electronics

MDPI

Article

Exploring Behavior Patterns for Next-POI Recommendation via Graph Self-Supervised Learning

Daocheng Wang, Chao Chen, Chong Di and Minglei Shu *

Shandong Artificial Intelligence Institute, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250014, China; wangdc2563@163.com (D.W.)
* Correspondence: shuml@sdas.org

Abstract: Next-point-of-interest (POI) recommendation is a crucial part of location-based social applications. Existing works have attempted to learn behavior representation through a sequence model combined with spatial-temporal-interval context. However, these approaches ignore the impact of implicit behavior patterns contained in the visit trajectory on user decision making. In this paper, we propose a novel graph self-supervised behavior pattern learning model (GSBPL) for the next-POI recommendation. GSBPL applies two graph data augmentation operations to generate augmented trajectory graphs to model implicit behavior patterns. At the same time, a graph preference representation encoder (GPRE) based on geographical and social context is proposed to learn the high-order representations of trajectory graphs, and then capture implicit behavior patterns through contrastive learning. In addition, we propose a self-attention based on multi-feature embedding to learn users' short-term dynamic preferences, and finally combine trajectory graph representation to predict the next location. The experimental results on three real-world datasets demonstrate that GSBPL outperforms the supervised learning baseline in terms of performance under the same conditions.

Keywords: graph self-supervised learning; contrastive learning; implicit behavior pattern; self-attention; next POI recommendation

Citation: Wang, D.; Chen, C.; Di, C.; Shu, M. Exploring Behavior Patterns for Next-POI Recommendation via Graph Self-Supervised Learning. *Electronics* **2023**, *12*, 1939. https://doi.org/10.3390/electronics12081939

Academic Editor: Domenico Ursino

Received: 22 March 2023
Revised: 13 April 2023
Accepted: 15 April 2023
Published: 20 April 2023

1. Introduction

In recent years, location-based social networking (LBSN) services such as Uber, Foursquare, and Gowalla have experienced significant growth with the prevalence of smartphones. Users can easily share their check-in locations, such as restaurants and shopping centers, on LBSN applications. As the amount of check-in data increases exponentially, point-of-interest (POI) recommendation systems have become critical in recommending places of interest to users based on their historical check-in records. The next-POI recommendation system is a subfield of POI recommendation that can learn personalized preferences based on users' historical check-in records and recommend places that they are likely to visit next [1,2]. Therefore, the model needs to have the ability to model complex behavior patterns.

Currently, next-location prediction methods have been extensively explored. Early research primarily focused on personalized extensions of Markov chain (MC) models [3–5]. However, MC-based models have strong independence between different events and cannot model preferences. With the development of deep learning, models based on recursive neural networks (RNNs), including long short-term memory (LSTM) and gated recurrent unit (GRU), are widely used in processing sequence problems due to their memory mechanism [6–9]. These models usually slice the check-in sequence based on a fixed time and learn the long-term and short-term preferences of the global or local check-in sequence. For instance, the STMLA [10] is a Mogrifier LSTM that couples the traditional LSTM with multiple context information in order to capture user preferences. Despite these methods

having achieved success, they are limited in capturing the multiple associations between different check-in data, since RNN models cannot handle non-Euclidean data.

Another method is based on the self-attention mechanism, which computes the correlation between non-consecutive check-ins and expands the self-attention mechanism, such as incorporating spatial-temporal context information [11]. However, these methods have larger computational costs and struggle to capture the high-order sequentiality of check-in data. Models based on a graph neural network (GNN) have the ability to capture the high-order associations between different instances [12]. Recently, some POI recommendation methods have used graph representation learning techniques to learn the interactions between POIs and update the node representations in low-dimensional space, followed by generating user preferences [13–15]. For example, FC-CF [16] establishes a stratified graph of users and POIs, facilitating the learning of collaborative filtering signals between users and POIs. Despite these breakthroughs, the existing methods have not explored users' complex behavior patterns behind the check-in data and lack a unified method that can capture both continuous and discontinuous visits or other behavioral patterns.

Our work is motivated by the real-world check-in behavior of users. By observation, we identified three common behavioral patterns: (1) In Figure 1, pattern A shows that the *user* tends to visit certain places in a fixed order. For example, he may visit workplace p_1 with high regularity on workdays and then go to restaurant p_2 after work. This is the most common sequential visit pattern. (2) From the routes containing pattern B, $p_1 \rightarrow p_4 \rightarrow p_5$ and $p_4 \rightarrow p_6 \rightarrow p_5 \rightarrow p_{12}$, we can see that although *user* may occasionally visit attraction p_6 due to temporary decisions, the *user*'s habitual behavior preferences of visiting mall p_4 and bookstore p_5 in order remains unchanged. In other words, the *user* tends to visit p_4 and p_5 continuously or in a skipping manner, which we call the skip visit pattern. (3) Pattern C in Figure 1 shows that the visit order between some locations is not fixed, and *user* can visit p_8 before going to p_9, or vice versa. This is the indefinite-order visit pattern. We define these three check-in behaviors as implicit behavioral patterns. It is worth noting that check-in data are one-dimensional data sorted by timestamps. In general, the model cannot learn users' complex behavioral patterns, so we need to generate some useful information from existing data to assist recommendation models. In recent years, with the ability to generate auxiliary supervisory signals through data augmentation operations to assist downstream tasks in training, graph contrastive learning (GCL) has gradually come to researchers' attention [17–19].

Inspired by the latest research progress in GCL in the recommendation field, in this paper, we propose a novel graph self-supervised behavior pattern learning model (GSBPL) for next-POI recommendation. GSBPL achieves unified learning of implicit behavior patterns using self-supervised graph contrastive learning methods. Specifically, GSBPL consists of two key parts: (1) Graph data augmentation, which uses data augmentation operations on check-in trajectory graphs to generate augmented subgraphs, and (2) graph contrastive learning, which maximizes the consistency between positive subgraphs. We propose two data augmentation operations: *edge masking and bridging*, and *edge inversion*, respectively, used for modeling skip and indefinite-order visit patterns, and the original check-in graph is used to model sequential visit pattern. Then, we propose the graph preference representation encoder (GPRE) to generate graph-level embeddings for sequential graphs and use graph contrastive learning on augmented subgraph embeddings to capture user personalized implicit behavior patterns. Additionally, we designed a multi-feature self-attention mechanism to learn a user's short-term dynamic preference representation and combine it with the original trajectory graph embedding for recommendation, improving the recommendation performance. It is noteworthy that GSBPL only utilizes user-initiated check-in behavior data for recommendation, without using any personal attributes or real-time location data of users, which is advantageous for protecting users' privacy.

In conclusion, the main contributions of this paper are summarized as follows:

- We propose the concept of implicit behavior patterns and the graph self-supervised learning model (GSBPL), which achieves unified modeling of users' multiple check-in behaviors. To our best recollection, GSBPL is the first next-POI recommendation model that applies graph contrastive learning for behavior pattern modeling.
- We propose a graph preference representation encoder (GPRE), which differs from methods that update node features using graph representation learning. GPRE updates higher-order node representation based on positional and social popularity context information and learns the graph-level embedding for sequential graphs.
- We propose a multi-feature self-attention mechanism that embeds temporal, spatial, and popularity features of POIs and captures user short-term dynamic decision-making information to generate short-term preference representation.
- We conducted extensive experiments on three real LBSN datasets, and the experimental results show that GSBPL has superior recommendation performance.

