Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Toward an accurate mobility trajectory recovery using contrastive learning*

Yushan LIU[1], Yang CHEN[†‡1], Jiayun ZHANG[1], Yu XIAO[2], Xin WANG[1]

[1]*Shanghai Key Lab of Intelligent Information Processing, School of Computer Science, Fudan University, China*

[2]*Department of Information and Communications Engineering, Aalto University, Finland*

[†]E-mail: chenyang@fudan.edu.cn

Received Sept. 21, 2023; Revision accepted Feb. 2, 2024; Crosschecked

Abstract: Human mobility trajectories are fundamental resources for analyzing mobile behaviors in urban computing applications. However, these trajectories, typically collected from location-based services, often suffer from sparsity and irregularity in time. To support the development of mobile applications, there is a need to recover or estimate missing locations of unobserved time slots in these trajectories at a fine-grained spatialâĂŞtemporal resolution. Existing methods for trajectory recovery rely on either individual user trajectories or collective mobility patterns from all users. The potential to combine individual and collective patterns for precise trajectory recovery remains unexplored. Additionally, current methods are sensitive to the heterogeneous temporal distributions of the observable trajectory segments. In this paper, we propose CLMove (where CL stands for contrastive learning), a novel model designed to capture multilevel mobility patterns and enhance robustness in trajectory recovery. CLMove features a two-stage location encoder that captures collective and individual mobility patterns. The graph neural network (GNN)-based networks in CLMove explore location transition patterns within a single trajectory and across various user trajectories. We also design a trajectory-level contrastive learning task to improve the robustness of the model. Extensive experimental results on three representative real-world datasets demonstrate that our CLMove model consistently outperforms state-of-the-art methods in terms of accuracy of trajectory recovery.

Key words: Human mobility; Mobility trajectory recovery; Contrastive learning.
https://doi.org/10.1631/FITEE.2300647 **CLC number:** TP39

1 Introduction

As location-based services gain widespread popularity, location-based data have become readily and abundantly available. Such data include various sources, such as user check-in data in location-based social networks (Yang et al., 2015), and geolocation records collected from devices during the use of location-based services (Chandio et al., 2016). These location-based data facilitate the designs of many applications in mobile and ubiquitous computing, such as next-location recommendation (Feng et al., 2018; Liu et al., 2016; Yang et al., 2015), urban function prediction (Noulas et al., 2015; Wang et al., 2020), and urban social diversity assessment (Hristova et al., 2016). However, the granularity of human mobility data from these sources is often not as good as the data from transportation systems, such as the global positioning system (GPS) module in taxis, primarily due to irregular and infrequent user contributions to location-based services.

The sparsity of human trajectory data inevitably hinders the performances of downstream applications. For example, with a limited number of mobility data entries, it is difficult to accurately recommend the next location or point of interest (POI)

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (Nos. 62072115 and 61971145), Shanghai Science and Technology Innovation Action Plan Project (No. 22510713600)
ORCID: Yang CHEN, https://orcid.org/0000-0003-4749-3060

for a user (Xi et al., 2019). Furthermore, for tasks such as human traffic prediction and mobility pattern recognition, the occurrence of missing data also weakens model performance (Li et al., 2013). The evolving landscape of mobile computing across diverse scenarios poses a pressing demand for accurate recovery of human mobility trajectory.

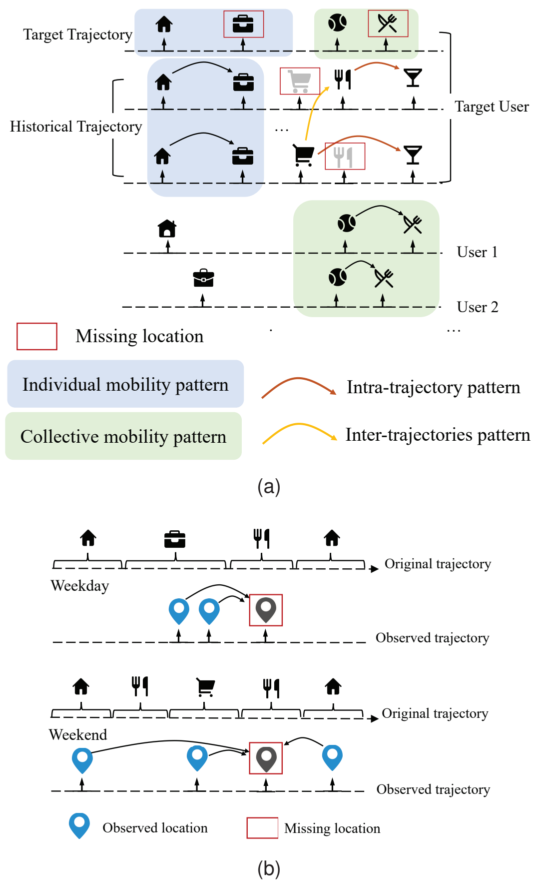

Fig. 1 An illustrative example of mobility patterns and temporal irregularity of observed trajectory points: (a) mobility patterns; (b) temporal irregularity of the observed trajectory points.

Existing recovery methods either extract mobility patterns from collective mobility data encompassing all users (Lin et al., 2021; Salakhutdinov and Mnih, 2007) or rely on user-specific historical data to capture individual mobility patterns (Dhont et al., 2022; Feng et al., 2018; Xia et al., 2021; Sun et al., 2021). The combined effect of both data sources is ignored.

We observe three limitations in existing trajectory recovery models. First, human mobility exhibits both periodicity and a high degree of freedom and variation (Cho et al., 2011; Fang et al., 2021). Fig. 1a presents examples of historical trajectories for both the target user and other users, in addition to showing the trajectory of the target user that requires recovery. As noted, individual mobility patterns can be observed from one's historical trajectories. The blue box highlights the distinct pattern of regular morning commutes to the workplace on weekdays in the target user's trajectories. This pattern is crucial for inferring the missing location of the workplace in the target trajectory. In addition, the target user typically engages in leisure activities, such as shopping, after work. However, relying solely on individual mobility patterns is inadequate for pinpointing locations visited after playing tennis in the target trajectory, especially if the user has no prior history of tennis activity. The green box demonstrates the collective transition patterns observed from the trajectories of other users (such as User 1 and User 2), for instance, moving from the tennis court to the canteen. These collective mobility patterns can facilitate the recovery of the missing canteen location in the target trajectory.

It is important to strike a balance between incorporating collective information and preserving personalized patterns (Fang et al., 2021).

Second, for each individual user, each daily trajectory shows a unique intratrajectory location transition pattern for a day. However, due to the sparsity of the trajectory, the information from a single trajectory alone might be insufficient for modeling individual patterns. Thus, aggregating multiple trajectories becomes necessary to compensate for missing information and to capture inter-trajectory transition patterns. For example, in Fig. 1a, when analyzing the historical trajectories of the target user, we can identify intratrajectory patterns, such as transitions from a restaurant to a bar or from a shopping mart to a bar, within each trajectory. Yet, these historical trajectories are often incomplete and sparse, which limits the utility of such intratrajectory patterns for reconstructing the entire trajectory. To address this challenge, we jointly consider the temporal and spatial information across trajectories and use the locations that appear in multiple trajectories (e.g., the bar in the figure) as a pivot to aggregate them. This enables the identification of intertrajectory patterns, such as transitions from a shopping mall to a restaurant, highlighted by a yellow arrow in the figure. Effective trajectory information aggregation is essential for leveraging other trajectory data without introducing noise.

Finally, the observable trajectory points are irregular and sparse, primarily because the location information is recorded only when users access the services. Fig. 1b presents an example of trajectory irregularity, displaying a user's simplified trajectories on weekdays and weekends. Although a person's actual daily trajectory is temporally coherent, the check-ins are recorded sparsely and at irregular intervals, posing challenges in accurately reconstructing the missing locations with a limited number of observed trajectory points.

To address these three limitations, we proposed our model CLMove, where CL stands for contrastive learning. To fully consider the periodicity and variability of trajectory data, we combine the collective and individual patterns by separating the learning of users' spatial patterns into two stages. (1) We train a location encoder through self-supervised learning on collective trajectories in the city. This pretraining step obtains location embeddings that capture general spatial patterns. (2) We apply a novel graph neural network (GNN)-based location encoder to fine-tune the location embeddings based on user-specific historical data. This fine-tuning process allows us to capture the specific individual patterns.

To tackle the second limitation, we design a GNN-based location encoder that jointly captures the location transition patterns within individual trajectories and across different trajectories in an iterative manner. Finally, to improve the robustness of the model when dealing with different incomplete trajectory contexts, we incorporate a trajectory-level contrastive learning task. This task lets the model learn similar trajectory embeddings for two subtrajectories from the same original trajectory. Based on the location embeddings learned in previous steps, we apply an attention-based mobility trajectory recovery module to reconstruct the trajectory. We conduct extensive experiments on three real-world datasets, namely, Foursquare, Geolife, and Porto Taxi. Our model CLMove consistently outperforms state-of-the-art methods in terms of accuracy of trajectory recovery.

