Geoinformatica
https://doi.org/10.1007/s10707-021-00448-9

# How do you visit: Identifying addicts from large-scale transit records via scenario deep embedding

**Canghong Jin[1] · Dongkai Chen[2] · Zhiwei Lin[3] · Zemin Liu[4] · Minghui Wu[1]**

Received: 6 November 2019 / Revised: 16 June 2021 / Accepted: 22 July 2021 /
Published online: 21 August 2021

**Abstract**
Identification of individuals based on transit modes is of great importance in user tracking systems. However, identifying users in real-life studies is not trivial owing to the following challenges: 1) activity data containing both temporal and spatial context are high-order and sparse; 2) traditional two-step classifiers depend on trajectory patterns as input features, which limits accuracy especially in the case of scattered and diverse data; 3) in some cases, there are few positive instances and they are difficult to detect. Therefore, approaches involving statistics-based or trajectory-based features do not work effectively. Deep learning methods also suffer from the problem of how to represent trajectory vectors for user classification. Here, we propose a novel end-to-end scenario-based deep learning method to address these challenges, based on the observation that individuals may visit the same place for different reasons. We first define a scenario using critical places and related trajectories. Next, we embed scenarios via path-based or graph-based approaches using extended embedding techniques. Finally, a two-level convolution neural network is constructed for the classification. Our model is applied to the problem of detection of addicts using transit records directly without feature engineering, based on real-life data collected from mobile devices. Based on constructed scenario with dense trajectories, our model outperforms classical classification approaches, anomaly detection methods, state-of-the-art sequential deep learning models, and graph neural networks. Moreover, we provide statistical analyses and intuitive explanations to interpret the characteristics of resident and addict mobility. Our method could be generalized to other trajectory-related tasks involving scattered and diverse data.

✉ Minghui Wu
mhwu@zucc.edu.cn

Canghong Jin
jinch@zucc.edu.cn

Dongkai Chen
Dongkai.Chen.GR@dartmouth.edu

[1] Zhejiang University City College, 51 Huzhou street, 310015, Hangzhou, China

[2] Dartmouth College Hanover, Hanover, NH 03755, USA

[3] Zhejiang University, 310027, Hangzhou, China

[4] Singapore Management University, 81 Victoria Street, Singapore, 188065, Singapore

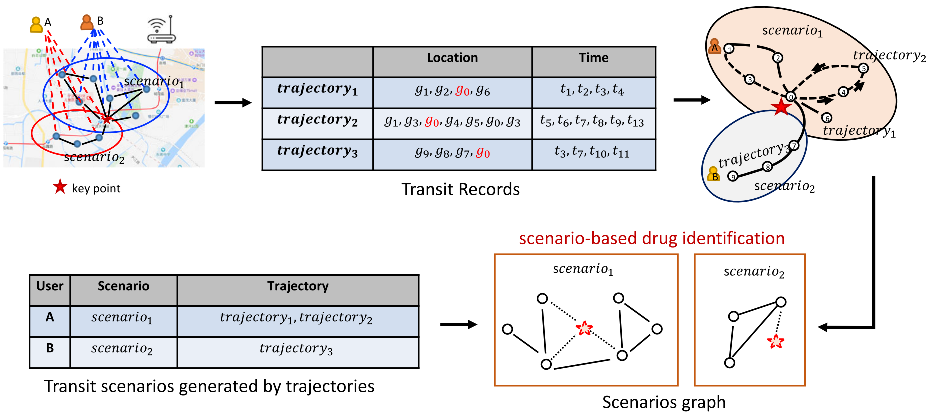
**Keywords** Addict detection · Graph construction · Scenario embedding · Neural network

## 1 Introduction

Drug addiction not only harms individual health but is also linked to criminal and offensive behavior, leading to public safety concerns and social problems. The National Narcotics Control Commission reported that there were more than 2.5 million drug users in China by the end of 2016, with 6.8 percent growth. Worldwide, the total is about 275 million people, representing roughly 5.6 percent of the global population aged 15–64 years, according to the 2018 World Drug Report [1]. Police departments, which have important roles in controlling drug problems, use "strong relationships" detection methods, such as call record analysis, trade record exploration, and human face recognition. In real life, since such strong relationships are rare and difficult to obtain, it is necessary to use relevant "weak" relationships such as transit patterns to solve the detection problem. With the development of mobile and sensor techniques, individual transit pattern analysis is widely used in the field of public security. For instance, the police use Google to find crime suspects by seeking data from mobile phones in target areas, and pickpocket suspects are identified by mining public transit records in Beijing [7]. Here, based on previous studies, we apply spatio-temporal movement records to the addict detection problem, given the following two phenomena. First, it has been reported [47] that most addicts are young and frequently use mobile applications; thus, their transit behaviors could be tracked by mobile devices. Second, according to various drug-related judgments and articles [37], addicts usually have low levels of education and are unemployed, self-employed, or social idlers. Therefore, their activities might be different from those of other residents, which may provide some discriminative features for classification (Fig. 1). Our experiments in real-life data also provide evidence of this phenomenon.

Typical modeling for user identification considers that most people follow a simple and reproducible pattern with respect to movement [25]; nearly 93 percent of personal transit is predictable, according to a study measuring the entropy of individuals' trajectories [32]. Based on this assumption, the current identification task could be addressed using several main approaches. The first approach is based on statistical features, using detection of transport modes (walk, bike, bus, etc.) in the daily activities of different groups of people [30, 44–46]. This enables classification of users with a transport model via machine learning methods. Other approaches focus on mining various patterns in speed, direction, and duration [28], as well as periodic patterns, from a spatial-temporal sequence [4, 5]. After extracting features of movement patterns, classifiers can be leveraged to distinguish users. *TraClass* proposes region-based and trajectory-based clustering methods to improve the accuracy of classification [22]. Deep learning (DL) methods are often applied to trajectory-related problems. For instance, STResNet [42] was designed to forecast the flow of a crowd, and the DeepMove [8] model predicts an individual's next location using a recurrent attention network.

Despite the great benefits of these methods in classifying or predicting human behaviors, they pose some challenges when applied to our problem. 1) The data are imbalanced and the number of target users is tiny. More importantly, we do not know whether a person is a suspect until they have been arrested, which means we cannot make use of any previous survey describing the movement behaviors of particular groups. Thus, statistics-based models [30, 44, 46] will not work well. 2) As movement records collected by WiFi sensors can only give an approximate range, their accuracy in terms of longitude and latitude is



| | Location | Time |
|---|---|---|
| $trajectory_1$ | $g_1, g_2, g_0, g_6$ | $t_1, t_2, t_3, t_4$ |
| $trajectory_2$ | $g_1, g_3, g_0, g_4, g_5, g_0, g_3$ | $t_5, t_6, t_7, t_8, t_9, t_{13}$ |
| $trajectory_3$ | $g_9, g_8, g_7, g_0$ | $t_3, t_7, t_{10}, t_{11}$ |

| User | Scenario | Trajectory |
|---|---|---|
| A | $scenario_1$ | $trajectory_1, trajectory_2$ |
| B | $scenario_2$ | $trajectory_3$ |

**Fig. 1** Motivation and process of scenario embedding model. There are three trajectories going through the key point $O_0$,(e.g., Station) in which trajectory $\mathcal{P}_1 = (O_1, O_2, O_0, O_6)$ and trajectory $\mathcal{P}_2 = (O_1, O_3, O_0, O_4, O_5, O_0, O_3)$ belong to $user A$ and trajectory $\mathcal{P}_3 = (O_9, O_8, O_7, O_0)$ belongs to $user B$. Although $user A$ and $user B$ go through the point $O_0$, their transit intention might be different. $user A$ treats "station" as a passing point while $user B$ treats it as destination. Moreover, the structure of $\mathcal{P}_1$ and $\mathcal{P}_2$ are more complicate than $\mathcal{P}_3$

lower than that of GPS data. Moreover, individuals' movements might be unobserved when they enter places without sensors, where trajectories might be lost. 3) Pre-processing operations such as *TraClass* [22] use region-based and trajectory-based approaches to increase classification accuracy after clustering trajectories in a homogeneous dataset; an example is shown in Fig. 2(a), where the different animals have separate activity regions.[1] However, in our case, most regions are heterogeneous and visited by both addicts and residents (Fig. 2(b)). 4) Suspect identification via public transit records [7] cannot be used to solve our problem, because urban travel in our dataset does not follow a fixed route such as bus or subway. Other DL methods for mobility prediction, such as STResNet [42] and DeepMove [8], attempt to generate features of trajectories, but their goal is to predict the next step, which is not our aim.

