MCN4Rec: Multi-Level Collaborative Neural Network for Next Location Recommendation

Anonymous Author(s)

ABSTRACT

Next location recommendation plays an important role in various location-based services, yielding great value for both users and service providers. Existing methods usually model temporal dependencies with explicit time intervals or learn representation from customized point of interest (POI) graphs with rich context information to capture the sequential patterns among POIs. However, this problem is perceptibly complex because various factors, *e.g.*, users' preferences, spatial locations, time contexts, activity category semantics, and temporal relations, need to be considered together, while most studies lack sufficient consideration of the collaborative signals. Toward this goal, we propose a novel Multi-Level Collaborative Neural Network for next location Recommendation (MCN4Rec). Specifically, we design a multi-level view representation learning with level-wise contrastive learning to collaboratively learn representation from local and global perspectives to capture complex heterogeneous relationships among user, POI, time, and activity categories. Then a causal encoder-decoder is applied to the learned representations of check-in sequences to recommend the next location. Extensive experiments on two real-world check-in mobility datasets demonstrate that our model significantly outperforms the existing state-of-the-art baselines for next location recommendation. Ablation study further validates the benefits of the collaboration of the designed sub-modules. The source code is available at https://anonymous.4open.science/r/MCN4Rec/.

KEYWORDS

Next Location Recommendation, Human Mobility, Representation Learning, Collaborative Learning

ACM Reference Format:
Anonymous Author(s). 2023. MCN4Rec: Multi-Level Collaborative Neural Network for Next Location Recommendation. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30 - May 4, 2023, Austin, Texas, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/xxxxxxx.xxxxxxx

1 INTRODUCTION

The popularity of location-based social networks (LBSN), such as Foursquare and Yelp, has attracted a large number of users to share their experiences on points of interest (POI) through mobile devices, thus generating a large number of user mobility check-in data. As a result, these large amounts of mobility data give rise to increasingly powerful next location recommendation systems [3]. Such systems aim to predict the next location a given user will be most likely to visit based on his/her current and historical footprints (*i.e.*, check-ins) [32]. These location recommendations are beneficial to both users and businesses. For example, an effective location recommendation can help users save the time and effort of searching for places of interest. On the other hand, they can also assist businesses in acquiring potential new customers and thus enable enterprises to achieve profit growth [11].

Existing methods towards this goal, have made significant efforts on next location prediction based on Markov chains, recurrent neural network (RNN) model, attention-based model, and graph-based model. Early studies mainly focus on sequential transition modeling, such as Markov chains [4, 24]. Based on the great success of RNN model in sequence modeling, a handful of RNN model-based location recommendation models [6, 19, 25, 36, 38, 39] have been developed to better capture the sequential regularities of use trajectories with rich contexts. Besides, self-attention network-based models [8, 17, 20, 22] have been proposed for location recommendation due to the great success of Transformer [26] in NLP field. They also stack extra modules to integrate additional information [8, 18, 20, 33]. Recently, graph-based models [9, 15, 23] have been proposed to learn POI representation for recommendation by capturing the users' visit preferences and/or POI transition patterns.

Despite the effectiveness of existing next location recommendation methods [17, 20, 22, 23, 33], they have the following limitations that significantly affect their effectiveness. *First, the correlations among user, POI, time, and activity semantics in user check-in records have not been learned effectively.* Many literatures on location recommendation systems have proved the potential of complex joint effects in improving recommendation accuracy. We note that, in the spatio-temporal recommendation, most existing studies have focused on modeling the sequence dependence of the trajectory. Although some methods have considered the impact of superficial relationships, few pay attention to comprehensively capturing multiple complex heterogeneous relationships, *e.g.*, users' preferences towards different POIs, alignment of POIs in different perspectives, and uniformity retention of POIs with their essential characteristics. *Second, most approaches suffer from the data sparsity issue in user mobility check-in data.* Compared with classic items, such as goods or songs, the data sparsity issue is particularly severe in the user check-in dataset [1]. The performance of most models drops significantly on short check-in trajectories compared to long check-in trajectories. *Third, the cold-start problem is common in real-life recommendation systems, but previous works extensively overlooked this problem.* The cold start problem is mainly caused by inactive users or new users. According to the statistics on Foursquare, the number of check-ins of 72.29% users is less than 100 in one year, indicating that there are a large number of inactive users with few check-in records. Due to the limited trajectory length, the provided scant spatiotemporal semantics lead to insufficient sequence modeling, which dramatically harms the recommendation performance.

WWW'23, April 30 - May 4, 2023, Austin, Texas, USA
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-xxxx-x/xx/xx…$15.00
https://doi.org/10.1145/xxxxxxx.xxxxxxx

To tackle the aforementioned challenges, we propose a **M**ulti-Level **C**ollaborative Neural **N**etwork for next location **Rec**ommendation, called MCN4Rec. To alleviate the pain of cold start, it incorporates a multi-level representation learning module by modeling the overall historical trajectory, and uses rich cooperation signals to learn the preferences of various users and the semantic information of POIs. To capture complex heterogeneous relationships, it contains a component of multi-level view representing learning to model check-in data from multiple perspectives, a component of level-wise contrastive representation learning to encourage learning to a consistent semantic space, and a component of category-aware temporal representation to incorporate POI categories into time embeddings. Then, the representations of the check-in sequence are sent to a causal encoder to further learn the sequence movement pattern, and the inference of the next location preference is constrained by multiple decoders. Experimental results on two real-world mobility datasets show that our model significantly outperforms several strong baselines (8.51% Acc@1 gain and 6.45% MRR gain on average) in next location recommendation. We also demonstrate the benefits of the collaboration of the designed sub-modules through an ablation study. We further illustrate the capability of our MCN4Rec model in handling cold-start issue on short trajectories and check-in data of inactive users.

The main contributions of this work are summarized as follows:

- We propose a novel Multi-Level Collaborative Neural Network model, MCN4Rec, to solve data sparsity, cold-start problem, and complex correlation learning issues for next location recommendation. Our model effectively leverages various rich cooperation signals in user mobility data to improve model performance.
- We design a novel multi-level contrastive representation learning framework. With the carefully-designed three sub-components, our model achieves effective representation learning for check-in data, capturing complex multiple heterogeneous interaction relations in user mobility data.
- Extensive experiments on two real-life mobility check-in datasets are performed. Results show that our model significantly outperforms state-of-the-art baselines by average 8.51% and 6.45% in terms of Acc@1 and MRR.

2 RELATED WORK

2.1 Next Location Recommendation

Most of the next location recommendation methods are based on sequence models, ranging from simple Markov chains to popular recurrent neural networks, as well as recently dominated methods based on self-attention models. Early works generally try to use the Markov chain-based method [4, 13, 24] to learn the transition probability between consecutive check-in, and some works use matrix factorization [14] and metric embedding [7] to alleviate the sparsity of check-in data. *However, these methods can not model high-order sequential patterns because they mainly learn the conversion rules between consecutive check-in.* Inspired by the success of RNNs in modeling sequence data, RNN-based methods have become popular. STRNN [19], STGN [38], and ASPPA [36] expand the gate mechanism in the recurrent neural network to learn users' potential interests and preferences based on time interval, geographical distance, subsequence patterns, and other factors. DeepMove [6], LSTPM, [25], and ST-PIL [5] propose spatio-temporal periodic interest learning networks with different types of long- and short-term modules, which use historical trajectories to capture users' long-term and short-term movement preferences. GeoSAN [17] and STAN [20] use well-designed transformer networks to model from the perspective of spatial and non-spatial proximity, respectively. AutoMTN [22] replaces the self-attention with auto-correlation in each channel and aggregates information at the sub-sequence level. CF-Rec [35] combines LSTM and Transformer encoder to help infer user future preference in a self-ensembling manner. *Nevertheless, all the above work only focuses on modeling the user's sequence pattern but ignores the multi factors cooperation signals of LBSN, so it cannot comprehensively capture multiple complex heterogeneous relationships in spatio-temporal recommendations.*

