HERMAS: A Human Mobility Embedding Framework with Large-scale Cellular Signaling Data

YIWEI SONG, Peking University
DONGZHE JIANG, Peking University, China
YUNHUAI LIU*, Peking University, China
ZHOU QIN, Rutgers University, United States
CHANG TAN, iFlytek, China
DESHENG ZHANG, Rutgers University, United States

Efficient representations for spatio-temporal cellular Signaling Data (SD) are essential for many human mobility applications. Traditional representation methods are mainly designed for GPS data with high spatio-temporal continuity, and thus will suffer from poor embedding performance due to the unique Ping Pong Effect in SD. To address this issue, we explore the opportunity offered by a large number of human mobility traces and mine the inherent neighboring tower connection patterns. More specifically, we design HERMAS, a novel representation learning framework for large-scale cellular SD with three steps: (1) extract rich context information in each trajectory, adding neighboring tower information as extra knowledge in each mobility observation; (2) design a sequence encoding model to aggregate the embedding of each observation; (3) obtain the embedding for a trajectory. We evaluate the performance of HERMAS based on two human mobility applications, i.e. trajectory similarity measurement and user profiling. We conduct evaluations based on a 30-day SD dataset with 130,612 users and 2,369,267 moving trajectories. Experimental results show that (1) for the trajectory similarity measurement application, HERMAS improves the Hitting Rate (HR@10) from 15.2% to 39.2%; (2) for the user profiling application, HERMAS improves the F1-score for around 9%. More importantly, HERMAS significantly improves the computation efficiency by over 30×.

CCS Concepts: • **Networks** → **Location based services**; • **Human-centered computing** → *Collaborative and social computing*; • **Computing methodologies** → *Model development and analysis.*

Additional Key Words and Phrases: Signaling Data, Representation Learning, Trajectory Embedding, Regular Pattern Exploration

ACM Reference Format:
Yiwei Song, Dongzhe Jiang, Yunhuai Liu, Zhou Qin, Chang Tan, and Desheng Zhang. 2021. HERMAS: A Human Mobility Embedding Framework with Large-scale Cellular Signaling Data. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 3, Article 128 (September 2021), 21 pages. https://doi.org/10.1145/3478108

*Corresponding author.

Authors' addresses: Yiwei Song, Peking University, No.5 Yiheyuan Rd. Haidian District, Beijing, China, yiwei.song@pku.edu.cn; Dongzhe Jiang, Peking University, No.5 Yiheyuan Rd. Haidian District, Beijing, China, dongzhe.jiang@pku.edu.cn; Yunhuai Liu, Peking University, No.5 Yiheyuan Rd. Haidian District, Beijing, China, yunhuai.liu@pku.edu.cn; Zhou Qin, Rutgers University, New Jersey, United States, zq58@cs.rutgers.edu; Chang Tan, iFlytek, Hefei, China, changtan2@iflytek.com; Desheng Zhang, Rutgers University, New Jersey, United States, desheng.zhang@cs.rutgers.edu.

2474-9567/2021/9-ART128 $15.00
https://doi.org/10.1145/3478108

1 INTRODUCTION

Currently, the proliferation of mobile phones and cellular networks and their resultant data brings opportunities for various human mobility applications[7]. In particular, **Signaling Data (SD)** are collected when the mobile phones interact with the cell towers, e.g., calling, sending messages, or switching to other cells. They thus provide high penetration rates and high spatio-temporal coverage [25] in an automatic and non-intrusive manner [39]. A deep understanding of human mobility with spatio-temporal data is essential for many mobility applications, ranging from transportation management [3], urban planning [4], to network infrastructure construction [24]. Among all these applications, representation learning is widely recognized as an effective approach to mine the inherent features of the human mobility and be beneficial for many use cases [8, 32]. For instance, computing the similarity among trajectories based on representation data rather than the raw data can help to reduce the computational complexity, save the storage space, and protect the user privacy [34], User profiling based on represented data can help forecast population and facilitate the network infrastructure construction [12].

Representation learning (RL) has been extensively studied and exploited in spatio-temporal data analysis. In latest researches, RL first learns a representation vector(embedding) for each spatio-temporal observation by mining context relationships between observations in spatio-temporal trajectories. And the trajectories could be embedded by a sequential model such as RNN[29]. For example, the Word2Vec technique [22, 23] is applied to transit the original spatio-temporal data(typically GPS records) to certain locations by mapping to urban infrastructure, e.g., road segments. As such, the embeddings of trajectories are approached by a sequence of approximated road segments [40]. These approaches are mainly designed for GPS because the localization error of GPS is widely assumed to follow a normal distribution [20], resulting in a reasonable approximation.

Different from GPS data, an SD observation can only tell that a cell phone user is associated with a cell tower at certain time, without the precise locations of the user. Moreover, cell phones are not always connected to the closest cell tower due to the load balance considerations, which means that they may connect to any connectable cell towers based on real-world network loads. We define this phenomenon as the **Ping Pong Effect** (**PPE**), and the potentially connectable towers as neighbor towers. As illustrated in Fig. 1, given an observed tower record, the potential location of the user may be anywhere within the coverage area centered on the observed tower. The cell towers within the radius of twice the coverage radius of the observed tower are therefore considered as the neighbor towers. Due to PPE, it is unknown whether the observed tower or which neighbor tower the user will connect to the next time when passing nearby. Hence, the state-of-the-arts methods are not enough to learn the mobility pattern of users by exclusively mining context relationships.

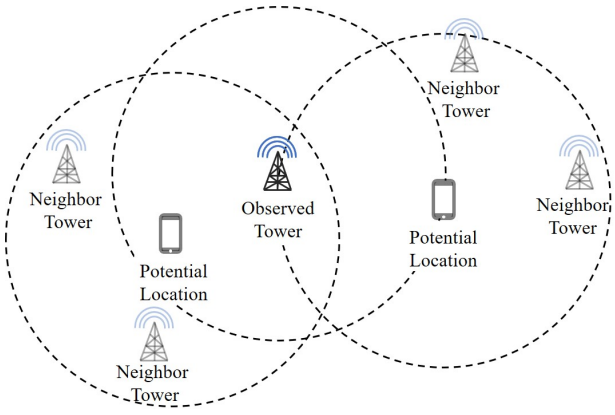

Fig. 1. An Illustration of Ping Pong Effect

Here we refer to the embedding of each cell tower as tower embedding, because spatio-temporal observations in SD data are cell towers. Although PPE increases the uncertainty of trajectories, it essentially consists of unclear connection patterns between cell towers. If we can make an exploration of the PPE from historical data, i.e., modeling the relationship between the observed tower and the corresponding neighbor towers, more human mobility related knowledge in SD would be added to each tower's embedding. As we mentioned before, the high penetration rates of SD indicate the potential for sensing human mobility, which reveals the PPE as connection patterns of each cell tower between neighbor towers. Thus the key idea of this paper is to model the connection patterns first and make use of it as extra information. With more embedded human mobility information, our embedding of SD enables better performance in downstream applications.

However, modeling the connection patterns between cell towers due to PPE is a very challenging task. It is almost impossible to exhaust all possible connection patterns between cell towers. Only considering a part of the data cannot provide full coverage of human mobility on city wide. There are indeed some studies that attempt to exploit the relationships between observations and neighbor observations to enhance the ability of spatio-temporal data embedding to represent human mobility. For example, [21] assumes that the probability of connection between observations of spatio-temporal data is inversely proportional to the geographical distance. But the combination of population mobility, environmental conditions, load balance design, and other factors make the PPE in SD not simply measurable by geographical distance. Here in this paper, we design a deep learning approach to investigate the connection patterns between cell towers to incorporate a deep understanding of human mobility.

In this paper, we design a representation learning framework called HERMAS for SD data dedicated to urban-wide human mobility-related applications, e.g. trajectory similarity measurement and user profiling. As a whole, HERMAS learns an expression for each cell tower on a basis of SD data, and uses tower embeddings to construct SD trajectory embeddings as input for downstream applications. Focusing on the unique characteristic of SD's PPE, HERMAS enhances the understanding of it via a deep model based on a large number of tower-to-tower connection patterns. The resulting tower embeddings would be more effective in representing SD data with PPE. Further, the construction of trajectory embedding is managed via a sequential model. We conduct the comparative experiments with several widely used representation learning frameworks. The evaluation shows that HERMAS can efficiently extract the user mobility using SD on a large spatio-temporal scale. The contribution of this paper is summarized below.

