effective way to learn graph representation and model node-to-node dependencies [41], [42]. In this study, we model the location-location, location-activity, and activity-activity dependencies using a hierarchical graph, which consists of three layers: the location layer, localized-activity layer, and activity layer. Here, the localized-activity layer is used to suppress noise aggregated from the location layer.

We formally describe the hierarchical graph with notation \( G = (\mathcal{V}, \mathcal{E}) \), where \( \mathcal{V} = \mathcal{V}^L \cup \mathcal{V}^C \cup \mathcal{V}^C \) and \( \mathcal{E} = \{A^L, A^C, A^{LC}, A^{CC}\} \). Specifically, \( \mathcal{V}^L \) and \( \mathcal{V}^C \) represent the sets of location nodes and activity nodes, respectively. \( \mathcal{V}^C \) indicates the set of localized-activity nodes, which is an identical copy of the activity node set. \( \mathcal{E} \) comprises four adjacency matrices denoting the dependencies between two location nodes (\( A^L \)), two activity nodes (\( A^C \)), a location node and a localized-activity node (\( A^{LC} \)), and an activity node and a localized-activity node (\( A^{CC} \)).

The location adjacency matrix \( A^L \) is defined based on the geographical distance between locations. Specifically, two locations \( i \) and \( j \) are linked with an edge if their haversine distance is within a threshold. \( A^L \in \mathbb{R}^{[L] \times [L]} \) is defined as:

\[ A^L_{ij} = \begin{cases} 1, & \text{Haversine}(i, j) < D^b \\ 0, & \text{otherwise} \end{cases} \]

where \( D^b \) is a hyperparameter denoting the distance threshold.

The construction of \( A^C \) is based on observed trajectories. Intuitively, the dependencies between activities can be measured by the frequency of co-occurrence in the same time interval. However, if we directly consider the activity co-occurrence frequency based on all trajectories from all users, it may lead to unrelated activities being linked (e.g., check-in at subway stations and gyms both often occur in the evening), due to the difference in user preferences. Instead, we propose to learn the inter-activity dependencies based on activity co-occurrence within individual-level trajectory sets. Therefore, we can traverse each user’s trajectories \( R_u \) and...
count the co-occurrence frequency \( M_{e,c} \) between each activity pair \((e^i, e^j)\). Based on activity co-occurrence frequencies, \( A^C \in \mathbb{R}^{C \times C} \) is defined as:

\[
A^C_{e,c} = \begin{cases} 
1, & \text{if } M^C_{e,c} > \text{mean} \left( M^C \right) \\
0, & \text{otherwise},
\end{cases}
\]

(4)

The adjacency matrix \( A^L \) defines the dependencies between location nodes and localized-activity nodes. Each node of \( V^L \) is linked to only one node of \( V^C \), representing the corresponding activity category at that location. In contrast, each node of \( V^C \) may be linked to multiple nodes of \( V^L \), as several locations can share the same activity type. Formally, we define \( A^C \in \mathbb{R}^{C \times C} \) based on the affiliations of locations and activities, with each row corresponding to a location and each column an activity. Additionally, we construct the adjacency matrix \( A^LC \in \mathbb{R}^{(L+|C|) \times (L+|C|)} \) based on \( A^L \) in the following block matrix form:

\[
A^LC = \begin{bmatrix}
O_L & A^L_C \\
(A^L_C)^\top & O_C
\end{bmatrix},
\]

(5)

where \((A^L_C)^\top\) is the transpose of \( A^L_C \), \( O_L \in \mathbb{R}^{L \times L} \) and \( O_C \in \mathbb{R}^{C \times C} \) are two zero matrices.

The localized-activity layer is designed to suppress noise from the location layer aggregated to the activity layer. Therefore, each node in the localized-activity layer is assumed to be connected to the node in the activity layer representing the same activity type. Similarly, we have \( A^{CC} \in \mathbb{R}^{2C \times 2|C|} \):

\[
A^{CC} = \begin{bmatrix}
O_C & I_C \\
I_C & O_C
\end{bmatrix},
\]

(6)

where \( I_C \in \mathbb{R}^{C \times C} \) is an identity matrix.

