Spatio-Temporal Hypergraph Learning for Next POI Recommendation

Xiaodong Yan*
Ant Group
Beijing, China
qiqing.yxd@antgroup.com

Tengwei Song*†
Ant Group
Beihang University
Beijing, China
songtengwei.stw@antgroup.com

Yifeng Jiao*
Ant Group
Hangzhou, China
jiaoyifeng.jyf@antgroup.com

Jianshan He
Ant Group
Beijing, China
yebai.hjs@antgroup.com

Jiaotuan Wang
Ant Group
Hangzhou, China
yunting.wjt@antgroup.com

Ruopeng Li
Ant Group
Beijing, China
ruopeng.lrp@antgroup.com

Wei Chu
Ant Group
Hangzhou, China
weichu.cw@antgroup.com

ABSTRACT

Next Point-of-Interest (POI) recommendation task focuses on predicting the immediate next position a user would visit, thus providing appealing location advice. In light of this, graph neural networks (GNNs) based models have recently been emerging as breakthroughs for this task due to their ability to learn global user preferences and alleviate cold-start challenges. Nevertheless, most existing methods merely focus on the relations between POIs, neglecting the higher-order information including user trajectories and the collaborative relations among trajectories. In this paper, we propose the Spatio-Temporal HyperGraph Convolutional Network (STHGCN). This model leverages a hypergraph to capture the trajectory-grain information and learn from user's historical trajectories (intra-user) as well as collaborative trajectories from other users (inter-user). Furthermore, a novel hypergraph transformer is introduced to effectively combine the hypergraph structure encoding with spatio-temporal information. Extensive experiments on real-world datasets demonstrate that our model outperforms the existing state-of-the-art methods and further analysis confirms the effectiveness in alleviating cold-start issues and achieving improved performance for both short and long trajectories.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Neural networks.*

KEYWORDS

Next POI Recommendation, Hypergraph, Graph Transformer

ACM Reference Format:
Xiaodong Yan, Tengwei Song, Yifeng Jiao, Jianshan He, Jiaotuan Wang, Ruopeng Li, and Wei Chu. 2023. Spatio-Temporal Hypergraph Learning for Next POI Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591770

1 INTRODUCTION

In recent years, the advancements of Location-Based Social Networks (LBSNs) bring the flourishing of Point-of-Interest (POI) recommender systems, which provide valuable geographical information for both service providers and users. In the next-POI recommendation scenario, when a user visits a POI at a certain time, a *check-in* record may be created. These check-in sequences then form the user's *trajectory*, depicting their generic movement patterns within a specific time window and providing higher-order information for the next-POI recommendation.

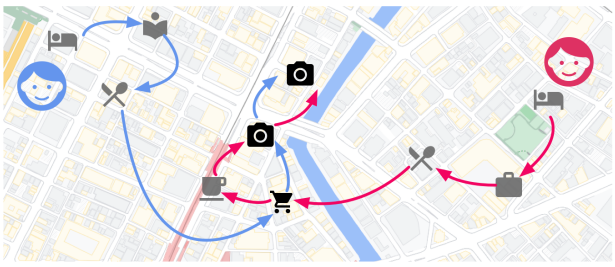


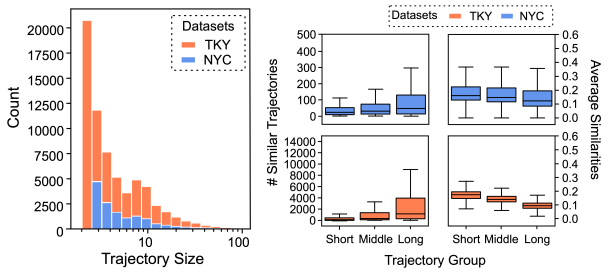
Figure 1: Trajectory overlapping between two users.

*The first three authors contributed equally to this research.
†This work was done during author's internship at Ant Group.

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591770

Trajectories play a pivotal role in next-POI recommendation. A user may generate different historical trajectories by visiting various locations at different times and if these trajectories share associated check-in clips, we say these trajectories have *intra-user* collaborative relations. Similarly, *inter-user* collaborative trajectories reflect similarities in movement behavior between individuals. As shown in Figure 1, the trajectories of two users can partially overlap, making it possible to predict one's next POI destination by referring to other similar trajectories. Inter-user trajectory information is especially essential for LBSNs that have diverse behavior patterns, such as short trajectories, or users with few check-in histories (known as the cold-start problem in recommender systems).



(a) Trajectory sizes (b) Trajectory similarities

Figure 2: Statistic of trajectory sizes and similarities in real-world datasets. We rank trajectories by length and label the top 30% as long trajectories, bottom 30% as short trajectories and others as middle ones.

In real-world LBSNs, short trajectories make up a large portion of user behaviors. Moreover, many trajectories may have strong correlations with one another. As shown in Figure 2(a), if 24-hour time intervals are used to construct trajectories, most trajectories contain fewer than 10 check-in records. The similarity of two trajectories is measured by Jaccard distance, which is the number of overlapped POIs divided by the total number of POIs within two trajectories, and two trajectories are similar when the metric is greater than 0.005. Figure 2(b) reveals the number of similar trajectories and the average similarity, where we can see that short trajectories have fewer similar trajectories yet with higher similarities between them, allowing for obtaining strong collaborative information through similar trajectories. Long trajectories have more similar trajectories, yet with lower similarities, containing much more noise as a consequence. The statistical results demonstrate the difficulties of modeling trajectory collaborations, however, they also highlight the potential of using the intricate collaborative relations among trajectories to generate higher-quality recommendation results.

Sequence-based approaches such as LSTPM [16] emphasize the learning of individual user sequences from a local view, making it difficult to leverage beneficial collaborative information from other users. In order to make use of the collaborative information across users in a global view, GNN-based methods offer a solution [10, 11, 20, 21, 27]. GETNext [27], in particular, proposes a trajectory flow map for learning user preferences from trajectories, but it only incorporates low-order POI-POI relations and is unable to leverage high-order information from trajectory collaboration.

Recently, hypergraph-based methods has attracted increasing attention in the field of recommendation [19, 23, 24] due to their ability to capture the high-order relations and enhance the efficacy of models. Conceptually, hypergraph is a generalization of a graph wherein a hyperedge links arbitrary number of vertices, while maintaining an inherent coherence between the nodes within this hyperedge. In the context of next-POI recommendation, check-in and trajectory information is invaluable, with each providing a distinctive level of granularity. Trajectories, composed of check-ins, depict users' higher-order movement patterns and can be regarded as hyperedges. Therefore, incorporating hypergraph into next-POI recommendation task is a challenging yet promising approach that could lead to improved model performance.

To effectively capture high-order relations between user movement patterns from fine-grained check-in level to coarse-grained trajectory level, we introduce hypergraph to the next-POI recommendation and propose a Spatio-Temporal HyperGraph Convolutional Network (STHGCN). STHGCN models high-order relations betwen check-ins within trajectories as well as the correlations between trajectories, while taking into account the spatio-temporal context. Specifically, we regard check-ins and trajectories as nodes and hyperedges in a hypergraph, and build heterogeneous relations between hyperedges including intra-user and inter-user types. Inspired by graph transformer [8, 15], we introduce a Hypergraph Transformer, which unifies the two types of message passing paradigms (nodes to hyperedges and hyperedges to hyperedges), and utilizes the spatio-temporal information when learning attention.