2.2 Multi Factors of LBSN for Recommendation

Since the data of LBSN contains a variety of complex heterogeneous relationships, some studies [1, 2, 9, 16, 21, 34] for location recommendation try to mine user preferences from user relationship influence, activity category influence, time influence, geographic influence, or partial combined information, to alleviate problems such as data sparsity and cold start. Graph-based models have natural advantages in modeling complex heterogeneous relationships and have recently been introduced into the field of location recommendation. SGRec [15], DRAN [33], and GETNext [30] pay attention to the intrinsic characteristics of POI, and consider the collaborative signals from cross-user check-in sequences to construct a graph learning module, infers the general preferences of users, and overcomes the sparsity of POI-level interactions. STGCAN [23] and Graph-Flashback [29] introduce knowledge graphs to address the limitations of previous methods for capturing heterogeneous relations. *Still and all, the above methods more or less ignore some factors of LBSN and do not explicitly emphasize the collaboration of multiple factors. In this work, we are the first attempt to comprehensively consider all potential factors and build a multi-level collaborative neural network to capture its cooperative signals.*

3 PROBLEM DEFINITION

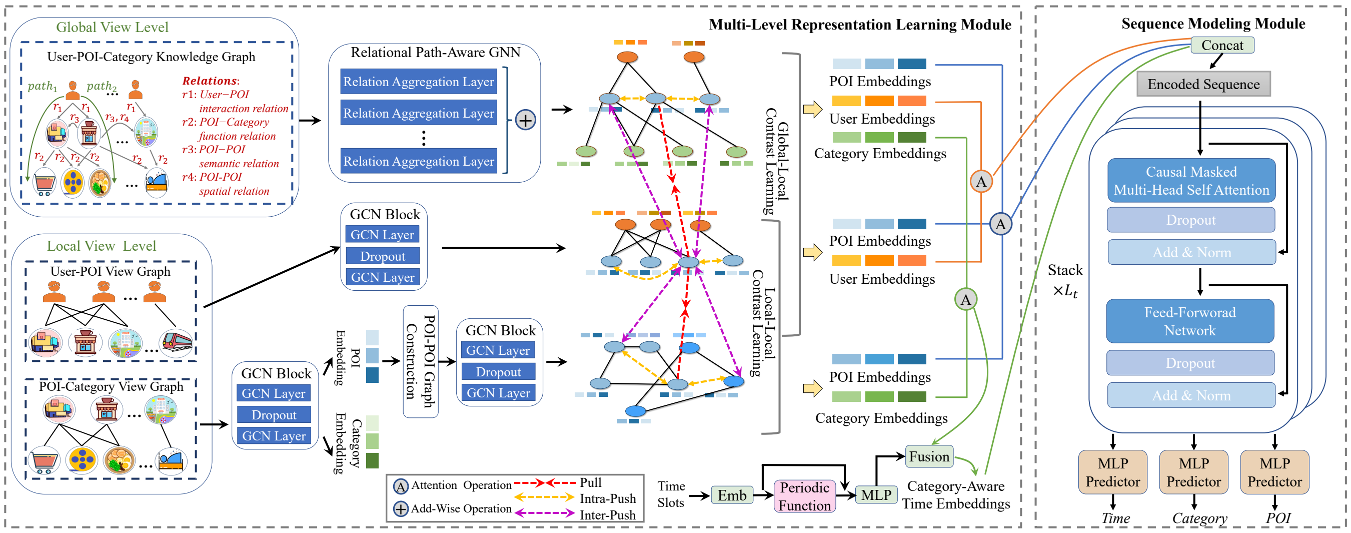
In this section, we first clarify key concepts used in this paper and then formally define the studied problem.

Let $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ represent the set of all users, $\mathcal{P} = \{p_1, p_2, ..., p_{|\mathcal{P}|}\}$ represent the set of all POIs and $C = \{c_1, c_2, ..., c_{|C|}\}$ denote the set of all activity categories.

Definition 1 (Check-in Record). *A check-in record γ is usually a quadruple $< u, p, c, t >$ that represents user u visiting a POI p with activity category c at time t, where $u \in \mathcal{U}, p \in \mathcal{P}, c \in C$.*

Definition 2 (Visit Sequence). *A visit sequence contains a complete check-in records $(\gamma_1, \gamma_2, ..., \gamma_n)$ generated by user u in chronological order during all recorded time, which is denoted by Tr_u.*

Definition 3 (Trajectory). *A meaningful session-based trajectory $Tr_u^s = \{\gamma_1, \gamma_2, ..., \gamma_m\}$ is a segment of visit sequence Tr_u, if $0 < t_{i+1} - t_i \leq \Delta t, \forall 1 \leqslant i < m$, where Δt is the pre-defined maximum time threshold. For the k-th session, we can regard it as the current*



Global View Level
User-POI-Category Knowledge Graph
path1
path2
Relations:
r1: User−POI interaction relation
r2: POI−Category function relation
r3: POI−POI semantic relation
r4: POI-POI spatial relation
Relational Path-Aware GNN
Relation Aggregation Layer
Relation Aggregation Layer
Relation Aggregation Layer
Local View Level
User-POI View Graph
POI-Category View Graph
GCN Block
GCN Layer
Dropout
GCN Layer
POI Embedding
Category Embedding
POI-POI Graph Construction
Multi-Level Representation Learning Module
Global-Local Contrast Learning
Local-Local Contrast Learning
POI Embeddings
User Embeddings
Category Embeddings
Attention Operation
Add-Wise Operation
Pull
Intra-Push
Inter-Push
Time Slots
Emb
Periodic Function
MLP
Fusion
Category-Aware Time Embeddings
Sequence Modeling Module
Concat
Encoded Sequence
Causal Masked Multi-Head Self Attention
Dropout
Add & Norm
Feed-Forword Network
Stack ×Lt
MLP Predictor
Time
Category
POI

Figure 1: The overall architecture of our proposed MCN4Rec.

trajectory $Tr_u^{s_k}$, *while contain the previous* $k-1$ *session list is regarded as the historical trajectory* $\{Tr_u^{s_j}|1 \leq j < k\}$.

Problem 1 (Next Location Recommendation). *The basic input of a next location recommendation is the partially known session information: the current trajectory* $Tr_u^{s_k} = \{\gamma_1, \gamma_2, ..., \gamma_{m-1}\}$ *of a user* $u \in \mathcal{U}$, *and a set of historical trajectories* $\{Tr_{u_i}^{s_j}|u_i \in \mathcal{U}, 1 \leq j < k\}$ *of all users. Our goal is to predict where the user* u *is most likely to visit by maximizing the utility score conditioned on a given context:*

$$\hat{\gamma}_m = \arg\max f(Tr_u^{s_k}, \{Tr_{u_i}^{s_j}|u_i \in \mathcal{U}, 1 \leq j < k\})$$

where $\hat{\gamma}_m$ *denotes the predicted next POI.*

4 METHODOLOGY

The overall architecture of the proposed MCN4Rec is presented in Figure 1. MCN4Rec consists of four major components: *multi-level view representation learning*, *level-wise contrastive representation learning*, *category-aware temporal representation*, and *sequence modeling module*. *Multi-level view representation learning* builds a knowledge graph to model various relationships among users, POIs, and activity categories, and learn their representation vectors through different embedding technologies from the global and local views, respectively. *Level-wise contrastive representation learning* uses local-level and global-level contrastive learning to constrain node representation learning between different local views and between global and local views to maximize the agreement of representations across different views. *Category-aware temporal representation* uses time2vec [10] to learn time periodicity and incorporates activity category semantics into temporal representation. The learned representations are used for sequence modeling for next location recommendation in *sequence modeling module*. We will describe the method in detail as follows.

4.1 Multi-Level View Representation Learning

In this section, we present the method to learn the embeddings of users, POIs and categories from multi-level views. Our method models various relations among users, POIs, and categories by building a related knowledge graph, and then obtains their embedding by performing different graph neural networks from the local and global views, respectively.