We choose a station as a scenario, as in the example in Fig. 3, to demonstrate data diversity. Here, we define the trajectory length as the number of points of the trajectory. In part ($a$) of the figure, the horizontal axis represents the trajectory length of users going through the *plaza*, and the vertical axis indicates the Jaccard similarity between trajectories with various length. Part ($b$) shows the diverse lengths of trajectories. According to these results, even if several people pass the same point, their trajectories vary and their next location is unpredictable.

In order to address the challenges of trajectory sparsity and diversity, in this work we construct a transit scenario that represents user movements rather than trajectories, where the trajectories are dense and points have special meanings. Figure 1 illustrates our motivation and the process of generating the scenario embedding model (SEM).

Regarding addicts' activities, we define two types of scenario: the user's favorite-spot scenario (by visit frequency) and a special-spot scenario (by place type). The former contains more trajectories, whereas the latter involves clearer intentions. One could choose

[1]https://www.fs.fed.us/pnw/starkey/data/tables/

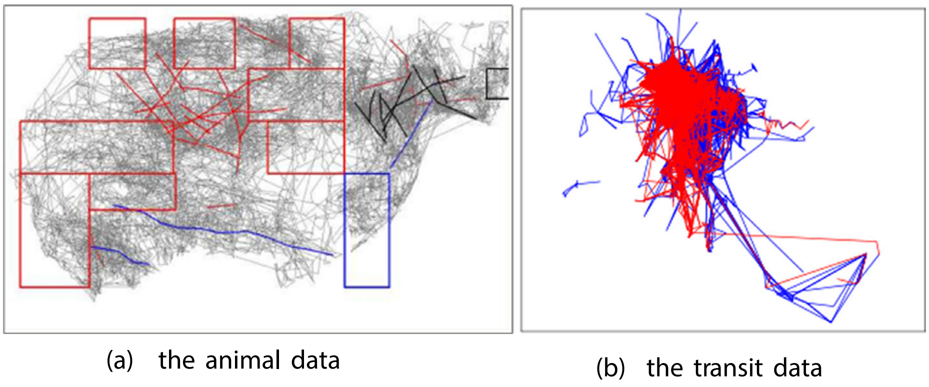


**Fig. 2** Comparison of homogeneity between two trajectory cases. (**a**) Animal movement dataset generated by the Starkey project [22], where red is elk, blue is deer, and black is cattle. (**b**) The urban transit dataset in our study, where red represents addicts and blue represents residents

other points to construct a scenario, depending on the data features and application functions. Moreover, we propose two strategies to demonstrate user intention in scenarios with different granularity: a path-based strategy from a sequential perspective and a graph-based strategy from a structural perspective.

Overall, we define a scenario $\mathcal{S}$ by its key place $\mathcal{K}$ (denoted by a star in Fig. 1) and design a two-level convolutional neural network (CNN) to capture the classification task. This can be used to optimize the model in the training step and avoid human influence.

Our contributions to the literature can be summarized as follows.

- We propose an end-to-end framework containing well-selected scenarios and embed this information to classify individuals based on their long-range and sparse trajectories. Our framework is generalizable to other trajectory classification problems and does not depend on human-selected features.
- We leverage novel embedding methods to represent the latent semantic intention of a scenario using two models: a path-based model and a graph-based model. We also use temporal information to provide joint features for learning. To the best of our

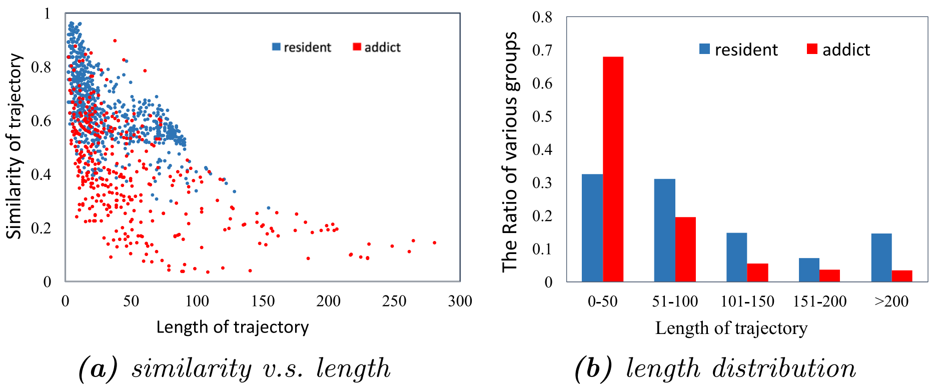


**Fig. 3** Trajectory diversity in scenario *plaza*. (**a**) trajectory *Jaccard* similarity measured in various length. (**b**) trajectory length distribution

knowledge, our framework is the first to identify users by learning their transit behaviors directly.

- We evaluate precision, recall, and F-value on a real-life mobility dataset. Compared with the results of classical classification methods, anomaly detection (AD) methods, state-of-the-art DL models including the sequential long short-term memory (LSTM) model, and graph neural networks including graph convolutional networks (GCN) or graph attention networks (GAT), the evaluation results demonstrate that our model outperforms all baselines.
- Although our model cannot solve the problem completely, that is, we cannot identify addicts by trajectories only, our findings point to a significant improvement in the classification of trajectories. We observe that describing transit behaviors by scenario embedding could partly address the data scattering challenge and could to be applicable to other trajectory problems.

## 2 Preliminaries

### 2.1 User mobility track

We constructed a real-life observational dataset using WiFi sensor equipment installed all over the city, includes MAC address, timestamp and geo-information. By using these transit data and location features, we could generate user track behaviors as Fig. 1 shows. Transit data represents transit track of people in the city and consists of time and location information. Basic location features provide additional semantic of location such as location name, location category, longitude and latitude.[2] Finally, we distinguished users based on a suspect list provided by the police security department. People on the list were treated as positive instances, and other residents were considered as negative instances. Clearly, the number of negative instances was much larger that of positive ones.

### 2.2 Problem formulation

Here, we introduce several basic concepts and provide a formal definition of the scenario-based user identification problem. Detailed descriptions of the notation used in our problem can be found in Table 1.

**Definition 1** A **trajectory** $\mathcal{P}^u = \{O_1, O_2, ..., O_n\}$ consists of a series of points belonging to user $u$, each of which can be represented as $O^u(g, t)$, where $g$ is the position information in terms of latitude and longitude and $t$ is the related transit time. All points in a trajectory are in chronological order, which means that $\forall O_i^u, O_j^u \in \mathcal{P}$, we have $t_i^u < t_j^u$ if $i < j$.

**Definition 2** A **scenario** $\mathcal{S}^u_{\mathcal{K},\langle\mathcal{P}\rangle}$ represents the movement characteristics of a user in a certain place and contains three components: user $u$, key point $\mathcal{K}$, and trajectories through the key point $\langle\mathcal{P}\rangle$. In each scenario, the key point represents the transit semantics and the trajectories describe the transit characteristic.

[2] https://github.com/jincanghong/traj_cls/tree/master

**Table 1** Notation used in our problem

| Notation | Descriptions |
|---|---|
| $u$ | A user $u$ is a resident or a suspect |
| $O^u(g, t)$ | Movement data of user $u$ consists of location $g$ and timestamp $t$ |
| $\mathcal{P}^u(O_i)$ | A trajectory $\mathcal{P}$ belongs to user $u$ and consists of a series of $O_i$ |
| $\mathcal{K}$ | Key points of a user's scenario |
| $\mathcal{S}^u_{(\mathcal{K},\langle\mathcal{P}\rangle)}$ | A user's scenario, with key points $\mathcal{K}$ and a set of trajectories |
| $\mathcal{N}(i, m, n)$ | The previous $m$ points and next $n$ points of index $i$ in a trajectory |
| $\mathcal{T}_\mathcal{K}$ | Location types of key points $k$ |
| $\mathcal{G} = (V, E)$ | Graph $\mathcal{G}$ with set of nodes $V$ and set of edges $E$ |
| $\mathcal{G}^c = (V^c, E^c)$ | Sub graph $\mathcal{G}$ with $c$ layers |
| $\mathcal{F}$ | A function to generate graph $\mathcal{G}$ with trajectories |
| $d$ | Dimensionality of the embedding |
| $\mathcal{Y}_i, \mathcal{Y}_\mathcal{P}, \mathcal{Y}_\mathcal{G}$ | Embedding of a node, trajectory ($\mathcal{P}$), and graph ($\mathcal{G}$) |

Unlike trajectory $\mathcal{P}$, which is represented as a single path, $\mathcal{S}$ contains multiple trajectories with certain semantics. For example, Fig. 1 shows two scenarios for users. Scenario 1 has two trajectories and has a more complex structure than Scenario 2.