The main contributions of our work are summarized as follows:

- We propose a spatio-temporal hypergraph convolutional network, which integrates complex high-order information and global collaborative relations among trajectories. To the best of our knowledge, this is the first work to leverage hypergraph for the next-POI recommendation task.
- We utilize a similarity-based paradigm to identify strong correlations among trajectories. Moreover, we introduce a hypergraph transformer to combine the inter-user and intra-user collaborative messages as well as spatio-temporal context information simultaneously. By doing so, our model can effectively alleviate the cold-start problem and improve prediction accuracy on both short and long trajectories.
- We conduct experimental validation on real-world LBSN datasets, and the results show that our approach outperforms state-of-the-art methods.

2 RELATED WORK

Next-POI Recommendation. The methods for next-POI recommendation can be categorized into two distinct streams of literature:

(1) Sequential-based methods regard the visit history of a user as a sequence. Traditional MC based methods [7] fuse Markov chain with latent pattern, and learn the transition probability between successive check-ins. Matrix Factorization (MF) based methods represented as PMF [14] factorize the user-POI interaction matrix to learn dense embeddings. FPMC [13] combines MF and MC and factorize the personalized transition matrix to capture the long term user preferences. However, these methods are limited

in modeling complex long term and short term patterns. With the boom of deep learning, there raised RNN based models. HST-LSTM [9] proposed the hierarchical spatio-temporal LSTM to combines spatio-temporal influence into LSTM. From the view of trajectory, LSTPM [16] learn the representation of historical trajectories of a user by LSTM to capture the long term information. Based on attention mechanism, STAN [12] uses a bi-layer attention architecture that models non-adjacent check-ins through attention of spatio-temporal interactions. Recently, CFPRec [29] further propose to split user preferences into past, current and future, and model them via Transformer layer, LSTM layer and a two-layer attention aggregation respectively.

(2) Graph-based methods for next-POI recommendation have been developed recently to address the limitations of sequential-based methods, enabling to learn across users in a global view. STP-UDGAT [11] first proposed a spatio-temporal-preference user dimensional graph attention network to learn from various graphs. To alleviate the sparsity of the User-POI matrix, HMT-GRN [10] facilitate the task by learning User-Region matrices, and transfer it into a sequence annotation task using beam search. The graph learning module of HMT-GRN simply build an undirected and unweighted POI-POI graph without consideration of global collaboration, which inadequately exploit the graph information. For better utilizing the trajectory flows information, GETNext [27] propose a global trajectory flow map to learn enhanced POI embedding for Transformer. DRAN [21] adopts both distance-based and transition-based POI-POI graph to learn the transfer of user behavior between POIs via disentangled representation.

Hypergraph Learning. Hypergraph offers a natural approach to express and capture heterogeneous high-order structures. HGNN [5] and HyperGCN [25] first apply graph convolution to hypergraph, however, these methods are not memory and computationally efficient due to the conventional graph convolution. AllSet[2] proposed a novel message passing based framework for hypergraph convolution, which integrates Deep Sets and Set Transformers, and gains more modeling flexibility.

Recently, hypergraph neural networks have received much attention in various tasks of recommendation systems. In sequence-based recommendation task, HyperRec [19] build a series of hypergraphs based on item correlations of users at different time periods to learn dynamic item embedding. In session-based recommendation scenario, DHCN [24] utilize hyperedges to aggregate agnostic items within sessions, and integrates self-supervised learning into a dual channel GCNs. In multi-behavior recommendation scenario, MBHT [28] construct hypergraph to capture behavior-aware interaction sequences for enhancing a transformer framework. These studies have demonstrated the impressive potential of hypergraph-based methods, highlighting the promising applications for next-POI recommendation.

3 THE PROPOSED METHOD: STHGCN

3.1 Preliminary and Notation

Next-POI problem formulation. Let $U = \{u_1, u_2, \ldots, u_{|U|}\}$ be the user set, $P = \{p_1, p_2, \ldots, p_{|P|}\}$ be the POI set, and $C = \{c_1, c_2, \ldots, c_{|C|}\}$ be the category set. Let $Q = \{q_1, q_2, \ldots, q_{|Q|}\}$ denote the set of all check-in records. Each check-in is denoted by $q =< u, p, c, g, t >$ with information of user u visited a POI p at timestamp t, with the POI category c and geometric information $g =< latitude, longitude >$. Let $S = \{s_1, s_2, \ldots, s_{|S|}\}$ be the set of all trajectories. Trajectory is a set of consecutive check-ins generated as follows: For each user u, let Q^u denote the history check-ins of u. Split Q^u by a certain time interval and get a trajectory sequence $S^u = \{s_1^u, s_2^u, \ldots, s_{|S^u|}^u\}$, where the mth trajectory s_m^u contains a sequence of user's check-ins, denoted by $s_m^u = \{q_1^u, q_2^u, \ldots\}$. The time interval we adopt in this paper is 24 hours. In the rest of paper, we will omit the user indices in S^u and s_m^u for simplicity.

For an arbitrary check-in indexed by $k (k = 0, 1, \ldots, |Q|)$, let s_m denote the trajectory that q_k belongs to, i.e., $q_k \in s_m$. We say that two trajectories are intra-user correlated, denoted $s_m \sim s_n$, if two trajectories s_m and s_n come from the same user. Similarly, we say two trajectories are inter-user collaborated, denoted $s_m \approx s_n$, if s_m and s_n come from different users and $Jaccard(s_m, s_n) \geq \eta$, where $Jaccard(\cdot, \cdot)$ is the Jaccard distance function defined in Eq. (1) and η is the threshold set manually. Here we only take POI element p into account when calculating this metric.

$$Jaccard(s, s') = \frac{|s \cap s'|}{|s \cup s'|}. \tag{1}$$

To formulate the aim of next-POI recommendation task in a trajectory-grain view, given historic trajectories S before current timestamp t_c, and a current trajectory s_c of a specific user, the task is to let the model rank the visiting confidence among all POIs $\hat{y}_i (i = 0, 1, \ldots, |P|)$ and predict the POIs that the user incline to visit next.

Hypergraph. Hypergraph is denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of vertices with size $|\mathcal{V}|$ and $\mathcal{E}$ is the set of hyperedges with size $|\mathcal{E}|$. For each hyperedge $e \in \mathcal{E}$, there are two or more than two vertices $v \in \mathcal{E}$. The hypergraph can be also represented by an incidence matrix $\mathcal{H}$, where $\mathcal{H}_{ve} = 1$ if $v \in e$ and $\mathcal{H}_{ve} = 0$ otherwise.

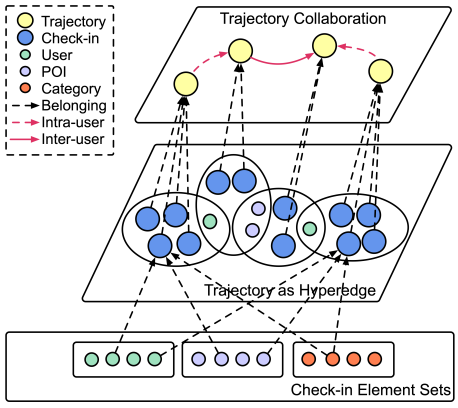


Figure 3: Multi-level hypergraph construction. Four trajectories have pairwise collaborative relations because of POI-overlapping or user-overlapping, which are denoted as inter-user (red solid arrows) or intra-user (red dotted arrows) collaborative relations respectively.