4.1.1 *User-POI-Category Knowledge Graph.*

We first design a novel knowledge graph called User-POI-Category Knowledge Graph (UPC-KG) as shown in Figure 1. UPC-KG integrates the traditional User-POI interaction and POI-Category relation with semantic relationships among POIs. Formally, UPC-KG is a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where $\mathcal{V}$ is the union of user set, POI set, and category set (*i.e.*, $\mathcal{U} \bigcup \mathcal{P} \bigcup \mathcal{C}$), and $\mathcal{E}$ is the union of edge sets of different types between $\mathcal{V}$. $\mathcal{R}$ is the set of relationship triples, each triple (h_e, r, t_e) representing a relationship from head entity h_e to tail entity t_e ($h_e, t_e \in \mathcal{V}$). In UPC-KG, we construct four types of relations respectively, and each edge belongs to one of the following four relations:

- **User-POI interaction relation**. It represents user's visiting behavior to POIs in the check-in activities.
- **POI-Category functional relation**. It depicts the functional mode carried by POIs, precisely the specific categories of each POI.
- **POI-POI semantic relation**. It indicates that different POIs can provide the same service, which is reflected by the co-occurrence of POI-Category-POI.
- **POI-POI spatial relation**. It represents the geospatial connection between POIs; specifically, two POIs have adjacent geographic locations.

4.1.2 *Relational Path-Aware Aggregation.*

The commonly used graph neural networks aggregate multi-hop neighbor information into the representation through the neighborhood aggregation scheme. Since they focus on aggregating neighborhood information, thus cannot distinguish the paths from which these neighbors come. To capture user-POI interactions with multiple semantic relations in UPC-KG, inspired by [28], we design a relational path-aware aggregation mechanism in GNN. The mechanism incorporates User-POI-Category connectivity from the global perspective in UPC-KG into the embeddings of user, POI, and category entities.

A complete path (*e.g.*, u_i-p_j-c_k) usually represents the underlying intention of a user activity. To capture the intentions, the relational path-aware GNN performs multiple-layer aggregations on UPC-KG. The aggregation process in the l-th layer is expressed as:

$$e_u^{(l+1)} = \frac{1}{|\mathcal{N}_u|} \sum_{(r,p)\in\mathcal{N}_u} e_r \odot e_p^{(l)}, \tag{1}$$

$$e_p^{(l+1)} = \frac{1}{|\mathcal{N}_p|} \sum_{(r,c)\in\mathcal{N}_p} \eta(p,r,c) e_r \odot e_c^{(l)} + \frac{1}{\left|\mathcal{N}_p^{sp}\right|} \sum_{(r,p)\in\mathcal{N}_p^{sp}} e_r \odot e_p^{(l)} + \frac{1}{\left|\mathcal{N}_p^{se}\right|} \sum_{(r,p)\in\mathcal{N}_p^{se}} e_r \odot e_p^{(l)}, \tag{2}$$

$$e_c^{(l+1)} = \frac{1}{|\mathcal{N}_c|} \sum_{(r,p)\in\mathcal{N}_c} e_r \odot e_p^{(l)}, \tag{3}$$

$$\eta(p,r,c) = \frac{\exp\left(\left(e_p||e_r\right)^{\mathsf{T}} \cdot \left(e_c||e_r\right)\right)}{\sum_{(r',c')\in\mathcal{N}_p} \exp\left(\left(e_p||e_{r'}\right)^{T} \cdot \left(e_{c'}||e_{r'}\right)\right)}, \tag{4}$$

where $e_u^{(l)}$, $e_p^{(l)}$ and $e_c^{(l)}$ are the representations of user u, POI p and category c for the l-th layer, respectively. $||$ denotes the merge operation (*e.g.*, concatenation). $\mathcal{N}_u$, $\mathcal{N}_p$ and $\mathcal{N}_c$ represent the corresponding neighbors, including connected relations and nodes. $\mathcal{N}_p^{sp}$ and $\mathcal{N}_p^{se}$ denote the neighbors with POI-POI spatial and semantic relations, respectively. $\eta(p,r,c)$ is the attention weight of category c relative to POI p, which indicates the importance of different categories to the aggregation of POI representation.

We then sum the outputs of all layers to get the representations e_{upc}^u, e_{upc}^p and e_{upc}^c for user, POI and category nodes in the global view, respectively:

$$\begin{aligned} e_{upc}^u &= e_u^{(0)} + \cdots + e_u^{(L_g)}, \\ e_{upc}^p &= e_p^{(0)} + \cdots + e_p^{(L_g)}, \\ e_{upc}^c &= e_c^{(0)} + \cdots + e_c^{(L_g)}. \end{aligned} \tag{5}$$

where L_g is the number of aggregation layers in the relational path-aware GNN.

4.1.3 *Representation Learning from Local View Graphs.*

To strengthen the relationship between users and POIs, and between POIs and categories, we further extract two important local views from UPC-KG: User-POI view graph and POI-Category view graph. The User-POI view contains the explicit interactions between users and POIs, which implies the users' spatial preference, the co-occurrence patterns among users, and the collaborative relationships between POIs. The POI-Category view plays a role at the semantic activity level, which retains the category correlations and reflects the semantic commonality between POIs.

To capture the correlations between users and POIs, and between POIs and categories, we further employ Graph Convolutional Network (GCN) [12] to learn embeddings of users, POIs and categories on the local User-POI and POI-Category view graphs, respectively. Specifically, we define $\mathbf{A}_{up}$ and $\mathbf{A}_{pc}$ as the adjacency matrices of the two local view graphs. We adopt a multi-layer GCN with the following message-passing rule:

$$\mathbf{H}^{(l)} = \mathrm{ReLU}\left(\widetilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \widetilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(l-1)} \mathbf{W}^{(l)} + b^{(l)}\right), \tag{6}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$, $\mathbf{A}$ is $\mathbf{A}_{up}$ or $\mathbf{A}_{pc}$, $\tilde{\mathbf{D}}$ is the degree matrix of $\tilde{\mathbf{A}}$, $\mathbf{H}^{(0)}$ can be the feature matrix of the input nodes or identity matrix, ReLU is the Relu activation function, $\mathbf{W}^{(l)}$ represents the weight parameter matrix of the l-th layer, and $b^{(l)}$ is the corresponding bias of the l-th layer. One can enhance the representational power of the model by stacking L GCN layers. The output of the last layer of GCN is:

$$e^* = \widetilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \widetilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{L-1} \mathbf{W}^{L} + b^{L}. \tag{7}$$

e^* can be e_{up}^u, e_{up}^p/e_{pc}^p, e_{pc}^c, which are the embeddings of users, POIs, and categories learned from two local view graphs, respectively.

Furthermore, to strengthen the semantic relevance of POI-POI, we construct a POI-POI semantic graph based on the learned POI embeddings e_{pc}^p from the POI-Category local view graph. To be specific, we first compute the correlation between each pair of POIs based on the cosine similarity of their embeddings. Then we perform a kNN-based sparsification operation (*i.e.*, The k most similar neighbors of each POI are kept.) to obtain a sparse POI-POI semantic graph. After that, a multi-layer GCN is then utilized to update the embeddings of POI nodes, denoted by e_{pcp}^p.

4.2 Level-Wise Contrastive Representation Learning

Contrastive learning has demonstrated its superiority. Inspired by that, we propose a level-wise contrastive learning framework, which consists of the local-level and global-level contrastive learning to empower the representation learning ability of the model that maximizes the agreement of representations learned across different views while capturing different relationships.

4.2.1 *Local-Level Contrastive Learning.*

The local-level contrastive learning method is designed between different local views. For any node, its learned embedding in one local view is regarded as the anchor, and the embedding of the same node learned in another local view is taken as a positive sample, while the embeddings of other non-connected nodes are taken as negative samples in both local views.

With the view-specific embeddings $e_{up}^{p_i}$ and $e_{pcp}^{p_i}$ of the i-th POI from the User-POI view and the POI-Category view, we have the following cross-view contrastive loss in local level:

$$\ell(e_{up}^{p_i}, e_{pcp}^{p_i}) = \log \frac{\varphi\left(e_{up}^{p_i}, e_{pcp}^{p_i}\right)}{\sum\limits_{k\neq N_i} \varphi\left(e_{up}^{p_i}, e_{up}^{p_k}\right) + \sum\limits_{k\neq N_i} \varphi\left(e_{up}^{p_i}, e_{pcp}^{p_k}\right)}, \tag{8}$$

where $\varphi(e_i, e_j) = \exp(\mathrm{sim}(g(e_i), g(e_j))/\tau)$ and τ is the temperature parameter. $\mathrm{sim}(\cdot,\cdot)$ is the cosine similarity function, and $g(\cdot)$ is the nonlinear projection operation, which is implemented through a two-layer perceptron model. In the denominator, $(e_{up}^{p_i}, e_{up}^{p_k})$ represents a pair of intra-view negative samples, and $(e_{up}^{p_i}, e_{pcp}^{p_k})$ represents a pair of inter-view negative samples in local level.