- To our knowledge, this is the first work to study a human mobility embedding framework via large-scale cellular signaling data. Conceptually, we cope with the Ping Pong Effect by extracting neighbor tower information into each tower's embedding. We further design the embedding model to obtain trajectory embedding. The learned embeddings are able to preserve human mobility with a strong expression capability in human mobility applications.
- We design a novel framework HERMAS for SD data with a purpose of efficiently solving mobility-related applications at an urban scale. In terms of the key characteristics of the geographical relationship of SD, we take the initiative to work out tower embedding first by combining context tower embedding and neighbor tower embedding. Then, the SD trajectory is also embedded consequently for practical applications by distinct consequential encoding modules.
- Our framework HERMAS is evaluated by a large-scale SD dataset with 257 million daily records generated from more than 3.6 million users. Two human mobility applications are adopted and compared with 3 competitive baselines. HERMAS has a maximum improvement of 7.3% in similarity measurement accompanied by dozens of times faster. Additionally, HERMAS performs averagely 3.6% better than the most competitive baseline model in the application of user profiling.

2 MOTIVATION

2.1 Background

In recent years, the development of positioning infrastructure such as cellphones has enabled the collection of massive spatio-temporal data that depict human mobility. Meanwhile, advancing machine learning techniques has made the feature-based models quite effective in human mobility modeling and inference[39] . Although these feature-based models perform well, they are time-consuming and labor-consuming. Rapid urbanization and the need for human mobility development require researchers to design spatio-temporal features that are less dependent on specific problems to solve multiple new problems quickly and efficiently. Therefore, there is a tendency to use representation learning to extract useful information from raw spatio-temporal data, generating embedding to serve as input features with a strong generalization ability to approach multiple applications.

2.2 Characteristics of SD

With the extension of cellular networks and cellphones, SD become a valuable spatio-temporal data. On the one hand, cellphones have become a necessity for the vast majority of people, making the participated user number very large. Given the highly-dense distributed cell towers cities, users can interact with cellular towers wherever they are, leading to the large spatial coverage of cellular data. On the other hand, as long as users turn on their cellphones or even in the standby mode, SD are recorded by all sorts of cellular activities that may happen anytime. Subsequently, the high penetration rates of SD benefits the researchers to investigate human mobility applications.

2.2.1 Dataset. In this paper, we utilize the SD dataset from CTCC, one of the three largest cellular network operators in China. The dataset was generated from Hefei, a large Chinese city with a population of more than 8 million. Our dataset contains data for 3.6 million users during June 1, 2017, and June 30, 2017, with 257 million records per day. In this city, the three operators have similar market share and share the cellular network infrastructure. In other words, our dataset records almost unbiasedly the mobility of users throughout the city.

2.2.2 Data Format. In general, the data format of the SD records contain time-stamp, cell ID, user ID (anonymous ID), as well as the signaling type. Among them, the cell ID is an identification of the cell tower associated with the cellular record, and the user ID is an identification of the corresponding user of the cellular record. And both IDs are unique. Signaling type records the type of services for the signaling record, which has little to do with our investigation.

2.2.3 SD Trajectory. Similar to the common definition of spatio-temporal trajectory [38], an SD trajectory is a sequence of consecutive observations over a period of time. SD trajectories provide a large quantity of information to understand human mobility. An SD trajectory differs from a GPS trajectory in two obvious ways.

- Each observation in the SD data is the location of a cellular tower at the time the cell phone interacts with the cellular tower. In a city, the number and location of cell towers are fixed. Therefore the possible locations of each observation in the SD trajectory are fixed and known.
- In addition, SD data is collected in a user-unaware manner, so pre-processing the raw SD data to extract the SD trajectories is an essential step in the study of SD trajectories.

2.2.4 PPE. As we mentioned earlier, the PPE depicts the pattern of each observation-generating process in the SD data. A combination of environmental conditions, load balancing, and other factors cause the phone to be potentially randomly connected to its connectable tower. The actual location of the user is unknown, but this does not mean that the SD data lacks research value. As we mentioned before, SD is favored by many researchers due to its wide penetration rates and spatio-temporal coverage. In this circumstance, SD observations provide

plenty of information for our understanding of human mobility. In the following, we elaborate on the differences between PPE and some common trajectory phenomena in spatio-temporal trajectories.

- **Outlier.** The observations generated due to PPE are not outliers. Outliers are data recording errors that occur from equipment errors, data storage, and other processes and exist in all spatio-temporal data [35]. Outliers do not provide valuable information for the study of human mobility and therefore are necessary to be removed.
- **Stay Point.** PPE are not happened only when the user stays around; PPE are happened when the user's cell phone interacts with the cell towers, regardless of whether the user is or not moving. Therefore, traditional stay point or stay area identification algorithms do not reasonably model PPE [36]. In this paper, we focus on modeling PPE in movement trajectories.
- **Trajectory Uncertainty.** Compared with GPS trajectories, PPE does increase the uncertainty of SD trajectories. Popular trajectory uncertainty modeling algorithms usually assume that the uncertainty of spatio-temporal data satisfies some general distributions, such as a normal distribution with geographical distance as a parameter [27]. However, the location of each observation in SD data does not satisfy some specific distribution, so the traditional assumption methods are limited in the current conditions.

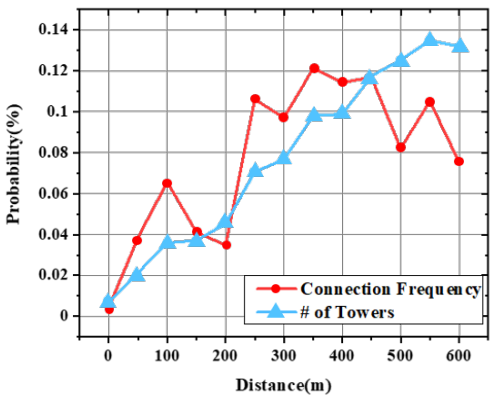

Fig. 2. Tower Distribution vs Connection Frequency

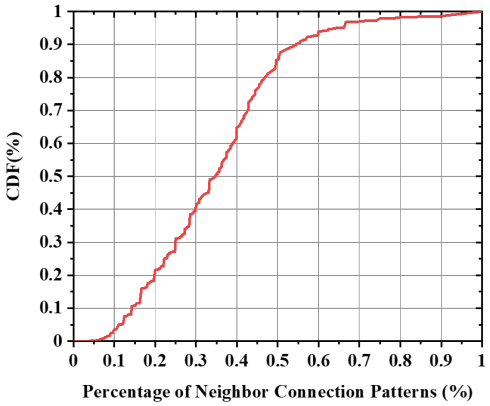

Fig. 3. Neighbor connection Pattern

2.3 Challenge

Essentially, the traditional methods are only applicable to precise spatio-temporal data such as GPS and learn the representation vector for each observation in this way. However, the PPE of SD data makes the uncertainty of the location of the observations not modelable in terms of distance, which makes the traditional approaches not applicable to do representation learning for SD. To illustrate this graphically, we extracted the connection frequency between a given tower and its neighbor towers respectively based on the SD trajectory (processing details is shown in Section 3.2).

Here, we refer to the given tower as the center tower, relative to its neighboring towers. In Fig. 2, we use the blue line to indicate the probability distribution of the number of neighboring towers at different distance slots from the center tower to the total number of neighboring towers. It is obvious that the more distant the

neighboring towers are from the center tower, the more they are distributed. In addition, we use the red line to indicate the connection probability of each center tower with its neighboring towers obtained based on the SD data. What can be found is that the connection pattern between the center tower and its neighboring towers does not correspond to the number of its neighbor towers. The neighbor towers with the highest connection probability to the center tower are located at about 350m. At the same time, this probability is only about 2/3 of the probability at 350m at a distance of 600m. This is because the connection patterns between the neighbor towers due to PPE are not directly correlated with the distance. Moreover, different towers do not have the same pattern as their neighboring towers, which also increases the difficulty of modeling PPE in our spatio-temporal data representation.

2.4 Opportunity

Although state-of-the-arts methods are limited due to the PPE of SD, the large number of neighbor connection patterns that exist in SD provides an opportunity to model it. We illustrate these neighbor connection patterns present in a large number of trajectories in Fig. 3. Given a center tower and a neighboring tower with which connections occur in a trajectory, Fig. 3 illustrates the cumulative probability of the frequency of connections occurring between that center tower and other neighboring towers as a proportion of the frequency of connections occurring between it and all neighboring towers. We observe that about 60% of the towers have at least 30% more connection patterns occurring in other trajectories. In addition, at least 40% of the towers have at least 40% or more connection patterns occurring in other trajectories. In other words, by mining a large number of neighbor connection patterns present in SD trajectories, more information reflecting human mobility can be learned than in the state-of-the-arts methods by only the connection patterns between towers in a single trajectory.