3.2 Hypergraph Construction

To capture relations beyond pairwise POI-POI in next-POI recommendation, we adopt a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ illustrated in Figure 3 to represent each trajectory as a hyperedge, *i.e.*, $\mathcal{V} \equiv Q, \mathcal{E} \equiv S$. To further introduce the collaborative relations among trajectories, we construct both $\mathcal{V} \to \mathcal{E}$ and $\mathcal{E} \to \mathcal{E}$ relation matrices.

We use $\mathcal{H}^{(1)}$ and $\mathcal{H}^{(2)}$ to denote the incidence matrix of $\mathcal{V} \to \mathcal{E}$ and hyperedge adjacent matrix of $\mathcal{E} \to \mathcal{E}$ respectively, where $\mathcal{H}^{(1)} \in \mathbb{R}^{|Q|\times|S|}$ and $\mathcal{H}^{(2)} \in \mathbb{R}^{|S|\times|S|}$. For $\mathcal{H}^{(1)}$, $\mathcal{H}^{(1)}_{km} = 1$ when a check-in q_k belongs to a trajectory s_m, *i.e.*, $q_k \in s_m$. For $\mathcal{H}^{(2)}$, $\mathcal{H}^{(2)}_{mn} = 1$ when two trajectories s_m and s_n satisfies $s_m \sim s_n$ or $s_m \approx s_n$. The relations in $\mathcal{E} \to \mathcal{E}$ are thus heterogeneous, which contain both intra-user and inter-user trajectory correlations. To remove data leakage when predicting the next-POI movement within a trajectory instead of the last visitation, we require that the end time of the source trajectory should be less than the start time of the target trajectory. Besides, to construct the heterogeneous information, we introduce edge type matrix $\mathbf{r}_{\mathcal{E}\to\mathcal{E}}$ between hyperedges. Specifically, we assign an unique edge type $\mathbf{r}_{mn} \in \mathbb{N}$ when $\mathcal{H}^{(2)}_{mn} = 1$, *e.g.*, $\mathbf{r}_{mn} = 0$ for intra-user correlation and $\mathbf{r}_{mn} = 1$ for inter-user collaboration.

To further involve spatio-temporal information on hypergraph, we assign relative time difference and spatial distance where $\mathcal{H}^{(1)}_{km} = 1$ and $\mathcal{H}^{(2)}_{mn} = 1$. Let $\Delta t_{\mathcal{V}\to\mathcal{E}}$ and $\Delta s_{\mathcal{V}\to\mathcal{E}}$ denotes the time difference and spatial distance between check-ins and their connected trajectories, and let $\Delta t_{\mathcal{E}\to\mathcal{E}}$ and $\Delta s_{\mathcal{E}\to\mathcal{E}}$ denotes the time difference and spatial distance between two connected trajectories respectively. In particular, for a check-in $q_k =< u, p, c, g, t >$, its timestamp $q^t_k = t$ and its location $q^l_k = g$. For a trajectory $s_m = \{q_k | k = 0, 1, \ldots, |s_m|\}$, we define its timestamp as the average timestamp of its check-ins, *i.e.*, $s^t_m = \Sigma_k^{|s_m|} \frac{q^t_k}{|s_m|}$, and its location as the average locations of its check-ins, *i.e.*, $s^l_m = \Sigma_k^{|s_m|} \frac{q^l_k}{|s_m|}$. As such, spatio-temporal information of s_m-related relations are

$$\begin{aligned}
\Delta t_{\mathcal{V}\to\mathcal{E}} &= \{q^t_{|s_m|} - q^t_k | \mathcal{H}^{(1)}_{km} = 1\} \\
\Delta t_{\mathcal{E}\to\mathcal{E}} &= \{s^t_m - s^t_n | \mathcal{H}^{(2)}_{mn} = 1\} \\
\Delta s_{\mathcal{V}\to\mathcal{E}} &= \{d(q^l_{|s_m|}, q^l_k) | \mathcal{H}^{(1)}_{km} = 1\} \\
\Delta s_{\mathcal{E}\to\mathcal{E}} &= \{d(s^l_m, s^l_n) | \mathcal{H}^{(2)}_{mn} = 1\}
\end{aligned} \tag{2}$$

where $d(\cdot, \cdot)$ is the Haversine [1] distance between two locations.

3.3 STHGCN

In this section, we present three main modules of our STHGCN framework: (1) neighbor sampler; (2) spatio-temporal encoder; and (3) hypergraph transformer. Firstly, the neighbor sampler fetches a sub-hypergraph of center trajectories which contains their collaborative trajectories and the check-in nodes belonging to each trajectory, thus benefiting mini-batch training and a more robust model. Secondly, spatio-temporal encoder encodes the time difference and the spatial distance to vectors and incorporates the information at the hypergraph convolution stage. Finally, we propose a novel hypergraph transformer based on Message Passing Neural Networks (MPNN) [6] paradigm, which is capable of learning the attention weights for different neighbors awaring of the spatio-temporal context information. The overall architecture is shown in Figure 5 (neighbor sampler and spatio-temporal encoder are not displayed for simplicity).

3.3.1 Neighbor Sampler. While both trajectories and check-ins are involved in every hop of neighbors, we propose a two-step neighbor sampling process to fetch their check-in and trajectory neighbors separately as is illustrated in Figure 4.

At the first step, we sample multi-hop trajectory neighbors for target trajectory using the hyperedge adjacent matrix $\mathcal{H}^{(2)}$, which means that we only fetch a fraction of multi-hop collaborative trajectories. To filter out weak inter-user collaborative relations, we define a hyperparameter η_{inter} only to keep the inter-user collaborative relations satisfying $Jaccard(s_m, s_n) \geq \eta_{inter}$. Similarly, we define η_{intra}. And these two values may differ from the threshold η when constructing the hypergraph. Then, we sample 1-hop check-in neighbors for all the trajectories (including target trajectories) sampled at the first step, using the incidence matrix $\mathcal{H}^{(1)}$. Thus, we finally get the sub-hypergraph for target trajectories.

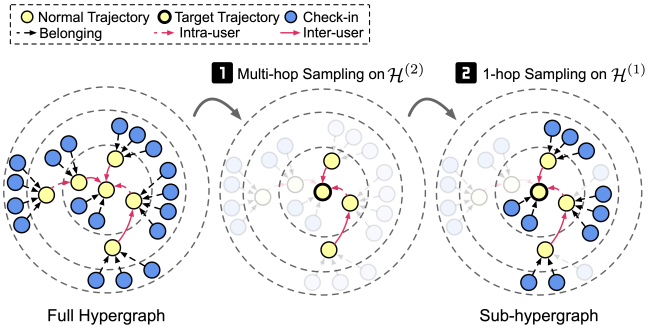


Figure 4: Two-step neighbor sampling process.

3.3.2 Spatio-Temporal Encoder. For the next-POI recommendation task, one of the main challenges is how to incorporate the spatio-temporal information in the model and empower the generalization ability of the representation. Here we propose a Spatio-Temporal Encoder to handle the spatio-temporal information.

For temporal information, introduced by [8], the relative temporal encoding mechanism is a proper way to map continuous time value to the d_t-dimensional vector space and has been proved to be capable of modeling the dynamic dependencies in the graph. Specifically, given the time difference Δt as we already mentioned in Eq. (2), this technique can be seen as a function Φ which utilizes a set of sinusoidal functions as basis:

$$\begin{aligned}
\Phi(\Delta t)_i &= \cos(\Delta t / 10^{\frac{i}{d_t}}), \\
\mathbf{t} &= \mathbf{W}_t \Phi(\Delta t),
\end{aligned} \tag{3}$$

where $\mathbf{t}$ is the resulting time vector, and $\mathbf{W}_t \in \mathbb{R}^{d_t \times d_t}$ is the trainable weight matrix of a linear projection.