Our contributions are summarized as follows:

- We propose a novel neural network-based model for trajectory recovery. Our model effectively captures the collective and individual spatial patterns using a pre-trained location encoder module and a novel fine-tuned GNN-based location encoder. An attention-based mobility trajectory recovery module is applied to fuse the historical trajectory into the target trajectory.
- We design a trajectory-level contrastive learning task to improve the robustness of the model when encountering unpredictable distribution of missing trajectory points. To the best of our knowledge, this is the first model that utilizes contrastive learning for accurate mobility trajectory recovery.
- We conduct extensive experiments on three representative real-world datasets to evaluate the performance of CLMove. The results demonstrate the superiority of CLMove over state-of-the-art baselines.

2 Related work

We review two important lines of related work: human mobility recovery; and contrastive learning and its applications in mining trajectories.

2.1 Human mobility recovery

There is increasing interest in understanding and recovering human trajectory. Proposed human mobility recovery approaches can be divided into two categories: human mobility recovery with road networks (Park et al., 2022; Fang et al., 2022; Ren et al., 2021; Chondrogiannis et al., 2022; Wu et al., 2016; Li et al., 2020; Zhang et al., 2022); and trajectory recovery without the limitation of road networks (Wei et al., 2012; Luo et al., 2018). The former studies consider the trajectories of objects moving in a road network, which are mostly recorded by the GPS of vehicles and have a relatively finer-grained temporal resolution. Our work falls within the latter category, which does not have road networks as input but pays attention to utilizing spatial-temporal patterns of users' trajectories for recovery.

Rule-based methods (Wei et al., 2012; González et al., 2008) infer and recover the trajectory of humans through construction of the moving relationship graph among locations. Since the transition relationship of locations in trajectories is complicated, deep learning technologies have been successfully applied in the task of human mobility recovery. The recovery of the missing value of general time series has been extensively studied (Luo et al., 2018). However, the general time series recovery model ignores the unique spatialâĂŞtemporal dependence in human mobility data. Apart from this type of models, mobility prediction models (Feng et al., 2018; Lin et al., 2021) for recovery are available for adoption. These models incorporate the human mobility periodicity patterns and consider the location transition relationship with recurrent neural networks (RNNs) and attention mechanisms. However, these models cannot make use of the mobility data that appear after the missing locations, and the performance of these models is acceptable only when the data are dense enough. Aiming at recovery, the context-enhanced trajectory reconstruction (CTR) (Chen et al., 2019) model extracts the features of users' mobility data and utilized a tensor factorization-based method, but the tensors are limited to be low-ranked ones. The bi-directional spatial and temporal dependence and usersâĂŹ dynamic preferences (Bi-STDDP) (Xi et al., 2019) model combines bidirectional temporal information to learn users' historical representation but fails to consider the different impacts of different historical trajectory points on recovery. Further considering the spatialâĂŞtemporal dependence of historical data of users, some recent studies (Xia et al., 2021; Sun et al., 2021) have designed attention-based methods to capture the multilevel periodicity of human mobility and fuse it into recovery. Considering users' collective mobility patterns, GETNext (Yang et al., 2022) proposes a user-agnostic global trajectory flow map and a novel graph-enhanced transformer (GET) model to predict the missing location in a trajectory. To alleviate the cold-start problem, Si et al. (2024) apply a transformer encoder in the TrajBERT model to capture bidirectional mobility patterns and have designed a novel spatialâĂŞtemporal-aware loss function to refine the prediction.

2.2 Contrastive learning

Contrastive learning is a useful representative self-supervised learning approach that utilizes large-scale datasets while avoiding the cost of manual annotation. Contrastive learning has been widely used in areas such as computer vision, natural language processing (NLP), and graph representation learning. The core idea of this approach is to train the model to maximize the similarity between positive sample pairs while retaining the dissimilarity of negative samples. To construct positive samples, when one sample from the training dataset is taken, data augmentation techniques generate a transformed version of the sample. The original sample and the transformed sample are considered a positive sample pair. Different data augmentation methods are applied in different areas. For trajectory data, point downsampling and point distorting are widely used (Deng et al., 2022; Zhou et al., 2022). Point downsampling (Li et al., 2018) models the nonuniform sampling rate of trajectories by randomly masking some trajectory points in a trajectory. Point distorting fits the characteristics of noisy points in trajectories by randomly distorting the coordinates of trajectory points.

Only considering maximization of the similarity between positive samples will cause a collapsing solution, i.e., all the input samples have the same output representations. This renders the model unable to learn any knowledge. To avoid a collapsing solution, many methods such as SimCLR (Chen et al., 2020), Moco (He et al., 2020), and Simsiam (Chen and He, 2021) have been proposed. SimCLR constructed negative sample pairs by combining one sample with the remaining samples in the batch or training dataset; Moco turned one branch of the Siamese network into a momentum encoder; and Simsiam (Chen and He, 2021) adopted the stop-gradient operation without negative samples.

3 Methodology

This section presents the design of our approach, as shown in Fig. 2. We first formally define the mobility trajectory recovery problem in Section 3.1.

For the key components of our approach, we first design an unsupervised pretrained location encoder that learns the pretrained location embeddings through the data from the whole user group in the same city (Section 3.2). Then, to capture user-specific location transition patterns through fine-tuning the location embeddings, we propose a GNN-based location encoder that can simultaneously consider the intra- and intertrajectory mobility patterns (Section 3.3). After obtaining the location embeddings, we improve the robustness of the model when encountering unpredictable incomplete trajectory contexts through a proposed trajectory-level contrastive learning task (Section 3.4). Finally, we apply an attention-based trajectory mobility recovery module to fuse the historical spatialâĂŞtemporal patterns of users into the target recovery trajectory and recover missing locations (Section 3.5).

3.1 Problem definition

In this subsection, we introduce the definition and notations we use in this paper and then define the problem to be addressed.

As in previous works (Feng et al., 2018; Xia et al., 2021; Sun et al., 2021), we define a trajectory as a time-ordered sequence of discrete trajectory points. We split 1 day into discrete time intervals (e.g., every 30 min) and form T time slots of one user's trajectory. Each trajectory point represents the location of the user in a certain time slot. To represent location, we partition the whole geographical space of the trajectories, denoted by L, into equally sized grid cells, and each grid cell is a location $l \in L$.

Instead of tracking the trajectory of users continuously, we can only observe the discrete location record when users access the location-based services and contribute their data. In one time slot t of day d, if the user has multiple records of $>$1 location, the location l with the most number of records is considered as the location of the user in that time slot, and the trajectory point is represented as $l_u^{d,t}$. If a user u does not have an observable record in the time slot t in day d, then $l_u^{d,t}$ is denoted by a predefined null, named âĂŸmissing locationâĂŹ.

Following the previous definition of trajectory points, each trajectory has T trajectory points, which represent a user's mobility trajectory in 1 day. Let $\mathcal{T}_u^d : l_u^{d,1} \to l_u^{d,2} \to \cdots \to l_u^{d,T}$ denote user u's d-th day's trajectory.

For a target day C, we define $\mathcal{T}_u^C$ as the target trajectory of the user and the trajectories in the past days $\{\mathcal{T}_u^1, \mathcal{T}_u^2, \cdots \mathcal{T}_u^{C-1}\}$ as the historical trajectories. Given a user's historical trajectories with the target trajectory $\mathcal{T}_u^C$, the goal of the model is to recover the target trajectory, by predicting the users' real locations for the missing null ones in $\mathcal{T}_u^C$.

3.2 Pretrained location encoder

To represent the trajectory points in a trajectory, we need to learn the time embedding and location embedding of each trajectory point. From the trajectories of the whole user group, we can learn the transition relationship of locations, which can be used to learn the pretrained location embeddings. Although different users have different user-specific mobility patterns, this module can incorporate the collective mobility patterns shared by all users into location embeddings.

Inspired by the impressive results attained through language modelings in NLP, we apply the mask location prediction task and utilize the transformer encoder module (Vaswani et al., 2017) with the time encoder (TE), which shares the same architecture of the time encoder in Section 3.5, to pretrain the location embeddings.

For each historical trajectory $\mathcal{T}^d = \{l^{d,1} \to l^{d,2} \cdots \to l^{d,T}\} \in \mathcal{T}_{his}$ from all users' historical trajectories, we randomly mask one of the not-missing trajectory points $l^{d,n}$ to be the null location l_{null}. Considering the temporal influence, we represent the trajectory point embedding as the sum of the time embedding from a time encoder, which is detailed in Section 3.5, and the pretrained location embedding. We utilize a transformer encoder that aims at reconstructing the masked location embedding with the input of all the trajectory point embeddings of the masked trajectory:

$$\begin{aligned}\{\hat{l}^{d,1},\hat{l}^{d,2},\cdots,\hat{l}^{d,T}\} = \\ \mathrm{TransEnc}(\{l^{d,1}+\mathrm{TE}(1), l^{d,2}+\mathrm{TE}(2),\cdots, \\ l_{null}+\mathrm{TE}(d,n),\cdots,l^{d,T}+\mathrm{TE}(T)\}),\end{aligned} \tag{1}$$

where the n-th trajectory point in the trajectory is masked, $\hat{l}^{d,n}$ is the reconstructed location embedding of the masked trajectory point, and location embeddings are trainable parameters. With the reconstructed location embedding $\hat{l}_{d,n}$, we calculate the probability of the masked location to be m as follows:

$$P^{d,n}(m) = \frac{\langle \hat{l}^{d,n}, l^m \rangle}{\sum_{k\in L}\langle \hat{l}^{d,n}, l^k \rangle} \tag{2}$$

where $\langle\cdot\rangle$ represents the inner product, l^m and l^k represent the location embeddings, and L represents all the locations. To train the encoder, we use cross-entropy as the loss function: $Loss_{pretrain} =$