2) Hierarchical Graph Attention Networks: GNNS have proven to be powerful in capturing dependencies on graphs. Both inter- and intra-layer nodes on the hierarchical graph have different dependencies on each other. Since the importance of locations within a certain distance are different to each other, we use GAT to model location-location dependencies:

\[
H^L = \text{GAT}_L \left( e_L, A^L \right),
\]

(7)

where \( \text{GAT}(\cdot) \) is an implementation of the original model [42]. \( H^L \in \mathbb{R}^{L \times d^L} \) is the learned representations as the output of the \( \text{GAT}_L \).

To integrate location information into representation learning of activities and suppress the noise aggregated to the nodes of activity layer, we introduce the localized-activity layer to pre-aggregate location embeddings. We first concatenate \( e_L \) and \( e_C \) to obtain the fused embedding matrix \( e_{LC} \in \mathbb{R}^{(L+|C|) \times d} \). Then the localized-activity process is implemented as:

\[
H^{LC} = \text{GAT}_{LC} \left( e_{LC}, A^{LC} \right),
\]

(8)

where \( H^{LC} \in \mathbb{R}^{(L+|C|) \times d} \) is the output of \( \text{GAT}_{LC} \). To obtain the pre-aggregated representation matrix \( H^{LC} \in \mathbb{R}^{C \times d} \), we remove the first \(|L|\) rows from \( H^{LC} \).

The learned representation \( H^C \) is then concatenated with activity embeddings \( e_C \) as the \( \text{GAT}_C \)'s input \( e_{CC} = [e_C \ H^{LC}]^\top \). It is worth noting that for all nodes in the activity layer, we can simultaneously aggregate information from neighbors in the localized-activity layer and neighbors in the same layer by simply modifying the matrix \( A^{CC} \) to:

\[
A^{CC}_{\text{new}} = \begin{bmatrix}
A^C & I_C \\
I_C & O_C
\end{bmatrix}.
\]

(9)

We employ a similar strategy to update the representation of the nodes in the activity layer:

\[
H^C = \text{GAT}_C \left( e_{CC}, A^{CC}_{\text{new}} \right),
\]

(10)

where \( H^C \in \mathbb{R}^{2|C| \times d} \) is the learned representation from \( \text{GAT}_C \). To obtain the updated activity node representation \( H^C \in \mathbb{R}^{C \times d} \), we remove the last \(|C|\) rows from \( H^C \).

The learned attention weights reflect the relative importance of each node to its neighbors, thereby demonstrating their influence within the network, and we provided more discussions in Section V-D.

C. Temporal Module

To model sequential dependencies of human mobility, the temporal module is designed to encode a user’s trajectory embeddings (from the embedding module) with learned graph representations (from the hierarchical graph attention module) through a recurrent structure. Given a user \( u \)'s trajectory, the learned representation for the \( i \)th activity or location can be denoted as:

\[
X^L_{ui} = e_u || H^C_{e^i_u},
\]

(11)

\[
X^L_{ui} = e_u || H^C_{e^i_u} \parallel H^L_{h^i_u},
\]

(12)

where \( || \) is the concatenation operation, \( H^C_{e^i_u} \) and \( H^L_{h^i_u} \) are the learned graph representations of the activity node \( e^i_u \) and location node \( h^i_u \), respectively.

Specifically, LSTM is used to encode both user activity trajectories and location trajectories. Therefore, the hidden state updating process at \( t \)th iteration is implemented as:

\[
e^c_{u}^{i,t}, h^c_{u}^{i,t} = \text{LSTM}(X^{C,i}_{u}, e^c_{u}^{i,t-1}, h^c_{u}^{i,t-1}),
\]

(13)

\[
e^l_{u}^{i,t}, h^l_{u}^{i,t} = \text{LSTM}(X^{L,i}_{u}, e^l_{u}^{i,t-1}, h^l_{u}^{i,t-1}),
\]

(14)

where \( h^c_{u}^{i,t} \) and \( h^l_{u}^{i,t} \) are the \( t \)th hidden states for user \( u \)'s activity and location sequences, respectively, \( e^c_{u}^{i,t} \) and \( e^l_{u}^{i,t} \) are the corresponding cell states.