3 SYSTEM DESIGN

An overview of our system is shown in Fig. 4. It is composed of three main parts: preprocessing, mobility embedding framework(HERMAS), and applications. Specifically, the SD data, as the input to the system, is transformed into trajectory data as the input of HERMAS after a preprocessing part. HERMAS is the representation learning framework of the system, in which the trajectory embedding is output through the tower representation learning module and a representation learning module in turn. Finally, trajectory embedding is used as the input to the downstream applications as an evaluation of HERMAS.

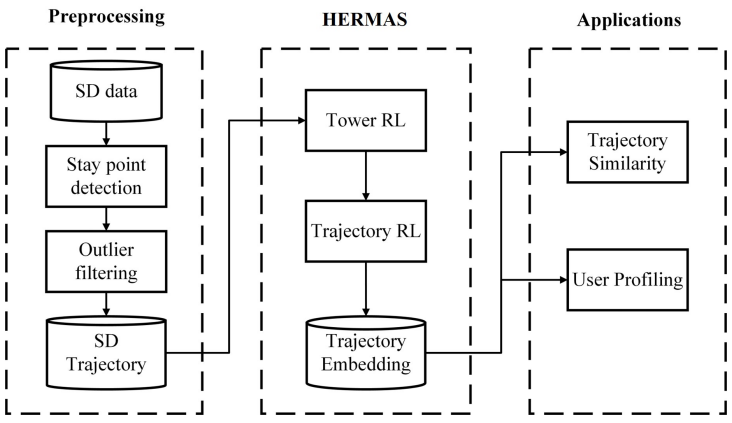

Fig. 4. System Overview

3.1 Terminology

In this paper, we use the following basic notions based on SD data.

Definition 1. ***Observation***. *Each observation o in our framework is the geographical location of a cell tower, represented as a tuple of <longitude, latitude >, each with a time stamp. Besides, the set of cell towers O is given in the dataset, so each observation o uniquely corresponds to a cell tower.*

Definition 2. ***SD Trajectory***. *Given a trajectory set S, the i^{th} trajectory $t_i = \{o_1, o_2, \ldots, o_{|t_i|}\}$ is one of the chronologically ordered observation sequence. o_k is the k^{th} observation of this trajectory for $k \in [1, |t_i|]$. $|t_i|$ is length of the trajectory t_i.*

Definition 3. ***Context Tower***. *Given an observed tower o_i, we define the set of its context towers $C(o)$ as the set of all its context towers. For example, in the SD trajectory $t = \{o_1, o_2, \ldots, o_{i-1}, o_i, o_{i+1}, \ldots, o_{|t_i|}\}$, o_{i-1} and o_{i+1} are selected as the context towers. Note that it does not make sense to gain context towers from all trajectories. We choose the towers within a radius of 1 kilometer centered on o_i, since the maximum distance between two observations in the trajectory is assumed to be 1 kilometer.*

Definition 4. ***Neighbor Tower***. *Given an observed tower o, we define the set of its neighbor towers $\mathcal{N}(o)$ as the set of all connectable towers to which the user may be connected. In this paper, considering all the coverage radius r of a cell tower is 300 meters, thus the user may be within $r = 300m$ centered on the observed tower o. Thus, all towers within a radius of $2 \times r = 600m$ centered on the observed tower o constitute the set of neighboring towers $\mathcal{N}(o)$.*

3.2 Preprocessing

SD data in the collection process produce a lot of errors and are almost impossible to be used directly by human mobility-related applications. Instead of using raw data directly, we have to make a preprocessing. Preprocessing reduces the negative impact of errors on the experiment on the one hand and prepares input data for subsequent models and tasks on the other. Overall, preprocessing is divided into two steps: stay point detection and outlier filtering.

3.2.1 Staying Point Detection. The trajectory generated when users are active is a primary component of depicting the user's mobility. However, SD data are collected continuously and insensibly, so we cannot directly obtain SD trajectories directly from the original SD data. A widely used technique to derive trajectories is to detect stay points as the basis for dividing the trajectories chronologically. Once the user's staying point is recognized, the trajectories are obtained. The detection of staying points has been widely studied, and in this paper, we make the extraction of trajectories based on a typical staying point detection method [36]. First, we define a trajectory with a minimum traverse distance of 1 kilometer and lasted at least 15 minutes. This definition is empirically validated and guarantees the integrity and significance of trajectories. Consequently, we segment the SD data according to the temporal gap of at least 10 minutes as a criterion to obtain candidate trajectories. Then, the final trajectories input to next the procedure are selected according to the above definition.

3.2.2 Outlier Filtering. The outlier in SD is closely related to cellular networks for ambiguous reasons involving operator strategies, the durability of the towers, etc. These anomalies are meaningless, so it is reasonable to remove them for the integrity of human mobility. In this paper, we use a two-stage approach to detect and clear anomalies.

- **Heuristic method.** We refer to T-Drive's method of heuristic exception detection [35] to remove detected outliers. We first set a maximum speed threshold (i.e. 150 km/h) and assume that all those that are faster than this threshold are outlier values. Specifically, we calculate the distance and time gap between each

observation and its previous one based on time and distance in the trajectory and delete observations that exceed the maximum velocity threshold.

- **Smoothing.** We use a two-stage smoothing algorithm to eliminate the effects of the ping-pong effect [17]. The first stage replaces some large outliers with a small sliding window through the means. This processing does not significantly change the prototype of the trajectory while making it more realistic. We then fill the trajectory with a 5-second interval of linear interpolation, which is based on a moving average of the second sliding window. Specifically, the tower closest to the geometric center point of several original points in the sliding window as the tower to be interpolated to the trajectory.

3.3 HERMAS Design

In this section, we introduce the details of HERMAS, which consists of two parts: tower representation learning and trajectory representation learning. The tower representation learning algorithm learns a latent embedding for each tower using SD trajectories. To model the uncertainty introduced by PPE in SD, neighbor tower information is dynamically added to each tower embedding. Specifically, we use an attention-based deep model to extract the neighboring connection patterns from a large number of trajectories as the neighbor tower information. Subsequently, we replace each observation in the SD trajectory with its corresponding tower embedding and use a GRU-based sequence model to obtain the trajectory embedding for each trajectory.

3.3.1 Tower Representation Learning. In this section we learns a latent vector for each tower, which can be seen as pre-training for the each tower's embedding. We first learn a context tower embedding for each tower that represents human mobility based on the context observations in the SD trajectory. Subsequently, we design an attention-based neighbor tower information extraction module to learn a neighbor information embedding for each tower based on the SD's property of PPE. Finally, we combine these two embeddings to obtain the embedding for each tower.

First, we introduce **Context Tower Embedding**. The key idea of context tower embedding is that embeddings with more information related to human mobility can achieve good results in downstream applications. We borrow the idea of skip-gram [22] here. The idea behind skip-gram is to represent human mobility by the sequential order of different towers in a trajectory. If several towers appear sequentially in an SD trajectory, then we use the center tower to predict the towers that appear before and after it and embed them in the same latent space.

Here in the SD trajectory $t = \{o_1, o_2, \ldots, o_{i-1}, o_i, o_{i+1}, \ldots, o_{|t_i|}\}$, we define the context tower embedding of a given observation o_i in this trajectory as E_i^c. We use the current observation o_i to predict its context observations, with the training goal of maximizing the prediction probability. We refer to o_{i-1} and o_{i+1} as o_{ic}, and the objective could be formulated as the following expression

$$\sum_{(o_{ic}, o_i) \in t} \log P\left(o_{ic} \mid o_i\right) \tag{1}$$

where $P\left(o_{ic} \mid o_i\right)$ measures the connection patterns between o_{ic} and o_i in trajectory t. We use the dot product of vectors to qualify it as u_c.

$$u_c = E_i^c \cdot E_{ic}^c \tag{2}$$

where E_{ic}^c is the context tower embedding of o_{ic}. And with the softmax function, the objective function could be represented as

$$\sum_{(o_{ic}, o_i) \in t} \log P\left(o_{ic} \mid o_i\right) = \sum_{(o_{ic}, o_i) \in t} \log \frac{\exp^{u_c}}{\sum_{k=1}^{|C(o_i))|} \exp^{u_k}} \tag{3}$$

where $C(o_i)$ means the set of context towers of o_i. Specifically, we calculate the set of context towers for all towers in advance.