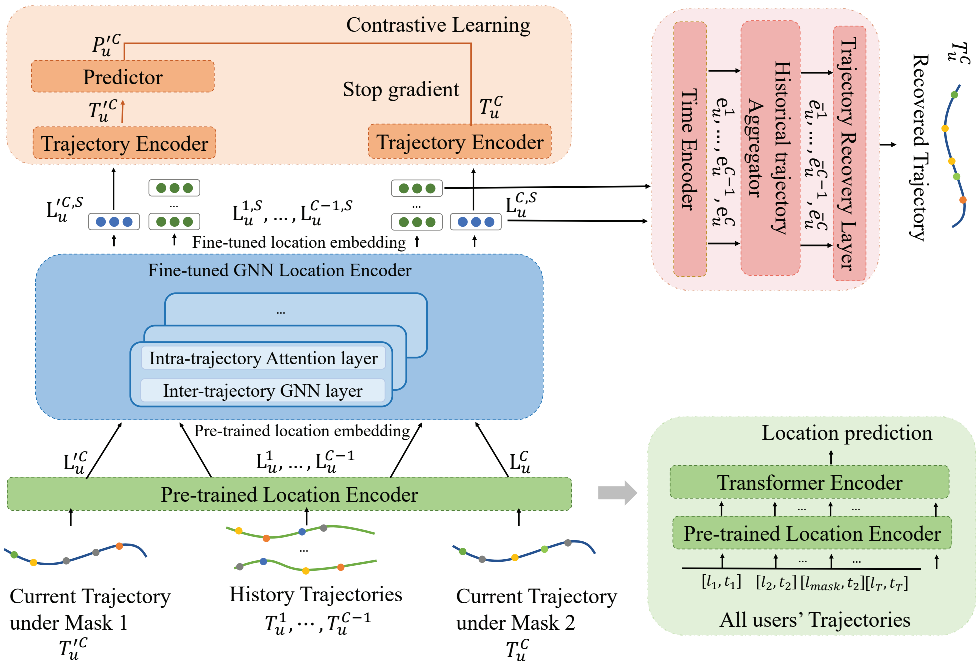

Fig. 2 Main architecture of CLMove.

$-\sum_{\mathcal{T}^d \in \mathcal{T}_{his}} y^{d,n} \log(P^{d,n}(l^{d,n}))$, where $\mathcal{T}_{his}$ represents all usersâĂŹ historical trajectories, $y^{d,n}$ is the one-hot representation of the masked trajectory point $l^{d,n}$. This module captures the collective temporalâĂŞspatial patterns by learning the pre-trained location embeddings.

3.3 Fine-tuned GNN location encoder

With the pretrained location encoder, we can train the location embedding with collective semantic information. However, the location embedding is not specific enough for the mobility trajectory recovery task because each user has his or her own temporalâĂŞspatial pattern and the pretrained location embeddings cannot capture it. To fully utilize the historical trajectories of users and capture the user-specific periodicity patterns, we design a GNN location encoder to fine-tune the location embeddings in the historical and target trajectories for the ensuing trajectory recovery task. It is worth noting that when training the Fine-tuned GNN location encoder, the input pretrained location embedding is frozen and is not updated during training. The model design for this part is discussed in Section 4.4.4.

Considering every single trajectory separately to capture the location transition patterns is insufficient due to the sparsity of trajectories. Some works (Feng et al., 2018; Xia et al., 2021) aggregate a user's historical trajectories into one dense trajectory by extracting the location with the highest visiting frequency in the corresponding time slot. However, this method cannot utilize all the historical data of users and may bring noise since the continuous trajectory points in the aggregated dense trajectory may not happen on the same day and cannot reflect the mobility patterns of the user.

To solve the above challenge, we design a fine-tuned GNN location encoder for simultaneously capturing the intra- and inter-trajectory location transition patterns in this module. The intratrajectory patterns consider the location transition patterns separately, which can totally utilize the historical trajectories of the user, while the intertrajectory location transition patterns can consider multiple trajectories together, which can address the sparsity of the data and learn higher-level location transition patterns.

First, we construct a weighted directed graph for each trajectory, where the nodes in the graph represent locations. Then, we capture the intra-trajectory location transition patterns by applying intragraph information propagation in each graph through a GNN model. At the same time, we consider the relationship among the nodes representing the same location in different trajectory graphs. We apply intergraph information propagation in multiple graphs through an attention-based method. The intragraph information propagation and intergraph information propagation steps are combined in an iterative manner to obtain the fine-tuned location embeddings of trajectories in this module.

3.3.1 Graph construction

First, for capturing the complex transition patterns hidden in each trajectory, we construct a graph for each trajectory. Given a trajectory $\mathcal{T}^d = \{l^{d,1} \rightarrow l^{d,2}, \cdots \rightarrow l^{d,T}\}$, we treat the nonrepeating locations on the trajectory as the nodes $\{l_1, \cdots, l_N\}$ and treat each $l^{d,i} \rightarrow l^{d,i+1}$ relationship as a direct edge. Considering the direction of the edge, we construct two directed graphs G_I^t, G_O^t for a trajectory. Since the location transition relationship can appear repeatedly, we calculate the weight of the corresponding edge as the frequency of the relationship. Then, we normalize the weights of edges with the same source node. The weighted adjacency matrix of the two graphs is A_I^t, A_O^t.

For example, for trajectory $T_3 = \{l_1 \rightarrow l_4 \rightarrow l_2 \rightarrow l_1 \rightarrow l_2\}$ in Fig. 3, the corresponding graph has three nodes, which represent $\{l_1, l_2, l_4\}$ respectively. For G_I, there will be four directed edges $\{l_1 \rightarrow l_4, l_4 \rightarrow l_2, l_2 \rightarrow l_1, l_1 \rightarrow l_2\}$, which consider the transition relationship between $l^{d,i} \rightarrow l^{d,i+1}$. Since l_1, as the source node, has two transition patterns with the same frequency, the weights of these two patterns are 0.5 after normalization. The corresponding weights of the edges are $\{0.5, 1, 1, 0.5\}$. Using the same graph construction method, G_O, which considers the relationship between $l^{d,i+1} \rightarrow l^{d,i}$, has four directed edges $\{l_4 \rightarrow l_1, l_2 \rightarrow l_4, l_1 \rightarrow l_2, l_2 \rightarrow l_1\}$ with weights $\{1, 0.5, 0.5, 1\}$.

For the historical trajectories $\mathcal{T}_u^1, \cdots, \mathcal{T}_u^{C-1}$ and target trajectory $\mathcal{T}_u^C$ of the user u, we can construct the corresponding graphs G_I^d, G_O^d and obtain the adjacency matrices A_I^d, A_O^d for each trajectory $\mathcal{T}_u^d$.

3.3.2 Node embedding update

To simultaneously capture the intra- and intertrajectory location transition patterns, we design an iterative node embedding update method to update the node embeddings.

First, for modeling the complex transition patterns among the graph's nodes, which correspond to locations in trajectories, we applied a GNN-based method. GNN has achieved impressive performance in many areas by aggregating adjacent nodes' embeddings to update node embeddings. We adopted gated GNN (Li et al., 2016), which has been proven to be able to capture complex transitions among nodes (Sun et al., 2021; Wu et al., 2019). In gated GNN, the propagation information from adjacent nodes is first calculated through a linear transformation in two directed graphs respectively. Then, the propagation information of the nodes in the two directed graphs A_I and A_O is concatenated in each step s as follows:

$$\begin{aligned} M^{c,s} = \mathrm{Concat}(&A_I^c \cdot (L^{c,s-1} W_I + b_I), \\ &A_O^c \cdot (L^{c,s-1} W_O + b_O)), \end{aligned} \tag{3}$$

where $c \in [1, C]$ represents each trajectory of the user, $L^{c,s} = [l_1^{c,s}, \cdots, l_N^{c,s}]$ is the calculated location embedding of the nodes in step s, and N is the number of nodes in the graph A_I. The location embedding in Step 0 is obtained from the pretrained location encoder which considers the collective mobility patterns. To update the information passed from the adjacent nodes from the previous step of node embedding, gated GNN uses the gated recurrent unit (GRU) (Chung et al., 2014; Seng et al., 2021). We calculate the intratrajectory updated node embedding in step s as follows:

$$\tilde{L}^{n,s} = \mathrm{GRU}(L^{n,s-1}, M^{n,s}). \tag{4}$$

The preceding part deals with intragraph information propagation, which can successfully model the location transition patterns in each trajectory. However, it ignores the relationship between multiple historical trajectories and learns the location embeddings in each trajectory independently. For example, a user tends to go to different restaurants for dinner after work. If we only consider the transition pattern âĂŸworkplace $\rightarrow$ restaurantâĂŹ, the relationship between different restaurants will be ignored.

At the same time, the sparsity of the trajectory makes independent consideration of each trajectory insufficient for modeling individual location transition patterns. We can consider multiple trajectories together to complement the missing information and capture the intertrajectory transition patterns. For example, in Fig. 1, the user has the mobility pattern âĂŸshopping $\rightarrow$ having dinner $\rightarrow$ drinkingâĂŹ. Due to the lack of records, both the trajectories cannot capture this pattern.