After obtaining final hidden states of activity and location encoder as \( h^c_{u} \) and \( h^l_{u} \), we implement our activity decoder as a multi-layer perceptron (MLP) to get the next activity logits \( h^C_{u} \in \mathbb{R}^{|C|} \):

\[
\tilde{h}^C_{u} = \text{MLP}_C \left( h^C_{u} \right),
\]

(15)

where the logits usually refer to the raw, unnormalized vectors output by the model, which are used as inputs to a Softmax function to obtain a predicted probability distribution.

Finally, we combine the obtained activity logits with the encoded representation \( h^L_{u} \) using a residual connection [43]. This results in the final location logits \( \tilde{h}^L_{u} \in \mathbb{R}^{|L|} \):

\[
\tilde{h}^L_{u} = \lambda_r \cdot \text{MLP}_L \left( h^L_{u} \right)
\]
where \( \lambda_r \) is a factor that trades off different features.

**D. Model-Agnostic History-Enhanced Confidence Label**

Existing models [7], [16] often try to learn temporal and periodic mobility patterns from collective trajectories from all users, overlooking the heterogeneity in individual preferences. Travel behavior theories suggest that individuals are more likely to revisit locations they have visited before, due to familiarity and established activity patterns [44]. To address this issue, we introduce a modified soft labeling approach known as the model-agnostic history-enhanced confidence (MAHeC) label. Unlike traditional labels which provide hard classifications, soft labels offer a probability distribution over classes, capturing uncertainty and allowing the model to learn more nuanced patterns [45]. The MAHeC label incorporates historical user trajectory information, enhancing the model’s ability to focus on relevant trajectories by assigning higher confidence to visited locations.

Specifically, for each location \( l_i \in L \), we differentiate its confidence for a user \( u \)’s next location as follows:

\[
\text{MAHeC}^u_{L} = \begin{cases} 
  w^c, & \text{if } l_i = l_u^{R_u + 1} \\
  (1 - w^c) \frac{f^u|l_i|}{|R_u|}, & \text{if } l_i \in R_u \text{ and } l_i \neq l_u^{R_u + 1} \\
  0, & \text{otherwise}
\end{cases}
\]

(17)

where \( w^c \in [0, 1] \) is a hyperparameter that indicates the confidence of \( u \)’s ground truth label, and \( f^u|l_i| \) denotes \( u \)’s frequency of visits to \( l_i \) in the observed trajectory \( R_u \). Then the MAHeC label for \( u \)’s next location is defined as:

\[
\text{MAHeC}^u_{L} = (\text{MAHeC}^u_{L})_{i=1}^{L} \in \mathbb{R}^{[L]},
\]

(18)

where each element in \( \text{MAHeC}^u_{L} \) represents the confidence that the user \( u \) decides to choose as their next location based on their past visits. Similarly, we conduct the same operations for user activity trajectories to obtain \( \text{MAHeC}^u_{A} \).

**E. Model Optimization**

Since next location prediction is a classification problem, we transform \( \hat{h}_u^L \) from Eq. (16) to the probability distribution of locations \( \text{Softmax}(\hat{h}_u^L) \) through \( \hat{h}_u^L \). Given \( \text{MAHeC}^u_{L} \) and \( \hat{h}_u^L \), we can compute the cross-entropy loss for next location prediction, denoted as \( \mathcal{L}_L \):

\[
\mathcal{L}_L = -\frac{1}{|U|} \sum_{u \in U} \sum_{i=1}^{[L]} \text{MAHeC}^u_{L} \cdot \log(\hat{h}_u^L(i)),
\]

(19)

where \( \hat{h}_u^L(i) \) is the \( i \)th element of \( \hat{h}_u^L \). Similarly, we compute the next activity loss \( \mathcal{L}_A \) based on the same operations. Finally, we can train our HGARN with a overall loss function:

\[
\mathcal{L} = \lambda_L \cdot \mathcal{L}_L + \lambda_C \cdot \mathcal{L}_C
\]

(20)

where \( \lambda_L \) and \( \lambda_C \) are hyperparameters that trade off different loss terms.