To conclude, we update the context tower embedding for each tower by predicting its context towers in the with the above loss function. After that, the learned embeddings are sent to **Neighbor Tower Information Extraction** described in the following paragraph.

To cope with the property of PPE of SD, we design a neighbor tower information extraction algorithm based on attention mechanism. Specifically, given an observation tower o_i and its neighbor tower set $\mathcal{N}(o_i)$, the neighbor tower information of o_i is dynamically added to its embedding in the process of tower representation learning. Here we illustrate our main intuition for designing the attention-based approach to explore the connection patterns between o_i and neighbor towers $o_{in}^n \in \mathcal{N}(o_i)$. The probability that a user connects to its different neighbor towers presented by PPE is different, i.e., each neighbor tower has a different connection pattern to a given tower. Therefore it is significant to distinguish the connection patterns of different neighbor towers to o_i and utilize those information. For the formation of the neighbor tower set $\mathcal{N}(o_i)$, we use the large number of neighbor connection patterns in the SD data to accomplish this. Specifically, we look for neighboring towers from the trajectory dataset with which connections have occurred with o_i. It is a fact that these connection patterns occurring in other trajectories can provide a large amount of neighbor tower information for PPE. Besides, for each selected neighbor tower, we additionally select an equal number of neighbor towers centered on it. These second-order neighbor towers enrich the neighbor tower information.

Considering that the size of neighbor tower set varies from different towers, here we group the neighboring towers using the distance Δd_i from the observation tower o_i, specifically dividing the neighboring towers into groups according to $\Delta d = 100m$. The vector w_d is randomly initialized and updated in the training phase. Each group corresponds to a weight vector w_d, and all the weight vectors form a weight matrix W^n for a shard transformation of tower embeddings. The attention module scans the neighbor towers, measuring the connection relationship with the corresponding weight vector. Here we refer the correspondence between o_i and one of its neighbor towers o_{in} as e_{in}

$$e_{in} = \sigma\left(W^n[E_i^c; E_{in}^c] + b^o\right). \tag{4}$$

where b^o is the bias and σ is the logistic sigmoid function. Further, the weight parameter α_{in} for each neighbor tower o_{in}^n is derived via a softmax function performed on e_{in}. It seeks to learn the importance of each neighbor tower in regard to the observed tower[28].

$$\alpha_{in} = \frac{\exp\left(e_{in}\right)}{\sum_{j \in \mathcal{N}(o_i)} \exp\left(e_{in}\right)} \tag{5}$$

Then the α_{in} are deployed to compute the weighted sum of the neighbor tower embeddings, resulting in a neighbor embedding E_i^n of o_i

$$E_i^n = \sigma\left(\sum_{n \in \mathcal{N}(c_i)} \alpha_{in} \mathbf{W} E_n^c\right). \tag{6}$$

In summary, we first extract the context information of the given tower o_i and embed it into the vector E_i^c to obtain the basic human mobility pattern. After that we dynamically extract the tower's neighbor tower information into E_i^n and combine it with E_i^c to get the tower's embedding E_i^o

$$E_i^o = \sigma\left(W^o[E_i^c; E_i^n] + b^n\right). \tag{7}$$

where b^n is the bias and W^o is the weighted function for transformation here. The tower embedding algorithm is presented in Algorithm 1.

Algorithm 1 Tower Embedding Algorithm

Input:
Trajectory set S; Cell tower set O; Weight matrix $\mathbf{W}$.
Output:
Tower embeddings E^c for all towers in O.

```
Initialize W, E^o
Find the context tower set C(o) and neighbor tower set N(o) for all towers in O.
for t ∈ T do
    for o_i ∈ t = {o_1, o_2, ..., o_|t|} do
        Perform context tower embedding algorithm of o_i to get E_i^c.
        for o_ij^n ∈ N(o_i) do
            Calculate α_ij using E_i^o, E_j^o and W^n by Equation 5.
        end for
        Perform neighbor tower embedding algorithm to get E_i^n.
        Calculate E_i^o using E_i^c and E_i^n by Equation 7.
    end for
end for
return E^o
```

3.3.2 Trajectory Representation Learning. In this part, considering that a trajectory is composed of a temporally continuous number of observations, we employ a model of Recurrent Neural Network (RNN)[29] to embed the trajectory. In the scenario of trajectory embedding, RNN serves as a encoding model consisted of sequential memory cells with learnable weights to learn expression [29]. To deal with the long-term memory of long trajectories, a well-known GRU (Gated Recurrent Unit) [10] skilled in distilling sequential information is adopted.

The framework of trajectory embedding is shown in Fig. 5. The model takes the trajectory of $t_i = \{o_1, o_2, \ldots, o_{|t_i|}\}$ as input, resulting a trajectory embedding E_i^t. Note that each input $\mathbf{x}_j$ here is the tower embedding of o_j, e.g. $\mathbf{x}_j = E_j^o$, $j \in [0, |t_i|)$.

Besides, due to the ambiguous mechanism of cellular activities, SD data have a very uncertain time granularity, accompanied by irregular time intervals, which brings challenges to the traditional RNN model. To alleviate the chaos in the temporal relationship caused by the instability of the time interval, we interpolate and fill the preprocessed trajectory sequence. The interpolation is described in Section 3.2. This is associated with two considerations: a) SD data are not spatially fine-grained and the trajectory with the outliers removed is very uncertain temporarily, thus the interpolation in terms of a temporally small range of original observations will not cause the user's mobile characteristics to impaired; b) Immobilizing the time interval makes full use of the potential of seq2seq models such as RNN to extract features [9].

A graphical depiction of GRU cell is shown in Fig. 6. The input of each GRU unit is the previous hidden output h_{t-1} and the input $\mathbf{x}_t$ at the current state. The entire gate control structure consists of two main gates: *reset* gate and *update* gate. The *reset* gate is given by

$$r = \sigma\left(W_r\left[\mathbf{x}_t; h_{t-1}\right]^T\right) \tag{8}$$

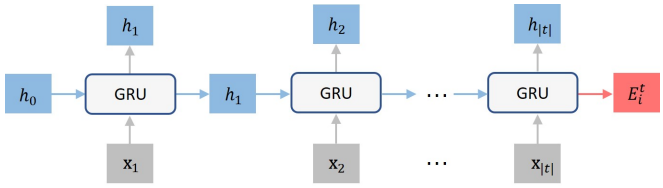

Fig. 5. Trajectory embedding

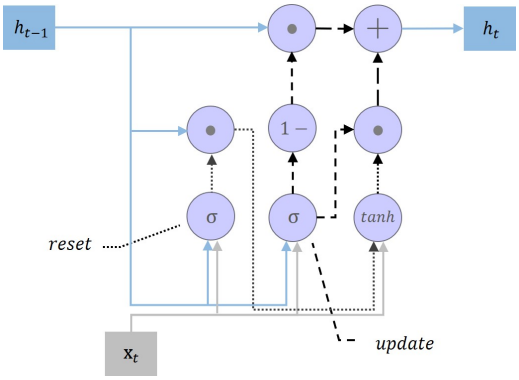

Fig. 6. A GRU cell

where r is the *reset* gate, and σ is the logistic sigmoid function. W_r represents the weighted matrix of the *reset* gate to synthesize the information of $\mathbf{x}_t$ and h_{t-1}. The sigmoid function maps the value into (0, 1) to become a memorization portion. The closer of memorization portion to 1 means the more part that should be memorized. The *reset* gate selectively forgets part of the hidden state and obtains a temporary hidden state h' with a hidden state worth remembering:

$$h' = \tanh\left(W_h\left[\mathbf{x}_t; h_{t-1} \odot r\right]\right) \tag{9}$$

where $\odot$ is the hadamard product. The *update* gate is resulted from the previous hidden state h_{t-1} and the temporary input $\mathbf{x}_t$. Equally, the computation of *update* gate is given by

$$z = \sigma\left(W_z\left[\mathbf{x}_t; h_{t-1}\right]^T\right) \tag{10}$$

where z is the *update* gate. Combining the information of h_{i-1} and h', the current hidden state of the GRU cell is computed as

$$h_t = z \odot h_{i-1} + (1 - z) \odot h' \tag{11}$$

where the *update* gate controls the retaining portion of the previous state and its relationship that conveys to the current state. The GRU cell ignores the information that learned to be irrelevant in the trajectory, resulting in a more compact representation for the spatio-temporal trajectory data[10].