Since two local views are symmetric, the loss is defined similarly for $\ell(e_{pcp}^{p_i}, e_{up}^{p_i})$. The overall loss function in local level to be minimized is denoted as the average over all positive pairs:

$$\mathcal{L}_{local}^{cl} = \frac{1}{2|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} (\ell(e_{up}^{p_i}, e_{pcp}^{p_i}) + \ell(e_{pcp}^{p_i}, e_{up}^{p_i})). \tag{9}$$

4.2.2 Global-Level Contrastive Learning. The global-level contrastive learning method is designed between the global view and local views. With the embeddings of the i-th POI in the global view and the local view, *i.e.*, $e_{upc}^{p_i}$ and $e_{local}^{p_i}$, we adopt the following contrastive loss between global and local views for each positive pair:

$$\ell(e_{upc}^{p_i}, e_{local}^{p_i}) = \log \frac{\varphi\left(e_{upc}^{p_i}, e_{local}^{p_i}\right)}{\sum_{k \neq N_i} \varphi\left(e_{upc}^{p_i}, e_{upc}^{p_k}\right) + \sum_{k \neq N_i} \varphi\left(e_{upc}^{p_i}, e_{local}^{p_k}\right)}, \tag{10}$$

where $e_{local}^{p_i}$ can be $e_{up}^{p_i}$ or $e_{pcp}^{p_i}$.

Similarly, the local-global across-view loss can be defined for user and category nodes. Then, the overall loss function in global-level contrastive learning is obtained as follows:

$$\begin{aligned} \mathcal{L}_{global}^{cl} = & \frac{1}{2|\mathcal{U}|} \sum_{i=1}^{|\mathcal{U}|} (\ell(e_{upc}^{u_i}, e_{up}^{u_i}) + \ell(e_{up}^{u_i}, e_{upc}^{u_i})) \\ & + \frac{1}{2|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} (\ell(e_{upc}^{p_i}, e_{up}^{p_i}) + \ell(e_{up}^{p_i}, e_{upc}^{p_i})) \\ & + \frac{1}{2|\mathcal{P}|} \sum_{i=1}^{|\mathcal{P}|} (\ell(e_{upc}^{p_i}, e_{pcp}^{p_i}) + \ell(e_{pcp}^{p_i}, e_{upc}^{p_i})) \\ & + \frac{1}{2|C|} \sum_{i=1}^{|C|} (\ell(e_{upc}^{c_i}, e_{pc}^{c_i}) + \ell(e_{pc}^{c_i}, e_{upc}^{c_i})). \end{aligned} \tag{11}$$

4.2.3 Attention-based Representation Fusion. We use the attention mechanism to aggregate the importance of corresponding embeddings in local and global levels to obtain more powerful representations:

$$\begin{aligned} \alpha_{e_{upc}^u}, \alpha_{e_{up}^u} &= \mathrm{Attn}\left(e_{upc}^u, e_{up}^u\right), \\ \alpha_{e_{upc}^p}, \alpha_{e_{up}^p}, \alpha_{e_{pcp}^p} &= \mathrm{Attn}\left(e_{upc}^p, e_{up}^p, e_{pcp}^p\right), \\ \alpha_{e_{upc}^c}, \alpha_{e_{pc}^c} &= \mathrm{Attn}\left(e_{upc}^c, e_{pc}^c\right), \end{aligned} \tag{12}$$

where $\alpha_*^* \in \mathbb{R}^{n \times 1}$ are the attention values of the corresponding embeddings.

Taking the i-th POI node as an example, its embedding is $e_{up}^{p_i} \in \mathbb{R}^{1 \times d}$ in User-POI local view. We use the softmax function to normalize the attention values on all views to get the final attention weights:

$$\begin{aligned} \alpha_{e_{up}^p}^i &= \mathrm{softmax}\left(e_{up}^{p_i}\right) \\ &= \frac{\exp\left(\omega(e_{up}^{p_i})\right)}{\exp\left(\omega(e_{up}^{p_i})\right) + \exp\left(\omega(e_{pcp}^{p_i})\right) + \exp\left(\omega(e_{upc}^{p_i})\right)}, \end{aligned} \tag{13}$$

where $\omega(e_{up}^{p_i}) = Q^{\mathsf{T}} \cdot \tanh\left(\mathbf{W} \cdot \left(e_{up}^{p_i}\right)^{\mathsf{T}} + \mathrm{b}\right)$ is the corresponding attention value in User-POI local view, which is implemented using a nonlinear transformation (*i.e.*, tanh) with a shared attention vector $Q \in \mathbb{R}^{d' \times 1}$. $\mathbf{W} \in \mathbb{R}^{d' \times d}$ is the weight matrix and $b \in \mathbb{R}^{d' \times 1}$ is the bias vector.

By this means, we can obtain the attention weights of different views: $\alpha_{e_{up}^p} = \mathrm{diag}\left(\left[\alpha_{e_{up}^p}^i\right]\right), \alpha_{e_{pcp}^p} = \mathrm{diag}\left(\left[\alpha_{e_{pcp}^p}^i\right]\right), \alpha_{e_{upc}^p} = \mathrm{diag}\left(\left[\alpha_{e_{upc}^p}^i\right]\right)$, and the final POI embedding is attained by combining these three embeddings:

$$e^P = \alpha_{e_{up}^p} \cdot e_{up}^p + \alpha_{e_{pcp}^p} \cdot e_{pcp}^p + \alpha_{e_{upc}^p} \cdot e_{upc}^p. \tag{14}$$

The final user embedding e^U and category embedding e^C can be similarly obtained.

4.3 Category-Aware Temporal Representation

This section presents the temporal representation learning method that fuses POI semantic categories. Generally, people participate in specific activities during specific periods. For example, most people flock to transportation facilities in the morning rush hour, enter catering places around noon and visit entertainment venues after 8 pm. Accordingly, temporal representations incorporating activity category semantics can boost the performance of predicting the next potential location.

We first split one day into 24 time slots with each of 60 minutes and project each time to one of the 24 time slots. We adopt *time2vec* to encode time to consider the effect of time periodicity:

$$e^t[i] = \begin{cases} \psi_i t + \phi_i, & \text{if } i = 0 \\ \mathcal{F}\left(\psi_i t + \phi_i\right), & \text{if } 1 \leq i \leq k \end{cases} \tag{15}$$

where $e^t[i]$ is the i-th element of the embedding vector of time slot t, $\mathcal{F}$ is the periodic activation function (*e.g.*, sin), and ψ_i and ϕ_i are the learnable parameters. We then get time embeddings through an MLP for all time slots, denoted by e^T.

Moreover, we use a dense layer to fuse the time embedding $e^t \in e^T$ and the corresponding category embedding $e^c \in e^C$ in check-in records, then the category-aware temporal embedding e^{tc} is calculated as follows:

$$e^{tc} = \sigma\left(\mathbf{W}_{tc}[e^t; e^c] + b_{tc}\right) \in e^{TC}. \tag{16}$$

where $\mathbf{W}_{tc}$ is the learnable weight vector and b_{tc} is the bias. σ is a leaky ReLU activation function with leaky rate 0.2.

Finally, a check-in record $< u, p, c, t >$ is represented by the concatenation of the resulting user embedding $e^u \in e^U$, POI embedding $e^p \in e^P$, and category-aware time embedding $e^{tc} \in e^{TC}$, *i.e.*, $[e^u; e^p; e^{tc}]$. An input trajectory is thus represented by a sequential embedding of check-in records.

4.4 Sequence Modeling Module

As the next location recommendation is a sequence prediction task, we adopt the Transformer to encode the check-in sequence and use three MLP predictors to decode it for predicting the time, POI, and category, respectively.