**Definition 3** A **graph** is given by $\mathcal{G} = (V, E)$, where $V$ is a set of points and $E$ is a set of edges between the points in $V$. We use $\mathcal{F}(\mathcal{K}, \langle\mathcal{P}\rangle)$ to generate $\mathcal{G}$. For example, in Fig. 1, $user_A$ goes through the station via two trajectories, generating a scenario graph $\mathcal{G}$, where $\mathcal{K}$=station, $\mathcal{P} = \{trajectory_1, trajectory_2\}$, and $u$=$user_A$, details of the generation process will be in Section 3.3.

**Definition 4** A **sub-graph** of $\mathcal{G}^c = (V^c, E^c)$ can be extracted from a scenario graph, where if a node $v \in V^c$ and an edge $e^c \in E^c$, then $v \in V$ and $e \in E$.

In our work, as suggested in [29, 38], we define a "graphlet" as a set of 30 sub-graph structures, as shown in Fig. 4. As a graph $\mathcal{G}$ consists of various graphlets, we use a *complex graphlet first* strategy to select the best one. For instance, the graph in Fig. 4 could be split into $G1$ and $G6$, but we prefer $G17$ because it contains more nodes. If two graphlets have the same number of nodes, we randomly select one of them.

**Definition 5** The **addict identification** problem is defined as judging whether or not a user $u$ is an addict based on their trajectories $\langle\mathcal{P}\rangle$.

In this work, we used an addict list provided by the police department to identify positive instances where at least 3 months of data had been collected before the person's arrest. To identify negative instances, we focused on residents for whom long-range and high-quality trajectory data were available, and we compressed some trajectories using PRESS [33] and TraClass [22].

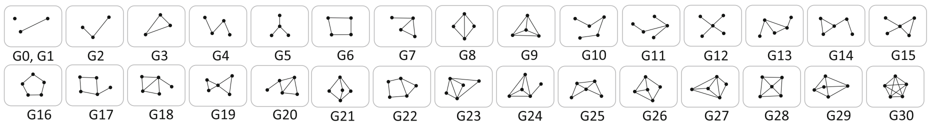


**Fig. 4** Connected, non-isomorphic induced graphlets of size $\leq 5$

## 3 Scenario construction process

In this section, we introduce how to construct our classification framework $S^u_{\mathcal{K},\langle\mathcal{P}\rangle}$ including 1) scenario selection strategies; 2) scenario embedding methods.

### 3.1 Scenario selection strategy

We define a scenario using two components: key points and trajectories. For example, in the case in Fig. 1, a scenario could help us to answer the following questions: *where and when does a user visit a place*? and *how did he/she get there*? That is, besides movement patterns such as velocity, duration, or distance of trajectory, the places visited in a trajectory could also be analyzed to improve our understanding of travel behavior. This could be achieved, for example, by using location-based social network applications [43].

The semantics of trajectories can be defined by one or more special points (key points $\mathcal{K}$). For example, the scenario "*Jack rides to work*" has $\mathcal{T}_{\mathcal{K}}$ *=(home, office)*, and the scenario "*Jennie likes to watch movies in a theater*" has $\mathcal{T}_{\mathcal{K}}$ *= (theater)*. Therefore, in our case study, it is better to design a scenario selection strategy by considering both significant semantic knowledge about trajectories and quality of trajectories. In this paper, we propose two key points selection strategies as a guide, there are also other strategies based on the characteristics of each dataset.

**Special spot strategy** In this strategy, we prefer the semantic meaning of a point to its geographical information. Most points, including *street*, *avenue*, or *crossroads*, do not contain particular or clear semantics, whereas others such as *hospital*, *Karaoke TV,KTV*, or *station* are more meaningful for our target analysis. As the target instances in our study are addicts, based on analysis of criminal records provided by the police, we chose certain points as special spots around which to monitor movement behaviors for modeling.

**Favorite spot strategy** In this strategy, we make use of the movement patterns in users' daily lives by observing the places they visit most frequently. A point is defined as a user's favorite spot if they visit it more frequently than other points. In this way, we can generate preferred places for each user, such as their home, office, or points on the way to work. Trajectories around these points are often more dense than those in other places, as shown in Table 2, and could thus supply more information.

These two strategies focus on different aspects and have different advantages. Special spots contain clear and rich semantics and are the same for all users. Therefore, these points can provide valid information in tasks such as *detecting the movement patterns of all visitors to a railway station*. However, as special points represent only a small proportion of all points, for most users, trajectories involving these points are not sufficient for our analysis. Favorite spots, by contrast, vary among different users, indicating the users' preferences. Sometimes, favorite spots are considered 'normal', i.e., without specific meanings; however, we can use them in scenarios to describe a user's daily life, i.e., *scope of present activity near home or work place*.

**Table 2** Statistics of real life dataset

| Dataset | Addict | Resident | Addict traj. | Resident traj. |
|---|---|---|---|---|
| All Spots | 270 | 38739 | 12391 | 2282630 |
| Special Spots | 154 | 21494 | 4764 | 3919972 |
| Favorite Spots | 179 | 19656 | 8701 | 4066495 |

**Trajectory selection** We consider a scenario with trajectories under two conditions. First, the path should contain at least one key point $\mathcal{K} \in \mathcal{P}$. Second, in order to construct a context graph for a scenario, there should be multiple trajectories passing through the points.

## 3.2 Path-based scenario model

Based on the above definitions, we first propose a path-based scenario model ($SEM_{\mathcal{P}}$) to present semantics in a scenario with multiple paths, as shown in Fig. 5. Here, we define a path as a segment of a trajectory. To model scenarios and also deal with the context of various paths, we have developed a combined path-embedding mechanism.

**Path embedding** We use the scenario shown in Fig. 5 as an example to provide an overview of the process of path embedding. This scenario contains two related trajectories. For each trajectory, two components, location embedding and time embedding, need be embedded.

1) *Location Embedding*. First, we embed points in our dataset in their sequential order in the related trajectories to obtain vector values [11]. Then, a given location-embedding function takes each trajectory in turn as its input and produces a vector space, with each unique point $O_i$ in the space corpus assigned a corresponding vector $v_i$ in an area with dimension $d$.
2) *Time Embedding*. Given a series of timestamps in a trajectory $\mathcal{P}^u$, represented as $(O_1(t), O_2(t), ..., O_n(t))$, we define a position-related method for time embedding. We use $O^u(g, h^{th})$ to represent a user-visited location $g$ at the $h^{th}$ hour in a day. Here, $h^{th}$ is a index of $t$ and describes an activity or behavior at hour $h$, that is, it represents human movements on the corresponding time dimension. A one-hot encoding method is used as follows to map $O(t)$ to an integer value and then to represent the activity time as a binary vector containing all zero values except the index of the integer:

$$t = [0...1...0], O(t) \in h_{th}, h \in [0, 23]. \tag{1}$$

For example, we use $O^u(g, 9_{th})$ to indicate user transit location $O$ during the time period 9:00 to 10:00 am.

As the time sections are ordered in sequence, we add "positional encoding" using the *sine* and *cosine* functions with different indices [35]:

$$TimeEmb_{(pos,2i)} = sin\left(\frac{pos}{1000^{\frac{2i}{d_{model}}}}\right) \tag{2}$$

$$TimeEmb_{(pos,2i+1)} = cos\left(\frac{pos}{1000^{\frac{2i}{d_{model}}}}\right) \tag{3}$$

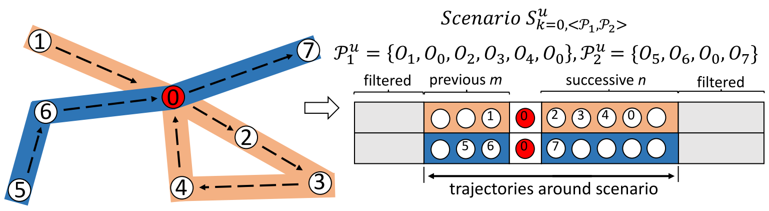

Scenario $S^u_{k=0,<\mathcal{P}_1,\mathcal{P}_2>}$
$\mathcal{P}^u_1 = \{O_1, O_0, O_2, O_3, O_4, O_0\}, \mathcal{P}^u_2 = \{O_5, O_6, O_0, O_7\}$
filtered
previous $m$
successive $n$
filtered
trajectories around scenario
1
2
3
4
5
6
7
0

**Fig. 5** Structure and process for $SEM_{\mathcal{P}}$. The selected path contains a key point $O_0$ and a window of length $[m, n]$ for embedding the scenario

Positional encoding has no a learning process, each word in the given sentence or each position in a trajectory in a trajectory in this case will automatically get a unique embedding representation based on their appearance order in a sequence. This method is used in natural language processing for processing long sequences, e.g., Transformer in [35]. We set the embedding size of positional encoding is 128.