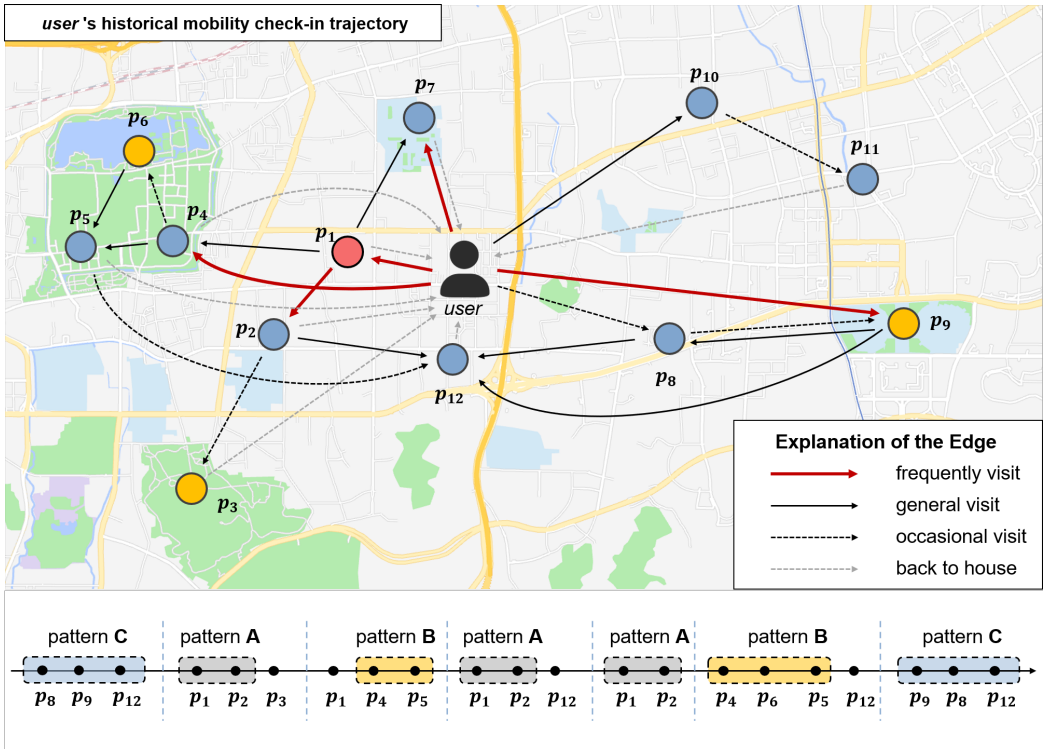


Figure 1. An example showing the implicit behavior patterns of users in complex spatial-temporal transition trajectories. The map illustrating the *user*'s historical movement trajectory. The visited locations are named from p_1 to p_{12}. The nodes in blue, red, and yellow represent shopping malls/restaurants/supermarkets, workplaces, and leisure venues/attractions, respectively. Below shows the *user*'s visit trajectory for a single week, in which the gray, yellow, and blue areas are the *user*'s implicit behavior patterns A, B, and C, respectively.

The remainder of this paper is organized as follows. In Section 2, we summarize previously conducted research related to our study. In Section 3, we give the preliminary preparation of the model. In Section 4, we describe our proposed GSBPL. In Section 5, we compare GSBPL with existing recommendation models and analyze the experimental results. Last, in Section 6, we summarize the work done in this paper.

2. Related Work

In this section, we introduce the current research work on next-POI recommendation, GNN-based recommendation, and graph contrastive learning in order.

2.1. Next-POI Recommendation

The major goal of the next-POI recommendation is to predict the user's next visit based on a given user's check-in list. Early work used the Markov chain (MC) for personalized recommendation [3,5], but Markov models have limitations in capturing high-order sequential dependencies. Deep learning was applied to the next location recommendation system afterwards, due to its excellent performance in modeling complex relationships between heterogeneous data and extracting latent features. The most widely used method is based on RNN. ST-RNN [20] defines a spatial-temporal matrix to extend the mobility of RNN in capturing local spatial-temporal intervals of a specific length. DeepMove [6] designed a multi-modal RNN that embeds user, time, and region information into the feature space to model long-term dependencies in check-in sequence. LSTPM [21] learns both long- and short-term interests. It uses LSTM to learn the daily sequence representation and uses time-weighted operations to measure the similarity between each subsequence and the latest visit. STGN [22] added time gates and distance gates to improve LSTM and improve the recommendation effect. However, these models ignore the learning of implicit behavior patterns behind user check-in action.

2.2. GNN-Based Recommendation

The graph neural network has powerful representation learning abilities. In the recommendation system, the interaction bipartite graph and social network of users and items are natural graph structures. Many excellent GNN recommendation methods [23–27] capture high-order connectivity at the graph level through message passing to perform representation learning based on the principle of neighbor similarity. GGLR [13] uses LightGCN [24] as a basic framework to learn incoming and outgoing geographic influences from user-POI and POI-POI graphs to evaluate user preference. SGRec [14] constructed daily sequences into sequence diagrams for session-level collaborative filtering while using location-aware attention to learn short-term preference. ASGNN [28] uses a gated graph neural network [29] to model a user's check-in behavior and embeds a vector-input hierarchical attention network to learn a user's long- and short-term interests. Despite being effective, these models ignore the close relationship between the entire motion graph structure and the user's preference.

2.3. Graph Contrastive Learning

Contrastive learning is an unsupervised learning method that can extract auxiliary supervised signal training models from the data themselves in the case of sparse data features and missing labels [30]. Contrastive learning is usually used in graph representation learning methods in graph classification tasks [31,32]. The two core modules of GCL are: (1) data augmentation—generating two graph augmentation views by randomly augmenting the graph through pruning and other strategies; (2) contrastive loss—the consistency of the two positive-pair views is maximized in the feature space by the loss function. GCC [33] uses contrastive learning to learn transferable graph structure representations for downstream tasks in the pre-training phase. The GCA framework [17] is an adaptive data augmentation scheme based on graph topology and node attributes. Recently, Wu proposed a self-supervised graph learning framework SGL [34], which proposes a multi-node self-distinguishing data expansion paradigm for GCL. Furthermore, [35–38] have successfully applied GCL to recommendation fields such as sequence recommendation and cross-domain recommendation.

3. Preliminaries

In this section, we briefly introduce the rich features and give the definition of key terms and a statement of the problem. Let $U = \{u_1, u_2, \ldots, u_{|U|}\}$ and $P = \{p_1, p_2, \ldots, p_{|P|}\}$ denote the set of users and POIs, respectively, where each POI $p \in P$ corresponds to an exact longitude and latitude coordinate, namely, $(lon,\ lat)$. Embedding vectors $e_u \in \mathbb{R}^d$ and $e_p \in \mathbb{R}^d$ describe user u and POI p, respectively, where d is the embedding dimension.

Historical Check-in Trajectory. Sequence of check-ins by user visiting locations in chronological order. For the target user $u \in U$, the tuple $c_i = (u, p_i, t_i)$ is the check-in of the user u at a certain time in the historical trajectory, where t_i is a timestamp in hours to represent the check-in time, and $p_i \in P$ is the POI visited by the user u at the time step t_i. We define the historical check-in trajectory of user u as $S(u) = \{c_1, c_2, \ldots, c_N\}$, where $N = |S(u)|$ is the check-in numbers of user u. Previous studies have shown that Google Maps search history data can be used to infer a user's interests and preferences, and their movements over time [39]. By analyzing patterns in the types of locations that a user searches for and visits, researchers can gain insights into their POIs. Since each POI has a unique latitude and longitude coordinate, the user's historical trajectory graph can be visually observed after mapping all the historical check-in trajectories to maps [40].

Correlation Matrix. According to the transfer information between each check-in in the user's historical trajectory, this paper explicitly models from three aspects: time slot, spatial distance, and social popularity. Specifically, we simply represent the time slot between the i-th and the j-th check-in as $r^t_{i,j} = |t_i - t_j|$ and calculate the spatial distance $r^s_{i,j} = haversine(loc_i, loc_j)$ between the i-th and the j-th check-in using the Haversine formula. It is worth noting that there are also popularity features with social attributes between POIs which dynamically determine the user's next visit object. We map the transfer frequency $freq_{i,j}$ between the i-th and the j-th check-in to the social popularity. A good rule of thumb is that the influence degree of location is a long tail distribution, and some locations have dense connectivity (i.e., very popular). Thus, we apply a natural logarithm to reduce this influence, and finally get the popularity $r^p_{i,j} = \ln(freq_{i,j})$. Based on the above rules, we generate the temporal, spatial, and popularity correlation matrices as $R^t = (r^t_{i,j})_{N\times N}$, $R^s = (r^s_{i,j})_{N\times N}$, and $R^p = (r^p_{i,j})_{N\times N}$, respectively, where $i, j = 1, 2, \ldots, N$.

Next-POI recommendation. Given all the historical check-in trajectories $S(u)$ of the target user u and the candidate POI set P, our goal is to output the TOP-K POIs that the user wants to visit at time t_{N+1}.

4. Proposed Framework

In this section, we will illustrate our proposed model in detail. The GSBPL model architecture is shown in Figure 2, which contains three important components: (1) the graph self-supervised pattern learning module; (2) the dynamic preference-learning module; (3) the prediction layer.