4 EXPERIMENTS

In this section, we carry out the data-driven evaluation in detail by two applications: trajectory similarity measurement and user profiling. The application of trajectory similarity measurement is used to study whether the trajectory embedding module efficiently extracts discriminative information from the moving trajectory, which is shown in Section 4.2. We also investigate the impact of the important parameters in 4.2.5. And in Section 4.3, two novel tasks related to user profiling are employed to verify trajectory embedding associated with user mobility. Besides, the impact of decisive factors in tower embedding is also evaluated in Section 4.4. We investigate whether the vectors of observations can continuously maintain the geographical relationship after being embedded into representation space to ensure the mobility regularity of users.

4.1 Setting & Baselines

As mentioned earlier, the statistics of SD data of different users vary greatly. We preprocess the SD data (detailed in Section 3.2) and describe some notable settings below.

- To ensure that user's mobility and movement trajectory are completely recorded, in the 30-day SD dataset from June 1, 2017 to June 30, 2017, we filter out the users with at least 20 days of SD records and two trajectories per day. And finally 130,612 users are selected with 2,369,267 trajectories entirely.
- The longitude range and latitude range of the Hefei map in the dataset are [117.00°, 117.60°] and [31.60°, 32.00°] respectively. Besides, the number of cell towers are 11,024, with a number of nearly 115 millions SD records per day.
- The spatio-temporal uncertainty of SD seriously affects the extraction of human trajectory. For the semantic integrity, we set a rule that a trajectory must keep a minimum traverse distance of 1 kilometer and last at least 15 minutes.

As for the comparison, we briefly choose a naive embedding method, two state-of-the-art representation learning methods in spatio-temporal data research, and a variant model as baselines. The **Naive** method only use the latitude and longitude information of the tower without other representation learning techniques. Besides, the state-of-the-art representation learning methods include the family of methods of Word2Vec[23] such as **Trembr** and methods designed for specific tasks such as **NeuTraj**. These baselines use different representation learning techniques to implement spatio-temporal data embedding. However, they mostly use limited GPS data as the data source for evaluation, so their effect on large-scale signaling data has not been evaluated. At last, as a variant of our framework, **HERMAS-freq** is designed to evaluate the performance gain of neighbor information extraction. We briefly introduce these baselines respectively.

- **Naive**: In the naive embedding model, we directly use the normalized longitude and latitude of cell tower as the tower embedding. More specifically, the embedding of each tower is a two-dimensional vector. And the trajectory embedding is implemented by the trajectory representation learning approach in Section 3.3.2.
- **Trembr** [13]: It is a representation learning framework designed for use in a variety of trajectory tasks. The basic intuition is to first match the spatio-temporal observations to the nearest road segment through road network matching. Then a representation learning method similar to Word2Vec is performed for the road network in terms of sequence of road segments. Finally, the spatio-temporal trajectory is represented as the sequence of embedded road segments and the embedding of the trajectory is obtained using a recurrent neural network.
- **NeuTraj** [34]: This method is elaborated on trajectory similarity measurement with a representation framework. NeuTraj focuses on accelerating trajectory similarity computation. Specifically, NeuTraj samples seed trajectories linked with the selected one to uses the pair-wise similarities as guidance in approximating that of traditional similarity measures.
- **HERMAS-freq**: In this model, we directly use the **freq**uency of the connection of a given tower with each neighbor tower as the weight of different neighbor towers in the neighbor information extraction module in Section 3.3.1. This model is designed to compare the impact of neighbor tower embedding in HERMAS.

In HERMAS and its variant, we set the dimension of context tower embedding as well as neighbor tower embedding by default as 16, resulting in the dimension of tower embedding to be 32 as default. Further, the dimension of trajectory embedding is set to be 64 as default, which would be discussed in Section 4.2.4 as a significant parameter. And in all baseline methods, we treat each cell tower as a basic spatio-temporal observation in embedding, and obtain optimal parameters through fine-tuning. The dimension of embeddings are set to be equal with that of HERMAS. Some necessary changes have also been adopted to adapt to the requirements of the evaluations.

The optimizer in our framework is Adam optimizer. The learning rate in training phrase is set as 0.001. The experiments are done on a deep learning platform with a 2.10GHz Intel Xeon Silver CPU with 78G as main

memory. After that data preprocessing, we use a NVIDIA TITAN V GPU with 12G memory to accelerate training in deep learning implemented by Pytorch 1.0.1.

4.2 Evaluation on Application I: Trajectory Similarity Measurement

4.2.1 Introduction. Investigating trajectory similarity is a kind of primary research content. A large number of online trajectory applications require instantly management of trajectory data for development. In view of the fact that the distance between trajectories is not naturally measured, a series of traditional trajectory distance calculation methods have been proposed and been referred by researchers, such as **Dynamic Time Wrapping (DTW)**[6], **Hausdorff distance**[2], and the **Fréchet distance**[1]. As is known to all, these methods are based on expert knowledge devoted to tricks like trajectory arrangement, interpolation and matching. However, they have different applicability and involve high time and space complexity, which greatly reduces the efficiency. This incident has inspired massive research aiming at accelerating calculations while ensuring correct similarity calculation results. Therefore, in this evaluation, we compare HERMAS with the baselines in terms of accuracy and efficiency.

4.2.2 Implementation Details. We collectively refer DTW, Hausdorff distance, and Fréchet distance as the **classic methods** here. Some implementation details are listed below.

- For the ground truth of trajectory similarity, we use classic methods to calculate the trajectory similarity among all the trajectories before conducting the following comparison evaluation.
- For the trajectory similarity calculation of HERMAS as well as baselines, a vector embedding is learned for each trajectory respectively. Subsequently, we use the cosine similarity between trajectory embeddings as the trajectory similarity.
- For the evaluation metrics, a line of similarity evaluation metrics has been designed for the evaluation. Hitting rate [34] provides a widely accepted measurement about the accuracy of similarity. Additionally, for the searching efficiency, we derive the sum of the time it takes to calculate the similarity of a given trajectory with respect to all trajectories in the dataset.

Table 1. Comparison of various models on trajectory similarity

Distance	DTW			Hausdorff			Fréchet		
	HR@10	**HR@50**	**R10@50**	**HR@10**	**HR@50**	**R10@50**	**HR@10**	**HR@50**	**R10@50**
Naive	0.269	0.278	0.506	0.257	0.324	0.536	0.289	0.326	0.587
Trembr	0.306	0.394	0.635	0.358	0.450	0.716	0.394	0.480	0.723
NeuTraj	0.388	0.512	0.720	0.398	0.512	0.804	0.616	0.757	0.928
HERMAS-freq	0.414	0.527	0.769	0.383	0.533	0.775	0.645	0.745	0.894
HERMAS	**0.458**	**0.552**	**0.793**	**0.422**	**0.591**	**0.838**	**0.681**	**0.770**	**0.951**

4.2.3 Evaluation on Hitting Rate. The hitting rate means that given a trajectory, whether the $top-k$ similar trajectories retrieved by the model can hit the trajectory with the closest distance to the given a trajectory. We use HR@k to represent the hitting rate (e.g. Recall k within 50 trajectories) of all trajectories in the trajectory set. In this article, we use HR@10 and HR@50 to measure the hit rate[34]. In addition, R10@50 (e.g. Recall 10 within 50 trajectories) is also adopted, which measures the level of the model hitting at least 10 $top-50$ trajectories among the trajectories with the closest distance.

We randomly select 1000 trajectories as a dateset to perform trajectory similarity, the comparison result is shown in Table 1. The overall performance of HERMAS is more prominent than the other three methods based on representation learning techniques. And NeuTraj, as a framework specifically designed for trajectory similarity

calculation, performs better than the other two baselines. However, its performance on DTW is still not as good as HERMAS-freq. We can find that the Naive model has the worst performance, because the node of cell towers lack enough semantic information. Similarly, the performance of Trembr, which follows the idea of Word2Vec, is also unsatisfactory, since the indirect measurement of SD makes it impossible to accurately associated with urban facilities. Compared with NeuTraj, HERMAS has the most obvious performance gain on DTW distance, which is an increase of 7.0%, 4.0% and 7.3% on the three metrics respectively. This is mainly because the DTW distance relates to the realignment between points on the trajectory, and this feature can be well adapted by HERMAS.

4.2.4 Evaluation on Searching Efficiency. We aware that the time cost is an important indicator of the practicality of trajectory similarity measurement. Therefore, we evaluate the efficiency of different models on trajectory similarity search.