To take the spatial-temporal information of a trajectory into account, we combine these two embedding vectors. The embedding value of $O_i \in \mathcal{P}$ is $X_i = [Y_i, Z_i]$, where $Y_i \in R^d$ is a vector of position in $\mathcal{P}$ and $Z_i \in \mathbb{R}^{d'}$ is a vector of activity time in $\mathcal{P}$; thus $_i \in \mathbb{R}^{d+d'}$. Given a trajectory $\mathcal{P} = \{O_1, O_2, ..., O_M\}$, we define $\mathcal{Y}_{\mathcal{P}} = [X_1, _2, ..., _M]$ and $\mathcal{Y}_{\mathcal{P}} \in \mathbb{R}^{M\times(d+d')}$.

**Scenario embedding** To differentiate path-embedding contributions, we introduce a key point context-aware mechanism to code the environment of a scenario. Taking Fig. 5 as an example, where node 0 is a key point with index i, we set the window [m,n] to extract m previous points and n subsequent points, and we pad them if there are not enough points. In our experiment, we set both $m$ and $n$ as 10. Therefore, for each trajectory $\mathcal{P} \in S^u_{K,<P>}$, if we use $N(i, m, n)$ to represent the context, the result of the embedding is $\mathcal{Y}_{\mathcal{P}} \in \mathbb{R}^{|m+n|\times(d+d')}$. Finally, a scenario with $k$ trajectories can be embedded as $[\mathcal{Y}_{\mathcal{P}_1}, \mathcal{Y}_{\mathcal{P}_2}, ..., \mathcal{Y}_{\mathcal{P}_k}]$. Algorithm 1 shows the whole process of building a scenario with multiple paths.

**Algorithm 1** Path-based model $SEM_{\mathcal{P}}$.

```
Input: Key point set $\mathcal{K}^u$, historical trajectory set $\langle\mathcal{P}^u\rangle$ for user $u$, activity window $m, n$;
Output: $\left[\mathcal{Y}_{\mathcal{P}_1}, \mathcal{Y}_{\mathcal{P}_2}, ..., \mathcal{Y}_{\mathcal{P}_k}\right]$
1:  Initialize: $Empty\ list$ []
2:  for each key point $\mathcal{K}_i \in \mathcal{K}$ do
3:      for each traj. $\mathcal{P}_i \in \langle\mathcal{P}^u\rangle$ do
4:          if $\mathcal{K}_i \in \mathcal{P}_i$ then
5:              idx = index of $\mathcal{K}_i$
6:              $start \leftarrow max(0, idx - m)$
7:              $end \leftarrow len(\mathcal{P}_i - 1, idx + n)$
8:              $\mathcal{Y}_{\mathcal{P}_i} = Concatenate(LocationEmb\ (\mathcal{P}_i[start : end]), TimeEmb\ (\mathcal{P}_i[start : end]))$
9:              $[] \leftarrow \mathcal{Y}_{\mathcal{P}_i}$
10: return: $\left[\mathcal{Y}_{\mathcal{P}_1}, \mathcal{Y}_{\mathcal{P}_2}, ..., \mathcal{Y}_{\mathcal{P}_k}\right]$
```

## 3.3 Graph-based scenario model

Although building scenario on path-view is easy to interpret and simple to implement, its ability of expression is limited because of ignoring the relationship of between different trajectories. In order to describe the activities around the key point, we propose a more complex graph-based model to describe this scenario, in which structures of graphs could be utilized as movement patterns. Our motivation of $SEM_{\mathcal{G}}$ is to represent a daily phenomenon of activities. In $SEM_{\mathcal{G}}$, multiple trajectories in a given scenario could be used for building a graph $\mathcal{G} = (V, E)$ , where a key point $k \in \mathcal{K}$ at the center of graph $\mathcal{G}$, $V$ is the vertex set and $E$ are edges between vertexes.

Briefly, for each user, we conduct a activity scenario based on the user's historical movements. First, all points except the key points are selected to form a candidate point set $CP$ (e.g., $O_8$, $O_{11}$ in Fig. 6). Then, for every pair of points in $CP$, we add an edge between them if they are consecutive points in any historical trajectory. Next, inspired by the influence propagation in social network [39], we introduce the concept of graph layers in our model as follows: given a pair of points $\langle O_k, O_j \rangle$, where $O_k$ is a key point and $O_j \in c - step$ layer if the length of the shortest path from $O_j$ to $O_k$ is $c$, we can generate a graph $\mathcal{G}^c = (V^c, E^c)$ in each layer, where $V^c$ represents the points in the $c - step$ layer and $E^c \in |V^c \times V^c|$. When there are multiple subgraphs in a scenario, there is an inclusive relation between two layers. Thus, for a given $\mathcal{G}^c, \mathcal{G}^{c'} \in S^u_{\mathcal{K},<\mathcal{P}>}$, if $c > c'$, we can infer that $\mathcal{G}^{c'} \subset \mathcal{G}^c$.

To describe the structure of graph $\mathcal{G}^c$, we define several basic graphlets as described in Fig. 4 and as proposed in a previous article [29], where a small graph can be divided into several smaller graphlets which are the basic component of a graph, and each graphlet can keep unique important information respectively because of their edge connectivity, so our goal is now representing a series of graphlets instead of a graph. Graphlets treated as substructures could be embedded by deep graph kernel method, which could build a corpus where the co-occurrence relationship between graphlets is meaningful [39].

Before dividing a subgraph into graphlets, we first need to calculate its modularity defined by equation 4. Modularity is designed to measure the strength of division of a graph into graphlets (modules) in our case, in other words, a graph with high modularity have dense connection between the nodes within modules but sparse connections between nodes in different modules. We then use graph segmentation methods (a community division algorithm [14] or the Depth First Search algorithm) to split $\mathcal{G}^c$ into a set of graphlets $g$. The following step is to generate a unique vector for each graphlet by network embedding methods applied to the graph structures. Here, we use node2vec [13] to embed graphlets and the embedding size of this is 128, more parameters details can be found in Section 5.4. The overall generative process is summarized in Algorithm 2. The graph-based embedding vector merges structure and layer information in a scenario, thereby potentially improving the identification performance of the model.

$$Q = \frac{1}{2m} \sum_{vw} \left[ A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w) \tag{4}$$

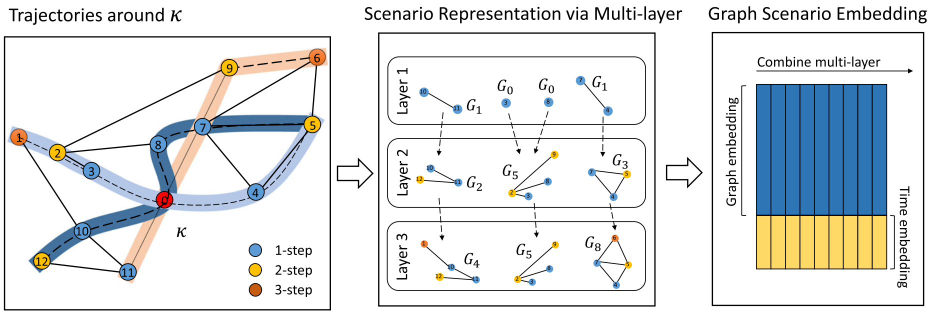


**Fig. 6 Structure of Graph-based Model** $SEM_{\mathcal{G}}$. On the left hand, there are three trajectories with different color going through the key point $O_0$. The points $O_3$, $O_7$, $O_8$, $O_4$, $O_{10}$, $O_{11}$ are in 1-step layer of $O_0$ and generated by $G_0$ and $G_1$. On the middle, we put external points(i.e., $O_2$, $O_9$) into the 1-step layer and regenerate graph $\mathcal{G}$. And on the right hand, scenario is embedded with visited time and graph structures

**Algorithm 2** Graph-based model $SEM_{\mathcal{G}}$.

```
Input: key point set $\mathcal{K}^u$, a historical trajectory set $\{\mathcal{P}^u\}_{i=1}^{number}$ for user $u$
Output: $[\mathcal{Y}_{\mathcal{G}_1}, \mathcal{Y}_{\mathcal{G}_2}, ..., \mathcal{Y}_{\mathcal{G}_n}]$
1: Initialize: Using trajectories $\langle\mathcal{P}^u\rangle$ to construct a graph $G = (V, E)$, layer size $C$, valid
   key points set $K$, empty list [] $graph\ G(V) = \mathcal{F}(\mathcal{P}^u)$
2: for $\mathcal{K}_i \in \mathcal{K}^u$ and $\mathcal{K}_i \in G(V)$ do
3:     $O_k \leftarrow the\ point\ \mathcal{K}_i \in G(V)$
4:     for $c \in [1, C]$ do
5:         for $O_j \in G(V)$ do
6:             if The distance of $(O_k, O_j) = c$ then
7:                 Remove edge of $E_{kj} \in G(E)$
8:                 Add $E_{kj} \rightarrow \mathcal{G}^{k_c}$
9:                 Add $\mathcal{K}_i \rightarrow K$
10: for $k \in K, c \in C$ do
11:     function SUBGRAPH SEGMENTATION($\mathcal{G}^{k_c}$)
12:         Modularity calculation (Section 3.3, Eq. 4)
13:         $Community\ partition$ (Section 3.3)
14:         return a series of graphlets $g$
15:     $\mathcal{Y}_{\mathcal{G}^{k_c}} \leftarrow NetworkEmbedding$(a sequence of graphlets $g$)
16:     [] $\leftarrow \mathcal{Y}_{\mathcal{G}^{k_c}}$
17: return: $[\mathcal{Y}_{\mathcal{G}_1}, \mathcal{Y}_{\mathcal{G}_2}, ..., \mathcal{Y}_{\mathcal{G}_n}]$
```

## 4 Learning process

Having constructing a well-conditioned representation of the scenario as a set of point vectors and the corresponding relative graphlets, in this section, we propose a neural network-based framework to detect addicts from large-scale movement records. Unlike other two-step classification methods, which rely on two separate classifier processes (a trajectory pattern classifier $TC$ and a user classifier $UC$ based on the results of $TC$), our framework uses an end-to-end method to describe user behaviors in structures of multiple granularities (path-based or graph-based) and to optimize models for the identification task.