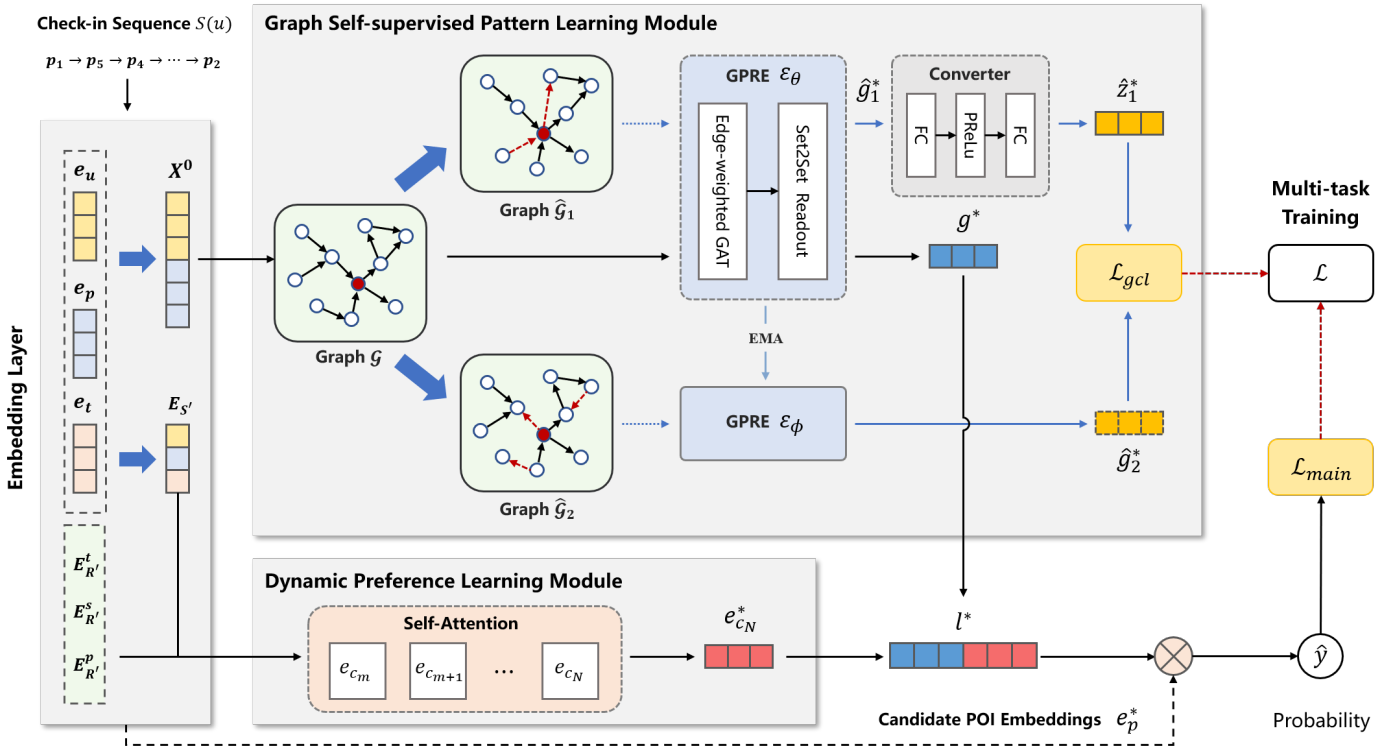


Figure 2. Our proposed GSBPL framework.

4.1. Graph Self-Supervised Pattern Learning Module

4.1.1. Trajectory Graph Construction

The sequential visits of the user between POIs can assist us in learning implicit behavior patterns and reasoning between instances. The graph neural network (GNN) has the powerful abilities of knowledge representation and relational reasoning, which can map the check-in trajectory to graph data in complex non-Euclidean space. We define a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to describe the user's historical trajectory, where $\mathcal{V}$ is the graph node representing the POI set P and $\mathcal{E}$ is the directed edge representing the order in which the user visits the POI (e.g., the directed edge $(i, j) \in \mathcal{E}$ represents the path from p_i to p_j).

The adjacency matrix $A \in \mathbb{R}^{|P| \times |P|}$ is used to model the graph $\mathcal{G}$, and the element $a_{i,j}$ represents the transfer relationship from p_i to p_j. Empirically, users are affected by the spatial distance and popularity between locations when making mobile visits. We normalize the spatial correlation matrix and the popularity correlation matrix by column and calculate the correlation weights as the element values of the adjacency matrix, which is shown in (1):

$$a_{i,j} = \begin{cases} (\tau \cdot r_{i,j}^{p\,\prime} + (1-\tau) \cdot r_{i,j}^{s\,\prime})/2 & i \neq j \\ 0 & i = j \end{cases}, \tag{1}$$

$$r_{i,j}^{s\,\prime} = max\left(\frac{r_{*,j}^{s,max} - r_{i,j}^{s}}{r_{*,j}^{s,max} - r_{*,j}^{s,min}}, p_\tau^s \right), \tag{2}$$

$$r_{i,j}^{p\,\prime} = max\left(\frac{r_{i,j}^{p} - r_{*,j}^{p,min}}{r_{*,j}^{p,max} - r_{*,j}^{p,min}}, p_\tau^p \right), \tag{3}$$

where $\tau \in [0, 1]$ is a hyperparameter to control the impact of spatial distance and social popularity. $r_{i,j}^{s\,\prime}$ and $r_{i,j}^{p\,\prime}$ are the column-normalized spatial weights and the column-normalized popularity weights, respectively. $r_{*,j}^{\cdot,max}$ ($r_{*,j}^{\cdot,min}$) is the maximum (minimum) of the j-th

column of the matrix. p_τ^s (p_τ^p) < 1 is the truncated probability for preserving the structural invariance of a graph. It is worth mentioning that we assume that the closer the geographical location, the greater the correlation between POIs. Therefore, we calculated the differential value between the maximum distance between the node j and all neighbors and the distance of the target neighbor to ensure that the spatial weight is inversely proportional to the distance.

4.1.2. Graph Data Augmentation

The purpose of graph data augmentation is to create a new and reasonable graph structure through certain operations to provide auxiliary supervision signals for the model without affecting the basic topology. At present, common data augmentation strategies for graph contrastive learning (GCL) [18] include random node (edge) dropping, feature masking, and random walk, but they have the following boundedness in our task: (1) The general graph augmentation paradigm targets static graph data with relatively stable node and edge attributes, but the user check-in trajectory graph will be dynamically expanded with the unstable factor of user interest. (2) In reality, the user's check-in data have a long-tailed distribution. Unprocessed, random data augmentation operations will reduce the ability of the model to fit user behavior.

In this paper, through the analysis of a large number of user behavior data, we summarize the three implicit behavioral patterns of users: sequential visit, skip visit, and indefinite order visit (refer to Figure 1). Additionally, two edge-based data augmentation operations have been developed. Algorithm 1 describes the process in which the target user u obtains the augmented graph from the trajectory graph. From the historical trajectory graph $\mathcal{G}$, we construct an augmented graph $\hat{\mathcal{G}} = f^a(\mathcal{G})$, where $\hat{A}$ and $\hat{\mathcal{E}}$ are the adjacency matrix and edge set of $\hat{\mathcal{G}}$, respectively. $f^a(\cdot)$ is the data augmentation operation. The details are listed as follows:

Algorithm 1 Graph data augmentation operation for user u.

Input: Historical trajectory graph $\mathcal{G}$, historical check-in data $S(u)$, parameter ρ^m, parameter ρ^c and operation type $\hat{t}$
Output: Augmented graph $\hat{\mathcal{G}}$

```
Extracting adjacency matrix A and in-degree matrix D from trajectory graph G;
Obtain the length N of S(u);
if t̂ == 1 then
    Edge Masking and Bridging: Â = f_1^a(A, ρ^m, N, D);
else if t̂ == 2 then
    Edge Inversion: Â = f_2^a(A, ρ^c, N);
end if
Get augmented graph Ĝ from augmented adjacency matrix Â;
return Ĝ;
```

- ***Edge Masking and Bridging***. By randomly masking the edges in the graph and bridging the nodes at both ends of the obscured edge with each other's forward nodes and backward nodes, the skip visit behavior pattern can be simulated. Specifically, the shadowing probability of the directed edge $(i, j) \in \mathcal{E}$ of $\mathcal{G}$ is defined as $1 - a_{i,j}$, and $\rho^m \times (N - 1)$ edges are randomly shadowed based on the shadowing probability, where $\rho^m < 1$ is a hyperparameter to control the number of edges to be shadowed. In the next step, we bridge the forward and backward nodes at both ends of the shading edge (e.g., Figure 3, assuming that the edge $p_2 \to p_3$ is obscured, p_1 is the forward node of p_2, and p_4 is the backward node of p_3, we will connect $p_1 \to p_3$ and $p_2 \to p_4$ if they do not exist). Finally, the centrality measure $\epsilon = A^\top D^{-1} \in R^{|P|}$ of each node is calculated, where $D \in R^{|P| \times |P|}$ is the in-degree matrix of the node of graph $\mathcal{G}$, and the

centrality score ϵ_j of the target node j is used to obtain the weight of the newly added edge $\hat{a} \in \hat{A}$. The formulas are:

$$\hat{a}_{i^{fore},j} = a_{i^{fore},i} \cdot a_{i,j}, \quad \hat{a}_{i,j^{rear}} = a_{i,j} \cdot a_{j,j^{rear}}, \tag{4}$$

where $*^{fore}$ and $*^{rear}$ are described as the forward and backward nodes of the target node, respectively. $a_{i^{fore},i}$ and $a_{j,j^{rear}}$ are attenuation factors to prevent new edges from being strongly related to all incoming edges of the sinks. Finally, we get the augmented graph $\hat{\mathcal{G}}_1$.