**V. EXPERIMENTS**

In this section, we compare HGARN with existing SOTAs on two real-world LBSN check-in datasets.

**A. Datasets**

We adopt two LBSN datasets [2] containing Foursquare check-in records in New York City (NYC) and Tokyo (TKY) from April 12, 2012 to February 16, 2013, including 227,428 check-ins for NYC and 573,703 check-ins for TKY. The location distributions of NYC and TKY datasets are shown in Figure 3. Users and locations with less than 10 records are removed following previous works. After cleaning, we obtain 308 and 286 activities for NYC and TKY, respectively. We divide the data into training and testing sets in a ratio of 8:2, following a chronological order (training first), in line with the conventions used in [7], [20], and [26]. Key data summary statistics are listed in Table I.

**B. Baselines & Experimental Details**

- **MC** [4] is a widely used sequential prediction approach which models transition patterns based on visited locations.
- **STRNN** [25] is an RNN-based model that incorporates the spatial-temporal contexts by leveraging transition matrices.
- **DEEPmove** [7] uses attention mechanisms and an RNN module to capture human mobility patterns.
- **LSTM** [27] introduces a non-local network and a geodilated LSTM to model human mobility patterns.
- **FLASHBACK** [8] is an RNN-based model that leverages spatial and temporal intervals to compute an aggregated hidden state for prediction.
- **PLSPL** [46] incorporates activity information to learn user preferences and utilizes two LSTMs to capture human mobility patterns.
- **PG2NET** [47] learns users’ group and personalized preferences with spatial-temporal attention-based Bi-LSTM.
- **GCDAN** [16] leverages graph convolution to learn spatial-temporal representations and use dual-attention to model the sequential dependencies.
Discounted Cumulative Gain), which are defined as

\[ \text{Recall@K} = \frac{1}{|U|} \sum_{u \in U} \frac{\sum_{l \in R_u} \min(|l|, K)}{|R_u|} \]  

\[ \text{NDCG@K} = \frac{1}{|U|} \sum_{u \in U} \frac{\sum_{l \in \hat{R}_u} \min(|l|, K)}{|R_u|} \]  

where \( \hat{R}_u \) indicates the top \( K \) predicted locations.

We port all the baselines to our run time environment for fair comparisons based on their open-source codes. We carefully tuned their hyperparameters to get the best results. Additionally, unlike previous works that only evaluate overall model performances (main setting), we also conduct experiments under the recurring and explorative settings for more comprehensive performance evaluation. For the main and recurring settings, we choose \( K = \{1, 5, 10\} \) for evaluation. As the performance is generally poorer under the explorative setting, we set \( K = \{10, 20\} \).

For the choice of hyperparameters, we set both \( \lambda_L \) and \( \lambda_C \) to 1, \( \lambda_r \) to 0.6 for both datasets. For embedding dimensions, we set \( d = 200 \), \( d^u = 10 \), \( d^l = 30 \), \( d^g = 50 \) and the dimension of encoders' hidden states are set to 600. Detailed reproducibility information can be found in Appendix B.

### C. Main Results

Table II shows the performance comparison between different methods for next location prediction. HGARN achieves state-of-the-art performance on both datasets across all metrics. Specifically, HGARN outperforms the best baseline approach by 12-19% on Recall@K and NDCG@K for NYC and 11-20% for TKY. Its advantages become more significant as \( K \) increases, validating the effectiveness of the hierarchical graph modeling and MAHeC label for the next location prediction task. We also provide model sizes (i.e., number of model’s trainable parameters) in Table III. Since model sizes are data-specific, we use the NYC dataset for demonstration.

In addition, we also evaluate different models separately in the recurring and explorative settings, with results shown in Tables IV and V.\(^1\) Respectively. In the recurring setting, HGARN outperforms all baselines significantly. In the explorative setting, the overall prediction performance is much lower than those in the main and recurring settings, which is intuitive because of the inherent difficulty of predicting unseen locations. A possible approach to improving the prediction in the explorative setting is to model the dependencies between

---

\(^1\)The Mc’s results are all zeros and thus deleted from the table.
locations. In addition, due to the larger number of locations in the TKY dataset, the hierarchical graph modeling may introduce noise, making our model less effective in ranking the predicted locations. The above hypotheses may also be why our model performs better in Recall but are not consistently better in NDCG.