To solve these disadvantages, we need to enable different trajectories to exchange information. We let the nodes representing the same location in different trajectories act as anchors and let the nodes propagate information among themselves. We implement intertrajectory information propagation after intragraph information propagation in each step.

In step s, the location l may acts as a node with node embedding $\{\tilde{l}^{1,s}, \cdots, \tilde{l}^{M,s}\}$ in historical trajectories $\{\mathcal{T}'^{1}, \cdots, \mathcal{T}'^{M}\}$ and the node embedding $\tilde{l}^{C,s}$ in the target trajectory $\mathcal{T}^{C}$ after intragraph information propagation. We update these node embeddings for the same location in different trajectories through an attention-based method. For each location embedding $\tilde{l}^{d,s}$ in the historical trajectories, we define the similarity between it and the corresponding location embedding in another historical trajectory $\tilde{l}^{i,s}$ as α_i^d and update the node embedding in historical trajectories as follows:

$$l^{d,s} = \sum_{i=1}^{M} \alpha_i^d (W_{l_3} \cdot \tilde{l}^{i,s})$$
$$\text{where } \alpha_i^d = \frac{\exp\left(\langle W_{l_1} \cdot \tilde{l}^{d,s}, W_{l_2} \cdot \tilde{l}^{i,s} \rangle\right)}{\sum_{j=1}^{M} \exp\left(\langle W_{l_1} \cdot \tilde{l}^{d,s}, W_{l_2} \cdot \tilde{l}^{j,s} \rangle\right)}. \tag{5}$$

When we update the node embeddings in the graph of historical trajectories, we use Eq.(5). For the node embeddings in the graph of the target trajectory, we consider both the historical trajectories and the target trajectory when calculating α_i^d.

For node embedding update, we propagate the adjacent nodes' information from the intratrajectory graph and the same location information from the intertrajectory graphs iteratively in each step. The iterative update of node embeddings enables the location embeddings to model the mobility patterns of both intra- and intertrajectory graphs.

A proof-of-concept example is illustrated in Fig. 3, which has five time slots $\{t_1, \cdots, t_5\}$, four locations $\{l_1, \cdots, l_4\}$, and three trajectories $\{T_1, T_2, T_3\}$ of a user. The three graphs on the right are constructed from these three trajectories. Due to space limitation, we show one of the two directed graphs of a trajectory in Fig. 3, with the same principle being applied to both. The update process for location embeddings involves both intra- and intergraph information propagation steps. Taking location l_1 as an example, each graph first updates its own embedding using adjacent node information within the graph, leading to trajectory-specific l_1 embeddings. In this step, GNN enables intratrajectory information propagation and captures the intratrajectory mobility patterns. Then, these trajectory-specific l_1 embeddings are aggregated across different graphs, enriching the l_1 embedding in one graph with information from all trajectories. The attention-based aggregation among multiple trajectories helps to capture intergraph information propagation. This update process, applied similarly to other locations, iteratively refines the embeddings to encapsulate both intra- and intertrajectory mobility patterns. After several steps, we can get the fine-tuned location embedding for each trajectory point.

This module considers the location transition patterns of historical trajectories to fine-tune the location embedding in the historical and target trajectories for the subsequent trajectory recovery task. The output node embedding $l^{d,S}$ of the last step S is considered as the final trajectory point embedding of the trajectory $\mathcal{T}^{d}$.

3.4 Contrastive learning task

With the pretrained location encoder and fine-tuned location encoder, we can obtain the trajectory point embeddings of historical and target trajectories, which are then used by the trajectory recovery module (Section 3.5) for recovering the missed trajectory point in the target trajectory. However, due to the requirement of users' contribution in location-based services, the distribution of the observable trajectory points in a trajectory is usually sparse, irregular, and unpredictable. Since the trajectory point embedding is closely related to the other observable trajectory points, the robustness of the model encountering different distributions of the observable trajectory points is essential for improving the quality of trajectory point embeddings.

To improve the robustness of the model when encountering different incomplete trajectory contexts, we design a trajectory-level embedding contrastive learning task. For one targeted recovery route of a user, due to the randomness of the time at which the user visits the location-based services and contributes the relevant information, the dis-

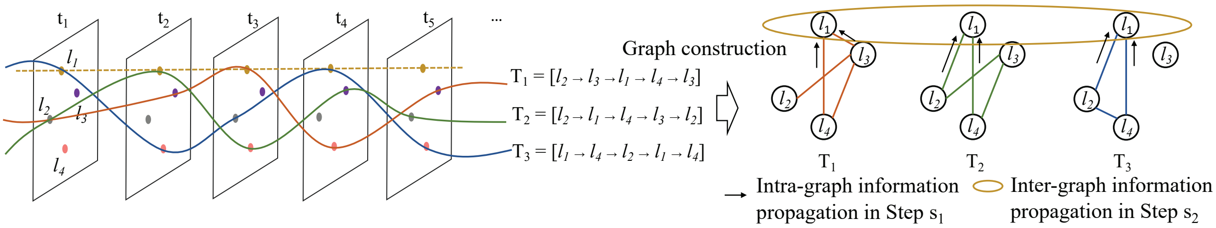

Fig. 3 A proof-of-concept example of node embedding update.

tribution of the observable trajectory points can be different. For one missing trajectory point, different observable trajectory point distributions have different influences on its recovery. The distribution shift reduces the robustness of the model. Unfortunately, directly minimizing the difference caused by the distribution shift is hard. Therefore, we replace it by indirectly maximizing the similarity of the trajectory representations modeled by the distributions of different observable trajectory points.

For one targeted recovery trajectory, we randomly mask the same number of trajectory points of the original trajectory to construct two subtrajectories T^C and T'^C. These two trajectories correspond to the same route while having different observable trajectory point distributions. With the pretrained and fine-tuned location encoders, we can obtain the location embeddings $\{l^1, \cdots, l^M\}$ and $\{l'^1, \cdots, l'^{M'}\}$ of the two different randomly masked target trajectories. Then, we want to learn the trajectory embeddings of T^C and T'^C and maximize the similarity between them. It is noteworthy that the location embeddings are the output of the fine-tuned GNN location encoder, which means that the location embeddings have integrated some historical information of the users. It makes constructing similar trajectory embeddings under different masking situations more feasible.

We aggregate the fine-tuned location embeddings of the target trajectory with an attention-based trajectory encoder to obtain the trajectory embeddings. The trajectory embedding t^C of the trajectory T^C can be calculated as follows:

$$t^C = \sum_{i=1}^{M} \alpha_i l^i \quad \text{where } \alpha_i = \frac{\exp(q^\top \sigma(W_c \cdot l^i))}{\sum_{j=1}^{M} \exp(q^\top \sigma(W_c \cdot l^j))}, \tag{6}$$

where q and W_c are learnable parameters and $\sigma(\cdot)$ represents the rectified linear unit (ReLU) function. The trajectory embedding t'^C of the trajectory T'^C can be calculated using Eq.(6) in the same way.

Inspired by Chen and He (2021), we maximize the similarity between the two trajectory embeddings while avoiding collapsing solutions by using the stop-gradient operations. With the two trajectory embeddings t^C and t'^C, we apply a multilayer perceptron (MLP) layer as the predictor, denoted as h, on one of the trajectory embeddings to predict the other one, while applying the stop-gradient operation on the other embedding to avoid the collapsing solutions:

$$p^C = h(t^C), p'^C = h(t'^C). \tag{7}$$

We minimize the negative cosine similarity between the predictor output embedding P^C and the trajectory encoder output embedding t'^C, as follows:

$$D(p^C, t'^C) = -\frac{p^C}{\|p^C\|_2} \cdot \frac{t'^C}{\|t'^C\|_2}, \tag{8}$$

where $\|\cdot\|_2$ is the $l2$-norm. To calculate a symmetrized loss with the stop-gradient operation, the loss of the trajectory embedding contrastive learning task is

$$\begin{aligned} Loss_c = \frac{1}{2}(D(p^C, \text{stopgrad}(t'^C)) + \\ D(p'^C, \text{stopgrad}(t^C))). \end{aligned} \tag{9}$$

This contrastive learning task then further trains the location embeddings to adapt different distribution scenarios of missing points.

3.5 Trajectory recovery module

In this module, we first use a time encoder to integrate spatialâĂŞtemporal patterns. Then, we fuse the historical spatialâĂŞtemporal patterns into the target trajectory recovery. Finally, a trajectory recovery layer generates locations as recovery.

3.5.1 Time encoder

To integrate the spatialâĂŞtemporal patterns, we add the time embedding to the corresponding location embedding as the trajectory point embedding.

The time embeddings should have the same dimension as the location embeddings. Following Vaswani et al. (2017), we calculate the time embedding for each time slot t as follows:

$$\begin{cases} \mathrm{TE}(t,2i) = sin(t/100000^{2i/d}), \\ \mathrm{TE}(t,2i+1) = cos(t/100000^{2i/d}), \end{cases} \tag{10}$$

where TE is the time encoder, and i is the dimension of the embedding. For a trajectory point with location l and time slot t in the trajectory d, the trajectory point embedding is $e^{d,t} = \mathrm{TE}(t) + l^{d,S}$, where $l^{d,S}$ is the output at the last step of the fine-tuned GNN location encoder.