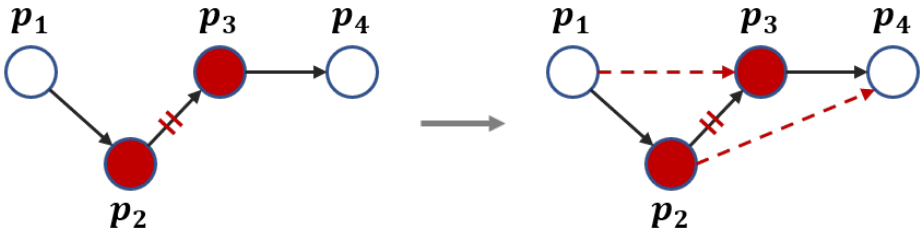


Figure 3. *Edge masking and bridging*. We shadow part of the edge and selectively bridge the forward and backward nodes at both ends.

- ***Edge Inversion***. By randomly selecting the edge and changing the direction of the edge, the indefinite order visit behavior pattern can be simulated. Specifically, the edge weight $a_{i,j}$ represents the importance of users tending to visit the i-th and j-th POIs, so we regard $a_{i,j}$ as the selection probability and randomly select $\rho^c \times (N-1)$ edges, where $\rho^c < 1$ is a hyperparameter to control the number of edges to be selected. We use the reverse operation $reverse(i,j)$ for the selected directed edge $(i,j) \in \mathcal{E}$ to obtain a new edge $(j,i) \in \hat{\mathcal{E}}$, and $\hat{a}_{j,i} = a_{i,j}$. Finally, we get the augmented graph $\hat{\mathcal{G}}_2$.

4.1.3. Graph Preference Representation Encoder (GPRE)

This section proposes GPRE, which consists of an edge-weighted graph attention network (GAT) layer and a graph readout layer. It is a graph encoder that learns the relationship between check-ins in a trajectory graph/augmented graph and outputs graph-level feature representations.

The edge-weighted GAT layer of GPRE will learn the representation vector of nodes from the transformation relationship of nodes in the user's trajectory graph and update it. Specifically, we first perform user-based personalized embedding on graph nodes and obtain a set of node features $X^0 = \{x_1^0, x_2^0, \ldots, x_{|P|}^0\}$. The specific embedding operation is:

$$x_i^0 = e_u \| e_{p_i}, \tag{5}$$

where $x_i^0 \in \mathbb{R}^{2d}$ is the initial node characteristic and $\|$ represents splicing two vectors.

The traditional GAT does not take edge weights into account. Therefore, we embed additional weights between nodes in the edge-weighted GAT layer of GPRE, which makes the updated graph node feature vector contain spatial attributes and popularity attributes. The attention score is nonlinearly activated using the LeakyRelu function, which is calculated as:

$$e_{ij} = LeakyRelu\left(W_a\left[Wx_i \| Wx_j \| W_e a_{i,j}\right]\right), \tag{6}$$

where W_a, W, and W_e are trainable weight matrices. $i \in \mathcal{N}^{(j)}$ is the neighbor node pointing to node j, and $\mathcal{N}^{(j)}$ is the set of neighbor nodes pointing to node j.

Then, in order to make the weights between nodes comparable, we use the softmax function to normalize the attention coefficient in the form of probability:

$$\alpha_{ij} = softmax_i(e_{ij}) = \frac{exp(e_{ij})}{\sum_{m \in \mathcal{N}^{(j)}} exp(e_{mj})}. \tag{7}$$

Finally, we apply the attention coefficient obtained in (7) to calculate the linear combination of the corresponding nodes, and potentially assign nonlinearity to the results to obtain the updated node features. Meanwhile, in order to learn more stable node features, we used the method of multi-head attention recommended in [41]. The specific operation is to perform the attention mechanism multiple times and splice the results. The final paradigm is as follows:

$$x'_j = \Big\Vert_{k=1}^{K} \sigma\Big(\sum_{i \in \mathcal{N}^{(j)}} \alpha_{ij}^k W^k x_i\Big), \tag{8}$$

where K denotes the number of heads, the parameters of each head are independent of each other, $\|$ in (8) denotes the splicing of all heads, and σ is the ReLU activation function.

The vector dimension of the node feature x' after using multi-head GAT is $2Kd$ rather than the original vector dimension $2d$. Therefore, after applying (6)–(8), we add a multi-head GAT layer and discard the stacking operation, instead of calculating the average value of the multi-head as the final graph node to embed x_j^{update}:

$$x_j^{update} = \frac{1}{K} \sum_{k=1}^{K} \sum_{i \in \mathcal{N}^{(j)}} \alpha_{ij}^k W^k x'_i. \tag{9}$$

The graph readout layer of GPRE aims to embed the feature-representation vector of the output graph based on the trajectory graph nodes. Obviously, according to the different user behavior patterns, POIs have personalized high-order order transfer characteristics. Commonly used graph readout operations such as *sum*, *mean*, and *max* cannot make the model learn this feature. Set2Set [42] is a graph pooling strategy based on attention for weighted summation of node embedding, which can learn the possible sequence relationship between nodes. The specific calculation process is:

$$q = \mathrm{LSTM}(0), \tag{10}$$

$$e_i^r = f(x_i^{update}, q), \tag{11}$$

$$\alpha_i^r = softmax_i(e_i^r), \tag{12}$$

$$r = \sum_i \alpha_i^r x_i^{update}, \tag{13}$$

$$g^* = \mathrm{MLP}(q \| r), \tag{14}$$

where $q \in \mathbb{R}^{2d}$ is the query vector for attention, derived from an implicit state generated by an empty input LSTM. The i index represents the i-th node in the user trajectory graph, and the function $f(\cdot)$ is used to calculate the similarity e_i^r between the embedding x_i^{update} of each node and the query vector (the function is the dot product in this model). α_i^r is a probabilistic attention coefficient after softmax normalization of similarity, which is used to calculate a linear combination with each node embedding x_i^{update}. The final output graph represents the vector g^*, which is obtained by concatenating the query vector q and the output result $r \in \mathbb{R}^{2d}$ of a forward propagation. In particular, in order to ensure the dimensional consistency between the graph representation vector and the POI embedding vector, we use a multi-layer perceptron to ensure $g^* \in \mathbb{R}^d$.

To date, we can simplify (6)~(14) to the general paradigm of GPRE:

$$g^* = encoder(X^0, \mathcal{A}), \tag{15}$$

where $\mathcal{A}$ is the adjacency matrix of any trajectory graph. By inputting the adjacency matrices A, $\hat{A}_1$, and $\hat{A}_2$ of $\mathcal{G}$, $\hat{\mathcal{G}}_1$, and $\hat{\mathcal{G}}_2$ into GPRE, respectively, we can obtain g^*, $\hat{g}_1^*$, and $\hat{g}_2^*$, which are described as the graph feature-representation vectors of $\mathcal{G}$, $\hat{\mathcal{G}}_1$, and $\hat{\mathcal{G}}_2$, respectively. Formally, for $\mathcal{G}$ and $\hat{\mathcal{G}}_1$, we use the online encoder ε_θ, and for $\hat{\mathcal{G}}_2$, we use the target encoder ε_ϕ.

4.1.4. Graph Contrastive Learning

An augmented graph based on the potential behavior pattern modeling plays an important role in learning users' preferences. Differently from the general contrastive learning paradigm, we consider the augmented graphs $\hat{\mathcal{G}}_1$ and $\hat{\mathcal{G}}_2$ as positive pairs, aiming to maximize the consistency of features of the two positive-pair views in the feature space through the loss function. Formally, we follow BGRL [43,44], which is non-contrastive and does not require any negative examples. It can be seen in Figure 2 that for the online encoder ε_θ, we use the nonlinear transformation module Converter (FC $\rightarrow$ PReLu $\rightarrow$ FC) to process $\hat{g}_1^*$ to obtain $\hat{z}_1^*$, which makes the structure of the online pipeline and the target pipeline asymmetric. We define a contrastive learning loss function:

$$\mathcal{L}_{gcl} = 2 - 2 \cdot \langle \hat{z}_1^*, \hat{g}_2^* \rangle / (\|\hat{z}_1^*\|_2 \cdot \|\hat{g}_2^*\|_2). \tag{16}$$

The loss function $\mathcal{L}_{gcl}$ updates the parameter θ of the online encoder ε_θ by following a gradient of cosine similarity.

On the other side, we update the momentum of parameter ϕ of the target encoder ε_ϕ to be the exponential moving average (EMA) of the online parameter θ:

$$\phi \leftarrow \beta\phi + (1 - \beta)\theta, \tag{17}$$

where $\beta \in [0, 1)$ is a hyperparameter, defined as the decay rate that controls the distance between ϕ and θ.