For the Transformer encoder, we stack several transformer layers, each layer consisting of a causal masked multi-head self-attention module and a position-wise feed-forward network (FFN) module. The vanilla multi-head self-attention sequentially models the correlation between check-ins, while the causal masked multi-head self-attention module adds an attention mask that can prevent the model from capturing sequence information after the predicted position during sequential modeling. Position-wise FFN will output a bag of embeddings, where the embedding at each position predicts the corresponding next check-in of the sequence. Residual connections and normalization have been applied to both modules.

For the l-th layer, the input $\mathbf{X}^{(l)} \in \mathbb{R}^{m\times d}$ is firstly transformed by the multi-head self-attention module. The output of the first attention head is:

$$\text{head}^{\#1} = \text{softmax}\left(\frac{\mathbf{X}^{(l)}\mathbf{W}^{Q}\left(\mathbf{X}^{(l)}\mathbf{W}^{K}\right)^{\mathsf{T}}}{\sqrt{d}}\right)\mathbf{X}^{(l)}\mathbf{W}^{V}, \tag{17}$$

where $\mathbf{W}^{Q}, \mathbf{W}^{K}, \mathbf{W}^{V} \in \mathbb{R}^{d\times d/h}$ are the weight matrices corresponding to "Query", "Key" and "Value", respectively. $\mathbf{X}^{(0)}$ is the learned check-in embedding matrix in the input trajectory. Then, we stack multiple attention heads and merge the outputs from different attention heads by performing a linear transformation operation:

$$\text{MultiHead}\left(\mathbf{X}^{(l)}\right) = \left[\text{head}^{\#1}; \text{head}^{\#2}; \cdots; \text{head}^{\#h}\right] \times \mathbf{W}_o, \tag{18}$$

where $\mathbf{W}_o \in \mathbb{R}^{d\times d}$ is the learnable parameter matrix.

Next, we perform layer normalization and residual connection on attention module to obtain the final output of attention module:

$$\mathbf{X}_{\text{ATT}}^{(l)} = \text{LayerNorm}\left(\mathbf{X}^{(l)} + \text{MultiHead}\left(\mathbf{X}^{(l)}\right)\right), \tag{19}$$

After attention module, it will pass through a position-wise FFN:

$$\mathbf{X}_{\text{FFN}}^{(l)} = \max\left(0, \mathbf{W}_1\mathbf{X}_{\text{ATT}}^{(l)} + b_1\right)\mathbf{W}_2 + b_2, \tag{20}$$

where $\mathbf{W}_1, \mathbf{W}_2$ are trainable weight matrices, and b_1, b_2 are biases. Lastly, we can get the output of the l-th encoder layer:

$$\mathbf{X}^{(l+1)} = \text{LayerNorm}\left(\mathbf{X}_{\text{ATT}}^{(l)} + \mathbf{X}_{\text{FFN}}^{(l)}\right), \tag{21}$$

After Transformer Encoder, we employ three MLP predictors to predict the next visit POI, as well as the visit time and the specific category. The MLP predictor is constructed as:

$$y_* = \mathbf{W}_*^{(2)}\,\text{ReLU}\left(\mathbf{W}_*^{(1)}\mathbf{X}^{(L_t)} + b_*^{(1)}\right) + b_*^{(2)}. \tag{22}$$

where $\mathbf{X}^{(L_t)}$ is the final output of Transformer encoder, L_t denotes the number of layers in Transformer encoder, and $\mathbf{W}_*^{(\cdot)}$ and $b_*^{(\cdot)}$ are the trainable parameters. y_* is the predicted output, *i.e.*, time, category or POI.

4.5 Model Learning

As for the sequence modeling module, each MLP predictor is preset with a corresponding loss function, cross-entropy for *POI* and *category*, and mean squared error (MSE) for *time*, in our study. The sum of the loss of each MLP predictor constitutes the total loss $\mathcal{L}_{\text{seq}}$ for sequence modeling, which is defined as follows:

$$\mathcal{L}_{seq} = \ell_{POI} + \ell_{time} + \ell_{category}, \tag{23}$$

By combining the contrastive loss at the global and local levels with the loss of sequence modeling, we minimize the following objective function for model learning:

$$\mathcal{L}_{model} = \beta\left(\alpha\mathcal{L}_{local}^{cl} + (1-\alpha)\mathcal{L}_{global}^{cl}\right) + \mathcal{L}_{seq}. \tag{24}$$

where α and β are the hyper-parameters to balance the contrastive loss on the local and global views.

Table 1: Statistics of Datasets

Dataset	#User	#Location	#Check-in	Duration
Foursquare-NYC	1083	5136	135,938	10 months
Foursquare-TKY	2290	7057	389,063	11 months

5 EXPERIMENT

To verify the effectiveness of our proposed model, we conduct extensive experiments on two real-world check-in mobility datasets.

5.1 Datasets

We evaluate MCN4Rec on two public real-world check-in datasets, which were collected from New York City (NYC) and Tokyo (TKY) and have been widely used for the next POI recommendation [32]. Check-in data are formed by users' arriving at functional locations and thus can be regarded as users' trajectories. These datasets include long-term check-in data in New York City, and Tokyo city collected from Foursquare from 12 April 2012 to 16 February 2013. Following [27], we discard the POIs which are visited by less than five users and the users with less than ten check-in records on two datasets. Then, the users' visit sequence is broken into trajectories according to the maximum threshold Δt of the time intervals between two consecutive check-in records. Here, we set Δt=24 hours. We split the datasets into training, validation, and testing sets by 6:2:2 in chronological order. The statistics of datasets after the preprocessing are summarized in Table 1.

5.2 Baseline Models

We compare our model against the following baseline models:

- **MF** [13].It is a classical method based on Matrix Factorization for interest point recommendation.
- **FPMC** [24]. It is a method that combines Matrix Factorization and Markov Chain together to model both long-term user preference and sequential behavior.
- **STRNN** [19]. It is a variant RNN model that incorporates spatio-temporal features between consecutive visits.
- **DeepMove** [6]. It is an attentional recurrent network that combines an attention layer with GRU to capture the multi-level periodicity for mobility prediction.
- **STGN** [37, 38]. It is a spatio-temporal gated network that adds spatio-temporal gates to LSTM to capture the spatio-temporal relationships between successive visits.
- **TALE** [27]. It is a state-of-the-art pre-training method based on the CBOW framework, which can incorporate temporal information into the learned embedding of locations.
- **GeoSAN** [17]. It is a self-attention-based model that uses hierarchical gridding of GPS locations for spatial discretization and self-attention layers to capture long-term sequential dependence.
- **STAN** [20]. It is a state-of-the-art attention-based model that uses self-attention layers to exploit relative spatiotemporal information of all POIs within user trajectory.
- **Flashback** [31]. It is an RNN-based model which explicitly uses spatiotemporal contexts to search past hidden states in RNNs for location prediction.
- **AutoMTN** [22]. It is an auto-correlation enhanced multi-modal Transformer network for next POI recommendation.

Table 2: Recommendation performance comparison with baselines.

	TKY					NYC				
	Acc@1	Acc@5	Acc@10	Acc@20	MRR	Acc@1	Acc@5	Acc@10	Acc@20	MRR
MF	0.0562	0.0875	0.1322	0.1889	0.0717	0.0886	0.1023	0.1449	0.2285	0.0992
FPMC	0.0892	0.2107	0.2789	0.3416	0.1384	0.1015	0.2238	0.2982	0.3362	0.1814
STRNN	0.1367	0.2978	0.3529	0.4643	0.2175	0.1448	0.3020	0.3590	0.4673	0.2213
DeepMove	0.1269	0.2884	0.3459	0.4592	0.2103	0.1350	0.3004	0.3547	0.4614	0.2225
STGN	0.1655	0.3384	0.3857	0.4427	0.2384	0.1743	0.3405	0.4177	0.5102	0.2619
TALE	0.1535	0.3343	0.3846	0.4719	0.2479	0.1674	0.3413	0.3897	0.4849	0.2541
GeoSAN	0.1973	0.3729	0.4473	0.5008	0.2803	0.2133	0.4469	0.5804	0.6152	0.3219
STAN	0.2103	0.3758	0.4667	0.5187	0.2929	0.2189	0.4590	0.5849	0.6285	0.3264
Flashback	0.1453	0.3363	0.4005	0.4916	0.2578	0.1508	0.3448	0.3877	0.4773	0.2502
AutoMTN	0.2341	0.3802	0.4687	0.5469	0.3242	0.2089	0.4370	0.5289	0.5914	0.3042
GETNext	0.2231	0.4345	0.5136	0.5874	0.3267	0.2361	0.5023	0.6056	0.6712	0.3443
CFPRec	0.2249	0.4470	0.5296	0.5947	0.3302	0.2404	0.5106	0.6175	0.6809	0.3513
Graph-Flashback	0.2268	0.4537	0.5300	0.5982	0.3325	0.2431	0.5201	0.6133	0.6827	0.3569
MCN4Rec	**0.2535**	**0.4850**	**0.5656**	**0.6146**	**0.3475**	**0.2643**	**0.5429**	**0.6405**	**0.7117**	**0.3868**
Improvement	8.29%	6.90%	6.72%	2.74%	4.51%	8.72%	4.38%	3.72%	4.25%	8.38%