### D. Ablation Study

To investigate the effectiveness of each component of HGARN, we conduct an ablation study considering the following 6 variants of HGARN:

- **HGARN w/o HG** is the variant that contains only the **temporal module** for next location prediction;
- **HGARN w/o AGAT** is the variant whose **hierarchical graph attention module** contains only the location layer and corresponding GATS;
- **HGARN w/o LAL** is the variant whose **hierarchical graph attention module** contains only the location layer and the activity layer;
- **HGARN w/o RES** is the variant that removes the residual connection in the **temporal module**;
- **HGARN w/o MAHEC** is the variant that leverages original labels to optimize our model;
- **HGCRN** is the variant replacing the hierarchical graph attention with hierarchical graph convolution

The ablation study results are shown in Figure 4. It is found that all performance metrics improve as more components are included. The gradual improvement in prediction results from HGARN w/o HG to HGARN w/o AGAT, and then to HGARN clearly demonstrates the effectiveness of GAT in each layer of the hierarchical graph. In addition, adopting the MAHEC label leads to a significant model performance improvements when K are large, verifying its effectiveness despite the simplicity.

The comparison between HGARN and HGCRN shows that GAT can outperform GCN for graph-based learning tasks through its self-attention mechanism. This has been supported by previous research [41] demonstrating that GAT could achieve competitive performance through proper hyperparameter tuning and configuration compared to GCN. Additionally, it provides more robust inductive capabilities that can incorporate newly available (unseen) nodes without retraining. Finally, GAT is more adaptable and scalable than GCN due to its dynamic scheme that can automatically learn the importance of each node from a graph structure. The hierarchical design can help to overcome the over-smoothing problem [49] of GNNs, and the dependencies between location nodes sharing the same activity across regions can be modeled. In contrast, the HGARN variant without the LAL layer (HGARN w/o LAL) suffers from “information overload” from the location layer, as it aggregates excessive information without proper filtering. This results in a reduced ability to effectively capture both activity-to-activity and fine-grained activity-to-location dependencies, as evidenced by the ablation study. The localized-activity layer (LAL) in HGARN addresses this issue by focusing the model on relevant activity-location relationships while suppressing noise from overly dense connections in the location layer, leading to more effective learning. This approach aligns with techniques explored in other hierarchical GNN models, where multi-layer aggregation schemes help filter irrelevant information and better capture hierarchical relationships in complex systems [38], [50].

### E. Hyperparameter Sensitivity Analysis

We further study the sensitivity of a few key parameters by varying each parameter while keeping others constant. The results in Figure 5 illustrate that the distance threshold $D^h$ affects the location dependencies; if the threshold is too high or too low, the prediction performance would be negatively
affected. The best results are obtained at $D^h = 1$km for NYC and $D^h = 0.1$km for TKY, probably due to the higher density in TKY, as shown in Figure 3. The MAHeC hyperparameter $w^c$ affects how the model treats the locations a user visited in the past. For both datasets, the results show an upward and then downward trend as $w^c$ increases, suggesting that a moderate amount of attention to the previously visited locations can produce the best model performance. This indicates that striking a balance between exploration and recurrence leads to optimal performance in overall mobility modeling. $\lambda_r$ from Eq. (16) affects how much activity information is fused in predicting the next location. Intuitively, a large value of $\lambda_r$ would introduce more noise, and a small value may result in ineffective utilization of activity information. The results confirm that the model achieves optimal performance with $\lambda_r$ at 0.6 for both the NYC and TKY datasets. To summarize, the proposed HGARN model demonstrates robustness across a range of parameter settings, with only small oscillations in performance as parameters vary. The results also demonstrate the model’s capacity to balance exploration and recurrence effectively, while integrating activity information, resulting in consistently strong performance across both datasets.