For spatial information, borrowed from [12], a unit embedding layer and an interpolation embedding layer are two options, and we refer to them as "simple linear" and "linear interpolation" respectively. Simple linear only uses one unit vector defined as $\mathbf{w}_s \in \mathbb{R}^{d_s}$

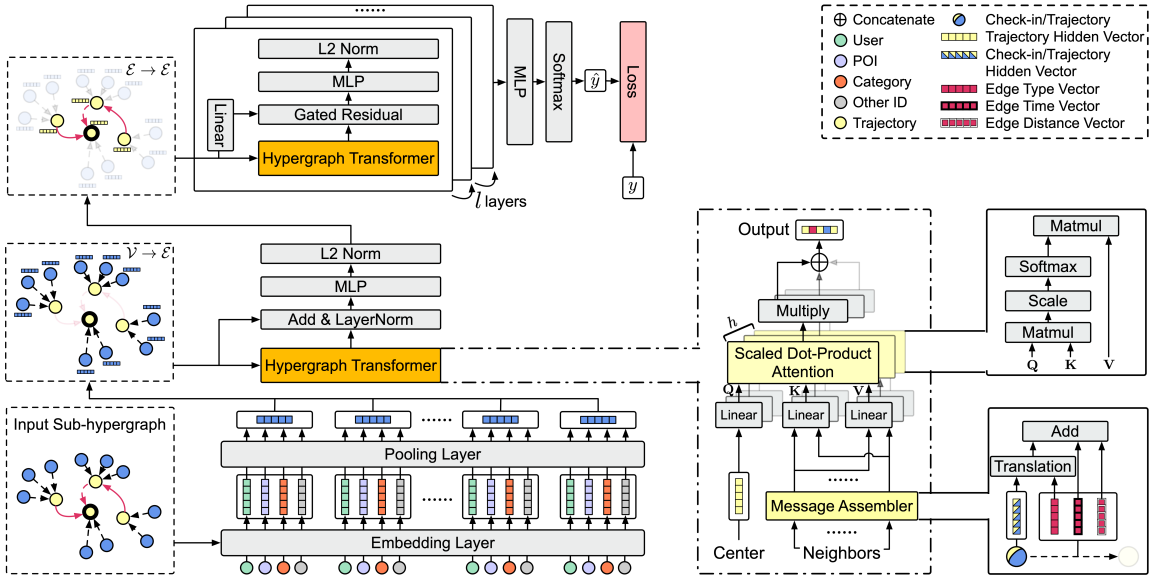


Figure 5: The overall STHGCN framework.

with d_s as the dimension, the result distance vector is formulated as

$$\mathbf{s} = \Delta s \times \mathbf{w}_s. \tag{4}$$

Linear interpolation initializes a trainable upper-bound unit vector $\mathbf{s}_u \in \mathbb{R}^{d_s}$ and a trainable lower-bound unit vector $\mathbf{s}_\ell \in \mathbb{R}^{d_s}$, and the interpolation distance vector is calculated as

$$\mathbf{s} = \frac{\mathbf{s}_u(\Delta s^u - \Delta s) + \mathbf{s}_\ell(\Delta s - \Delta s^\ell)}{\Delta s^u - \Delta s^\ell}, \tag{5}$$

where Δs^u and Δs^ℓ are upper-bound and lower-bound of the spatial distance value respectively.

Meanwhile, both $\Delta t_{\mathcal{V}\to\mathcal{E}}$ and $\Delta t_{\mathcal{E}\to\mathcal{E}}$ share the same time encoder, and both $\Delta s_{\mathcal{V}\to\mathcal{E}}$ and $\Delta s_{\mathcal{E}\to\mathcal{E}}$ share the same distance encoder in our model. We also use the relative temporal encoding to deal with the spatial information in our experiments (see Section 4.2.3), which surprisingly achieves better performances than the other two methods.

3.3.3 Hypergraph Transformer. To aggregate neighbor information during the message passing process while incorporating rich edge information such as type vector $\mathbf{r}$, time vector $\mathbf{t}$ and distance vector $\mathbf{s}$, we propose a novel hypergraph transformer layer as is illustrated at the bottom right of the Figure 5.

Generally, hypergraph transformer involves message assembling and propagation stages and can be formulated as

$$\mathbf{h}_i^{(l+1)} = \underset{\forall j \in \mathcal{N}(i)}{\textbf{Propagate}}\{\textbf{Message}(\mathbf{h}_j^{(l)}, \mathbf{r}_{ij}, \mathbf{t}_{ij}, \mathbf{s}_{ij})\}, \tag{6}$$

where i and j are the indices of the target node (regard hyperedges as special nodes in realization) and the source node, respectively. Set $\mathcal{N}(i)$ contains neighbors of the node i. $\mathbf{h}_i^{(l+1)} \in \mathbb{R}^{d_h}$ is the hidden embedding of the target node at $(l+1)$-th layer, and $\mathbf{h}_j^{(l)} \in \mathbb{R}^{d_h}$ are hidden embedding of neighbors at the l-th layer. Suppose $\mathbf{x}_j \in \mathbb{R}^d$ is original features of source node j, then we have $\mathbf{h}_j^{(0)} = \mathbf{x}_j$. Note that $d = d_h$. Also, $\mathbf{r}_{ij}$, $\mathbf{t}_{ij}$ and $\mathbf{s}_{ij}$ are edge vectors between i and j.

At the message assembling stage, inspired by [17] which introduces non-parametric composition operators as GCN encoder and has been shown success on link-prediction task, we utilize four composition operators termed as "translation" to combine node hidden representation and edge type vector as follows:

$$\begin{aligned}
&\textbf{Addition (Add)} : \phi(\mathbf{h}_j, \mathbf{r}_{ij}) = \mathbf{h}_j + \mathbf{r}_{ij},\\
&\textbf{Subtraction (Sub)} : \phi(\mathbf{h}_j, \mathbf{r}_{ij}) = \mathbf{h}_j - \mathbf{r}_{ij},\\
&\textbf{Multiplication (Mult)} : \phi(\mathbf{h}_j, \mathbf{r}_{ij}) = \mathbf{h}_j * \mathbf{r}_{ij},\\
&\textbf{Circular-correlation (Corr)} : \phi(\mathbf{h}_j, \mathbf{r}_{ij}) = \mathbf{h}_j \star \mathbf{r}_{ij}.
\end{aligned} \tag{7}$$

Then we add the time vector and the distance vector, and thus get message vector $\mathbf{m}_{ij} \in \mathbb{R}^{d_h}$ for every $(source\ node, edge)$ pair as

$$\mathbf{m}_{ij}^{(l)} = \phi(\mathbf{h}_j^{(l)}, \mathbf{r}_{ij}) + \mathbf{t}_{ij} + \mathbf{s}_{ij}. \tag{8}$$

At the propagation stage, we first estimate the importance of each message based on a multi-head scaled dot-product attention module which is commonly used in the designing of Transformer[18] architecture, and then aggregate the neighborhood message with a weighted summation by the attention weight. To obtain query, key, and value vector, we use three learnable projection matrices $\mathbf{W}_Q^{(k,l)}, \mathbf{W}_K^{(k,l)}, \mathbf{W}_V^{(k,l)}$ with the same dimension of $d_h \times d_h$, where k is the head index. Finally, the hidden embedding is updated by

$$\begin{aligned}
&\mathbf{Q}^{(k,l)} = \mathbf{W}_Q^{(k,l)}\mathbf{h}_i^{(l)}, \mathbf{K}^{(k,l)} = \mathbf{W}_K^{(k,l)}\mathbf{m}_{ij}^{(l)}, \mathbf{V}^{(k,l)} = \mathbf{W}_V^{(k,l)}\mathbf{m}_{ij}^{(l)}\\
&\mathbf{h}_i^{(l+1)} = \Big\|_{k=1}^{h} \textbf{Softmax}\left(\frac{\mathbf{Q}^{(k,l)}(\mathbf{K}^{(k,l)})^T}{\sqrt{d_h}}\right)\mathbf{V}^{(k,l)},
\end{aligned} \tag{9}$$

here, $\|$ denotes concatenation and h is the number of heads. Note that the output dimension of $\mathbf{h}^{(l+1)}$ is equal to $\mathbb{R}^{hd_h}$.