- **GETNext** [33]. It is a novel Graph Enhanced Transformer model to use collaborative signals for next POI prediction.
- **CFPRec** [35]. It is a state-of-the-art self-ensembling method that incorporates past, current and future preferences for next POI recommendation.
- **Graph-Flashback** [23]. It is a state-of-the-art model incorporating GCN and POI transition knowledge graph into RNN-based models to capture sequential transition patterns better.

5.3 Evaluation Metrics

We use commonly used metrics in recommender systems, Accuracy@K (Acc@K) and Mean Reciprocal Rank (MRR), to judge the performance of models. Given a test dataset with N samples, the evaluation metrics can be defined as:

$$Acc@K = \frac{1}{N}(\#hit@K) \tag{25}$$

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank} \tag{26}$$

where $(\#hit@K)$ is the number of samples with the correct predictions made within the top K of the ranked set for $K \in \{1, 5, 10, 20\}$, *rank* represents the rank of the true next POI in the recommended ordered list. *Acc@K* helps to understand the performance of the recommender system for the top K recommendations, while *MRR* gives the overall performance of the predicted ranking set. For all these metrics, the higher the value, the better the performance.

5.4 Experimental Settings

We implement our model MCN4Rec in the PyTorch framework and conduct experiments on NVIDIA GeForce RTX 3090. For baselines, we use the source code released by their authors, adopt the parameter settings recommended in their papers, and fine-tune them to be optimal. Notice that detailed experimental settings can be found in Appendix A. The sensitivity study w.r.t. hyperparameters L_g, L, L_t, h, α and β can be found in Appendix A.4.

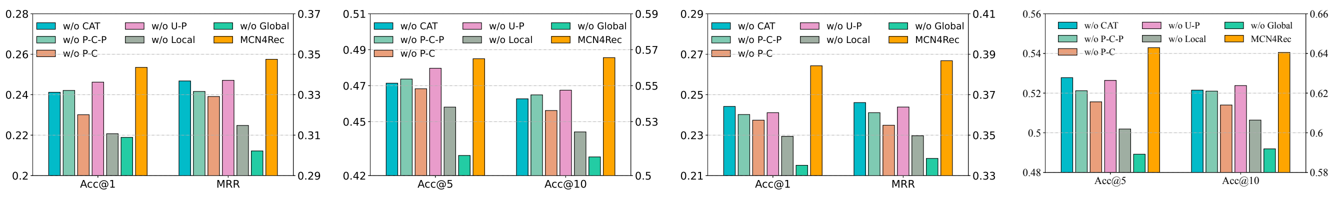
5.5 Overall Performance

The performance comparison of MCN4Rec with all baselines on two real-world mobility datasets is shown in Table 2, where the best is shown in bold and the second best is underlined.

We can observe from the experimental results that our proposed MCN4Rec achieves the best recommendation performance compared to state-of-the-art baseline models. In particular, the relative performance improvement of our MCN4Rec over the best-performed baseline Graph-Flashback is 10.25% and 6.45% in terms of Acc@1 and MRR across two datasets. Although Graph-Flashback incorporates knowledge graph embedding and GCN to learn location embeddings and POI transition patterns, it only relies on RNN model to learn temporal dependencies. Our MCN4Rec not only utilizes causal masked Transformer encoder to learn the long-term spatiotemporal dependencies efficiently but also utilizes multi-level contrastive representation learning to capture multiple correlations effectively among user, POI, time, and activity semantics in user check-in sequences. In addition, MCN4Rec significantly outperforms state-of-the-art LSTM-Transformer ensemble baseline CFPRec by an average of 11.33% and 7.68% in terms of Acc@1 and MRR on two datasets, respectively. This may be because, although CFPRec learns past and current preferences through Transformer and LSTM, respectively, and uses future preference extractor to predict future behaviors for next location prediction, more attention is paid to sequence modeling while ignoring effective representation learning of users and POIs. In fact, in user mobility data, the complex correlations among users, POIs, and various semantics are crucial for location recommendation, but it is difficult to capture only through sequence modeling. Additionally, the prediction performance superiority can be observed in comparisons of our MCN4Rec with all competitive methods, which validates the effectiveness of our designed model with the integration of comprehensive multi-level contrastive representation learning and requisite sequence modeling.

5.6 Ablation Study

To validate the effectiveness of each component of our model, we further conduct an ablation study on different MCN4Rec variations.



(a) Acc@1, MRR on TKY (b) Acc@5, Acc@10 on TKY (c) Acc@1, MRR on NYC (d) Acc@5, Acc@10 on NYC

Figure 2: Experimental results of Ablation study

We report the experimental results of the ablation study on two datasets in Figure 2, where the performance in terms of *Acc@K* and *MRR* refers to the right-ordinate axis. We drop different components to form variants, which are listed as:

- **w/o Global**: This variant removes UPC-KG components, *i.e.*, the global view.
- **w/o Local**: This variant drops two local views, including User-POI and POI-Category views.
- **w/o User-POI (U-P) view** : This variant only drops the User-POI local view.
- **w/o POI-Category (P-C) view**: This variant only drops the POI-Category local view.
- **w/o POI-POI semantic graph (P-C-P)**: This variant removes the POI-POI semantic graph in the POI-Category local view.
- **w/o Category-aware time representation (CAT)**: This variant use time embedding to replace the category-aware time embedding directly.

From the results in Figure 2, we can see that all key components contribute to the performance improvement of our MCN4Rec. The comparison between w/o Global and MCN4Rec highlights the effectiveness of the importance of knowledge graph modeling and representation learning. We can observe that w/o Global performs the worst on all datasets in terms of all evaluation metrics, reducing 13.65% and 18.62% performance in Acc@1 on TKY and NYC, which demonstrates the crucial role of our designed User-POI-Category knowledge graph in capturing the complex correlations among user, POI, and category for representation learning. The comparison between w/o Local and MCN4Rec reflects the importance of our local view representation learning module in enhancing local relation learning (*i.e.*, User-POI, POI-category, and POI-category-POI). To be specific, MCN4Rec improves 14.86% and 15.21% over w/o Local in terms of Acc@1 on TKY, and NYC, respectively. Additionally, w/o Local performs worse than w/o U-P, w/o P-C, and w/o P-C-P, illustrating the effectiveness and necessity of the three sub-modules of local view learning.

5.7 Cold-Start Study

Our model effectively mitigates the impact of data sparsity for POI recommendation by considering correlations between data from different hierarchical views. We perform experiments with inactive users and short trajectories to verify this compared with state-of-the-art cold-star recommendation baseline GETNext [33].

Following [33], we first split the dataset into training, validation, and testing sets. Then we count the number of trajectories of all users in the training set, set the first 15% as active users, the last 15% as inactive users, and the rest as normal users, and then use the testing set for evaluation. In addition, we also calculate the length of trajectories in the training set and set the first 15% as long trajectories, the last 15% as short trajectories, and the rest as normal trajectories. Table 3 and Table 4 show the performance comparison of our MCN4Rec and GETNext on user and trajectory using NYC dataset, respectively.