\subsection*{F. Interpretability Analysis}

1) How Does the MAHeC Label Work?: To understand the mechanism of the MAHeC label, we select an example user trajectory from the NYC dataset, visualize the predicted location probabilities based on “HGARN” and “HGARN w/o MAHeC,” and compare their differences. Figure 6 demonstrates the changes in predicted probabilities, where purple bars represent the locations in the user’s observed trajectory (i.e., visited) and blue bars indicate unvisited locations. The results reveal that the use of MAHeC labels increases the model’s predicted probability for the next location across previously visited locations (i.e., the probability difference for visited locations remains positive). This indicates that MAHeC labels effectively guide the model to consider the user’s past movements when predicting the next location through the model learning process. The effectiveness of MAHeC labels can also partially explain why the prediction performance of HGARN significantly exceeds that of the baseline methods, especially under the recurring setting.

2) What the Hierarchical Graph Learned?: Unlike other deep learning methods that may suffer from limited interpretability, HGARN can be used to reveal the dependencies between activities through the learned Hierarchical Graph. We visualize one attention head of GATC’s sliced attention matrix to analyze the learned activity-activity dependencies. In Figure 7, we select four activity pairs to show the related activities and their corresponding attention scores. These activity pairs are consistent with common sense, such as the high dependencies between Gyms and Stadiums, or Bus Stops and Travel Lounges. These results have important implications for
understanding human activity patterns and predicting future mobility behavior.

VI. CONCLUSION

Both travel behavior theories and empirical evidence suggest that human mobility patterns largely depend on the need to participate in activities at different times of the day. Therefore, it is crucial to consider the latter when modeling the former. In this paper, we propose a Hierarchical Graph Attention Recurrent Network (HGARN) for activity-aware human mobility prediction. Specifically, HGARN introduces hierarchical graph attention mechanisms to model time-activity-location dependencies, and uses next activity prediction as an auxiliary task to further improve the main task of next location prediction. Furthermore, we propose a simple yet effective MAHeC label that can guide our model to flexibly weigh the importance of a user’s previously visited locations when predicting their future locations. Finally, based on two real-world LBSN datasets, we perform comprehensive experiments to demonstrate the superiority of HGARN, considering both the recurring and explorative settings. We find that introducing activity information can effectively improve the model’s prediction performance, and the learned attention weights can reveal meaningful behavioral insights.

Future work should prioritize improving human mobility prediction in explorative settings, where users visit new, unvisited POIs. Developing models that can better infer these complex time-activity-location relationships, even without prior visitation history, will be crucial. Another key challenge is the cold-start problem, where the model must handle new users, locations, or activities introduced into the system. Addressing this could involve leveraging shared features like activity or contextual embeddings for initializing new entities, minimizing retraining needs. From a system design perspective, future models should aim for a lightweight and modular structure. These advancements, combined with a deeper understanding of human decision-making in travel behavior, will not only improve the model’s interpretability but also contribute to more intelligent, user-centered transportation systems that offer personalized and efficient travel recommendations.

APPENDIX

A. Data Preprocessing

The adopted dataset contains check-in data in New York city (NYC) and Tokyo (TKY) collected from Foursquare from 12 April 2012 to 16 February 2013. The data format is shown in Table VI. NYC contains 227428 check-ins in New York city, TKY contains 573703 check-ins in Tokyo.

Figure 8 shows the frequency distribution across different activity types during two selected time periods in NYC and TKY. We removed TKY’s top-1 activity Train Station (with 12468 check-ins) at 11:00-13:00 for better visualization. The results demonstrate a strong temporal dependency between activities.

The data preprocessing flow is illustrated in Figure 9. Based on the raw check-in data, we identify users, locations, and activities and filter out elements with fewer than 10 records. Next, we convert the continuous time to discrete time intervals, and merge mobility records by users to form trajectories. Lastly, we split all trajectories into training and testing data for model fitting and evaluation.

B. Reproducibility

For reproducibility of our study, we provide the specific information about computing devices and detailed hyperparameter settings used in our experiments. The source codes of this study are available at https://github.com/YihongT/HGARN.