3.3.4 Hyperedge Learning and Prediction. The hyperedge (trajectory) representation learning consists of three parts (as is shown at the left of Figure 5): check-in representation by its original id feature, initialing trajectory representation by aggregating message from neighbor check-ins, and updating trajectory hidden representation by message passing of collaborative trajectories.

Firstly, each check-in $q =< u, p, c, g, t >$ in sub-hypergraph basically has three id features: user id u, POI id p, and category id c. Besides, we add two extra time-related id features to capture the temporal specificity of the user mobility such as people are in favor of bars in the evening. One is the hour-of-day id feature t_h with 24 categories, and the other one is the day-of-week id feature t_d with 7 categories. The embedding layer f_{embed} transforms every id feature into a d_{id}-dimensional dense vector, then the pooling layer combines all the five embeddings to generate the check-in representation. The representing process of check-ins is formulated as

$$\mathbf{x} = \Big\|_{\forall i \in \{u,p,c,t_h,t_d\}} f_{embed}(i), \tag{10}$$

hence $d_h = 5d_{id}$.

Secondly, we produce initial representations of trajectories by hypergraph transformer. Specifically, when applying hypergraph transformer to represent trajectories by aggregating messages from check-ins ($\mathcal{V} \rightarrow \mathcal{E}$), we should replace $\mathbf{Q}^{(k)}$ with a learnable weight matrix $\theta^{(k)} \in \mathbb{R}^{d_h}$ in Eq. (9) because trajectories don't have original features at the first place. Hence $\theta \triangleq \|_{k=1}^{h} \theta^{(k)}$ means that all trajectories share the same initial embedding before aggregating messages from their neighbors. Formally,

$$\begin{aligned}
&\mathbf{m}_{ij}^{(0)} = \phi(\mathbf{x}_j, \mathbf{r}_{ij}) + \mathbf{t}_{ij} + \mathbf{s}_{ij}, \\
&\mathbf{Q}^{(k,0)} = \theta^{(k)}, \mathbf{K}^{(k,0)} = \mathbf{W}_K^{(k,0)} \mathbf{m}_{ij}^{(0)}, \mathbf{V}^{(k,0)} = \mathbf{W}_V^{(k,0)} \mathbf{m}_{ij}^{(0)}, \\
&\mathbf{h}_i^{(1)} = \Big\|_{k=1}^{h} \mathbf{Softmax}\left(\frac{\mathbf{Q}^{(k,0)}(\mathbf{K}^{(k,0)})^T}{\sqrt{d_h}}\right)\mathbf{V}^{(k,0)}.
\end{aligned} \tag{11}$$

We use the residual connection and the layer normalization skills to avoid the loss of the initial information, and then utilize multi-layer perceptrons (MLP) with a bias term that compresses the representation back into a d_h feature space. These processes are formulated as

$$\begin{aligned}
&\mathbf{h}_i^{(1)} = \mathbf{LN}(\theta + \mathbf{h}_i^{(1)}), \\
&\mathbf{h}_i^{(1)} = \mathbf{Norm}(\sigma(\mathbf{h}_i^{(1)}\mathbf{W}_0^{(0)} + \mathbf{b}_0^{(0)})\mathbf{W}_1^{(0)} + \mathbf{b}_1^{(0)})
\end{aligned} \tag{12}$$

where **LN** and **Norm** denotes the layer normalization and L2 normalization, respectively. $\mathbf{W}_0^{(0)} \in \mathbb{R}^{hd_h \times d_h}$, $\mathbf{W}_1^{(0)} \in \mathbb{R}^{d_h \times d_h}$ are weight matrices and $\mathbf{b}_0^{(0)}, \mathbf{b}_1^{(0)} \in \mathbb{R}^{d_h}$ are bias terms. σ is the nonlinear activation function, here we use ReLU.

To update trajectory hidden representation with rich multi-hop collaborative messages ($\mathcal{E} \rightarrow \mathcal{E}$), we apply a stack of $(L-1)$ hypergraph transformers and thus our model is capable of aggregating messages from long-range neighbors. Similarly, we keep the MLP and L2 normalization used in Eq. (12) at each layer, except that we use a linear projection and gated residual module instead of "Add & LayerNorm" to balance the information between the representation of previous layer $\mathbf{h}_i^{(l)}$ and the output of hypergraph transformer $\mathbf{g}_i^{(l+1)}$. So the processes above can be expressed as

$$\begin{aligned}
&\mathbf{g}_i^{(l+1)} = \mathbf{Hypergraph\ Transformer}(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, \mathbf{r}_{ij}, \mathbf{t}_{ij}, \mathbf{s}_{ij}), \\
&\mathbf{h}_i^{(l+1)} = \beta(\mathbf{h}_i^{(l)}\mathbf{W}_2^{(l)} + \mathbf{b}_2^{(l)}) + (1-\beta)\mathbf{g}_i^{(l+1)}, \\
&\mathbf{h}_i^{(l+1)} = \mathbf{Norm}(\sigma(\mathbf{h}_i^{(l+1)}\mathbf{W}_0^{(l)} + \mathbf{b}_0^{(l)})\mathbf{W}_1^{(l)} + \mathbf{b}_1^{(l)}),
\end{aligned} \tag{13}$$

where **Hypergraph Transformer** denotes formulas of Eq. (8) and Eq. (9) for simplicity and $l = \{1, 2, \ldots, L-1\}$. β is a hyperparamter and denotes the residual weight, and $\mathbf{W}_2^{(l)} \in \mathbb{R}^{d_h \times hd_h}$ is a linear projection to align the dimension when using gated residual module. $\mathbf{b}_2^{(l)} \in \mathbb{R}$ is the bias term.