As shown in Fig. 7, we first collect trajectory records of users and pre-process these records, including data cleaning and trajectory compression, and define special spot and favorite spot scenarios for different groups of users. We use approaches suggested in [18, 27, 34] to embed spatial and temporal features with different granularities. For the path-based model, we obtain an embedding vector corresponding to each point in a path, whereas for the graph-based model, each graphlet extracted from a trajectory is represented as a unique vector. Owing to the differences in the lengths of trajectories, the embedded trajectories from the proposed model form a variable input to the CNN. Thus, we represent the trajectory embedding as a matrix, where each row represents a point or graphlet embedding and the number of columns corresponds to the number of points or graphlets in the trajectory. We reshape this matrix to have a fixed size, splitting it if it is too large or padding it with zeros if it is too small. In our experiments, the input data are in the form of three-dimensional matrices of size $N \times s \times d$, where $N$ is the number of channels after reshaping each matrix, $s$ is the sequence length (50 in our experiment), and $d$ is the sum of the dimensions of the point/graphlet embedding and time embedding. After reshaping the input matrices, we feed

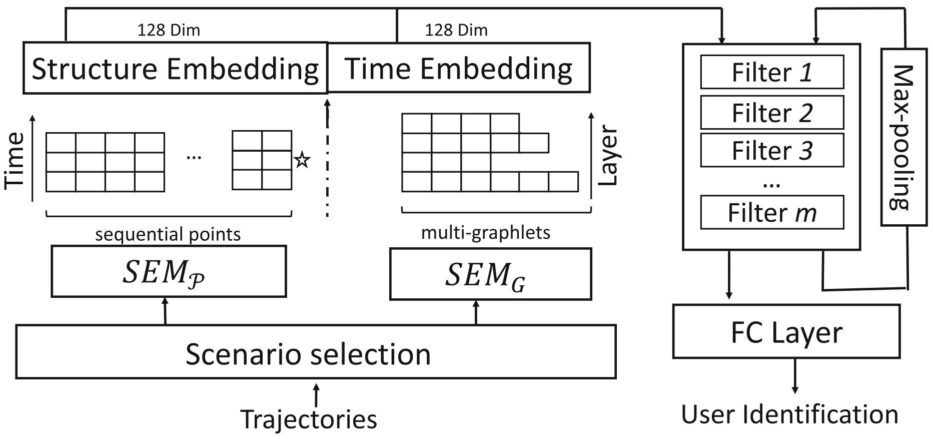


**Fig. 7** The architecture of framework

them into a CNN including two different convolution groups and two max-pooling layers. The first group, $Conv1$, is used to extract features in the embedding dimension, where the filter size is $c_1 \in \mathbb{R}^{o \times r \times d}$, $o$ is the number of output channels, $r$ is a hyperparameter that we have designed, and $d$ is the embedding dimension mentioned above. A feature $f_i$ is calculated from the convolution operation, and we apply the $Max\text{-}Pooling$ operation to extract the maximum value $\hat{f} = max(f)$. The second group, $Conv2$, extracts features through the previous convolution output channel, and applying a $Max\text{-}Pooling$ layer to the channel dimension makes the data channel-free. Finally, a fully connected layer with dropout and softmax output is obtained to identify users.

## 5 Experiment results

In this section, we present the experimental results obtained with our framework. First, we describe our dataset and the experimental environments. Then, two case studies are used to illustrate movement characteristics in different scenarios. Finally, we evaluate our models with respect to precision, recall, and F-score and compared them with several baseline methods.

### 5.1 Dataset and platform

**Dataset** We collect user movement records from sensors all over the city and do preprocessing to choose actual residents in three steps. First, In order to filter out data not generated by mobile phones, we check if the MAC belongs to those android brands according to the top-10 best-selling and most popular phones in China, such as Huawei, OPPO, Vivo, Xiaomi, Meizu, Gionee, Samsung, Letv, and Lephone, which accounted for over 75% market. And then, we select qualified residents, who have relatively more activity track records in our sensor system and have at least two weeks data in a month. Finally, we conduct three datasets for our experiments. 1). the whole dataset; 2). Special Spot dataset is part of all dataset, which only contains users who visit crucial spots. This dataset could describe users' scenarios for some clear semantics; 3). Favorite Spot dataset is generated from users' the most frequency visiting spots. The overview of three datasets are in Table 2, in the two sub-dataset, the number of trajectory of addicts is smaller than the whole dataset while that of the resident is higher, this is because that a single path may contain several special spots or favorite spots and we will separate it to multiple segments, which will finally produce more trajectories.

Table 2 shows that all of three datasets are imbalanced, such as 1:143 examples in the minority class to the majority class in all Spots dataset. According to article [3, 9], we split positive instances and utilize imbalanced-learn Python library[3] to undersample majority class(resident) with setting sampling_strategy value to 0.2 for training. In order to evaluate the effectiveness of our model in imbalanced data, we set the ratio of positive and negative instances as 1:100 for testing.

**Experiment environment** The platform is a Dell server 64-bit system (16 core CPU, each with 2.6GHz, four GPUs GTX 1080ti, 32G main memory). The algorithms and models in our paper were implemented by Python 3.6.

### 5.2 Case study: scenario with special spots in city

Taking the location categories supplied by the police department into account, we choose four scenarios: *hotel*, *park*, *station*, and *plaza*, for two reasons. First, these places were expected to provide sufficient data for analysis. Based on previous statistic, we set threshold of the minimum number of trajectories for constructing a scenario is 16. Second, as these are places that suspects often visit for trading or substance abuse, sensors are already installed there. We can use unsupervised learning methods to detect significant scenarios if we have more data.

In order to investigate the structure of a graph, we visualize the output of graphlet structures for special spot scenarios as in Fig. 8. We first choose the special spots and build the scenarios as described in Section 3, and then we normalize the weights of the number of graphlets based on all the addicts and some of the residents. Finally, we calculate the co-occurrence matrix of the graphlets and illustrate them as a heatmap. The horizontal axis and vertical axis in each square matrix represent both graphlet indices.

Figure 8 shows that simple structure graphlets (e.g., $G_0$, $G_1$, $G_2$) have more weight than complex structure graphlets (e.g., $G_{28}$, $G_{29}$, $G_{30}$) and simple graphlets are easier to combine with other simple or complex graphlets like $G_0$ and $G_1$. To see the dense of the heatmap, we study the difference of scenario complexity of two kinds of users. Among nearly 900 concurrence tuples, we note that residents have richer graph structures than addicts have. Moreover, residents have more simple graphlets than addicts have and have more concurrence graphlets tuples. Thus, we can infer that residents prefer to travel with simpler patterns, but some addicts visiting those special places with more complicated patterns. This phenomenon is consistent with our assumption that some places are attracted to addicts.