4.2. Dynamic Preference-Learning Module

The dynamic preference-learning module plays a key role in capturing users' short-term sequential action patterns and context-aware patterns, such as the periodicity of daily behavior within a week. In this paper, we benefit from the help of the user's check-in timestamp and intercept the latest week's visit from the user's complete check-in trajectory as a short-term check-in sequence $S'(u) = \{c_m, c_{m+1}, \ldots, c_N\}$, where $S'(u) \subseteq S(u)$ and $t_m \geq t_N - (7 \times 24) > t_{m-1}$. Here, the timestamp t in hours is modulo 24, mapping each check-in to 24 dimensions, representing 24 h in a day to capture the periodicity of short-term behavior patterns. Naturally, we get the embedding $e_{c_i} = e_u + e_{p_i} + e_{t_i}$ of check-in $c_i \in S'(u)$ by linear summation; e_u and e_{p_i} are the embeddings of user and POI, respectively; and $e_{t_i} \in \mathbb{R}^d$ denotes check-in time embedding. The embedding of a short-term check-in sequence is denoted by $E_{S'} = \{e_{c_m}, e_{c_{m+1}}, \ldots, e_{c_N}\} \in \mathbb{R}^{N' \times d}$, where N' is the length of $S'(u)$.

Benefiting from the success of self-attention mechanism in sequence recommendation, we propose an adaptive self-attention-based dynamic preference-learning module, aiming to learn the high-order correlations among user check-ins in the short term. Self-attention can capture the dependencies between check-ins within a trajectory and calculate the weights of each visit relative to the most recent one in an end-to-end manner. Moreover, by taking into account the influence of the check-in time interval, geographical distance between POIs, and social popularity of POIs on user decision making, such as users preferring to go to nearby and popular places, we incorporate multiple feature embeddings into weight calculation. Specifically, in order to model the complex context relationship between check-ins, a mask layer of length N is set, and the check-ins belonging to $S'(u)$ are set to one (otherwise zero). The purpose is to extract the short-term preference correlation matrix from the temporal, spatial, and popularity correlation matrices, which are $R^{t'}, R^{s'}, R^{p'} \in \mathbb{R}^{N' \times N'}$. The d-dimensional embedding matrix $E^t_{R'}, E^s_{R'}, E^p_{R'} \in \mathbb{R}^{N' \times N' \times d}$ corresponding to the short-

term preference correlation matrix is obtained by interpolation embedding method [45]. Furthermore, the short-term check-ins are embedded into a matrix $E_{S'}$, multiplied by three different trainable parameter matrices V_1, V_2, and V_3 to obtain query vector Q, key vector K, and value vector V, and then update the embedded short-term check-ins:

$$E^*_{S'} = softmax\left(\frac{QK^\top + E^t_{R'} + E^s_{R'} + E^p_{R'}}{\sqrt{d}}\right)V. \tag{18}$$

Here, $E^*_{S'} \in \mathbb{R}^{N' \times d}$ is the embedded representation of the updated short-term check-in trajectory of the user. Since the last check-in c_N is closest to the user's next visit, we use the last check-in embedded representation $e^*_{c_N}$ as the user's short-term preference.

4.3. Prediction Layer

As shown in Figure 2, the trajectory graph feature representation g^* and the short-term preference feature $e^*_{c_N}$ are spliced to obtain the user preference representation $l^* \in \mathbb{R}^{2\times d}$, and then l^* and candidate POI are embedded into $e^*_p \in \mathbb{R}^{|P|\times d}$ as input into the next location prediction module. The prediction module will output the probability that the candidate location is visited:

$$\hat{y} = softmax_p(W_o(e^*_p l^{*\top})). \tag{19}$$

Here, W_o is the weight matrix that can be learned, and the softmax function outputs the result in a probabilistic form. Finally, we apply the popular cross entropy as the loss of our supervised task to optimize the model's parameters:

$$\mathcal{L}_{main} = -\sum_{u\in U}\sum_{i=1}^{|P|} y_i \cdot \log(\hat{y}_i) + (1-y_i)\cdot \log(1-\hat{y}_i), \tag{20}$$

where the index i represents the i-th candidate POI and $y_i \in 0, 1$ is the label of the i-th candidate POI. In order to optimize our recommendation model through graph self-supervised tasks, we use a multi-task training strategy for collaborative optimization of parameters.

$$\mathcal{L} = \mathcal{L}_{main} + \lambda_1 \mathcal{L}_{gcl} + \lambda_2 \|\eta\|_2^2 + \lambda_3 \|\theta\|_2^2. \tag{21}$$

In this formula, η and θ are the parameter sets of the supervised model and the self-supervised model, respectively; and λ_1, λ_2, and λ_3 are the hyperparameters which control the influences of graph self-supervised learning and L_2 regularization, respectively.

5. Experiments

In this section, the experimental results are presented. All experimental results are displayed in the form of tables and charts.

5.1. Experimental Setup

5.1.1. Datasets

We evaluated our proposed BPGCL recommendation framework on three real-world LSBN datasets, the popular Gowalla, TKY, and NYC, respectively; and all three benchmark datasets are publicly accessible. In the experiment, given the differences in the rich context categories provided by the dataset, we only used the four common raw data of user, POI, check-in time, and spatial coordinates (longitude and latitude) of POI in the dataset. In order to ensure data quality, based on previous research work experience, we applied a 10-core setting for each dataset, that is, to retain users who have interacted with at least 10 POIs and POIs that have interacted with at least 10 users. Table 1 summarizes the detailed information of the dataset. In terms of dataset partitioning, we selected the top 90% check-in records of each user as the training set and the rest as the test set based on the

timestamp. In addition, 10% of the check-in records from the training set were selected as the validation set to adjust the hyperparameters.

Table 1. Experimental dataset's statistical details.

Dataset	#User	#POI	#Check-in
Gowalla	6265	19532	512332
TKY	2195	5491	303591
NYC	1063	5135	147938

5.1.2. Baselines

In order to verify the effectiveness of the proposed model, we compare it with the following supervised task baselines:

- **LSTM** [46]: A widely used classical sequential modeling method.
- **ST-RNN** [20]: An effective recommendation method for extending the RNN by modeling local temporal and spatial contexts.
- **DeepMove** [6]: An attention-based recursive network for mobility prediction from long and sparse trajectories.
- **STGN** [22]: This is a state-of-the-art approach that extends the time and distance gates in LSTM to personalize recommendations by modeling users' long- and short-term preferences in a spatial-temporal context.
- **ASGNN** [28]: This is an attention sequence model based on GNN that models user check-in behavior as a graph and updates the check-in feature representation locally through a gated graph neural network for recommendation.
- **LSTPM** [21]: A state-of-the-art model for joint recommendation through a non-local network for long-term preference modeling and a geographically extended RNN for short-term preference learning.
- **STAN** [47]: This is the state-of-the-art model that multi-modally models personalized trajectory and uses a self-attention combined with a spatial-temporal margin matrix to model the association between non-adjacent locations and discontinuous visits.

5.1.3. Evaluation Matrices

To evaluate the effectiveness of the model's TOP-K recommendation and recommendation ranking, we applied two widely used evaluation protocols: Recall@K and NDCG@K. The recall rate is described as the proportion of POI hits in all recommendation results. The larger the recall rate, the better the recommendation effect. NDCG is a normalized discounted cumulative gain, which is often used in recommendation and search tasks. It can measure the ranking order of result in the recommendation list. The larger the NDCG value, the better the effect. In this paper, we use $K \in \{5, 10\}$ to evaluate the index parameters, and the evaluation results are averages.

5.1.4. Implementation Details

To ensure fairness, we applied the same benchmark dimension d to all embedding and weighting matrices. Specifically, for the datasets Gowalla, TKY, and NYC, we set d to 50. The truncation probabilities p_τ^s and p_τ^p were set to 0.01. Following the setting in [43], we set the momentum temperature coefficient β to 0.99. Note that the number of heads K is 8 by default. In our model, we implemented GSBPL with Pytorch and carefully tuned its key parameters. For the remaining hyperparameters, we performed grid searching: for λ_2 and λ_3, we took values in the range of $[0, 1]$; the masking control ratios ρ^m and ρ^c in the data augmentation were searched in the range of $[0, 0.2]$ to change the structure of the original trajectory as little as possible. We took λ_1 in $\{0.01, 0.05, 0.1, 0.5, 1.0\}$; τ took values in $\{0, 0.2, 0.5, 0.8, 1\}$ and was analyzed experimentally. We used Adam as the model optimizer. The learning rate was 0.002, and the training epoch was 30.