3.5.2 Historical trajectory aggregator

We next proceed to combine the information from the properly processed historical trajectories and target trajectory to recover the missing locations. A direct solution to recover a trajectory point in the target trajectory is to aggregate the related representation of the location of the trajectory point and obtain the historical-related trajectory point embedding, which can provide additional information for recovery. To implement the solution, we calculate the similarity of the recovered trajectory point and trajectory points embeddings in all the historical trajectories and aggregate all trajectory point embeddings according to their similarity levels. For clarity, we separate the aggregation into two steps. First, we aggregate information at the trajectory point level. For each trajectory, we apply the cross-attention layer to compare the representation of the recovered trajectory point with all the points in the trajectory and gain the trajectory-related embeddings. Then, we apply the soft attention layer to aggregate all the trajectory-related embeddings for recovery.

In the cross-attention layer, we calculate the similarity between the trajectory point of the t-th time slot in the target trajectory $\mathbf{T}^C$ and the k-th time slot of the historical trajectory $\mathbf{T}^d$ as $\alpha_{t,k,d}$ and the trajectory-related embedding as follows:

$$\hat{e}_{t,d} = \sum_{i=1}^{M} \alpha_i l^i$$
$$\text{where } \alpha_{t,k,d} = \frac{\exp\left(\langle W_{c_1} \cdot e^{C,t}, W_{c_2} \cdot e^{d,k} \rangle\right)}{\sum_{i=0}^{T} exp\left(\langle W_{c_1} \cdot e^{C,t}, W_{c_2} \cdot e^{d,i} \rangle\right)}, \tag{11}$$

where $\hat{e}_{t,d}$ represents the additional information that the trajectory $\mathbf{T}^d$ can provide for the recovery of the t-th trajectory point in the target trajectory; $e^{C,t}$ is the representation of the t-th time slot in the target trajectory $\mathbf{T}^C$; $e^{d,k}$ is the representation of the k-th time slot in the historical trajectory $\mathbf{T}^d$; T is the number of time slots in the trajectory; and $< \cdot >$ represents the inner product.

To stabilize the learning process and enable the model to jointly attend to information at different positions, we apply the multihead attention mechanism. We concatenate all the output embeddings of H heads and apply a transformation matrix for the final embedding with historical trajectory information. We also add a residual connection to preserve the current information of the target trajectory point:

$$\tilde{e}^{t,d} = ReLU(W_{c_4} \cdot \mathrm{Concat}(\hat{e}^1_{t,d}, \ldots, \hat{e}^H_{t,d}) + e^{C,t}). \tag{12}$$

To further combine all the historical information, the soft attention layer leverages the soft attention mechanism to calculate the similarity between the representation $e^{C,t}$ of the t-th trajectory point in the target trajectory $\mathbf{T}^C$ and the corresponding historical trajectory embedding $\hat{e}^{t,d}$ in $\mathbf{T}^d$. With the similarity $\alpha_{t,d}$, we aggregate the historical-related embedding as follows:

$$\bar{e}'_t = \sum_{i=1}^{C-1} \alpha_{t,i} \hat{e}^{t,i},$$
$$\text{where } \alpha_{t,d} = q^\top \sigma(W_{s_1} e^{C,t} + W_{s_2} \hat{e}^{t,d} + b), \tag{13}$$

where $\bar{e}'_t$ is the aggregated embedding with all the historical information, q is a learnable parameter, and $\sigma(\cdot)$ represents the sigmoid function; $C-1$ is defined as the number of historical trajectories. With a residual connection for preserving the current information, the final trajectory point embedding of the target trajectory is

$$\bar{e}_t = W_{s_3} \cdot \mathrm{Concat}(\bar{e}'_t, e^{C,t}). \tag{14}$$

3.5.3 Trajectory recovery layer

With the predicted trajectory point embedding $\bar{e}_t$, we apply a trajectory recovery layer for calculating the probability that the missing location is l at time slot t in the target trajectory:

$$P_t(s) = \frac{\langle \bar{e}_t, l^s \rangle}{\sum_{k \in L} \langle \bar{e}_t, l^k \rangle}, \tag{15}$$

where $P_t(s)$ is the probability of the location of the t-th trajectory point to be location s, $k \in L$ denotes all the locations, and l^k is the corresponding pre-trained location embedding. The location with the maximum probability is identified as the recovery result.

3.6 Model training

To train the model, we use cross-entropy as the loss function:

$$Loss_{rec} = -\sum_{\mathcal{T} \in D} \sum_{t \in T^M} y_n^t log\left(p_t\right), \tag{16}$$

where y_n^t is the one-hot representation of the ground-truth location of the t-th trajectory point in the trajectory $\mathcal{T}$, T^M represents all the missing trajectory points of trajectory $\mathcal{T}$, and D represents all the trajectories to be recovered.

To better capture the spatial proximity, as in the work of Sun et al. (2021), we add a noise contrastive estimation (NCE)-based distance loss, as follows:

$$L_{dis} = \sum_{\mathcal{T} \in D} \sum_{t \in \mathbf{T^M}} \sum_{\substack{p \in N(l_n^t),\\ q \notin N(l_n^t)}} w_p^{\mathcal{T},t} \cdot max(\|\bar{e}^{\mathcal{T},t} - e^{l^p}\|_2 - \|\bar{e}^{\mathcal{T},t} - e^{l^q}\|_2, 0), \tag{17}$$

$$w_p^{\mathcal{T},t} = \frac{\exp(Dist(l_p, l_{\mathcal{T}}^t))}{\sum_{i \in N(l_{\mathcal{T}}^t)} \exp(Dist(l_i, l_{\mathcal{T}}^t))}, \tag{18}$$

where $l_{\mathcal{T}}^t$ represents the ground-truth location of the t-th trajectory point in trajectory $\mathcal{T}$, $N(l_{\mathcal{T}}^t)$ represents the top K locations nearest to the target location $l_{\mathcal{T}}^t$, and $Dist(\cdot)$ represents the distance between two locations. For each missing trajectory point, we treat $p \in N(l_{\mathcal{T}}^t)$ as positive neighbors and randomly sample K locations $q \notin N(l_n^t)$ as negative samples. Then, we randomly choose K pairs of positive and negative samples; further, we calculate the NCE-based distance loss as in Eq.(18). With this loss, we aim to increase the embedding similarity between positive pairs and decrease the embedding similarity between negative pairs.

Finally, we have the total loss as $Loss = Loss_{rec} + \lambda_1 Loss_c + \lambda_2 Loss_{dis}$, where λ_1 and λ_2 are hyperparameters to balance the three losses.

4 Experiments

4.1 Datasets

We use three representative datasets for our study. (1) The Geolife dataset (Zheng et al., 2008, 2010, 2009) was collected from Microsoft Research AsiaâĂŹs Geolife project, which had data from 182 users over a period of > 5 years (from April 2007 to August 2012), which is commonly used in the learning of trajectories (Sun et al., 2021; Xia et al., 2021). Each trajectory can be represented by a sequence of time-stamped points, each of which contains information on latitude and longitude. (2) The Foursquare dataset (Yang et al., 2015) was collected from the Foursquare social network from April 12, 2012, to February 16, 2013, which is also widely used in the learning of trajectories (Feng et al., 2018; Sun et al., 2021). Each record in the dataset represents a check-in event, and it contains the anonymized user ID, the latitude and longitude of the POI, and the corresponding time stamp. We normalize the collection period into 14 days while keeping the original order of the trajectory. (3) The Porto Taxi dataset[1] provides the accurate trajectories of 442 taxis running in the city of Porto, in Portugal, from July 1, 2013 to June 30, 2014. It is widely used in the learning of trajectory prediction (Gao et al., 2023; Xu et al., 2022; Deng et al., 2022). It provides the anonymous identifiers of taxi drivers and the latitudes and longitudes of the taxis. Although the trajectory of the taxi is partly determined by the passenger, we assume that the taxi's trajectory is also related to the driver's personal driving habits, and we hope that our model can capture this pattern.

For location representation, we partition the city into equally sized grids. As in the work of Sun et al. (2021), we set the side length as 500 m for each grid in both datasets. Following previous works (Xia et al., 2021; Sun et al., 2021), we set the time interval as 30 min, which means that each trajectory has 48 trajectory points. For model training and testing, we filter out the trajectories with < 12 observed trajectory points and filter out the users with < 5 trajectories for both datasets. The final detailed statistics of the preprocessed datasets are summarized in Table 1.

[1] https://www.kaggle.com/competitions/pkdd-15-predict-taxi-service-trajectory-i/overview

Table 1 Basic statistics of mobility datasets

Dataset	City	Duration	No. of users	No. of locations	No. of trajectories
Geolife	Beijing	>5 years	61	2280	1653
Foursquare	New York	11 months	315	2435	2146
Porto Taxi	Porto	1 year	433	1088	90,253

4.2 Baselines

To demonstrate the effectiveness of our proposed method, we compare it with some rule-based models, classic sequential data processing models, and state-of-the-art deep learning models.

- Top is an intuitive counting-based model. The top K popular locations in the training set are used as the recovery result for each missing location.

- Time Top recovers the missing location as the top K popular locations in the corresponding time slot in the training set.