### 5.3 Case study: scenario with user history behavior

In this scenario, we first collect users' historical records and extract the most frequent locations as their favorite spots, with all addicts and a subset of residents as our target users. The

[3]https://github.com/scikit-learn-contrib/imbalanced-learn

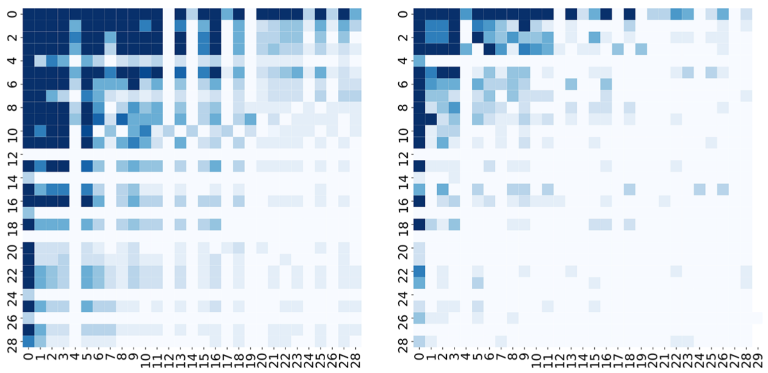

*(a) Residents* *(b) Addicts*

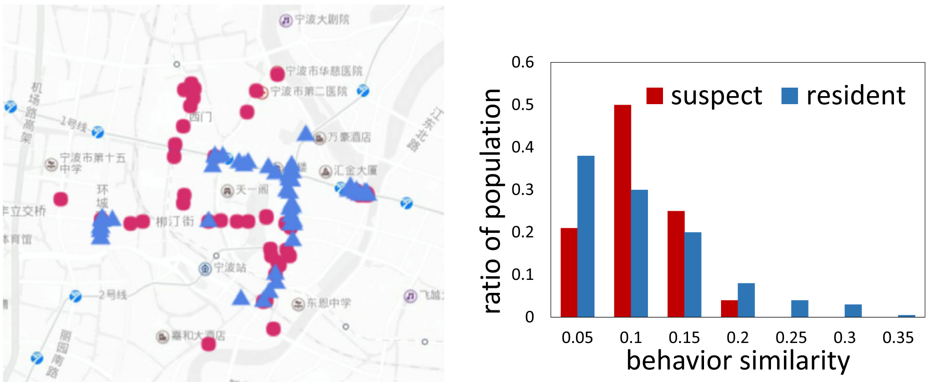
**Fig. 8** The heatmap of graphlet concurrence

positions of the top 50 favorite spots are shown on the map in Fig. 9, where the red spots correspond to addicts and blue spots correspond to residents. We select the top 50 locations in this scenario because the distribution of frequent locations followed a power law, and the trajectories are very sparse for most users when the number of positions is greater than 50. As shown in Fig. 9, the favorite spots of residents are concentrated close to main streets and central business districts (CBD), whereas addicts' favorite spots are scattered all over the city. Therefore, we would obtain entirely different scenarios for the two groups.

Next, we calculate the transit similarity for each group using the *Jaccard* formula. As shown in Fig. 9, from an individual perspective, most addicts or residents have diverse trajectories; on the horizontal axis, which represents individual behavior similarity, the ratio of each interval decreased and soon became zero.

## 5.4 Experimental evaluation

Based on the All Spot and Special Spot scenario which are the same to all users and Favorite Spot scenario which is various for every user, we compare our method with a variety of



*(a) Favorite places* *(b) Transit similarity*

**Fig. 9** Favorite places and behavior similarity of users

competing methods grouped into the three categories: Classification Method, Anomaly Detection Method and Deep Learning Method.

**Feature selection strategy** In order to evaluate traditional classifications for competition, we generate some features from the trajectories of the two groups of users. Here, we do not claim that our strategy is the best while it is a reasonable strategy as evaluated by various experiments.

Our feature generation approach has good usability for the following reasons. First, our dataset containing a total of 2582 sensor points and the timestamp is continuous; our experiments show that these data are too sparse for these dimensions to be treated directly as features. Second, based on *TraClass* [22], we attempt to use region-based and trajectory-based cluster method. We calculate the homogeneous regions, that is, those containing trajectories mostly of the same class. We found that more than 96 percent of points visited by addicts are also visited by residents. Therefore, the region-based method doesn't not suit our data. We therefore use a trajectory-based cluster approach to reduce the number of trajectories in the pre-processing stage. Then, we define the features by considering both trajectory patterns and activity time. As described in previous work by Zheng [28, 30, 45], we generate 12 trajectory-level features: *active scope*, *trajectory duration*, *trajectory length*, *speed*, *velocity change rate*, *head change rate*, *track curve*, *activity pattern*, *minimum speed*, *maximum speed*, *variance of speed*, and *number of large-angle rotations*. Besides the above human-defined features, we also introduce time sections as additional features for each trajectory, where a day of 24 hours is split into 96 sections that represents both hourly and daily preference variance. Overall, we obtain 108 features for classification and tag each trajectory with an addict or resident label.

**Classification method (CM)** We use a two-step framework to perform the identification task. In the first step, classification methods including support vector machine (SVM), naive Bayes (NB), random forest (RF), logistic regression (LR), gradient-boosting decision tree (GBDT) and k-nearest neighbor (KNN) are fitted to the training set using the above features to classify whether a trajectory belonged to a suspect. In the second step, the prediction results for each trajectory of the first step is used to predict user groups by applying the same classification method as in the previous step, similar to a voting mechanism.

**Anomaly detection (AD)** AD methods are unsupervised and find outliers by measuring the deviation of a given data point from its neighbors. In this work, we use one-class SVM (OCSVM) and isolation forest (iForest) to identify addicts, where only negative instances are in the training set. The features of AD methods are the same as those of CF methods.

**Deep learning (DL)** These methods which use the basic idea of human mobility prediction, including DeepMove [8], LSTM [16] and HST-LSTM [20]. LSTM-CNN uses LSTM (long short-term memory) architecture to generate trajectories and time sections as joint features. In order to do justice to the classification part and to demonstrate that our model could extract and learn latent solid representations through graphs, CNN, as in $SEM_{\mathcal{P}}$, is used to classify users. We also use FastText as a baseline which is used by Facebook and is a library for efficient learning of word representations and text classification [17], as one of our baselines to compute vector representations of points (as words) or movement paths (as text) for classification. We feed our data to FastText without time information, because our aim is to examine whether the model would work when solving the problem from the perspective of locations only. We also use some semi-supervised learning learning models on graph-structured data, including GCN (graph convolutional networks) [19], GraphSAGE [15], and GAT (graph attention networks) [36], which can learn graph structures by leveraging node feature information. Although these models show excellent performance in public datasets, they cannot solve the dynamic network problem effectively. Thus, we construct graphs consisting of a person's total trajectories using all points in the dataset, where the size of the adjacency matrix is $2615 \times 2615$.

**Scenario embedding model (SEM)** Our path-based model $SEM_{\mathcal{P}}$ and graph-based model $SEM_{\mathcal{G}}$ are implemented in the special spots and favorite spots datasets. We do not implement them in the all spots dataset as it would not have been possible to form graphs effectively owing to the scattered data.

**Hyperparameters** Python libraries including Pytorch 1.0.1, torch_geometric 1.0.3, and NetworkX 2.2 were used to build our models. The embedding dimensions of each graphlet and position are both 128 with regard to time. The first convolution group of the neural network includes two convolution layers where both of them use $C$ filters, of size $2 \times$ *embedding size* and $3 \times$ *embedding size*, respectively. The second convolution group also includes two convolution layers, one of size $C \times 2 \times C$ and the other of size $C \times 3 \times C$. $C$ in our experiments is set to 100. A max-pooling layer is used to maintain a fixed matrix size despite variable inputs [18]. All of the three graph neural networks we implemented use two convolutional layers: the first layer projects the feature dimension from 80 to 16, and the second layer converts the feature dimension to 1 then we flat the matrix. Finally, a fully connected layer is used to classify if the user is an addict. All other parameters in the three models are set to the default values in torch_geometric.

Afterward, we continue to train the model on the full training data for a fixed number of epochs (e.g., 10, 100 epochs). We optimize the parameters with 10-fold cross-validation by further dividing the training set into 70% for model fitting and 30% for validation. And since the number of positive instances (addicts) is extremely scarce in our experiment, we use under-sample method on negative instances (residents) to balance the data in the training process, the ratio of under-sampling is 0.2.

**Result analysis** The precision, recall, and F-score of classification are measured on a real-life dataset. The results are shown in Table 3. First, the AD methods are less capable than other methods, which means their trajectories are diverse and difficult to cluster. Second, the precision of CF methods based on human-selected features is much lower than that of our $SEM$. This indicates that the movement pattern features used in previous trajectory classification studies are not discriminative in our dataset.