5.2. Performance Comparison

Table 2 shows the recommended performance comparison between the proposed model and the baseline. We used a T-test with a p-value of 0.01 to evaluate the statistical significance of GSBPL improvement. We set the best result as bold, and in the last line, we list the performance improvements of our model. We observe the following conclusions from the statistics:

- Our proposed model exceeded the baseline in terms of each metric on three datasets with different densities. For example, in the check-in-data-rich NYC dataset, GSBPL's Recall@5 and NDCG@5 were 18.94% and 13.92% higher than those of our competitors, respectively. In particular, we achieved 14.99% and 14.06% improvements in the TKY dataset with comparable check-in density, which verifies the superiority of our proposed method in active datasets. In addition, the performance improvement of almost 7.3% in the Gowalla dataset also proves the effectiveness of GSBPL at dealing with sparse data. We consider that the significant improvement of the model is due to the powerful ability of self-supervised learning of implicit behavior patterns.
- As a classical deep learning method for processing sequences, LSTM has stable performance in this recommendation scenario, but it does not outperform and ST-RNN, DeepMove, or STGN. This was not a surprise, since these three models take into account the impacts of time slices and relative distances on users' future visits in the long-term preference modeling of the RNN framework. In addition, the recommendation performances of these methods are far less than that of LSTPM based on the RNN. It is reasonable, since LSTPM fine-grained divides users' preferences into long- and short-term preferences, and geographically expands the short-term preference RNN framework with a stronger correlation. However, these RNN-based methods cannot fit the complex behavior patterns of users due to their concatenated structure, resulting in sub-optimal performance.
- ASGNN is a simple GNN-based model, and its result is similar to that of LSTPM. This demonstrates the ability of GNN to capture the transfer patterns between POIs. Additionally, we believe that ASGNN can be improved to have a performance level beyond that of LSTPM. It applies the gated graph neural network only for the most basic node-level message propagation, without embedding the POI distance context. However, ASGNN ignores the key role of trajectory representation, which limits its performance.
- The state-of-the-art model STAN expresses significant advantages compared to the baseline. This is reasonable because the previous model ignores the correlation among non-adjacent locations and discontinuous visits and user behavior, and STAN uses the self-attention to calculate the spatial-temporal correlation between each node, which solves the defects caused by the series structure of the RNN-based method. Although effective, STAN dilutes the order dependence between POIs and cannot model the implicit behavior patterns of users. In addition, STAN ignores the impact of a location's social popularity on users' short-term dynamic decisions.

Table 2. Recommended performance comparison with baseline.

	Gowalla				**TKY**				**NYC**			
	Recall@5	**Recall@10**	**NDCG@5**	**NDCG@10**	**Recall@5**	**Recall@10**	**NDCG@5**	**NDCG@10**	**Recall@5**	**Recall@10**	**NDCG@5**	**NDCG@10**
LSTM	0.1576	0.1733	0.1388	0.1491	0.1703	0.1884	0.1501	0.1673	0.2123	0.2741	0.1788	0.1835
ST-RNN	0.1791	0.2279	0.1398	0.1426	0.1987	0.2161	0.1776	0.1877	0.2367	0.2803	0.1991	0.2082
STGN	0.1817	0.2381	0.1408	0.1513	0.2178	0.2387	0.1848	0.1954	0.2608	0.3262	0.2193	0.2405
DeepMove	0.1973	0.2475	0.1487	0.1607	0.2398	0.2587	0.1998	0.2175	0.2909	0.4084	0.2276	0.2554
ASGNN	0.2087	0.2510	0.1754	0.1909	0.2622	0.3241	0.2205	0.2498	0.3476	0.4209	0.2544	0.2801
LSTPM	0.2121	0.2563	0.1626	0.1826	0.2894	0.3479	0.2174	0.2411	0.3369	0.4119	0.2516	0.2776
STAN	0.2386	0.2811	0.1854	0.2087	0.3147	0.3941	0.2353	0.2613	0.4012	0.4953	0.2909	0.3209
GSBPL	**0.2561**	**0.3016**	**0.2007**	**0.2231**	**0.3619**	**0.4478**	**0.2684**	**0.2947**	**0.4772**	**0.5798**	**0.3314**	**0.3615**
Improvement	7.33%	7.29%	8.25%	6.89%	14.99%	13.62%	14.06%	12.78%	18.94%	17.06%	13.92%	12.65%

5.3. Ablation Study

We conducted ablation studies to analyze the effects of the modules in our model. We removed the core components of GSBPL in turn to form different derivative models. These derivative models were as follows:

- GSBPL$_{r-gsl}$: This model removes the graph data augmentation and contrastive loss and only retains the representation vector of the GPRE generated trajectory graph.
- GSBPL$_{r-g}$: This model removes the whole trajectory learning module and only considers the influence of users' short-term dynamic preferences.
- GSBPL$_{r-spop}$: This model removes the embedding of POI social popularity features in the short-term dynamic preference module.
- GSBPL$_{r-s}$: This model removes the entire short-term dynamic preferences module and only uses the trajectory graph learning module (including the part of graph contrastive learning) for recommendation.

Table 3 illustrates the results of the ablation study. We can draw the following conclusions from the observation of the results. Firstly, after we discarded the entire trajectory-learning module, the performance of the derived model GSBPL$_{r-g}$ was significantly different from that of GSBPL, which proves that the trajectory-learning module plays a key role in performance improvement. Secondly, the performances of GSBPL$_{r-spop}$ and GSBPL$_{r-s}$ on the two datasets were lower than that of GSBPL, and GSBPL$_{r-spop}$ obtained 2–6% higher results than GSBPL$_{r-s}$. The data not only prove the effectiveness of the short-term dynamic preference module, but also express that the social popularity of POI can dynamically affect users' temporary decisions. Finally, we can also observe from the performance reduction after removing the GSBPL$_{r-gsl}$ module that the optimization of our recommendations by the graph self-supervised contrastive learning framework is obvious. In addition to the above conclusions, we can also note that the performance degradation of GSBPL$_{r-gsl}$ on the Gowalla dataset was lower than that on the NYC dataset. One of the reasons we guessed is that it is relatively difficult for GSBPL to capture users' implicit behavior patterns in sparse datasets.

Table 3. Ablation study of different derivative models of GSBPL.

Model	Gowalla		NYC	
	Recall@5	**Recall@10**	**Recall@5**	**Recall@10**
GSBPL$_{r-gsl}$	0.2416	0.2927	0.4193	0.5166
GSBPL$_{r-g}$	0.1797	0.2183	0.2399	0.2954
GSBPL$_{r-spop}$	0.2378	0.2787	0.4511	0.5382
GSBPL$_{r-s}$	0.2241	0.2623	0.4216	0.5239
GSBPL	0.2561	0.3016	0.4772	0.5798

5.4. Impact of Hyperparameters

Figure 4 demonstrates the experimental results regarding the effects of parameters. We study the influences of three key hyperparameters on the proposed model, including the hyperparameter λ_1 that controls the intensity of the contrastive loss in the loss function, the influence coefficient τ of the control space distance and social popularity in the forward propagation of nodes, and the embedding dimension d. The settings of other hyperparameters were relatively fixed, so they are no longer described.

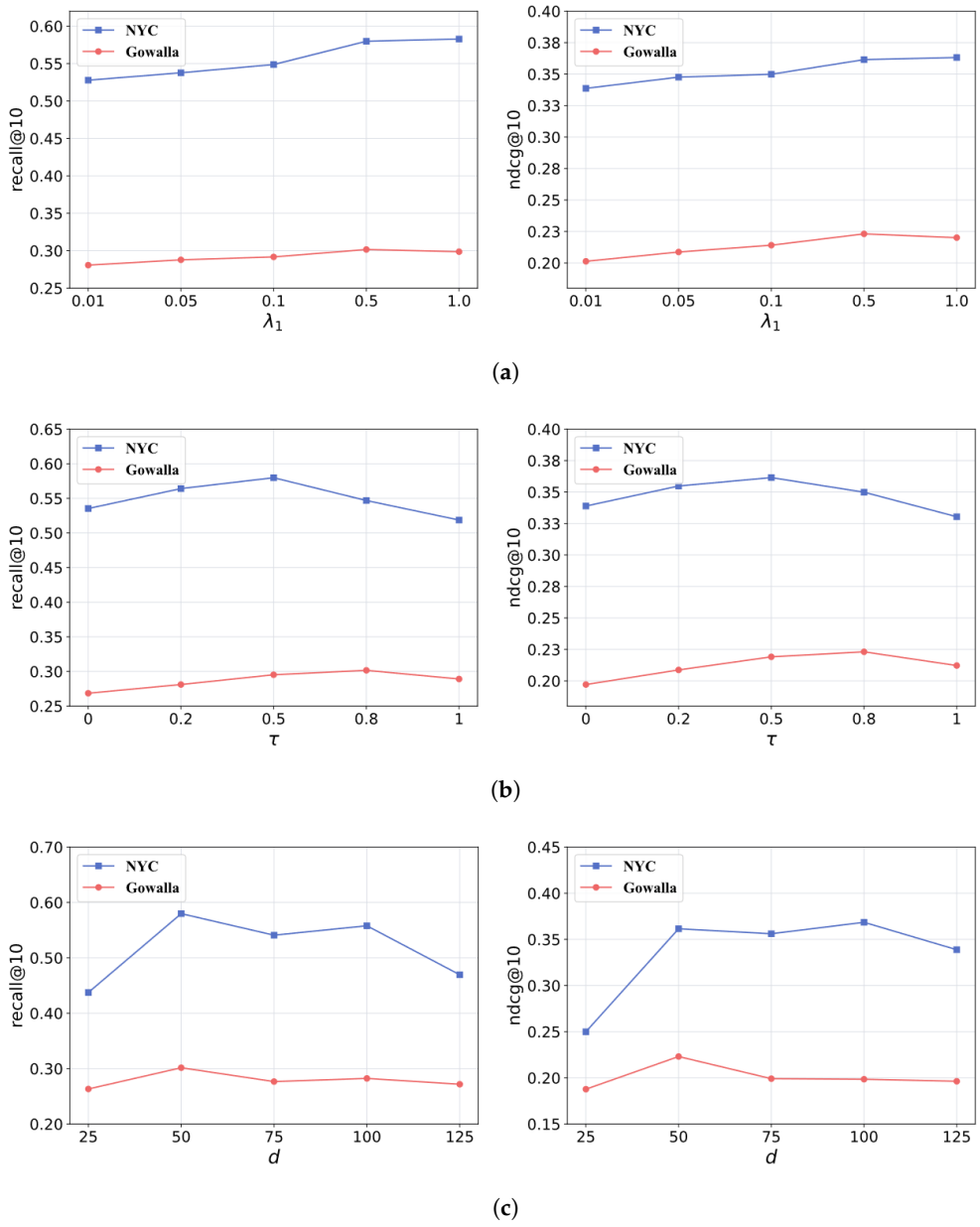


Figure 4. Impact evaluation of hyperparameters: (**a**) Impact of Parameter λ_1; (**b**) Impact of Parameter τ; (**c**) Impact of Parameter d.