The results are converted into how many trajectories the model can search per second to make it easier to understand. While, it is also obvious that the size of the trajectory dataset has a significant impact on the search efficiency, which is also considered. To explore the efficiency of the model in different scale datasets, we reconstructed the datasets in different scales. We randomly selected 1000 and 5000 trajectories to form two datasets $Dataset_{1000}$ and $Dataset_{5000}$.

Table 2. Comparison of model efficiency within different scales of dataset

Model	DTW		Hausdorff		Fréchet	
	$Dataset_{1000}$	$Dataset_{5000}$	$Dataset_{1000}$	$Dataset_{5000}$	$Dataset_{1000}$	$Dataset_{5000}$
Classic Method	24.95	4.52	8.43	1.43	0.58	0.01
Trembr	1525.36	310.65	1534.16	301.27	1489.28	303.85
NeuTraj	54.75	10.52	54.12	10.95	52.20	10.41
HERMAS-freq	1805.55	**367.86**	1835.86	379.25	**1877.74**	361.53
HERMAS	**1870.83**	365.13	**1846.89**	**374.17**	1860.35	**388.07**

It is worth noting that the classic methods are also added to the experiment to explore how much efficiency the proposed framework can make progress. For example, if DTW is used as the method to calculate the distance, then the classic method in the table temporarily represents the DTW method itself. Hausdorff and Fréchet are the same. As can be seen from Table 2, HERMAS and other baselines based on representation learning techniques significantly improve the efficiency than classic methods. In addition, HERMAS has similar performance at the listed distances, proving that it is a very versatile acceleration method. Take DTW as an example for analysis. We compare HERMAS with the classic method and baselines method respectively.

- HERMAS manage the similarity calculation of 1, 870.83 trajectories on average in $Dataset_{1000}$ in one second, and that result is 365.13 in $Dataset_{5000}$. The latter is about 5 times the former, because it is related to the size of the dataset. More practically, the sum of 5000 trajectories is likely to be generated by all the users who passed a street within an hour in the city. HERMAS can be 70 or 80 times faster than DTW. In other words, the number of trajectories processed by HERMAS in one second is more than the number of trajectories processed by classic methods in one minute.
- The primary difference between HERMAS and HERMAS-freq lies in tower embedding, so these models use the same module to implement the trajectory embedding. This leads to similar efficiencies. The implementation of Trembr involves map matching of spatio-temporal observations and thus the model is more time consuming. However, the results of Table 1 show that Trembr and HERMAS are at a significant disadvantage in hitting rate. Besides, the efficiency of NeuTraj is about 30 times slower than HERMAS.

This is because NeuTraj works out the trajectory embedding by sampling auxiliary guide trajectories from the dataset, and therefore bears a very expensive calculation cost.

4.2.5 Impact of Dimension of Tower Embedding. The dimension of the trajectory embedding is an important factor that has a great impact on the experimental results. Here we investigate the effect of different dimensions on evaluation results. We empirically chose 32, 64, 128 and 256 dimensions for the evaluation on HERMAS and other three baselines. As shown in the Fig. 7, the larger the trajectory embedding is, the fewer trajectories can be searched per second for HERMAS. This is because the time spent by the trajectory representation learning module increases with the dimension. We can also find that HERMAS is more efficient than baseline models such as NeuTraj. This was explained in detail in the Section 4.2.4. As shown in the Fig. 8, we choose the HERMAS model to compute the trajectory similarity for different dimension of trajectory embedding separately. From the results, the overall score is highest when the trajectory embedding is 64. This may be because the sparsity of the data also increases when the dimension is high, which leads to the limitation of the model. The human mobility information is not fully embedded when the dimension is not enough, which leads to the lack of representation capability of embedding vectors.

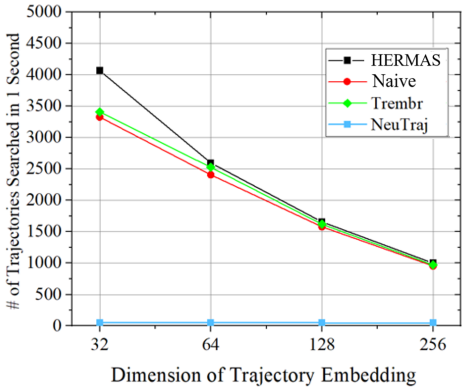

Fig. 7. Impact of Dimension of Trajectory embedding on Similarity Search Efficiency

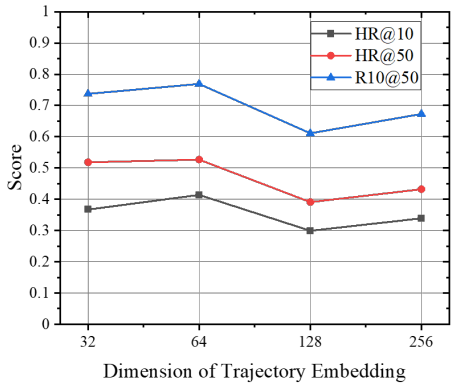

Fig. 8. Impact of Dimension of Trajectory embedding on Trajectory Similarity Search

4.3 Evaluation on Application II: User Profiling

4.3.1 Introduction. Here we make an evaluation in user profiling. Compared with other spatio-temporal data with notable collection cost, SD are provided by cellular provider in a city-wide way without extra efforts. This low-cost method with high penetration rates provide new application scenarios for user profiling. A large number of studies have proved that user mobility is closely related to user profiling, and the bridge between user mobility and SD data is also provided by HERMAS.

We import an additional user profiling dataset, which is provided by one of China's largest voice input software companies. The data comes from the voice input of trip destination collected by a navigation software company cooperating with Hefei CTCC. After trivial preprocessing, we can dig out user profiling information related to user mobility from the destination information. Among them, **transit user** and **traveling user** are two tags designed by us as the ground truth in evaluation. a) **transit user**: According to the latest human mobility survey [16], almost 80% of transit users take public transportation at least 4 times within a week. Based on this, we tag

the users as transit users with this transit pattern from user profiling data, expecting to benefit decision makers in improving the public transportation environment. b) **traveler user**: We use a frequency pattern of at least once a week to filter out users whose navigation destinations are outside the city from user profiling data. Identifying such users helps operators optimize cellular network equipment deployment and tariff package pricing strategies.

4.3.2 Implementation Details. The implementation details of evaluation of user profiling are listed below.

- This part of the evaluation is composed of two independent classification tasks with different labels, i.e. transit user and traveler user. The label of each trajectory is inherited from the label of the user to which it belongs.
- For the classifying issue, **Random Forest(RF)**, **Decision Tree(DTree)** and **XGBoost** are employed as the classifier because these classifiers have been demonstrated their applicability in tasks of classification. And the input is provided by the trajectory embeddings generated by each model. We then fine-tune each classifier of each model separately.
- **Precision**, **recall**, and **F1-score** are adopted as evaluation metrics. For robustness, we randomly split the dataset and perform a 5-fold cross-validation.

Table 3. Performance comparison on user profiling

Model		Transit user			Traveler user		
		Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Naive	**RF**	0.636	0.632	0.634	0.562	0.558	0.560
	DTree	0.591	0.598	0.594	0.534	0.524	0.529
	XGBoost	0.558	0.563	0.560	0.542	0.523	0.532
Trembr	**RF**	0.681	0.687	0.684	0.588	0.606	0.597
	DTree	0.678	0.615	0.645	0.526	0.559	0.542
	XGBoost	0.558	0.582	0.570	0.554	0.561	0.557
NeuTraj	**RF**	0.653	0.667	0.660	0.579	0.586	0.582
	DTree	0.664	0.587	0.623	0.589	0.575	0.582
	XGBoost	0.598	0.596	0.597	0.609	0.622	0.615
HERMAS-freq	**RF**	0.709	0.709	0.709	0.615	**0.635**	0.625
	DTree	0.684	0.635	0.659	0.568	0.593	0.580
	XGBoost	0.614	0.602	0.608	0.579	0.577	0.578
HERMAS	**RF**	**0.715**	**0.726**	**0.720**	**0.628**	0.632	**0.630**
	DTree	0.681	0.668	0.674	0.594	0.623	0.608
	XGBoost	0.618	0.611	0.614	0.624	0.633	0.628

4.3.3 Evaluation on User Profiling. According to the results presented in Table 3, we draw the following observations and make reasonable explanations.