All models (including HGARN and other baselines) with learnable parameters are trained on a desktop with Intel(R) Xeon(R) Platinum 8375C CPU @2.90GHz × 64, 125Gi RAM, NVIDIA GeForce RTX 3090 × 8, 4TB SSD. We implement HGARN based on Pytorch. Parameters of HGARN are ran-
domly initialized and optimized using the Adam optimizer with a learning rate of 2e-4, decaying by 0.8 with each epoch. HGARN is trained in 80 epochs.

For hyperparameter settings, we set \( \lambda_L = \lambda_C = 1 \), and dimensions \( d^g = 50 \), \( d = 200 \), \( d^u = 20 \), \( d' = 30 \). Location and activity encoders have hidden states with dimensions of 600. The above settings remain the same for all experiments. For the main and recurring settings, we set the number of attention heads to 2, 2 attention heads and set the dropout to 0.1. For the TKY’s main and recurring settings, they are set to 0.1, 0.6 and 0.6, respectively; for the NYC’s main and recurring settings, they are set to 0.1, 0.9, and 1, respectively.

C. An Example of User Embedding

In Figure 10, we demonstrate a user \( u \)’s embedding process, where \( e_u \) is the learned user embedding vector. Similar processes apply to the location \( l \), activity \( c \), and time slot \( t \) to obtain corresponding embedding vectors \( e_l \), \( e_c \), and \( e_t \), respectively.

D. Training Algorithm

The training process of HGARN is detailed in Algorithm 1.

E. Full Numerical Results

In this section, we show the complete numerical results for the figures presented in the main text. Table VII contains
complete numerical results for our ablation study. Tables IX, X and VIII shows the complete sensitivity experiment results for $D^h$, $\lambda_r$ and $w^c$, respectively.
[25] Q. Liu, S. Wu, L. Wang, and T. Tan, “Predicting the next location: A recurrent model with spatial and temporal contexts,” in Proc. 30th AAAI Conf. Artif. Intell., vol. 30, Feb. 2016, pp. 194–200.

[26] P. Zhao et al., “Where to go next: A spatio-temporal gated network for next POI recommendation,” IEEE Trans. Knowl. Data Eng., vol. 34, no. 5, pp. 2512–2524, May 2022.

[27] K. Sun, T. Qian, T. Chen, Y. Liang, Q. V. H. Nguyen, and H. Yin, “Where to go next: Modeling long- and short-term user preferences for point-of-interest recommendation,” in Proc. AAAI Conf. Artif. Intell., vol. 34, no. 1, 2020, pp. 214–221.

[28] R. Li, Y. Shen, and Y. Zhu, “Next point-of-interest recommendation with temporal and multi-level context attention,” in Proc. IEEE Int. Conf. Data Mining (ICDM), Nov. 2018, pp. 1110–1115.

[29] D. Lian, Y. Wu, Y. Ge, X. Xie, and E. Chen, “Geography-aware sequential location recommendation,” in Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2020, pp. 2009–2019.

[30] J. Manotumrukua, C. Macdonald, and I. Ounis, “A deep recurrent collaborative filtering framework for venue recommendation,” in Proc. ACM Conf. Inf. Knowl. Manage., Nov. 2017, pp. 1429–1438.

[31] B. Chang, Y. Park, D. Park, S. Kim, and J. Kang, “Context-aware hierarchical point-of-interest embedding model for successive POI recommendation,” in Proc. 27th Int. Joint Conf. Artif. Intell., Jul. 2018, pp. 3301–3307.

[32] N. Lim, B. Hooi, S.-K. Ng, Y. L. Goh, R. Weng, and R. Tan, “Hierarchical multi-task graph recurrent network for next POI recommendation,” in Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr., Jul. 2022, pp. 1133–1143.

[33] Y. Yuan, J. Ding, H. Wang, D. Jin, and Y. Li, “Activity trajectory generation via modeling spatiotemporal dynamics,” in Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining, Aug. 2022, pp. 4752–4762.

[34] J. Bao, Y. Zheng, and M. F. Mokbel, “Location-based and preference-aware recommendation using sparse geo-social networking data,” in Proc. 20th Int. Conf. Adv. Geogr. Inf. Syst., Nov. 2012, pp. 199–208.