5.4.1. Impact of Graph Contrastive Learning

In order to further verify the importance and effectiveness of contrastive learning in GSBPL, we conducted in-depth analysis on two datasets from Recall@10 and NDCG@10 by adjusting the value of hyperparameter λ_1. We assigned λ_1 in $\{0.01, 0.05, 0.1, 0.5, 1.0\}$. By observation, the performance improvement of our model was not obvious when $\lambda_1 \leq 0.1$, because our model is mainly guided by the supervised task to propagate the gradient back. A good result was obtained when λ_1 was set to 0.5 and 1.0. It is worth noting that we can see that the performance in the Gowalla dataset will decrease slightly when $\lambda_1 = 1.0$, which confirms our conclusion in Section 5.3 that it is difficult to capture implicit behavior patterns through contrastive learning on sparse check-in datasets. Thus, we set λ_1 to 0.5 in the experiment.

5.4.2. Impact of Spatial Distance and Social Popularity

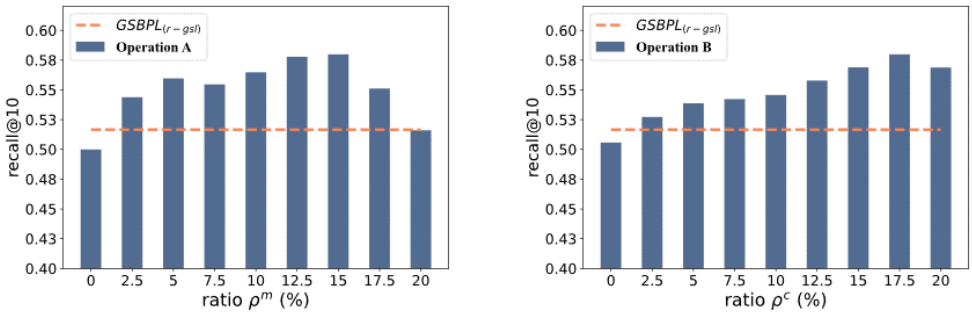
We observed the influences of spatial distance and social popularity on recommendation performance by adjusting the control coefficient τ in the node propagation of the trajectory graph. Taking the value of τ in turn as $\{0, 0.2, 0.5, 0.8, 1\}$, this setting enabled us to accurately evaluate the extreme states and equilibrium states of the two factors. By observation, when the τ values were zero and one, the performance was relatively low. The best performance was obtained when $\tau = 0.5$ in the NYC dataset, and the performance was better when $\tau = 0.8$ in the Gowalla dataset. This reflects that spatial distance and social popularity have different effects on user decision making in different datasets. We attribute this change to the different groups and strategic positioning of different LBSNs. Therefore, users are more inclined to check in popular locations in one LBSN and check in daily trajectories in another LBSN.

5.4.3. Impact of the Embedding Dimension

The influence of the embedding dimension on the performance of proposed model is significant. In order to explore the effect of the embedding dimension on GSBPL, we selected a group of values within a reasonable range for experimentation. Specifically, we assigned $\{25, 50, 75, 100, 125\}$ to the embedding dimension d sequentially. As shown in Figure 4c, the comprehensive performance of GSBPL is optimal when the embedding size is 50. Specifically, the model is insensitive to changes in hyperparameters in the range of $[50, 100]$, though the performance of the model cannot be fully exerted when $d = 25$, owing to the fact that a too-low embedding size cannot fully represent the characteristics of POIs. It is worth noting that the model's recommendation performance on both datasets showed a declining trend when d reaches 125, which may be caused by overfitting due to the large embedding dimensions and many parameters.

5.5. Effectiveness of Graph Augmentation Operation

In order to further investigate whether the graph expansion strategy has an impact on the GSBPL recommendation results and the impact degree, we evaluated it on a representative NYC dataset. We set *Edge Masking and Bridging* to graph augmentation operation A and *Edge Inversion* to graph augmentation operation B. Figure 5 illustrates our experimental results. The yellow dotted line is the effect of the derivative model GSBPL$_{r-gsl}$ in Section 5.3.



(**a**) (**b**)

Figure 5. Effectiveness visualization of data augmentation: (**a**) Operation A; (**b**) Operation B.

Figure 5a shows that the excessive enhancement of operation A will lead to the deterioration of the model's performance, though the effect is lower than that of the derivative model GSBPL$_{r-gsl}$. It works best when $\rho^m = 0.15$. On the other hand, Figure 5b illustrates that the Recall@10 of operation B is generally higher than that of GSBPL$_{r-gsl}$, which indicates that the user's visit order to some POIs is not fixed, which verifies the effectiveness of operation B. In particular, we found that the performance improvement of operation A is generally higher than that of operation B, even if the line chart of operation A

is unstable. This demonstrates that the implicit behavior pattern of skip visit has a stronger impact on the next check-in than indefinite order visiting.

6. Conclusions

In this paper, we proposed a graph self-supervised learning model GSBPL for recommending the next point of interest. Firstly, based on real check-in data, we proposed the concept of implicit behavior patterns and designed two graph data augmentation operations to generate augmented trajectory graphs for modeling implicit behavior patterns. Then, we proposed a graph preference representation encoder (GPRE), to learn the high-order representation of the enhanced trajectory graphs and adopted an asymmetric graph contrastive learning architecture based on positive examples to learn personalized behavior patterns of users. Finally, we designed a multi-feature self-attention mechanism to learn users' short-term dynamic preferences and recommend the next POI by combining user trajectories' graph representations. We conducted extensive experiments on three popular LBSN datasets, and the results prove the great competitiveness and effectiveness of GSBPL compared to supervised learning tasks.

The following improvements are listed as future work to address the limitations of the model: (1) Introduction of POI side information (e.g., POI categories) into the trajectory graph representation learning process to enhance performance in sparse datasets and alleviate the cold start problem. (2) Exploration of the possibility of moving the graph contrastive learning module to the model pre-training stage to achieve model decoupling and improve recommendation efficiency.

Author Contributions: Conceptualization D.W. and M.S.; methodology D.W.; software, D.W. and C.C.; validation, D.W., C.C. and C.D.; formal analysis, D.W.; investigation, M.S.; resources, M.S.; data curation, D.W.; writing—original draft preparation, D.W.; writing—review and editing, D.W., C.C., C.D. and M.S.; visualization, D.W.; supervision, M.S.; project administration, D.W.; funding acquisition, C.C., C.D. and M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the project ZR2020QF020 and ZR2022QF042 supported by Shandong Provincial Natural Science Foundation, and in part by the National Natural Science Foundation of China (Grant No. 52205020 and 62206144).