As we can see, MCN4Rec significantly outperforms GETNext in all test cases in both experiments. Especially on inactive users and short trajectories, our model achieves a significant performance improvement of 28.19% and 10.16% in Acc@1, respectively.

Table 3: Cold-Start (due to inactive users) on NYC

User Groups	Model	Acc@1	Acc@5	Acc@10	Acc@20
Inactive	GETNext	0.1224	0.3471	0.4394	0.4529
Normal	GETNext	0.2421	0.4739	0.5422	0.6430
Very active	GETNext	0.2692	0.5639	0.6995	0.7762
Inactive	MCN4Rec	0.1569	0.3871	0.4802	0.5011
Normal	MCN4Rec	0.2608	0.5214	0.5933	0.6890
Very active	MCN4Rec	0.2902	0.6324	0.7417	0.8005

Table 4: Cold-Start (due to short trajectory) on NYC

Trajecotory	Model	Acc@1	Acc@5	Acc@10	Acc@20
Short trajs	GETNext	0.2186	0.4561	0.5269	0.5782
Middle trajs	GETNext	0.2441	0.4927	0.5881	0.6519
Long trajs	GETNext	0.2452	0.5378	0.6698	0.7681
Short trajs	MCN4Rec	0.2408	0.5009	0.5683	0.6197
Middle trajs	MCN4Rec	0.2617	0.5342	0.6124	0.6874
Long trajs	MCN4Rec	0.2703	0.5837	0.7026	0.7983

6 CONCLUSION

In this paper, we propose a multi-level collaborative neural network (MCN4Rec) to solve the next location recommendation problem for sparse check-in mobility data. MCN4Rec effectively learns the multi-semantic correlation among user, POI, time, and activity category in check-in sequence data for next location recommendation by combining multi-level contrastive representation learning with Transformer-based sequence modeling. Experiments on two real-life check-in datasets show that MCN4Rec significantly outperforms state-of-the-art baseline approaches in all evaluation metrics on the next location recommendation task.

REFERENCES

[1] Mohammad Aliannejadi, Dimitrios Rafailidis, and Fabio Crestani. 2019. A joint two-phase time-sensitive regularized collaborative ranking model for point of interest recommendation. *IEEE Transactions on Knowledge and Data Engineering* 32, 6 (2019), 1050–1063.

[2] Anirban Chakraborty, Debasis Ganguly, and Owen Conlan. 2020. Relevance models for multi-contextual appropriateness in point-of-interest recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 1981–1984.

[3] Yudong Chen, Xin Wang, Miao Fan, Jizhou Huang, Shengwen Yang, and Wenwu Zhu. 2021. Curriculum meta-learning for next POI recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining.* 2692–2702.

[4] Chen Cheng, Haiqin Yang, Michael R Lyu, and Irwin King. 2013. Where you like to go next: Successive point-of-interest recommendation. In *Twenty-Third international joint conference on Artificial Intelligence.*

[5] Qiang Cui, Chenrui Zhang, Yafeng Zhang, Jinpeng Wang, and Mingchen Cai. 2021. ST-PIL: Spatial-Temporal Periodic Interest Learning for Next Point-of-Interest Recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management.* 2960–2964.

[6] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. Deepmove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the 2018 world wide web conference.* 1459–1468.

[7] 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.*

[8] Qing Guo, Zhu Sun, Jie Zhang, and Yin-Leng Theng. 2020. An attentional recurrent neural network for personalized next location recommendation. In *Proceedings of the AAAI Conference on artificial intelligence*, Vol. 34. 83–90.

[9] Peng Han, Zhongxiao Li, Yong Liu, Peilin Zhao, Jing Li, Hao Wang, and Shuo Shang. 2020. Contextualized point-of-interest recommendation. International Joint Conferences on Artificial Intelligence.

[10] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. 2019. Time2vec: Learning a vector representation of time. *arXiv preprint arXiv:1907.05321* (2019).

[11] Minseok Kim, Hwanjun Song, Doyoung Kim, Kijung Shin, and Jae-Gil Lee. 2021. PREMERE: Meta-Reweighting via Self-Ensembling for Point-of-Interest Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4164–4171.

[12] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR.*

[13] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer Society* 42, 8 (2009), 30–37.

[14] Xutao Li, Gao Cong, Xiao-Li Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval.* 433–442.

[15] Yang Li, Tong Chen, Hongzhi Yin, and Zi Huang. 2021. Discovering collaborative signals for next POI recommendation with iterative Seq2Graph augmentation. *arXiv preprint arXiv:2106.15814* (2021).

[16] Zeyu Li, Wei Cheng, Haiqi Xiao, Wenchao Yu, Haifeng Chen, and Wei Wang. 2021. You Are What and Where You Are: Graph Enhanced Attention Network for Explainable POI Recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management.* 3945–3954.

[17] Defu Lian, Yongji Wu, Yong Ge, Xing Xie, and Enhong Chen. 2020. Geography-aware sequential location recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining.* 2009–2019.

[18] Yan Lin, Huaiyu Wan, Shengnan Guo, and Youfang Lin. 2021. Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4241–4248.

[19] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Thirtieth AAAI conference on artificial intelligence.*

[20] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. Stan: Spatio-temporal attention network for next location recommendation. In *Proceedings of the Web Conference 2021.* 2177–2185.

[21] Pramit Mazumdar, Bidyut Kr Patra, and Korra Sathya Babu. 2020. Cold-start point-of-interest recommendation through crowdsourcing. *ACM Transactions on the Web (TWEB)* 14, 4 (2020), 1–36.

[22] Yanjun Qin, Yuchen Fang, Haiyong Luo, Fang Zhao, and Chenxing Wang. 2022. Next Point-of-Interest Recommendation with Auto-Correlation Enhanced Multi-Modal Transformer Network. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 2612–2616.

[23] Xuan Rao, Lisi Chen, Yong Liu, Shuo Shang, Bin Yao, and Peng Han. 2022. Graph-flashback network for next location recommendation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining.* 1463–1471.

[24] 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.* 811–820.

[25] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2020. Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 214–221.

[26] 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).

[27] Huaiyu Wan, Yan Lin, Shengnan Guo, and Youfang Lin. 2021. Pre-training time-aware location embeddings from spatial-temporal trajectories. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[28] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. 2021. Learning intents behind interactions with knowledge graph for recommendation. In *Proceedings of the Web Conference 2021.* 878–887.

[29] 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.* International Joint Conferences on Artificial Intelligence Organization.

[30] 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.* 1154–1163.

[31] Dingqi Yang, Benjamin Fankhauser, Paolo Rosso, and Philippe Cudre-Mauroux. 2020. Location Prediction over Sparse User Mobility Traces Using RNNs. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence.* 2184–2190.

[32] 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.

[33] 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.* 1144–1153.

[34] Fuqiang Yu, Lizhen Cui, Wei Guo, Xudong Lu, Qingzhong Li, and Hua Lu. 2020. A category-aware deep model for successive POI recommendation on sparse check-in data. In *Proceedings of the web conference 2020.* 1264–1274.

[35] 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.* International Joint Conferences on Artificial Intelligence Organization, 3751–3757.

[36] Kangzhi Zhao, Yong Zhang, Hongzhi Yin, Jin Wang, Kai Zheng, Xiaofang Zhou, and Chunxiao Xing. 2020. Discovering Subsequence Patterns for Next POI Recommendation.. In *IJCAI.* 3216–3222.

[37] Pengpeng Zhao, Anjing Luo, Yanchi Liu, Fuzhen Zhuang, Jiajie Xu, Zhixu Li, Victor S Sheng, and Xiaofang Zhou. 2022. Where to go next: A spatio-temporal gated network for next poi recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34, 5 (2022), 2512–2524.

[38] Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S Sheng, and Xiaofang Zhou. 2019. Where to go next: A spatio-temporal gated network for next POI recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence.* 5877.

[39] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to Do Next: Modeling User Behaviors by Time-LSTM.. In *IJCAI*, Vol. 17. 3602–3608.

A SUPPLEMENTAL MATERIAL

A.1 Notations

We summarize the key notations used in the paper and their definitions in Table 5.