First of all, the naive model is not able to extract deep-level features about human mobility and is therefore the least performance in this validation. Trembr and NeuTraj did not achieve the desired results in two tasks. In addition to the characteristics of indirect measurement of SD, Trembr only focused on the context relationships within the trajectory, ignoring the neighbor information of SD in user mobility. And NeuTraj cannot make progress based on the similarity of trajectories to understand the overall user mobility. In contrast, HERMAS-freq's evaluation performance is relatively improved, because the neghbor information is integrated about users' semantic information and visit frequency.

Secondly, HERMAS outperforms other methods in both tasks. In the task of predicting whether a user is a transit user, HERMAS achieved an precision of 0.715, an recall of 0.726 and a f1-score of 0.720. The results are $0.6\% - 1.7\%$ higher than HERMAS-freq, and $3.3\% - 3.9\%$ higher than other best baseline models. In the task of predicting whether the user is a traveler user, HERMAS achieves the best experimental results in accuracy and F1-score, while the best results in recall are achieved by HERMAS-freq. HERMAS has $1.5\% - 1.9\%$ performance improvement over other best baselines in accuracy and F1-score respectively. A plausible reason is that the tower embeddings of HERMAS highly retains the geographic attributes of the observation, so that the trajectory constituted consequently reflects the character istics of the user's mobility. Furthermore, the mobility information inherited in the large amount of trajectories of users is very efficiently collected and embedded. Benefiting from the excellent functions of each module, HERMAS is evaluated to have excellent application prospects.

4.4 Evaluation on Tower Embedding

In this section, we make an supplementary evaluation of tower embedding. Our intuition is that tower embeddings should maintain as much as possible the geographical relationship of the towers while obtaining contextual and neighborhood information. Given two locations, we measure the *Geographical Distance* between them as the geographical relationship. For simplicity, the Euclidean distance is used by us to calculate the geographical distance between two towers. Similarly, the Euclid distance between the embeddings are figured out as *Embedding Distance*.

Specifically, we randomly selected 100 towers and calculated the geographical distance between them pair by pair. We also compute the Embedding distance between their corresponding embeddings. as shown in the figure, we normalized these two distances separately. It can be seen from the figure that the Embedding distance and the Geographical distance almost show a one-to-one ratio. That is, the closer the points are in geographic space, the closer they are in embedding space. This indicates that our proposed tower embedding module can maintain the geographic connection of spatio-temporal data very well.

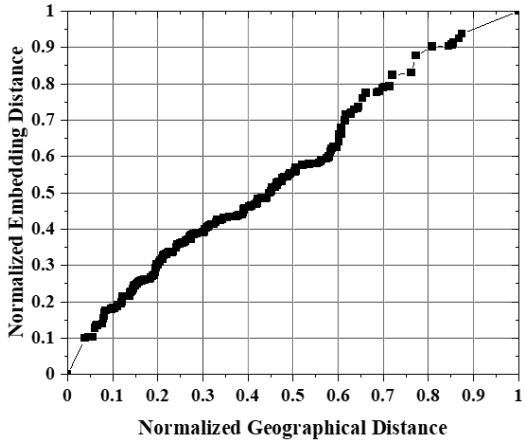

Fig. 9. Evaluation of Tower Embedding

5 DISCUSSION

5.1 Lessons Learned

We summarize our lessons as follows.

- We found that the PPE phenomenon in SD data strains the traditional spatio-temporal data embedding methods that applied to accurate spatio-temporal data like GPS data. Therefore, we propose HERMAS to exploit the large amount of neighbor connection patterns in order to extract the neighbor tower embedding and synthesize it to the tower embedding.
- Our framework outperforms the baselines in the trajectory similarity in both accuracy and efficiency, as shown in Table 1 and Table 2. Although the results of hitting rate on Hausdorff distance are not much better than baselines, great progress has been made in a more efficient fashion. Additionally, we found that HERMAS performs better in user profiling, as indicated in Table 3.
- Some key factors evolved in observation embedding also have been discussed in Section 4.2 with numerical results shown in Fig. 7 and Fig. 8. This suggests that the efficiency of tower embedding is closely related to the dimension of trajectory embedding as well as the size of dataset.

5.2 Generalization

The evaluation results is evaluated by the data collected from a single city in China with high penetration rates. We believe that the designed framework is adaptive to other cities with similar features. Further, the observation embedding module is able to derive any location in the given map via geographical estimation. However, there are certain limitations in the framework. For example, the geographical embedding module could only be trained in a small scale limited by the computational cost. Hence for the dataset with a broad range in longitude and latitude, it might be better to first train within a small scale and then transfer the parameters to the entire scale.

5.3 Privacy and Ethics:

Due to the prosperity of cellular networks, we have the opportunity to cooperate with cellular network operators to conduct research on smart cities with the help of cellular network data. As the part of the cellular service contract, all users involved agreed that the data will be adopted for comprehensive analysis of usage control, malicious cellular usage detection/blocking, access patterns, business opportunities, etc. Particularly, combining cellular data with the user's mobility makes it possible for many applications that are significantly related to the enhancement of user's quality of services. Furthermore, all users' IDs have been hashed into global identifiers by the operator's staff, which cannot be leveraged to trace users or other purposes. We envision the majority of users may not be against this work. As a result, our work is exempted from the institutional IRB process.

6 RELATED WORK

On the one hand, thanks to the high penetration rate, SD data have been widely studied in urban computing, transportation planning and so on. On the other hand, processing large-scale spatio-temporal data in human mobility studies benefits from representation learning for its high efficiency in capturing moving patterns. In this section, we review the related work from two aspects: representation learning for human mobility and SD-based human mobility studies.

6.1 Human mobility applications with SD

The recent development of cellular networks is producing a hot topic in SD, which provides an opportunity to investigate urban-scale human mobility in real-world applications from a novel point of view [7]. Based on the high penetration rate of SD, the researchers explore the inherent characteristics and relationship of human

mobility at a larger spatio-temporal scale. Rose [26] utilize SD to estimate traffic conditions based on treating mobile phones as traffic probes for the pervasiveness in urban areas. Janecek et al. [18] employ SD to predict traffic jam on urban scale by incorporating both active and inactive state of cell phone via a two-stage estimation. Traffic dynamics within cities or parts of the cities are influenced by individual users, which incentivize exploration that dig deeper into user-level. Qin et al. [24] investigated cellular data usage prediction with a behavior-aware scheme called CellPred. They divide users into several groups based on mobility patterns figured out by an additional dataset. Hence CellPred is able to accurately predict user data usage depending on the differences among characteristics of different categories of user. Xu et al. [33] inferred user demographics by a semantic-enhanced urban mobility embedding (SUME). They enhance user's profiling by the semantic information of the area through which the user passes, for the semantic information is deeply influenced by user demographics. Zhao et al. [37] proposed CellTrans to infer users' main transportation modes, which involves user's fine-grained measurement of movement that SD data do not provide. They believe the coarse spatio-temporal uncertainty challenge could be ironed out by well-designed features extracted from the expansion of data of each user. Similarly, we also take advantage of high penetration rates of SD in our framework.

6.2 Representation learning for human mobility

Representation learning yield representations of the data that efficiently extract meaningful information[5] of the data. Because of its widespread use in fields such as natural language processing[11, 22, 23, 30] and graph data mining [15, 19], researchers hope to bring its efficiency to applications related to spatio-temporal data. Some studies attempted to apply typical representing learning algorithms directly to scenarios with spatio-temporal data. Cao et al. [8] designed a representation learning framework called habit2vec to encode trajectory consisted of check-in data. Habit2Vec treats the POI (Point of Interest) category where the user is in as a word, the trajectory as a sentence, and finally uses Word2Vec as an embedding algorithm with a prediction experiment to verify the effectiveness. Gao et al. [14] studied a novel problem of linking trajectories to users with a solution model called TULER. They regard POI category as word trained in sentences generated by POI check-ins on social networks over a period of time to be encoded by Word2Vec. Wu et al. [31] designed two RNN based models with the aim of addressing the limitations of topological structure on trajectory modeling. They use a legal transition set to define costs and training progress, and the model works better than state-of-art methods in location prediction.

6.3 Summary

Human mobility has been extensively studied in recent years. With the widespread applications, using representation learning algorithms to study human movement has gradually become popular. With more or fewer approximations, some methods successfully apply typical representing learning algorithms to human mobility applications. But these models are mainly designed for a single application. It would be fulfilling to leverage representation learning in more applications with its intrinsic applicability [5]. For this goal, Trembr [13] coped with trajectory embedding by encoding both the spatio-temporal features and road segments based on Word2Vec techniques. The model is evaluated by three tasks, which prove its capability in the applications at the trajectory level. Nevertheless, the low penetration rates of fine-grained data, such as GPS, limit these methods to small-scale user and trajectory applications. In this paper, we explore representing SD data with high penetration rates in human mobility related applications.