Data Availability Statement: Data sharing not applicable. No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, Z.; Li, C.; Wu, Z.; Sun, A.; Ye, D.; Luo, X. Next: A neural network framework for next poi recommendation. *Front. Comput. Sci.* **2020**, *14*, 314–333. [CrossRef]
2. Ren, J.; Gan, M. Mining dynamic preferences from geographical and interactive correlations for next-POI recommendation. *Knowl. Inf. Syst.* **2023**, *65*, 183–206. [CrossRef]
3. Rendle, S.; Freudenthaler, C.; Schmidt-Thieme, L. Factorizing personalized markov chains for next-basket recommendation. In Proceedings of the 19th International Conference on World Wide Web, Raleigh, NC, USA, 26–30 April 2010; pp. 811–820.
4. Cheng, C.; Yang, H.; Lyu, M.R.; King, I. Where you like to go next: Successive point-of-interest recommendation. In Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, Beijing, China, 3–9 August 2013.
5. Zhang, J.D.; Chow, C.Y.; Li, Y. Lore: Exploiting sequential influence for location recommendations. In Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Dallas/Fort Worth, TX, USA, 4–7 November 2014; pp. 103–112.
6. Feng, J.; Li, Y.; Zhang, C.; Sun, F.; Meng, F.; Guo, A.; Jin, D. Deepmove: Predicting human mobility with attentional recurrent networks. In Proceedings of the 2018 World Wide Web Conference, Lyon, France, 23–27 April 2018; pp. 1459–1468.
7. Cui, Q.; Zhang, C.; Zhang, Y.; Wang, J.; Cai, M. 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, Virtual Event, 1–5 November 2021; pp. 2960–2964.
8. Wu, Y.; Li, K.; Zhao, G.; Qin, X. Personalized long-and short-term preference learning for next POI recommendation. *IEEE Trans. Knowl. Data Eng.* **2020**, *99*, 1. [CrossRef]

9. Zhao, P.; Zhu, H.; Liu, Y.; Li, Z.; Xu, J.; Sheng, V.S. Where to go next: A spatio-temporal LSTM model for next POI recommendation. *arXiv* **2018**, arXiv:1806.06671.
10. Zhang, Y.; Lan, P.; Wang, Y.; Xiang, H. Spatio-Temporal Mogrifier LSTM and Attention Network for Next POI Recommendation. In Proceedings of the 2022 IEEE International Conference on Web Services (ICWS), Barcelona, Spain, 10–16 July 2022; pp. 17–26.
11. Wang, E.; Jiang, Y.; Xu, Y.; Wang, L.; Yang, Y. Spatial-Temporal Interval Aware Sequential POI Recommendation. In Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE), Virtual Event, 9–12 May 2022; pp. 2086–2098.
12. Zhou, H.; Tan, Q.; Huang, X.; Zhou, K.; Wang, X. Temporal augmented graph neural networks for session-based recommendations. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, 11–15 July 2021; pp. 1798–1802.
13. Chang, B.; Jang, G.; Kim, S.; Kang, J. Learning graph-based geographical latent representation for point-of-interest recommendation. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, Virtual Event, 19–23 October 2020; pp. 135–144.
14. Li, Y.; Chen, T.; Yin, H.; Huang, Z. Discovering collaborative signals for next-POI recommendation with iterative Seq2Graph augmentation. *arXiv* **2021**, arXiv:2106.15814.
15. Li, Z.; Cheng, W.; Xiao, H.; Yu, W.; Chen, H.; Wang, W. 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, Virtual Event, 1–5 November 2021; pp. 3945–3954.
16. Cai, Z.; Yuan, G.; Qiao, S.; Qu, S.; Zhang, Y.; Bing, R. FG-CF: Friends-aware graph collaborative filtering for POI recommendation. *Neurocomputing* **2022**, *488*, 107–119. [CrossRef]
17. Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; Wang, L. Graph contrastive learning with adaptive augmentation. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 2069–2080.
18. You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; Shen, Y. Graph contrastive learning with augmentations. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 5812–5823.
19. You, Y.; Chen, T.; Shen, Y.; Wang, Z. Graph contrastive learning automated. In Proceedings of the International Conference on Machine Learning. PMLR, Virtual Event, 18–24 July 2021; pp. 12121–12132.
20. Liu, Q.; Wu, S.; Wang, L.; Tan, T. Predicting the next location: A recurrent model with spatial and temporal contexts. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
21. Sun, K.; Qian, T.; Chen, T.; Liang, Y.; Nguyen, Q.V.H.; Yin, H. 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, New York Hilton Midtown, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 214–221.
22. Zhao, P.; Luo, A.; Liu, Y.; Zhuang, F.; Xu, J.; Li, Z.; Sheng, V.S.; Zhou, X. Where to go next: A spatio-temporal gated network for next poi recommendation. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, s2512–s2524. [CrossRef]
23. Wang, X.; He, X.; Wang, M.; Feng, F.; Chua, T.S. Neural graph collaborative filtering. In Proceedings of the 42nd International ACM SIGIR Conference on Research and development in Information Retrieval, Paris, France, 21–25 July 2019; pp. 165–174.
24. He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; Wang, M. Lightgcn: Simplifying and powering graph convolution network for recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, China, 25–30 July 2020; pp. 639–648.
25. Mao, K.; Zhu, J.; Xiao, X.; Lu, B.; Wang, Z.; He, X. UltraGCN: Ultra simplification of graph convolutional networks for recommendation. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Virtual Event, QLD, Australia, 1–5 November 2021; pp. 1253–1262.
26. Wang, X.; Jin, H.; Zhang, A.; He, X.; Xu, T.; Chua, T.S. Disentangled graph collaborative filtering. In Proceedings of the 43rd international ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, China, 25–30 July 2020; pp. 1001–1010.
27. Yinwei, W.; Xiang, W.; Liqiang, N.; Xiangnan, H.; Tat-Seng, C. GRCN: Graph-Refined Convolutional Network for Multimedia Recommendation with Implicit Feedback. *arXiv* **2021**, arXiv:2111.02036.
28. Wang, D.; Wang, X.; Xiang, Z.; Yu, D.; Deng, S.; Xu, G. Attentive sequential model based on graph neural network for next poi recommendation. *World Wide Web* **2021**, *24*, 2161–2184. [CrossRef]
29. Li, Y.; Tarlow, D.; Brockschmidt, M.; Zemel, R. Gated graph sequence neural networks. *arXiv* **2015**, arXiv:1511.05493.
30. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9729–9738.
31. Zeng, J.; Xie, P. Contrastive self-supervised learning for graph classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Event, 22 February–1 March 2021; Volume 35, pp. 10824–10832.
32. Wu, D.; Luo, X.; Guo, X.; Chen, C.; Deng, M.; Ma, J. Concordant Contrastive Learning for Semi-supervised Node Classification on Graph. In Proceedings of the International Conference on Neural Information Processing, Sanur, Bali, Indonesia, 8–12 December 2021; pp. 584–595.
33. Qiu, J.; Chen, Q.; Dong, Y.; Zhang, J.; Yang, H.; Ding, M.; Wang, K.; Tang, J. Gcc: Graph contrastive coding for graph neural network pre-training. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual Event, 6–10 July 2020; pp. 1150–1160.

34. Wu, J.; Wang, X.; Feng, F.; He, X.; Chen, L.; Lian, J.; Xie, X. Self-supervised graph learning for recommendation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, 11–15 July 2021; pp. 726–735.
35. Li, H.; Luo, X.; Yu, Q.; Wang, H. Session-based Recommendation via Contrastive Learning on Heterogeneous Graph. In Proceedings of the 2021 IEEE International Conference on Big Data (Big Data), Orlando, FL, USA, 15–18 December 2021; pp. 1077–1082.
36. Yu, J.; Yin, H.; Xia, X.; Cui, L.; Nguyen, Q.V.H. Graph Augmentation-Free Contrastive Learning for Recommendation. *arXiv* **2021**, arXiv:2112.08679.
37. Xie, R.; Liu, Q.; Wang, L.; Liu, S.; Zhang, B.; Lin, L. Contrastive cross-domain recommendation in matching. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 14–18 August 2022; pp. 4226–4236.
38. Liu, Z.; Ma, Y.; Hildebrandt, M.; Ouyang, Y.; Xiong, Z. CDARL: A contrastive discriminator-augmented reinforcement learning framework for sequential recommendations. *Knowl. Inf. Syst.* **2022**, *64*, 2239–2265. [CrossRef]
39. Kosinski, M.; Stillwell, D.; Graepel, T. Private traits and attributes are predictable from digital records of human behavior. *Proc. Natl. Acad. Sci. USA* **2013**, *110*, 5802–5805. [CrossRef] [PubMed]
40. Wang, Z.; Zhu, Y.; Zhang, Q.; Liu, H.; Wang, C.; Liu, T. Graph-enhanced spatial-temporal network for next POI recommendation. *ACM Trans. Knowl. Discov. Data (TKDD)* **2022**, *16*, 1–21. [CrossRef]
41. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
42. Vinyals, O.; Bengio, S.; Kudlur, M. Order matters: Sequence to sequence for sets. *arXiv* **2015**, arXiv:1511.06391.
43. Thakoor, S.; Tallec, C.; Azar, M.G.; Munos, R.; Veličković, P.; Valko, M. Bootstrapped representation learning on graphs. In Proceedings of the ICLR 2021 Workshop on Geometrical and Topological Representation Learning, Virtual Event, 7 May 2021.
44. Grill, J.B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.; Buchatskaya, E.; Doersch, C.; Avila Pires, B.; Guo, Z.; Gheshlaghi Azar, M.; et al. Bootstrap your own latent-a new approach to self-supervised learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21271–21284.
45. Liu, Q.; Liu, Z.; Zhang, H. An empirical study on feature discretization. *arXiv* **2020**, arXiv:2004.12602.
46. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput*. **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
47. Luo, Y.; Liu, Q.; Liu, Z. Stan: Spatio-temporal attention network for next location recommendation. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 2177–2185.