[35] F. Yu, L. Cui, W. Guo, X. Lu, Q. Li, and H. Lu, “A category-aware deep model for successive POI recommendation on sparse check-in data,” in Proc. Web Conf., Apr. 2020, pp. 1264–1274.

[36] W. Zhang, H. Liu, Y. Liu, J. Zhou, and H. Xiong, “Semi-supervised hierarchical recurrent graph neural network for city-wide parking availability prediction,” in Proc. AAAI Conf. Artif. Intell., vol. 34, 2020, pp. 1186–1193.

[37] J. Xu, L. Chen, M. Lv, C. Zhan, S. Chen, and J. Chang, “HighAir: A hierarchical graph neural network-based air quality forecasting method,” 2021, arXiv:2101.04264.

[38] N. Wu, X. W. Zhao, J. Wang, and D. Pan, “Learning effective road network representation with hierarchical graph neural networks,” in Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2020, pp. 6–14.

[39] W. Zhang et al., “MugRep: A multi-task hierarchical graph representation learning framework for real estate appraisal,” in Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining, Aug. 2021, pp. 3937–3947.

[40] Z. Zhou, Y. Liu, J. Ding, D. Jin, and Y. Li, “Hierarchical knowledge graph learning enabled socioeconomic indicator prediction in location-based social network,” in Proc. ACM Web Conf., Apr. 2023, pp. 122–132.

[41] Q. Lv et al., “Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks,” in Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining, Aug. 2021, pp. 1150–1160.

[42] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” 2017, arXiv:1710.10903.

[43] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[44] S. F. Sönnem and A. R. Graefe, “Determining future travel behavior from past travel experience and perceptions of risk and safety,” J. Travel Res., vol. 37, no. 2, pp. 171–177, Nov. 1998.

[45] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015, arXiv:1503.02531.

[46] Y. Wu, K. Li, G. Zhao, and X. Qian, “Personalized long- and short-term preference learning for next POI recommendation,” IEEE Trans. Knowl. Data Eng., vol. 34, no. 4, pp. 1944–1957, Apr. 2022.

[47] H. Li, B. Wang, F. Xia, X. Zhai, S. Zhu, and Y. Xu, “PG2Net: Personalized and group preferences guided network for next place prediction,” 2021, arXiv:2110.08266.

[48] Y. He, W. Zhou, F. Luo, M. Gao, and J. Wen, “Feature-based POI grouping with transformer for next point of interest recommendation,” Appl. Soft Comput., vol. 147, Nov. 2023, Art. no. 110754.

[49] L. Zhao and L. Akoglu, “PairNorm: Tackling oversmoothing in GNNs,” 2019, arXiv:1909.12223.

[50] K. Guo, Y. Hu, Y. Sun, S. Qian, J. Gao, and B. Yin, “Hierarchical graph convolution network for traffic forecasting,” in Proc. AAAI Conf. Artif. Intell., vol. 35, 2021, pp. 151–159.

Yibong Tang received the bachelor’s degree in computer science and technology from Beijing University of Posts and Telecommunications (BUPT) and the Master of Philosophy (M.Phil.) degree from the Department of Urban Planning and Design, The University of Hong Kong (HKU). His research interests focuses on using data-driven approaches, statistical methods, and artificial intelligence to design more connected, autonomous, intelligent, and user-centric urban and transportation systems, with the aim of achieving broad social benefits.

Junlin He received the bachelor’s degree in computer science and technology from Beijing University of Posts and Telecommunications (BUPT). He is currently pursuing the Doctor of Philosophy (Ph.D.) degree with the Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University (PolyU). His research interests include machine learning and intelligent transport systems.

Zhan Zhao received the bachelor’s degree from Tongji University, the master’s degree from the University of British Columbia, and the Ph.D. degree from the Massachusetts Institute of Technology. He is an Assistant Professor with the Department of Urban Planning and Design, The University of Hong Kong (HKU), and the HKU Muskeeters Foundation Institute of Data Science. His research interests include human mobility modeling, public transportation systems, and urban data science.