Finally, we use one-layer perceptrons to map the trajectory representation into POI id space and make predictions of user's next movement, which is formulated as

$$\hat{\mathbf{y}}_i = \mathbf{Softmax}(\mathbf{h}_i^{(L)}\mathbf{W}_o + \mathbf{b}_o) \tag{14}$$

where $\mathbf{W}_o \in \mathbb{R}^{d_h \times |P|}$ and $\mathbf{b}_o \in \mathbb{R}^{|P|}$ are the weight matrix and bias, respectively. We use cross-entropy loss as the loss function for mini-batch training, and the loss is expressed as

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{p=1}^{|P|}\mathbf{y}_i \log \hat{\mathbf{y}}_i. \tag{15}$$

4 EXPERIMENTS

4.1 Experimental Setup

4.1.1 Datasets. We conduct our experiment on three public real-world datasets: Foursquare-NYC [26], Foursquare-TKY [26][1], and Gowalla-CA [3][2]. All datasets are collected from location-based social media platforms. The Foursquare-NYC dataset is collected in New York city and The Foursquare-TKY dataset is collected in Tokyo city over 11 months from Foursquare. The Gowalla-CA dataset with a longer time period and broader geographical coverage are collected in California and Nevada on the Gowalla platform. All check-in records in datasets contain user, POI ID, POI category, longitude, latitude, and timestamp. During preprocessing, we follow the next three steps: 1) filter out POIs with fewer than 10 visit records in history, 2) filter out users with fewer than 10 visit records in history, 3) divide user check-in records into several trajectories with 24-hour intervals and exclude the trajectories with only one check-in record. Then we sequence all of the check-in records in chronological order, the first 80% check-ins are defined as the training set, the middle 10% check-ins are defined as the validation set and the last 10% check-ins are defined as the test set. The validation/test set has to contain all the users and POIs in the training set. Otherwise, the extra user or POI will be removed from the validation/test set. Also, we only evaluate the last check-in record of each trajectory in the validation/test set. The implementation of our model is available via https://github.com/ant-research/Spatio-Temporal-Hypergraph-Model.

4.1.2 Baseline Models.

- FPMC [13]: A method based on Bayesian Personalized Ranking framework, using a common Markov chain and normal matrix factorization model to estimate location transition.

[1] https://sites.google.com/site/yangdingqi/home/foursquare-dataset
[2] https://snap.stanford.edu/data/loc-gowalla.html

- PRME [4]: A personalized ranking model uses a pairwise ranking metric embedding algorithm to learn POI sequential transition and user-POI preference in the latent space.
- STGCN [30]: An LSTM-based model employs the gate mechanism to model time and distance intervals over check-in sequences, which can capture both users' short-term and long-term preferences.
- PLSPL [22]: A recurrent model learns short-term preference by attention mechanism and long-term preference by two parallel LSTM models separately, while the two modules are fused via a user-based linear combination unit.
- STAN [12]: A state-of-the-art model uses a bi-layer attention architecture to aggregate spatio-temporal correlation within user trajectory, which can learn the regularities between non-adjacent locations and non-contiguous visits
- GETNext [27]: A transformer-based model utilizes a user-agnostic global trajectory flow map to enhance the next-POI prediction and proposes a GCN model to generate POI embedding.

4.1.3 Evaluation Metrics. We use two evaluation metrics: Top-k accuracy rates (Acc@k) and Mean Reciprocal Rank(MRR), which are widely used in the next-POI recommendation task. Acc@k represents the rate of whether the true POI exists in the Top-k predicted POIs. MRR indicates the index of true POI in the predicted POI list. All these metrics measure both classification precision and ranking quality of models.

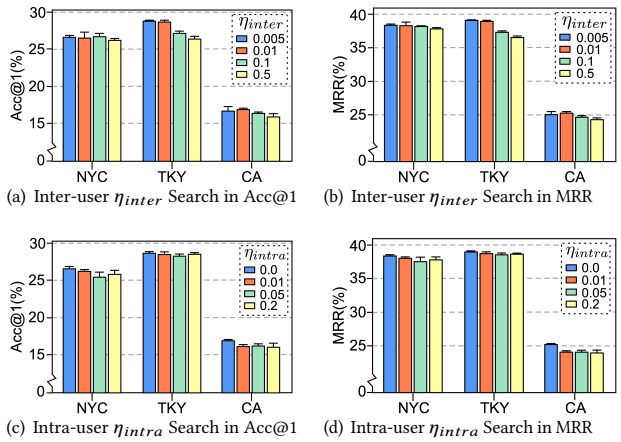
4.1.4 Parameter Settings. The key hyperparameters and training settings of our model are listed below. Adam optimizer is applied with default betas, the learning rate of $1e^{-4}$, and the epoch of 20. The strategy of learning rate decay is also applied in our training procedure to prevent overfitting. The id embedding dimension of all check-ins features d_{id} is searched from $\{32, 64, 128, 256\}$. When building hyperedge, the Jaccard threshold η_{inter} and η_{intra} are selected from $\{0.005, 0.01, 0.1, 0.5\}$ and $\{0.0, 0.01, 0.05, 0.2\}$. When doing hypergraph sampling, the one-hop hyperedge sampling size is tuned from $\{100, 200, 300, 400\}$. Also, we set the check-ins sampling size to the max size of all trajectories in each dataset respectively. Moreover, we employ a 2-layers hypergraph convolution model with a residual beta rate of 0.5 and attention head number of 4.

4.2 Main Results

4.2.1 Overall Performance. We conduct experiments on three datasets to compare the performance between our STHGCN model and baselines. We report the results on Acc@k ($k = 1, 5, 10$) and MRR metrics. The results are shown in Table 1 and the results of baseline models are taken from Song et al [27]. We can see that STHGCN outperforms the baseline models across three datasets in terms of all evaluation metrics. In particular, STHGCN obtains a large improvement of 12.28%, 30.88%, and 27.49% of Top-1 accuracy on NYC, TKY, and CA respectively compared with the-state-of-art baseline GETNext. Additionally, the significant improvement of STHGCN on TKY over competitive models compared with that on NYC and CA further indicates the effectiveness of our hyperedge collaboration mechanism, since Figure 2(b) shows there are more inter-user trajectories in TKY. Furthermore, the superiority of STHGCN on CA demonstrates that our model also shows strong capability to learn from sparse data.

4.2.2 Global Collaborative Effect. Inter-user and intra-user collaborative relations play important roles in our STHGCN model. In order to further analyze the effect of collaborative information in STHGCN, we conduct experiments with different η_{inter} and η_{intra} separately. The larger the value of the η_{inter} and η_{intra}, the more collaborative information is filtered. The global collaborative effect results are shown in Figure 6.

Specifically, Figure 6(a) and 6(b) show that STHGCN performs better when $\eta_{inter} = 0.005$ and $\eta_{inter} = 0.01$, which is particularly evident on TKY. Since there are more inter-user trajectories in TKY, this further demonstrates the effectiveness of our global inter-user collaboration. Furthermore, the result with $\eta_{inter} = 0.01$ on CA shows the best performance. This indicates that a low Jaccard threshold may also introduce noise information along with the collaborative message, which is more likely to have an impact on sparse data. Figure 6(c) and 6(d) present the results with η_{intra}, where we find from the histogram that our model shows overall robustness to the Jaccard threshold η_{intra}, with $\eta_{intra} = 0$ slightly performing better than other η_{intra} values across all three datasets. It can be explained that the historic trajectories of users do make contributions to the prediction of their future behavior.



(a) Inter-user η_{inter} Search in Acc@1 (b) Inter-user η_{inter} Search in MRR

(c) Intra-user η_{intra} Search in Acc@1 (d) Intra-user η_{intra} Search in MRR

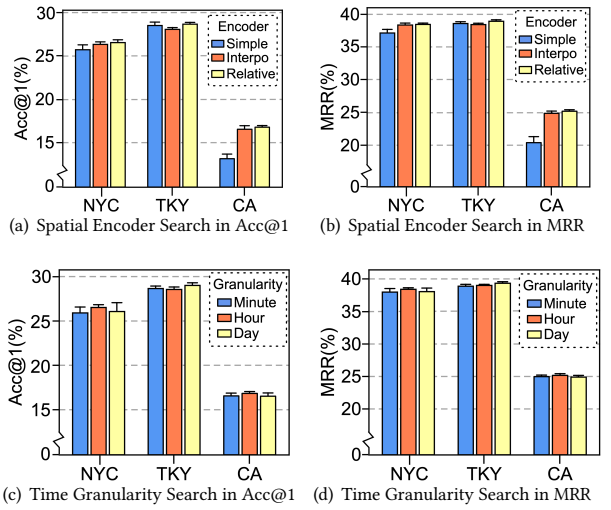
Figure 6: Global collaborative effect analysis. The mean and standard deviation of the test performance under every setting are achieved out of 10 independent runs.