Table 5: Summary of key notations

Notation	Definition
$\mathcal{U}, \mathcal{P}, C$	the set of all Users, POIs, categories
γ	a check-in record
$\mathcal{G}$	the knowledge graph
$\mathcal{V}, \mathcal{E}$	the set of nodes and edges in $\mathcal{G}$
$\mathcal{R}$	the set of relationship triples in $\mathcal{G}$
L_g	the number of aggregation layers in path GNN
L	the number of GCN layers in GCN
$\mathbf{A}_{up}, \mathbf{A}_{pc}$	the adjacency matrices in two local view graphs
$\mathbf{H}^{(l)}$	the hidden representation for the l-th layer
e^u, e^p, e^c	the embeddings of user, POI, and category
e^{tc}	the category-aware time embedding
e^*_{upc}	the embedding in UPC-KG view
$e^*_{up}, e^*_{pc}, e^p_{pcp}$	the embedding in two local view graphs
$\mathbf{W}^{(l)}, b^{(l)}$	the learnable weight matrix and bias
τ	the temperature parameter
$\mathcal{F}$	the periodic activation function
$\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V$	the weight matrices
h	the number of attention heads
L_t	the number of layers in Transformer encoder

A.2 Baselines

The public source codes of baselines can be available at the following URLs:

- **MF** – https://github.com/harshraj11584/Paper-Implementation-Matrix-Factorization-Recommender-Systems-Netflix
- **FPMC** – https://github.com/khesui/FPMC
- **STRNN** – https://github.com/yongqyu/STRNN
- **DeepMove** – https://github.com/vonfeng/DeepMove
- **TALE** – https://github.com/Logan-Lin/TALE
- **GeoSAN** – https://github.com/libertyeagle/GeoSAN
- **STAN** – https://github.com/yingtaoluo/Spatial-Temporal-Attention-Network-for-POI-Recommendation
- **Flashback** – https://github.com/eXascaleInfolab/Flashbackcode/
- **GETNext** – https://github.com/songyangco/GETNext
- **CFPRec** – https://github.com/wuziqi2/CFPRec
- **Graph-Flashback** – https://github.com/kevin-xuan/Graph-Flashback

For STGN and AutoMTN, their authors do not release the code addresses. Nevertheless, we contacted the corresponding authors to obtain the source code and compare it with our approach. In addition, we also remove the friendship between users in Graph-Flashback for a fair comparison (Unable to get user explicit relationship on our dataset).

A.3 Detailed Experimental Settings

For STRNN, we set the dimensionality to be $d = 13$ and $d = 7$ respectively, and set the window width to be $w = 6h$. For DeepMove, we set L2 penalty as 0.00005, the gradient clip as 1.0. For STGN, we set the radius is 0.5 and the window size is set to 3, the cell size and the batch size are set to 128 and 10. For TALE, we set the window size to 2, and set the Stochastic Gradient Descent optimizer with an initial learning rate of 0.001, set the time slice length to 240, set the influence span length to 60 minutes. For GeoSAN, we set the dimension of location embedding to 50, and we set the number of negative samples for training to 5. For STAN, we set the embedding dimension to 50, and set the maximum length for trajectory sequence of 100. For Flashback, we set the temporal decay α as 0.1, and set the spatial decay factor β as 1000. For AutoMTN, we set the category loss weight η as 0.4, and set the batch size as 32. For GETNext, we set the batch size as 20, and set the dimensions of feed-forward network in the transformer encoder layer as 1024. For CFPRec, we set the embedding size as 20, and set the number of Transformer block as 1, and set the number of LSTM layer is 3. For Graph-Flashback, we set the dimension of hidden states and POI embedding as 10.

For our model, we set the dimension of all embeddings to 256 and the batch size to 128. We use the Adam optimizer to optimize our model and adjust the learning rate in {0.0001,0.0003,0.001,0.003}; the GCN model has three hidden layers, each with 32, 64, and 128 channels. L_g, L, L_t, and h are all set to 2. α and β are set to 0.25 and 0.15, respectively, to balance the local and global contrastive learning.

A.4 Additional Experimental Results

In this section, we evaluate the impact of six hyper-parameters in the multi-level representation learning module on model performance: 1) the number of layers of Relational Path-Aware GNN L_g; 2) the number of layers of GCN in the GCN Block L; 3)the number of encoding layers in Transformer L_t; 4) the number of attention head h; 5) coefficient α and β.

A.4.1 Effect of the number of layers of Relational Path-Aware GNN L_g. In order to analyze the effect of relational path-aware GNN layers on model performance, we set L_g to transform in the range of {1, 2, 3, 4}, and the specific performance changes are shown in Figure 3. Experiments show that when $L_g = 2$, it is more suitable for the model to perform relational path-aware aggregation.

A.4.2 Effect of the number of GCN layers in the GCN Block L. To study the effect of the number of GCN layers on the model, we vary L in the range of {1, 2, 3, 4}, and the specific performance is shown in Figure 3. We notice that when the number of layers is greater than 2, the model performance starts to decline, which proves that too many layers can hurt the model performance.

A.4.3 Effect of the number of encoding layers in Transformer L_t. In order to study the influence of the number of coding layers in Transformer on the model performance, we set Lt to vary in the range of {1, 2, 3, 4}, and the specific performance changes are shown in Figure 3. When the number of layers is 2, the model has the best performance.

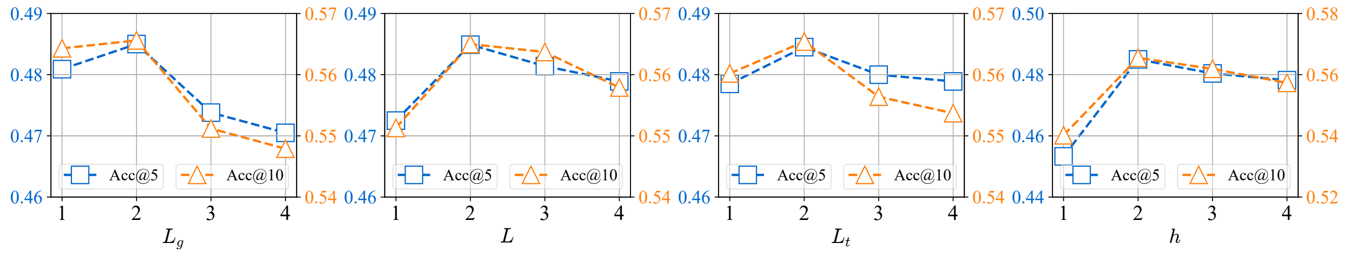


Figure 3: Parameter sensitivity *w.r.t.* L_t, L, L_t and h on TKY.

A.4.4 Effect of the number of attention head h. In order to explore the effect of the number of attention heads on the model performance, we set h to vary in the range of {1, 2, 3, 4}, and the specific performance changes are shown in Figure 3, and setting h to 2 will make the model have the best performance.

A.4.5 Effect of coefficient α and β. We also evaluate the sensitivity of MCN4Rec *w.r.t.* different settings of coefficient α and β in our loss function. The coefficient α is used to control the effect of the local loss on the overall contrastive loss, and β determines the importance of the contrastive loss. Results on TKY dataset are shown in Figure 4. We can observe that: (1) When $\alpha = 1$ (*i.e.*, ignoring $\mathcal{L}^{cl}_{global}$), the performance is the worst, this emphasizes the importance of the global contrastive loss. (2) As α increases from 0, the model performance is significantly improved, which verifies the effectiveness of local-level contrastive learning. When $\alpha = 0.25$, the model achieves the best performance. (3) The model achieves the best performance when $\beta = 0.15$, and as β increases to 1, the model performance drops sharply, even worse than when β equals 0. This suggests that focusing too much on representation learning over sequence modeling can significantly harm model performance.

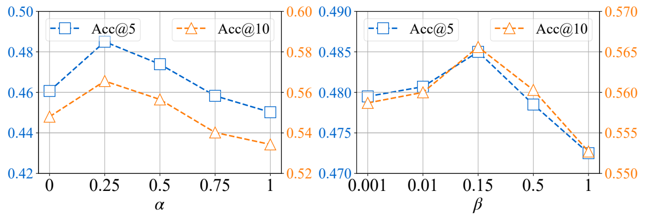


Figure 4: Parameter sensitivity *w.r.t.* α and β on TKY.