Third, LSTM-CNN performs more worse than CF methods, as the target of the LSTM process is a single trajectory tag, which may lose large amounts of information regarding trajectories during training. Therefore, the output vectors of LSTM are not effective during the CNN classification process. LSTM-CNN can be treated as another two-step method that learns features by modeling instead of by human definition. Furthermore, the performance of FastText indicates that trajectory data cannot not be considered simply as context. Unlike words organized in paragraphs, here, the number of distinct points is relatively small and the composition patterns of points are totally different from natural language patterns. The three graph neural networks perform better than classical CF and AD, as they attempt to learn graph structures and have a more powerful ability to represent nodes; however, they

**Table 3** Performance evaluations with classification categories

| Category | Method | Favorite points | | | Special points | | | All points | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Precision | Recall | F-score | Precision | Recall | F-score | Precision | Recall | F-score |
| CF | RF | 0.032 | 0.675 | 0.062 | 0.020 | 0.758 | 0.039 | 0.028 | 0.786 | 0.054 |
| | NB | 0.016 | 0.723 | 0.031 | 0.016 | 0.812 | 0.032 | 0.017 | 0.768 | 0.032 |
| | SVM | 0.012 | 0.793 | 0.023 | 0.016 | 0.937 | 0.031 | 0.013 | 0.784 | 0.022 |
| | GBDT | 0.037 | 0.681 | 0.071 | 0.027 | 0.684 | 0.052 | 0.030 | 0.802 | 0.058 |
| | LR | 0.056 | 0.708 | 0.103 | 0.033 | 0.708 | 0.103 | 0.012 | 0.684 | 0.023 |
| | KNN | 0.060 | 0.232 | 0.0905 | 0.017 | 0.573 | 0.033 | 0.037 | 0.778 | 0.023 |
| AD | OCSVM | 0.008 | 0.407 | 0.015 | 0.009 | 0.442 | 0.017 | 0.018 | 0.912 | 0.036 |
| | iForest | 0.012 | 0.971 | 0.024 | 0.013 | 0.961 | 0.022 | 0.027 | 0.981 | 0.052 |
| DL | LSTM | 0.018 | 0.742 | 0.035 | 0.031 | 0.851 | 0.060 | 0.018 | 0.915 | 0.035 |
| | LSTM-CNN | 0.024 | 0.956 | 0.047 | 0.034 | 0.963 | 0.067 | 0.021 | 0.892 | 0.041 |
| | FastText | 0.012 | 0.489 | 0.024 | 0.011 | 0.506 | 0.022 | 0.013 | 0.637 | 0.026 |
| | GCN | 0.091 | 0.352 | 0.146 | 0.052 | 0.470 | 0.095 | 0.027 | 0.338 | 0.051 |
| | GraphSAGE | 0.087 | 0.191 | 0.120 | 0.056 | 0.154 | 0.083 | 0.045 | 0.27 | 0.077 |
| | GAT | 0.09 | 0.400 | 0.148 | 0.083 | 0.428 | 0.139 | 0.081 | 0.437 | 0.137 |
| Ours | $SEM_{\mathcal{P}}$ | 0.136 | 0.601 | 0.221 | 0.281 | 0.175 | 0.215 | 0.191 | 0.650 | 0.295 |
| | $SEM_{\mathcal{G},all}$ | 0.034 | 0.689 | 0.065 | 0.085 | 0.350 | 0.137 | – | – | – |
| | $SEM_{\mathcal{G},complex}$ | 0.227 | 0.250 | 0.238 | 0.082 | 0.57 | 0.143 | – | – | – |

perform worse than our models, because our models consider graph structures around key points, locally but precisely, even when the graphs are dynamic.

To demonstrate the effectiveness of our SEMs, we test $SEM_{\mathcal{G}}$ in both of the favorite points and the special points datasets while ignore the all points dataset as it does not contain scenarios. For graph-based model $SEM_{\mathcal{G}}$, as nearly half of the total graphlets are one-node and two-node graphlets, we consider two options: either using all graphlets, denoting $SEM_{\mathcal{G},all}$, or omitting one-node and two-nodes graphlets, denoting $SEM_{\mathcal{G},complex}$. As shown in Table 2, addicts have many more trajectories in favorite spots than in special spots, such that trajectories in favorite spots are denser and more complex. Therefore, $SEM_{\mathcal{G},complex}$ shows the best performance in the favorite spots scenario because it contains more complex structural features, whereas the precision of $SEM_{\mathcal{P}}$ is better than that of $SEM_{\mathcal{G}}$ on the special spots data because the scenarios are the same for all users in this case.

Overall, end-to-end methods have better performance than two-step methods with either human-selected features or learned features. Moreover, the types of scenarios and the density of trajectories affect the effectiveness of the $SEM$. Finally, we do not claim that our $SEM$ is better than all other methods; however, our results demonstrate that it is a more suitable strategy for use in our raw dataset, in which the data are scattered and sparse.

## 6 Related work

Spatial-temporal pattern mining has emerged as an active research field, like urban traffic network analysis, automatic intersection recognition, and movement behavior mining.

In this section, we provide a brief review of the related works, including two categories: movement pattern mining and behavior understanding.

### 6.1 Movement pattern mining

The enormous amount of spatial-temporal data could be used to mine movement pattern. Gong [12] proposes a methodology to detect five travel models (walk, car, bus, subway and commuter rail) from the amount of data generated by GPS in New York. In article [10], Pinelli proposes an extension of the sequential pattern mining paradigm to analyze the trajectories of moving objects. REMO (Relative Motion) [21] method is based on a traditional cartographic approach of comparing snapshots and develops a comparison method based on motion parameters to reveal the movement patterns. Article [23] presents a complete and computationally tractable model for estimating and predicting trajectories based on sparse sampling, anonymous GPS land-marks that called GPS snippets. For example, Chen et al. [6] identifies spatio-temporal patterns from GPS traces of taxis for night bus route planning. Luo et al. [26] tries to reflect the common routing preference of past passengers by finding the most frequent path of a certain period. da Silva et al. [31] discovers and explains movement patterns of a set of moving objects (e.g. track management, bird migration, disease spreading). These previous works give us inspirations for representation of a trajectory, and traditional machine learning approaches proposed in these works have been built baselines for comparison in Section 5.

### 6.2 Behavior interpretation

A number of techniques for understanding user behaviors have also been proposed. For example, article extracts user features from subway transit records and explores abnormal traveling behaviors to discovery the pickpocket suspects [7]. Along the line of location-based anomaly detection, a framework that learns the context of different functional regions in a city is presented, which provides the basis of our feature extraction approach [40]. Traditional trajectory-based similarity calculations use the longest common substring to calculate the similarity of user history trajectories [24]. Abul proposes a W4M (wait for me) method, which uses edit distance to measure the similarity of different paths [2]. Considering the mobility similarity between user group, Zhang et al. [41] proposes GMove modeling method to share significant movement regularity. In recent research, some deep learning methods are applied to encode the trajectory. ST-ResNet [42] is designed to forecast the flow of the crowd. DeepMove [8] model predicts human mobility with recurrent attention network, while HST-LSTM [20] capture location prediction by Spatial-Temporal LSTM. Our work refers some ideas to above mentioned embedding techniques, but unlike classical classification models, our model describes users' behaviors from their movement scenarios and classifies users by convolutional neural network without selecting effective features.

## 7 Discussion

Here, we summarize our findings based on the experimental results and also discuss the differences between the datasets.

1) The behavior records are sparse and scattered. We found that different types of users have different behaviors: residents usually choose to visit main streets and CBD,

whereas trajectories of addicts are scattered irregularly across the city. Moreover, even people in the same group (residents or addicts) showing little similarity in their activity behaviors. The quality of sensor records is not as high as that of GPS data. Hence, the performance of traditional machine learning methods based on points is relatively poor (in Section 5.3).

2) The trajectory patterns are complex and diverse. Trajectories of addicts are shorter and more straightforward than those of ordinary residents. Given a specific scenario, the transit mode preferences of suspects are similar to those of residents. For example, addicts are also likely to go to a station between 8 and 12 am (in Section 5.2).
3) Positive instances are quite limited, and the dataset is very imbalanced. We collect user movement records from WiFi sensors all over the city for 3 months, together with a suspect list provided by local security departments. However, many circumstances may result in loss of tracking of users, such as a user changing their phone, powering off the phone, and leaving the phone at home. Thus, the periodicity is not apparent, in other words, the extraction of trajectory features is difficult (in Section 5.1).
4) The SEM is insufficient to describe user intentions. We assume that people visiting each place with a clear purpose, but in reality we cannot understand an individual's mind based simply on their tracking records. Therefore, in many cases, our model could only have a supplementary role in detecting addicts. We will require stronger connection information such as call, trading, and social network records in order to further develop the model.

## 8 Conclusion and future work

In this paper, we investigate the problem of user identification from a scenario perspective. We propose a framework based on neural networks and scenario embedding using trajectories and graph structures. Extensive experiments shows that our end-to-end model significantly outperformed all the baselines, including classification models and AD models in a real dataset.

There are several possible future directions for our work. First, we only use geographical information and time information to embed user behaviors; other information such as functional region in the city could be used in the future. Second, our current work does not consider the influence of group activities. We detect addicts based only on their individual behavior; however, addicts may prefer to travel with other addicts. We plan to add these features to our model to better describe user movement patterns. Furthermore, our classification framework is generalizable; we plan to apply it to other trajectory-based problem in region function design and public security prediction.