4.2.3 Spatio-temporal Encoder Variants. We further examine the model representation quality on different spatio-temporal encoders. We first conduct experiments on three spatial encoders, which are simple linear encoder in Eq. (4); linear interpolation encoder in Eq. (5), and relative temporal encoder in Eq. (3) respectively. The experimental results on Acc@1 and MRR are shown in Figure 7(a) and 7(b). The results show that the relative temporal encoder and linear interpolation encoder have generally better performance than the simple linear distance encoder, and the simple linear method has relatively worse performance in CA due to the sparsity of POIs in the Gowalla dataset.

Table 1: Performance comparison in Acc@k and MRR on three datasets.

	NYC				TKY				CA			
	Acc@1	Acc@5	Acc@10	MRR	Acc@1	Acc@5	Acc@10	MRR	Acc@1	Acc@5	Acc@10	MRR
FPMC	0.1003	0.2126	0.2970	0.1701	0.0814	0.2045	0.2746	0.1344	0.0383	0.0702	0.1159	0.0911
LSTM	0.1305	0.2719	0.3283	0.1857	0.1335	0.2728	0.3277	0.1834	0.0665	0.1306	0.1784	0.1201
PRME	0.1159	0.2236	0.3105	0.1712	0.1052	0.2278	0.2944	0.1786	0.0521	0.1034	0.1425	0.1002
STGCN	0.1799	0.3425	0.4279	0.2788	0.1716	0.3453	0.3927	0.2504	0.0961	0.2097	0.2613	0.1712
PLSPL	0.1917	0.3678	0.4523	0.2806	0.1889	0.3523	0.4150	0.2542	0.1072	0.2278	0.2995	0.1847
STAN	0.2231	0.4582	0.5734	0.3253	0.1963	0.3798	0.4464	0.2852	0.1104	0.2348	0.3018	0.1869
GETNext	0.2435	0.5089	0.6143	0.3621	0.2254	0.4417	0.5287	0.3262	0.1357	0.2852	0.3590	0.2103
STHGCN	**0.2734**	**0.5361**	**0.6244**	**0.3915**	**0.2950**	**0.5207**	**0.5980**	**0.3986**	**0.1730**	**0.3529**	**0.4191**	**0.2558**
Improvement (%)	12.28	5.34	1.64	8.12	30.88	17.88	13.11	22.19	27.49	23.74	16.74	21.64

We then conduct experiments to explore the time encoder with the influence of different temporal granularities, and the results are shown in Figure 7(c) and 7(d). These results indicate that STHGCN shows relatively strong robustness to temporal granularities. The granularity of hour performs better in NYC and CA and the granularity of day performs better in the TKY dataset, while STHGCN still maintains good overall performance on different temporal granularities.



(a) Spatial Encoder Search in Acc@1 (b) Spatial Encoder Search in MRR

(c) Time Granularity Search in Acc@1 (d) Time Granularity Search in MRR

Figure 7: Comparison of different spatial encoding methods and different time granularities. "Simple" and "Interpo" in (a) and (b) denote the simple linear and linear interpolation spatial encoders, respectively. "Relative" means dealing with spatial distance with the relative temporal encoding method

4.2.4 User cold-start Analysis. For location-based recommendation models, the issue of user cold-start has been recognized as an essential topic. It is usually difficult to predict the following behavior of users lacking historic information. STHGCN conceptually alleviates the cold-start challenge by leveraging collaborative information. To verify the effectiveness of STHGCN on the cold-start challenge, we split the users into three groups: inactive users, normal users, and active users. Generally, we count the number of trajectories of each user in the training set. We then define the top 30% as the group of active users, the bottom 30% as the group of inactive users, and the rest as the group of normal users.

Table 2: Results of user cold-start on NYC and TKY dataset.

		NYC		TKY	
User Groups	Model	Acc@1	MRR	Acc@1	MRR
Inactive	GETNext	0.1473	0.2319	0.1225	0.2133
Normal	GETNext	0.3108	0.4254	0.1652	0.2687
Very active	GETNext	0.2577	0.3933	0.2642	0.3859
Inactive	Ours	0.1460	0.2247	0.2164	0.3053
Normal	Ours	0.3050	0.4265	0.2659	0.3596
Very active	Ours	0.3085	0.4402	0.3464	0.4618

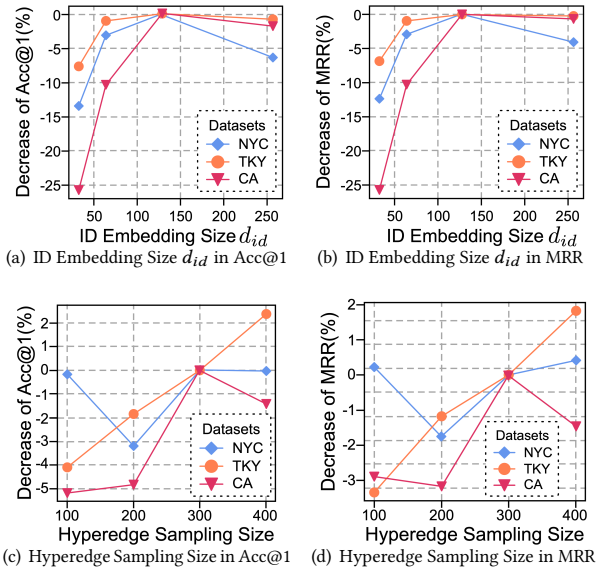
We take GETNext as the baseline model since it has achieved high predicting accuracy on inactive users [27]. Table 2 presents the comparison results of STHGCN and the baseline method. In the NYC dataset, the improvement of STHGCN is limited since there are no sufficient hyperedge collaborations, but the results are still comparable with the baseline model. In the TKY dataset, our model achieves a significantly great performance of all three groups and gains huge relative improvements of beyond 20% in all metrics compared with the baseline model. The main reason for the significant improvements in the TKY dataset is that the number of trajectories is much larger than those in the NYC dataset, which means more collaborative information facilitates prediction accuracy.

4.2.5 Behavior Diversity Analysis. The trajectory defined in STHGCN represents a user's continuous check-ins sequence over a while. Intuitively, short trajectories consisting of a small number of check-ins have limited spatiotemporal messages and lead to poor performance. The motivation of STHGCN is to merge effective messages of similar trajectories by hypergraph. The collaborative hyperedge is a kind of additional message except for the historic message of users. We divide the trajectories in the test set by length and mark the top 30% as long trajectories and the bottom 30% as short trajectories.

Table 3: Behavior diversity analysis on NYC and TKY dataset.