- LSTM (Liu et al., 2016), long short-term memory network, considers the trajectory points earlier than the predicted trajectory point and uses a forward RNN for recovery.

- BiLSTM (Zhao et al., 2018), bidirectional long short-term memory network, extends LSTM by using a bidirectional LSTM network to consider the bidirectional trajectory points for prediction.

- DeepMove (Feng et al., 2018) utilizes a historical attention module to aggregate the historical trajectories information into the target trajectory and considers the user preferences. We use the prediction of the next location as the recovery result.

- AttnMove (Xia et al., 2021) designs various attention mechanisms to model the mobility regularity and the historical pattern of users for trajectory recovery.

- PeriodicMove (Sun et al., 2021) is a state-of-the-art mobility trajectory recovery model that leverages a GNN-based attention mechanism for trajectory recovery.

- GETNext (Yang et al., 2022) is a state-of-the-art POI recommendation model. It proposes a user-agnostic global trajectory flow map and a novel graph enhanced transformer model to capture the collaborative mobility patterns. Similar to DeepMove, we use the prediction of the next location as the recovery result.

- TrajBERT (Si et al., 2024) applies the transformer encoder to capture the bidirectional mobility patterns and to design a novel spatial-temporalâĂŞaware-loss function to refine the prediction.

4.3 Experimental settings

For both datasets, we sort each user's trajectories by time and take the first 70% trajectories excluding the first three trajectories as the training dataset. The first three trajectories (Xia et al., 2021) of a user can guarantee that each recovered trajectory has at least three historical trajectories. The following 10% and the remaining 20% trajectories make up the validation dataset and test dataset. To evaluate the performance, we randomly mask 10 observed trajectory points in the trajectory and use the observed location as the ground truth. For the contrastive learning task, we construct a positive sample for each target trajectory by applying a different random mask policy on the ground truth.

To implement our model, we use PyTorch, a widely used open source machine learning framework. For our model, we set the hidden size to be 128. The Adam optimizer is used to optimize our model, with a default learning rate of 0.001 and minibatch size of 50. We set the dropout rate to be 0.3 in the trajectory recovery layer to prevent overfitting. The contrastive learning loss weight λ_1 and distance loss weight λ_2 loss weight are set to be 0.1.

To evaluate the performance of the model, we apply three key metrics, namely, Recall@K (Liu et al., 2016; Sun et al., 2021; Xi et al., 2019; Xia et al., 2021), Distance@K (Sun et al., 2021; Xia et al., 2021), and mean average precision (MAP) (Liu et al., 2016; Sun et al., 2021; Xi et al., 2019; Xia et al., 2021). For one trajectory point to be recovered, if the ground-truth location is in the predicted top-K probability ranked list, the Recall@K is âĂŸ1âĂŹ; otherwise, the value is âĂŸ0âĂŹ. The average value of Recall@K for all the trajectory points to be recovered is Recall@K. A Larger Recall@K value implies better performance. The Distance@K of a trajectory point is the smallest geographical distance between the center of the ground-truth location and the center of the locations in the predicted top-K probability-ranked list. A smaller Distance@K implies better performance. MAP is a widely used metric for ranking tasks that can consider the global ranked list. Therefore, we use it to evaluate the whole ranked list of all the locations. A larger MAP implies better performance.

4.4 Experimental result

4.4.1 Overall performance

We compare our model with baselines, and the experimental results are shown in Table 2. We have the following observations.

Firstly, the performance of rule-based models has the worst performance for almost all evaluation metrics on all datasets. Although the most frequently visited location could reflect the mobility patterns, the performance of rule-based models is still not acceptable for recovery. The RNN-based models achieve better performance than rule-based models because they can characterize location transition patterns. The bidirectional model performs better than the single-direction one because the former can consider more information in recovery.

Secondly, the models that consider historical trajectories have smaller distances than the ones that just consider the target trajectory. A possible reason is that historical data can basically reflect the geographical mobility patterns of users, which makes the model more user-specific.

Finally, our model outperforms all the baselines for all the chosen evaluation metrics on all datasets (except for the distance metrics in the Porto Taxi dataset and the Recall@5 metric in the Geolife dataset, where our model slightly underperformed by 4.8%, 0.3%, 0.2%, and 4% compared to the best results). Specifically, recall of our model outperforms the best baseline by 3%âĂŞ29.5% on the Geolife dataset and 1.8%âĂŞ2.9% on the Porto Taxi dataset. Distance of our model outperforms the best baseline by 17.2%âĂŞ19.4% on the Geolife dataset and 7.5%âĂŞ17.6% on the Foursquare dataset. The MAP of our model outperforms the best baseline by 22.3% on the Geolife dataset, 4.3% on the Foursquare dataset, and 2.3% on the Porto Taxi dataset. These great improvements indicate that our approach can well model human mobility patterns.

4.4.2 Ablation study

To evaluate the effectiveness of each component of the model, we perform a series of ablation studies by removing them one by one (Sun et al., 2021; Xi et al., 2019; Xia et al., 2021). The results are shown in Table 3. We gradually remove the contrastive learning loss in training, use randomly initialized location embeddings instead of the pretrained location embeddings, and remove the intergraph location embeddings update. The result shows that our model outperforms all the ablations and proves the effectiveness of our model designs.

4.4.3 Robustness analysis

One challenge in trajectory recovery is the sparsity of data. To assess the model's performance under different degrees of data sparsity, we conduct a robustness analysis experiment following the practices in prior studies (Sun et al., 2021; Xia et al., 2021). We control data sparsity by altering the ratio of missing trajectory points in historical trajectories from 0% to 80% to simulate potential sparse data scenarios in the real world. In the 0% scenario, all historical trajectories are used for recovery. For the 20%, 40%, 60%, and 80% scenarios, the corresponding percentages of trajectory points in each historical trajectory are randomly masked as missing. The higher the missing rate in historical trajectories, the sparser are the data.

We compare CLMove with the two baselines specifically designed for trajectory recovery tasks, namely, PeriodicMove and TrajBERT. Figure 4 shows the Recall@1 and MAP of the three models under five different data sparsity levels. As expected, the recovery performance of all models decreases with increase of data sparsity. Additionally, CLMove maintains the best performance across all levels of data sparsity compared to the other two models, which demonstrates its robustness in handling various sparsity scenarios in mobility data.

4.4.4 Effect of pretrained location embedding

During the training of the fine-tuned GNN location encoder, we freeze the pretrained location embedding. We compared the freeze mode with the not-freeze mode in Table 4. For most metrics and datasets, we can find that the freeze mode is better, which is consistent with the results of previous work (Deng et al., 2022). We think the reason might be that if the pretrained location embedding is not frozen during the training of the fine-tuned GNN location encoder, then the collective pattern information stored in the pretrained location embedding

Table 2 Overall performance comparison

Dataset	Model	Recall@1	Recall@3	Recall@5	Dis@1	Dis@3	Dis@5	MAP
Geolife	Top	0.016	0.071	0.100	8451	7444	7351	0.069
	Time Top	0.032	0.078	0.116	8628	6908	6158	0.080
	LSTM	0.050	0.105	0.160	8971	6775	5884	0.109
	Bi-LSTM	0.156	0.242	0.291	8628	5998	5128	0.221
	DeepMove	0.130	0.282	0.351	8037	5852	4982	0.233
	AttnMove	0.130	0.219	0.260	7900	5842	5071	0.191
	PeriodicMove	0.173	0.295	0.359	7860	5063	4146	0.260
	GETNext	0.214	0.300	0.338	11328	6284	4788	0.271
	TrajBERT	0.217	0.285	0.309	13613	7706	6025	0.260
	CLMove	**0.224**	**0.362**	**0.421**	**6331**	**4174**	**3434**	**0.318**
Foursquare	Top	0.021	0.051	0.067	9338	8560	7359	0.055
	Time Top	0.018	0.045	0.064	9708	8312	7552	0.048
	LSTM	0.022	0.039	0.045	9209	8037	7059	0.037
	Bi-LSTM	0.113	0.184	0.221	9321	5946	4924	0.172
	DeepMove	0.253	0.385	0.435	5428	3037	2402	0.340
	AttnMove	0.235	0.317	0.350	6493	4349	3632	0.294
	PeriodicMove	0.259	0.397	0.451	5686	3057	2418	0.349
	GETNext	0.241	0.445	**0.530**	12201	5462	3739	0.368
	TrajBERT	0.192	0.248	0.269	12982	7584	5789	0.230
	CLMove	**0.272**	**0.452**	0.510	**5247**	**2581**	**1991**	**0.384**
Porto Taxi	Top	0.093	0.207	0.260	1240	1144	1100	0.189
	TimeTop	0.103	0.217	0.286	1236	1114	1047	0.201
	LSTM	0.106	0.217	0.288	1265	1146	1092	0.203
	Bi-LSTM	0.150	0.262	0.335	1250	990	868	0.247
	DeepMove	0.167	0.281	0.345	1259	951	843	0.261
	AttnMove	0.160	0.275	0.346	1260	**917**	**789**	0.257
	PeriodicMove	0.169	0.281	0.349	1245	925	799	0.264
	GETNext	0.112	0.218	0.283	**1186**	1093	1052	0.197
	TrajBERT	0.154	0.269	0.338	1270	997	886	0.249
	CLMove	**0.172**	**0.289**	**0.359**	1243	920	791	**0.270**

would be modified and dropped, and it thus goes against the recovery of the trajectory.