**Acknowledgements** Our research is supported by the Natural Science Foundation of Zhejiang Province of China under Grant (No. LY21F020003), Zhejiang Provincial Key Research and Development Program of China (NO. 2021C01164).

## References

1. World drug report (2019) http://www.unodc.org/doc/wdr2018/WDR_2018_Press_ReleaseENG.PDF, Accessed 1 Feb 2019
2. Abul O, Bonchi F, Nanni M (2010) Anonymization of moving objects databases by clustering and perturbation. Inf Syst 35(8):884–910

3. Branco P, Torgo L, Ribeiro R (2015) A survey of predictive modelling under imbalanced distributions. arXiv:1505.01658
4. Cao H, Mamoulis N, Cheung DW (2007) Discovery of periodic patterns in spatiotemporal sequences. IEEE Trans Knowl Data Eng 19(4):453–467
5. Cao H, Mamoulis N, Cheung DW (2005) Mining frequent spatio-temporal sequential patterns. In: Fifth IEEE international conference on data mining (ICDM'05)
6. Chen C, Zhang D, Zhou Z, Li N, Atmaca T, Li S (2013) B-planner: Night bus route planning using large-scale taxi gps traces. In: 2013 IEEE international conference on pervasive computing and communications (PerCom), pp 225–233
7. Du B, Liu C, Zhou W, Hou Z, Xiong H (2018) Detecting pickpocket suspects from large-scale public transit records. IEEE Trans Knowl Data Eng :1–1
8. Feng J, Li Y, Zhang C, Sun F, Meng F, Guo A, Jin D (2018) Deepmove: Predicting human mobility with attentional recurrent networks. In: WWW '18 international world wide web conferences steering committee, pp 1459–1468
9. Fernández A, García S, Galar M, Prati RC, Krawczyk B, Herrera F (2018) Learning from imbalanced data sets. Springer, New York
10. Giannotti F, Nanni M, Pinelli F, Pedreschi D (2007) Trajectory pattern mining. In: KDD '07. ACM, pp 330–339
11. Goldberg Y, Levy O (2014) Word2vec explained: deriving mikolov others.'s negative-sampling word-embedding method. arXiv:1402.3722
12. Gong H, Chen C, Bialostozky E, Lawson CT (2012) A gps/gis method for travel mode detection in New York city. Comput Environ Urban Syst 36(2):131–139. special Issue: Geoinformatics 2010
13. Grover A, Leskovec J (2016) Node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. KDD '16, Association for Computing Machinery, New York, pp 855–864. https://doi.org/10.1145/2939672.2939754
14. Guangyu Z, Gao K (2015) Research on community division algorithm with directed and weighted network in pervasive sensing environment. In: (SKG'15), pp 105–111
15. Hamilton W, Ying Z, Leskovec J (2017) Inductive representation learning on large graphs. In: Advances in neural information processing systems, pp 1024–1034
16. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
17. Joulin A, Grave E, Bojanowski P, Mikolov T (2017) Bag of tricks for efficient text classification. Association for Computational Linguistics, pp 427–431
18. Kim Y (2014) Convolutional neural networks for sentence classification. arXiv:1408.5882
19. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks
20. Kong D, Wu F (2018) Hst-lstm: A hierarchical spatial-temporal long-short term memory network for location prediction. In: IJCAI'18. AAAI Press, pp 2341–2347
21. Laube P, Imfeld S (2002) Analyzing relative motion within groups oftrackable moving point objects. In: Egenhofer MJ, Mark DM (eds) Science, geographic information. Springer, Berlin, pp 132-144
22. Lee JG, Han J, Li X, Gonzalez H (2008) Traclass: trajectory classification using hierarchical region-based and trajectory-based clustering. PVLDB 1(1):1081–1094
23. Li M, Ahmed A, Smola AJ (2015) Inferring movement trajectories from gps snippets. In: WSDM '15. ACM, pp 325–334
24. Li Q, Zheng Y, Xie X, Chen Y, Liu W, Ma WY (2008) Mining user similarity based on location history. In: GIS '08. ACM, pp 34:1–34:10
25. Lin M, Hsu WJ (2014) Mining gps data for mobility patterns: A survey. Pervasive Mobile Comput 12:1–16
26. Luo W, Tan H, Chen L, Ni LM (2013) Finding time period-based most frequent path in big trajectory data. In: SIGMOD '13. ACM, pp 713–724
27. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv:1301.3781
28. Morency C, Trepanier M, Agard B (2006) Analysing the variability of transit users behaviour with smart card data. In: 2006 IEEE intelligent transportation systems conference, pp 44–49
29. Przulj N (2007) Biological network comparison using graphlet degree distribution. Bioinformatics 23(2):e177–e183
30. Reddy S, Mun M, Burke J, Estrin D, Hansen M, Srivastava M (2010) Using mobile phones to determine transportation modes. ACM Trans Sen Netw 6(2):13:1–13:27
31. da Silva TLC, de Macêdo JAF, Casanova MA (2014) Discovering frequent mobility patterns on moving object data. In: MobiGIS'14. ACM, pp 60–67
32. Song C, Qu Z, Blumm N, Barabási AL (2010) Limits of predictability in human mobility. Science 327(5968):1018–1021
33. Song R, Sun W, Zheng B, Zheng Y (2014) Press: A novel framework of trajectory compression in road networks. Proc VLDB Endow 7(9):661–672

34. Van Brummelen G (2013) Heavenly mathematics: The forgotten art of spherical trigonometry. Princeton University Press, Princeton
35. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. arXiv:1706.03762
36. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2017) Graph attention networks
37. Wang Y, Jiang WW, Zhang D (2017) A study on drug-taking behavior based on big data: Taking guizhou province as an example. Jouranl of Shandong police Colldege
38. Yanardag P, Vishwanathan S (2015) Deep graph kernels. In: KDD '15. ACM, pp 1365–1374
39. Yanardag P, Vishwanathan S (2015) Deep graph kernels. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1365–1374
40. Yuan J, Zheng Y, Xie X (2012) Discovering regions of different functions in a city using human mobility and pois. In: KDD '12. ACM, pp 186–194
41. Zhang C, Zhang K, Yuan Q, Zhang L, Hanratty T, Han J (2016) Gmove: Group-level mobility modeling using geo-tagged social media. In: KDD '16. ACM, pp 1305–1314
42. Zhang J, Zheng Y, Qi D (2016) Deep spatio-temporal residual networks for citywide crowd flows prediction. arXiv:1610.00081
43. Zheng Y (2015) Trajectory data mining: An overview. ACM Trans Intell Syst Technol
44. Zheng Y, Chen Y, Li Q, Xie X, Ma WY (2010) Understanding transportation modes based on gps data for web applications. ACM Trans Web 4(1):1:1–1:36
45. Zheng Y, Li Q, Chen Y, Xie X, Ma WY (2008) Understanding mobility based on gps data. In: UbiComp '08. ACM, pp 312–321
46. Zheng Y, Liu L, Wang L, Xie X (2008) Learning transportation mode from raw gps data for geographic applications on the web. In: WWW '08. ACM, pp 247–256
47. Zhonghua (2005) A longitudinal survey of patterns and prevalence on addictive drug use in general population in five or six areas with high-prevalence in China from 1993 to 2000 Chinese. J Drug Depend

**Canghong Jin** is an associate professor of computer science at Zhejiang University City College. His research focuses on mining and modeling large social and information networks, Spatio-temporal series mining, and Big data platform. Problems he investigated are motivated by large-scale transit records, the web, and online media.

**Dongkai Chen** is an intelligent researcher and engineer in data mining and machine learning. He is now a first-year graduate student at Dartmouth College Computer Science Department. He has a solid foundation in programming who participated in several algorithm contests and won awards. He also led several projects on spatial-temporal data mining and multi-label text classification and those finished projects published to conferences and journals. He is currently a research assistant under the supervision of Professor V.S. Subrahmanian.

**Zhiwei Lin** is a master student at Zhejiang University. His is interested in data mining and its application in real life.

**Zemin Liu** is now a research scientist at Singapore Management University. His is interested in graph embedding, especially semantic proximity search on heterogeneous graphs, generative adversarial networks on graphs and meta-learning. 1). Semantic Proximity Search: we exploit network structures among nodes to represent the semantic proximities among them. In particular, we suggest various structural embedding models to capture their proximity relations. 2). GANs based graph embedding: by generating fake ingredients for graph, we enhance GCNs by employing GANs, and achieve powerful performance. 3). Meta-learning: investigating the usage of meta-learning on graph embeddings.

**Minghui Wu** is a professor of Zhejiang University City College. His key areas of expertise are Mobile Application and Artificial Intelligence. He is interested to make the information easily accessible and useful, in various application domains. His work has been published at AAAI, KDD, WWW, VLDB.