		NYC		TKY	
Trajectoy Type	Model	Acc@1	MRR	Acc@1	MRR
Short	GETNext	0.2407	0.3570	0.1921	0.2914
Middle	GETNext	0.2524	0.3732	0.2060	0.3215
Long	GETNext	0.2419	0.3695	0.2428	0.3650
Short	Ours	0.2703	0.3783	0.2787	0.3710
Middle	Ours	0.2545	0.3795	0.2823	0.3850
Long	Ours	0.3184	0.4401	0.3116	0.4246

We take GETNext as the baseline model, and Table 3 shows the user behavior analysis results of STHGCN and GETNext on NYC and TKY datasets. Our method outperforms the baseline across all the groups. Specifically, our method has a significant improvement in the group of short trajectories. The Top-1 accuracy improvement is 12.3% in the NYC dataset and 45.08% in the TKY dataset over the baseline. Moreover, the performance of our model on short trajectories shows significant superiority to the baseline model, which indicates the strong capability of our STHGCN to learn collaborative information from short trajectories.



(a) ID Embedding Size d_{id} in Acc@1 (b) ID Embedding Size d_{id} in MRR

(c) Hyperedge Sampling Size in Acc@1 (d) Hyperedge Sampling Size in MRR

Figure 8: Hyperparameter Sensitivity Study.

4.2.6 Hyperparameter Sensitivity Study. We select two key hyperparameters to study the sensitivity of STHGCN: the id embedding dimension of check-ins features d_{id} and the hyperedge sampling size. Figure 8 presents the result of the sensitivity study in all three datasets. Figure 8(a) and Figure 8(b) show that the best performance can be achieved with $d_{id} = 128$. Small d_{id} limits the modeling expressiveness while larger d_{id} leads to redundant model parameters with performance degradation.

Figure 8(c) and 8(d) show the impact of hyperedge sampling size on the overall performance of SYHGCN, where STHGCN shows different influences with hyperedge sampling size on different datasets. In TKY, STHGCN performs better as the parameter increases, since the number of collaborative hyperedges is large in TKY. In NYC and CA, STHGCN performs best at hyperedge sampling sizes of 100 and 300, respectively. Nevertheless, our model still remains robust to different hyperedge sampling sizes with the maximum decrease rate of -5%.

4.3 Ablation Study

We consider that the superiority of STHGCN can be seen as the result of the joint effect of 1) hypergraph modeling; 2) spatio-temporal information; 3) hyperedge collaboration.

To investigate the contributions of each module in STHGCN, we replace hypergraph modeling with Transformer, and remove the spatio-temporal information and hyperedge collaboration component respectively. As is shown in Table 4.

Table 4: Ablation study result over three components on NYC and TKY.

	NYC		TKY	
	Acc@1	MRR	Acc@1	MRR
Full Model	**0.2734**	**0.3915**	**0.2950**	**0.3986**
w/o Hypergraph	0.2579	0.3757	0.2365	0.3330
w/o ST Information	0.2318	0.3594	0.2600	0.3643
w/o Hyperedge Collaboration	0.2550	0.3695	0.2496	0.3479

As can be observed in Table 4, each component impacts the full model performance to a certain extent, which indicates the necessity of STHGCN to incorporate these mechanisms. Furthermore, it is worth noting that the result without hypergraph modeling decreased significantly on TKY (-19.83%) compared with that on NYC (-5.67%). The main reason could be found in Figure 2(b), where the number of inter-user trajectories in TKY was very large. Therefore, it illustrates that the hypergraph modeling component efficiently learns the trajectory collaborations and significantly improves the model performance.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose STHGCN, which is as far as we know the first work utilizing hypergraph convolution networks to model diverse user behaviors in the next-POI recommendation task. In particular, we introduce a hypergraph to construct the complex structure of check-ins and trajectories. We develop hypergraph transformer layers to capture high-order heterogeneous inter-user and intra-user trajectories correlations while incorporating spatio-temporal contexts. Comprehensive experiments conducted on three real-world datasets have demonstrated the superiority of STHGCN in the task of next-POI recommendation, outperforming baseline models by a large margin.

Looking to the future, we will investigate incorporating multi-view trajectory information, such as breaking trajectories into Area-Of-Interest (AOI) segments within a specified spatial distance interval.

REFERENCES

[1] Glen Robert Brummelen. 2013. Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry. *Princeton University Press* (2013).

[2] Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. 2021. You are allset: A multiset function framework for hypergraph neural networks. *arXiv preprint arXiv:2106.13264* (2021).

[3] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1082–1090.

[4] 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 *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[5] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2018. Hypergraph Neural Networks. *AAAI 2019* (2018).

[6] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*. PMLR, 1263–1272.

[7] 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 *(AAAI'16)*. 137–143.

[8] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. In *Proceedings of The Web Conference 2020 (WWW '20)*. 2704–2710.

[9] Dejiang Kong and Fei Wu. 2018. HST-LSTM: A Hierarchical Spatial-Temporal Long-Short Term Memory Network for Location Prediction. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*. 2341–2347.

[10] 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 International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 1133–1143.

[11] 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 *Proceedings of the 29th ACM International Conference on Information amp; Knowledge Management (CIKM '20)*. 845–854.

[12] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. STAN: Spatio-Temporal Attention Network for Next Location Recommendation. In *Proceedings of the Web Conference 2021 (WWW '21)*. 2177–2185.

[13] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-Basket Recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW '10)*. 811–820.

[14] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo *(ICML '08)*. 880–887.

[15] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. 2021. Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. 1548–1554. Main Track.

[16] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Nguyen Hung, and Hongzhi Yin. 2020. Where to Go Next: Modeling Long- and Short-Term User Preferences for Point-of-Interest Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 214–221.

[17] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2019. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082* (2019).

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[19] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-Item Recommendation with Sequential Hypergraphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*. 1101–1110.

[20] Xiaolin Wang, Guohao Sun, Xiu Fang, Jian Yang, and Shoujin Wang. 2022. Modeling Spatio-temporal Neighbourhood for Personalized Point-of-interest Recommendation. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. 3530–3536.

[21] Zhaobo Wang, Yanmin Zhu, Haobing Liu, and Chunyang Wang. 2022. Learning Graph-Based Disentangled Representations for Next POI Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 1154–1163.

[22] Yuxia Wu, Ke Li, Guoshuai Zhao, and Xueming Qian. 2020. Personalized long-and short-term preference learning for next POI recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34, 4 (2020), 1944–1957.

[23] Lianghao Xia, Chao Huang, Yong Xu, Jiashu Zhao, Dawei Yin, and Jimmy Huang. 2022. Hypergraph Contrastive Collaborative Filtering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 70–79.

[24] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. 2021. Self-Supervised Hypergraph Convolutional Networks for Session-based Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (May 2021), 4503–4511.

[25] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. *HyperGCN: A New Method of Training Graph Convolutional Networks on Hypergraphs.*

[26] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. 2014. Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2014), 129–142.

[27] Song Yang, Jiamou Liu, and Kaiqi Zhao. 2022. GETNext: Trajectory Flow Map Enhanced Transformer for Next POI Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. 1144–1153.

[28] Yuhao Yang, Chao Huang, Lianghao Xia, Yuxuan Liang, Yanwei Yu, and Chenliang Li. 2022. Multi-Behavior Hypergraph-Enhanced Transformer for Sequential Recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*. 2263–2274.

[29] Lu Zhang, Zhu Sun, Ziqing Wu, Jie Zhang, Yew Soon Ong, and Xinghua Qu. 2022. Next Point-of-Interest Recommendation with Inferring Multi-step Future Preferences. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. 3751–3757.

[30] Pengpeng Zhao, Anjing Luo, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S Sheng, and Xiaofang Zhou. 2020. Where to go next: A spatio-temporal gated network for next poi recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34, 5 (2020), 2512–2524.