4.4.5 Sensitivity of hyperparameter hidden size

We further investigate the sensitivity of the hidden size of the hyperparameter . We change the hidden size in the range of $\{8, 16, 32, 64, 128\}$. From the result in Fig. 5, we can observe that as the hidden size increases, the performance of the model increases first; when it reaches a threshold, the performance starts to be stable and then declines. Experimental results show that for the three datasets, a hidden size of 128 is enough.

5 Conclusion and future work

In this paper, we propose CLMove, a novel human mobility trajectory recovery model based on contrastive learning. CLMove encodes individual and collective mobility patterns into location embeddings by the action of a pretrained and a fine-tuned location encoder. For the fine-tuned location encoder, the encoded information is propagated in both inter- and intra-trajectory modes through a GNN-based network in an iterative way. We further design a trajectory-level contrastive learning task to improve the robustness of CLMove. Extensive experimental analysis of three real-life datasets shows the effectiveness of our proposed model.

In the future, we plan to apply our model to other datasets with different granularities to evaluate the capability of our model when facing a finer-granular scenario. Furthermore, we plan to extend the proposed model by integrating more kinds of data sources, such as POI information, into our model.

Table 3 Ablation study

Model	Geolife			Foursquare			Porto Taxi		
	Rec@1	Dis@1	MAP	Rec@1	Dis@1	MAP	Rec@1	Dis@1	MAP
CLMove	**0.224**	**6331**	**0.318**	**0.272**	**5247**	**0.384**	**0.172**	**1243**	**0.270**
w/o Contrastive learning	0.216	6553	0.310	0.267	5428	0.376	0.171	1243	0.269
w/o Pretrained location encoder	0.178	7593	0.260	0.261	5438	0.360	0.164	1261	0.260
w/o Intergraph embeddings update	0.173	7860	0.260	0.259	5686	0.349	0.162	1260	0.259

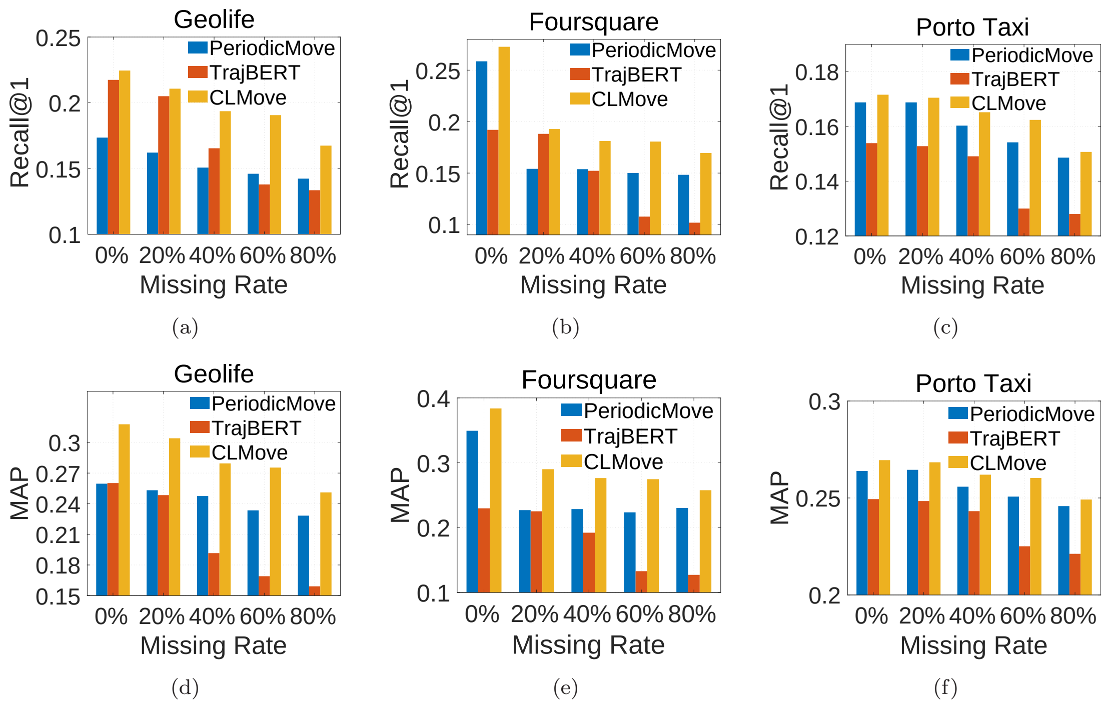

Fig. 4 Comparison of model performance under different missing rates of observable trajectory points: (a),(c) Geolife dataset; (b),(d) Foursquare dataset; (c),(f) Porto Taxi dataset.

Contributors

Yushan LIU and Yang CHEN designed the research. Yushan LIU processed the data. Yushan LIU drafted the manuscript. Yang CHEN and Jiayun ZHANG helped organize the manuscript. Yu XIAO and Xin WANG revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

Data available on request from the authors.

References

Chandio AA, Tziritas N, Zhang F, et al., 2016. Towards adaptable and tunable cloud-based map-matching strategy for GPS trajectories. *Frontiers Inf Technol Electron Eng*, 17(12):1305-1319. https://doi.org/10.1631/FITEE.1600027

Chen GS, Viana AC, Fiore M, et al., 2019. Complete trajectory reconstruction from sparse mobile phone data. *EPJ Data Sci*, 8(1):30. https://doi.org/10.1140/epjds/s13688-019-0206-8

Chen T, Kornblith S, Norouzi M, et al., 2020. A simple framework for contrastive learning of visual representations. Proc 37th Int Conf on Machine Learning, p.1597-1607.

Chen XL, He KM, 2021. Exploring simple siamese representation learning. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.15745-15753, https://doi.org/10.1109/CVPR46437.2021.01549

Cho E, Myers SA, Leskovec J, 2011. Friendship and mobility: user movement in location-based social networks. Proc 17th ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining, p.1082-1090, https://doi.org/10.1145/2020408.2020579

Chondrogiannis T, Bornholdt J, Bouros P, et al., 2022. History oblivious route recovery on road networks. Proc 30th Int Conf on Advances in Geographic Information Systems, article 44, https://doi.org/10.1145/3557915.3560979

Table 4 Effect of pretrained location embedding

Model	Geolife			Foursquare			Porto Taxi		
	Recall@1	Dis@1	MAP	Recall@1	Dis@1	MAP	Recall@1	Dis@1	MAP
Freeze	**0.224**	6331	**0.318**	**0.272**	**5247**	**0.384**	**0.172**	**1243**	**0.270**
Not freeze	0.211	**6056**	**0.318**	0.263	5386	0.373	0.160	1265	0.257

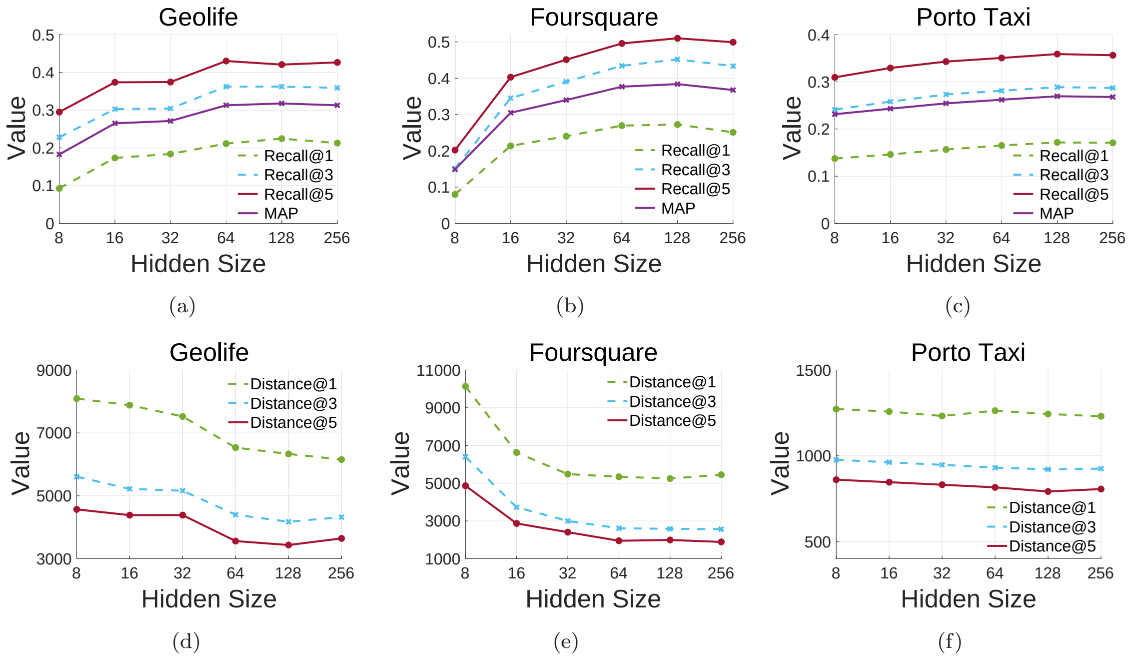

Fig. 5 Performance of the model under different hidden sizes: (a),(c) Geolife dataset; (b),(d) Foursquare dataset; (c),(f) Porto Taxi dataset.