7 CONCLUSION AND FUTURE WORK

In this paper, we design a novel representation learning framework HERMAS to embed SD data for large-scale human mobility related applications. Aiming at efficiently embedding SD data into high-dimensional vectors while retaining the characteristics of human mobility, we design a representation learning algorithm for the

characteristics of PPE of SD, which distinguish our framework from the previous works. The evaluation is conducted on two prospective applications with a city-wide SD dataset collected by cellular service provider in Hefei, China. The first evaluation on the application of trajectory similarity shows the performance enhancement on the similarity measurement as well as computational efficiency. In particular, HERMAS outperforms the most competitive baseline model for more than 30× efficiency with a performance promotion from 15.2% to 39.2%on similarity measurement. The second evaluation of user profiling also demonstrate the ability of our framework in extracting and representing useful information associated with user mobility.

In the future, we will further investigation on the study for the analysis of the PPE in SD data. Besides, a more extensive research on the human mobility applications is also attractive for our study.

ACKNOWLEDGMENTS

This work is supported partly by the National Key R&D Program of China 2018YFB2100300, 2018YFB0803400, and National Natural Science Foundation of China (NSFC) 61925202, 61772046.

REFERENCES

[1] Helmut Alt and MICHAEL GODAU. 1995. Computing the Fréchet Distance between Two Polygonal Curves. *Int. J. Comput. Geometry Appl.* 5 (1995), 75–91.

[2] S. Atev, G. Miller, and N. P. Papanikolopoulos. 2010. Clustering of Vehicle Trajectories. *IEEE Transactions on Intelligent Transportation Systems* 11, 3 (2010), 647–657.

[3] Rajesh Krishna Balan, Khoa Xuan Nguyen, and Lingxiao Jiang. 2011. Real-Time Trip Information Service for a Large Taxi Fleet. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services (MobiSys '11)*. 99–112.

[4] Jie Bao, Tianfu He, Sijie Ruan, Yanhua Li, and Yu Zheng. 2017. Planning Bike Lanes Based on Sharing-Bikes' Trajectories. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. 1377–1386.

[5] Y. Bengio, A. Courville, and P. Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 8 (2013), 1798–1828.

[6] Byoung-Kee Yi, H. V. Jagadish, and C. Faloutsos. 1998. Efficient retrieval of similar time sequences under time warping. In *Proceedings 14th International Conference on Data Engineering*. 201–208.

[7] Francesco Calabrese, Laura Ferrari, and Vincent D. Blondel. 2014. Urban Sensing Using Mobile Phone Network Data: A Survey of Research. *ACM Comput. Surv.* 47, 2 (2014).

[8] H. Cao, F. Xu, J. Sankaranarayanan, Y. Li, and H. Samet. 2020. Habit2vec: Trajectory Semantic Embedding for Living Pattern Recognition in Population. *IEEE Transactions on Mobile Computing* 19, 5 (2020), 1096–1108.

[9] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent Neural Networks for Multivariate Time Series with Missing Values. *Scientific Reports* 8 (2018).

[10] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. arXiv:arXiv:1406.1078

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:arXiv:1810.04805

[12] Zhihan Fang, Fan Zhang, Ling Yin, and Desheng Zhang. 2018. MultiCell: Urban Population Modeling Based on Multiple Cellphone Networks. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3 (2018).

[13] Tao-Yang Fu and Wang-Chien Lee. 2020. Trembr: Exploring Road Networks for Trajectory Representation Learning. *ACM Trans. Intell. Syst. Technol.* 11, 1 (Feb. 2020), 25.

[14] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Xucheng Luo, and Fengli Zhang. 2017. Identifying Human Mobility via Trajectory Embeddings. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*. 1689–1695.

[15] William L. Hamilton, Rex Ying, and Jure Leskovec. [n. d.]. Inductive Representation Learning on Large Graphs *(NIPS'17)*. 1025–1035.

[16] Hugh M. Clark. 2017. Who Rides Public Transportation. https://www.apta.com/wp-content/uploads/Resources/resources/reportsandpublications/Documents/APTA-Who-Rides-Public-Transportation-2017.pdf.

[17] Tiziano Inzerilli, Anna Maria Vegni, Alessandro Neri, and Roberto Cusani. 2008. A location-based vertical handover algorithm for limitation of the ping-pong effect. In *Networking and Communications, 2008. WIMOB'08. IEEE International Conference on Wireless and Mobile Computing,*. IEEE, 385–389.

[18] A. Janecek, D. Valerio, K. A. Hummel, F. Ricciato, and H. Hlavacs. 2015. The Cellular Network as a Sensor: From Mobile Phone Data to Real-Time Road Traffic Monitoring. *IEEE Transactions on Intelligent Transportation Systems* 16, 5 (2015), 2551–2572.

[19] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. arXiv:arXiv:1609.02907

[20] Quannan Li, Yu Zheng, Xing Xie, Yukun Chen, Wenyu Liu, and Wei-Ying Ma. 2008. Mining User Similarity Based on Location History. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '08)*. 10.

[21] X. Li, K. Zhao, G. Cong, C. S. Jensen, and W. Wei. 2018. Deep Representation Learning for Trajectory Similarity Computation. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. 617–628.

[22] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:arXiv:1301.3781

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. arXiv:arXiv:1310.4546

[24] Zhou Qin, Fang Cao, Yu Yang, Shuai Wang, Yunhuai Liu, Chang Tan, and Desheng Zhang. 2020. CellPred: A Behavior-Aware Scheme for Cellular Data Usage Prediction. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1 (2020).

[25] Zhou Qin, Zhihan Fang, Yunhuai Liu, Chang Tan, Wei Chang, and Desheng Zhang. 2018. EXIMIUS: A Measurement Framework for Explicit and Implicit Urban Traffic Sensing. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems (SenSys '18)*. 1–14.

[26] Geoff Rose. 2006. Mobile Phones as Traffic Probes: Practices, Prospects and Issues. *Transport Reviews* 26, 3 (2006), 275–291.

[27] Han Su, Kai Zheng, Haozhou Wang, Jiamin Huang, and Xiaofang Zhou. 2013. Calibrating Trajectory Data for Similarity-Based Analysis. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13)*. 833–844.

[28] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*. 2440–2448.

[29] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need *(NIPS'17)*. 6000–6010.

[31] Hao Wu, Ziyang Chen, Weiwei Sun, Baihua Zheng, and Wei Wang. 2017. Modeling Trajectories with Recurrent Neural Networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*. 3083–3090.

[32] Ding Xiao, Li Song, Ruijia Wang, Xiaotian Han, Yanan Cai, and Chuan Shi. 2020. Embedding geographic information for anomalous trajectory detection. *World Wide Web* 23, 5 (2020), 2789–2809.

[33] Fengli Xu, Zongyu Lin, Tong Xia, Diansheng Guo, and Yong Li. 2020. SUME: Semantic-Enhanced Urban Mobility Network Embedding for User Demographic Inference. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3 (2020), 25.

[34] D. Yao, G. Cong, C. Zhang, and J. Bi. 2019. Computing Trajectory Similarity in Linear Time: A Generic Seed-Guided Neural Metric Learning Approach. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. 1358–1369.

[35] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. 2010. T-Drive: Driving Directions Based on Taxi Trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '10)*. 99–108.

[36] N. J. Yuan, Y. Zheng, X. Xie, Y. Wang, K. Zheng, and H. Xiong. 2015. Discovering Urban Functional Zones Using Latent Activity Trajectories. *IEEE Transactions on Knowledge and Data Engineering* 27, 3 (2015), 712–725.

[37] Yi Zhao, Xu Wang, Jianbo Li, Desheng Zhang, and Zheng Yang. 2019. CellTrans: Private Car or Public Transportation? Infer Users' Main Transportation Modes at Urban Scale with Cellular Data. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3 (2019).

[38] Yu Zheng. 2015. Trajectory Data Mining: An Overview. *ACM Trans. Intell. Syst. Technol.* 6, 3 (2015), 41.

[39] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban Computing: Concepts, Methodologies, and Applications. *ACM Trans. Intell. Syst. Technol.* 5, 3 (2014).

[40] Yu Zheng, Like Liu, Longhao Wang, and Xing Xie. 2008. Learning Transportation Mode from Raw Gps Data for Geographic Applications on the Web. In *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*. 247–256.