Chung J, Gülçehre Ç, Cho K, et al., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling.
https://arxiv.org/abs/1412.3555

Deng LW, Zhao Y, Fu ZD, et al., 2022. Efficient trajectory similarity computation with contrastive learning. Proc 31st ACM Int Conf on Information & Knowledge Management, p.365-374,
https://doi.org/10.1145/3511808.3557308

Dhont M, Tsiporkova E, González-Deleito N, 2022. Mining of spatiotemporal trajectory profiles derived from mobility data. IEEE Int Conf on Data Mining Workshops, p.1-9,
https://doi.org/10.1109/ICDMW58026.2022.00133

Fang ZH, Yang Y, Yang G, et al., 2021. Cellsense: Human mobility recovery via cellular network data enhancement. *Proc ACM Interact Mob Wearable Ubiquitous Technol*, 5(3):100.
https://doi.org/10.1145/3478087

Fang ZQ, Du YT, Zhu XJ, et al., 2022. Spatio-temporal trajectory similarity learning in road networks. Proc 28th ACM SIGKDD Conf on Knowledge Discovery and Data Mining, p.347-356,
https://doi.org/10.1145/3534678.3539375

Feng J, Li Y, Zhang C, et al., 2018. Deepmove: Predicting human mobility with attentional recurrent networks. Proc World Wide Web Conf, p.1459-1468,
https://doi.org/10.1145/3178876.3186058

Gao Q, Wang XH, Liu CR, et al., 2023. Open anomalous trajectory recognition via probabilistic metric learning. Proc 32nd Int Joint Conf on Artificial Intelligence, p.2095-2103,
https://doi.org/10.24963/ijcai.2023/233

González MC, Hidalgo CA, Barabási AL, 2008. Understanding individual human mobility patterns.
https://arxiv.org/abs/0806.1256

He KM, Fan HQ, Wu YX, et al., 2020. Momentum contrast for unsupervised visual representation learning. Proc IEEE/CVF Conf Computer Vision and Pattern Recognition, p.9726-9735,
https://doi.org/10.1109/CVPR42600.2020.00975

Hristova D, Williams MJ, Musolesi M, et al., 2016. Measuring urban social diversity using interconnected geo-social networks. Proc 25th Int Conf on World Wide Web, p.21-30,
https://doi.org/10.1145/2872427.2883065

Li L, Li YB, Li ZH, 2013. Efficient missing data imputing for traffic flow by considering temporal and spatial dependence. *Transp Res Part C Emerg Technol*, 34:108-120.
https://doi.org/10.1016/j.trc.2013.05.008

Li XC, Zhao KQ, Cong G, et al., 2018. Deep representation learning for trajectory similarity computation. Proc IEEE 34th Int Conf on Data Engineering, p.617-628,
https://doi.org/10.1109/ICDE.2018.00062

Li XC, Cong G, Cheng Y, 2020. Spatial transition learning on road networks with deep probabilistic models. Proc

IEEE 36th Int Conf on Data Engineering, p.349-360, https://doi.org/10.1109/ICDE48307.2020.00037

Li YJ, Tarlow D, Brockschmidt M, et al., 2016. Gated graph sequence neural networks. Proc 4th Int Conf on Learning Representations.

Lin Y, Wan HY, Guo SN, et al., 2021. Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction. Proc 35th AAAI Conf on Artificial Intelligence, p.4241-4248, https://doi.org/10.1609/aaai.v35i5.16548

Liu Q, Wu S, Wang L, et al., 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. Proceedings of the 30th AAAI Conference on Artificial Intelligence, p.194-200, https://doi.org/10.1609/aaai.v30i1.9971

Luo YH, Cai XR, Zhang Y, et al., 2018. Multivariate time series imputation with generative adversarial networks. Proc 32nd Int Conf on Neural Information Processing Systems, p.1603-1614.

Noulas A, Shaw B, Lambiotte R, et al., 2015. Topological properties and temporal dynamics of place networks in urban environments. Proc 24th Int Conf on World Wide Web, p.431-441, https://doi.org/10.1145/2740908.2745402

Park D, Kang J, Song H, et al., 2022. Multi-view poi-level cellular trajectory reconstruction for digital contact tracing of infectious diseases. Proc IEEE Int Conf on Data Mining, p.1137-1142, https://doi.org/10.1109/ICDM54844.2022.00144

Ren HM, Ruan SJ, Li YH, et al., 2021. Mtrajrec: Map-constrained trajectory recovery via seq2seq multi-task learning. Proc 27th ACM SIGKDD Conf on Knowledge Discovery & Data Mining, p.1410-1419, https://doi.org/10.1145/3447548.3467238

Salakhutdinov R, Mnih A, 2007. Probabilistic matrix factorization. Proc 20th Int Conf on Neural Information Processing Systems, p.1257-1264.

Seng D, Lv FS, Liang ZY, et al., 2021. Forecasting traffic flows in irregular regions with multi-graph convolutional network and gated recurrent unit. *Frontiers Inf Technol Electron Eng*, 22(9):1179-1193. https://doi.org/10.1631/FITEE.2000243

Si JJ, Yang J, Xiang Y, et al., 2024. Trajbert: Bert-based trajectory recovery with spatial-temporal refinement for implicit sparse trajectories. *IEEE Trans Mob Comput*, in press. https://doi.org/10.1109/TMC.2023.3297115

Sun H, Yang CJ, Deng LW, et al., 2021. Periodicmove: Shift-aware human mobility recovery with graph neural network. Proc 30th ACM Int Conf on Information & Knowledge Management, p.1734-1743, https://doi.org/10.1145/3459637.3482284

Vaswani A, Shazeer N, Parmar N, et al., 2017. Attention is all you need. Proc 31st Int Conf on Neural Information Processing Systems, p.6000-6010.

Wang YY, Jiang WH, Pu SL, et al., 2020. Learning embeddings of a heterogeneous behavior network for potential behavior prediction. *Frontiers Inf Technol Electron Eng*, 21(3):422-435. https://doi.org/10.1631/FITEE.1800493

Wei LY, Zheng Y, Peng WC, 2012. Constructing popular routes from uncertain trajectories. Proc 18th ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining, p.195-203, https://doi.org/10.1145/2339530.2339562

Wu H, Mao JY, Sun WW, et al., 2016. Probabilistic robust route recovery with spatio-temporal dynamics. Proc 22nd ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining, p.1915-1924, https://doi.org/10.1145/2939672.2939843

Wu S, Tang YY, Zhu YQ, et al., 2019. Session-based recommendation with graph neural networks. Proc 33rd AAAI Conf on Artificial Intelligence, p.346-353, https://doi.org/10.1609/aaai.v33i01.3301346

Xi DB, Zhuang FZ, Liu YC, et al., 2019. Modelling of bi-directional spatio-temporal dependence and users' dynamic preferences for missing POI check-in identification. Proc 33rd AAAI Conf on Artificial Intelligence, p.5458-5465, https://doi.org/10.1609/aaai.v33i01.33015458

Xia T, Qi YH, Feng J, et al., 2021. Attnmove: History enhanced trajectory recovery via attentional network. Proc 35th AAAI Conf on Artificial Intelligence, p.4494-4502, https://doi.org/10.1609/aaai.v35i5.16577

Xu Y, Xu JJ, Zhao J, et al., 2022. Metaptp: An adaptive meta-optimized model for personalized spatial trajectory prediction. Proc 28th ACM SIGKDD Conf on Knowledge Discovery and Data Mining, p.2151-2159, https://doi.org/10.1145/3534678.3539360

Yang DQ, Zhang DQ, Zheng VW, et al., 2015. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Trans Syst Man Cybern Syst*, 45(1):129-142. https://doi.org/10.1109/TSMC.2014.2327053

Yang S, Liu JM, Zhao KQ, 2022. Getnext: Trajectory flow map enhanced transformer for next POI recommendation. Proc 45th Int ACM SIGIR Conf on Research and Development in Information Retrieval, p.1144-1153, https://doi.org/10.1145/3477495.3531983

Zhang WY, Xia DW, Chang GY, et al., 2022. APFD: an effective approach to taxi route recommendation with mobile trajectory big data. *Frontiers Inf Technol Electron Eng*, 23(10):1494-1510. https://doi.org/10.1631/FITEE.2100530

Zhao J, Xu JJ, Zhou R, et al., 2018. On prediction of user destination by sub-trajectory understanding: A deep learning based approach. Proc 27th ACM Int Conf on Information and Knowledge Management, p.1413-1422, https://doi.org/10.1145/3269206.3271708

Zheng Y, Li QN, Chen YK, et al., 2008. Understanding mobility based on GPS data. Proc 10th Int Conf on Ubiquitous Computing, p.312-321, https://doi.org/10.1145/1409635.1409677

Zheng Y, Zhang LZ, Xie X, et al., 2009. Mining interesting locations and travel sequences from GPS trajectories. Proc 18th Int Conf on World Wide Web, p.791-800, https://doi.org/10.1145/1526709.1526816

Zheng Y, Xie X, Ma WY, 2010. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng Bull*, 33(2):32-39.

Zhou F, Wang PY, Xu X, et al., 2022. Contrastive trajectory learning for tour recommendation. *ACM Trans Intell Syst Technol*, 13(1):4. https://doi.org/10.1145/3462331