9-2020

# Extract human mobility patterns powered by City Semantic Diagram

Zhangqing SHAN
*Fudan University*

Weiwei SHAN
*Fudan University*

Baihua ZHENG
*Singapore Management University*, bhzheng@smu.edu.sg

Published in IEEE Transactions on Knowledge and Data Engineering, September 2020, Advance online. DOI: 10.1109/TKDE.2020.3026235

# Extract Human Mobility Patterns Powered by City Semantic Diagram

Zhangqing Shan,Weiwei Sun*,and Baihua Zheng

**Abstract**—With widespread deployment of GPS devices, massive spatiotemporal trajectories became more accessible. This booming trend paved the solid data ground for researchers to discover the regularities or patterns of human mobility. However, there are still three challenges in semantic pattern extraction including semantic absence, semantic bias and semantic complexity. In this paper, we invent and apply a novel data structure namely *City Semantic Diagram* to overcome above three challenges. First, our approach resolves semantic absence by exactly identifying semantic behaviours from raw trajectories. Second, the design of semantic purification helps us to detect semantic complexity from human mobility. Third, we avoid semantic bias using objective data source such as ubiquitous GPS trajectories. Comprehensive and massive experiments have been conducted based on real taxi trajectories and points of interest in Shanghai. Compared with existing approaches, *City Semantic Diagram* is able to discover fine-grained semantic patterns effectively and accurately.

**Index Terms**—Human mobility; fine-grained semantic pattern; GPS trajectory; Point of Interest.

## 1 Introduction

The number of GPS-enabled mobile devices is increasing globally. According to Ericsson, there were 5.5 billion smartphone subscriptions in 2019, and this number is expected to reach 7.5 billion by 2025, significantly larger than the number of active fixed line subscriptions worldwide which was reported to be close to 1 billion in 2019. This growth has a big impact in more ways than one. For one, a smartphone, in addition to serve purposes such as keeping in touch with family members, conducting business, and for entertainment, has become a vehicle for data collection. GPS trajectory tracked by smart phones is one example.

Raw GPS trajectories capture how residents move in a city, while there are lots of rich information that could be inferred and learned from the massive raw trajectory data collection. In this paper, we aim at discovering the regularities of human mobility and identifying commuters' daily activity patterns via a data-driven approach, by analyzing a large collection of taxi trajectories collected in Shanghai. To be more specific, raw GPS trajectories tell us when a commuter makes a trip via taxi, together with the source and the destination of the trip. Unfortunately, a raw trajectory does not contain any semantic information so we do not understand why a commuter makes a trip via taxi, and whether the purpose of the trip is common to many commuters.

However, understanding the real purpose of trips and the travel patterns is crucial to many applications and services. First, it allows us to apply business intelligence for improved decision making [1], [2], [3]. For example, patterns such as *Residence* $\rightarrow$ *Shop* or *Residence* $\rightarrow$ *Supermarket* can help estimate the popularity level and the purchasing power around certain commercial center and the demand for commodities and services revealed from patterns is valuable for site selection of new shops. Second, as cities grow and attract more diverse populations, public transport agencies are faced with growing ridership, changing demand patterns and higher commuters' expectations. Understanding the real and regular travel patterns allows agencies to better respond to the changes of urban travel demand [4], [5], [6]. For example, common travel patterns shared by a large number of taxi commuters imply traffic congestion or certain shortages in public transport, which provides valuable input for the authorities to expand or revise the train/bus network. Last but not least, it enables a deeper understanding of commuters' real needs which can help mobile apps to deliver more relevant and more useful services. For example, commuters traveling from *Office* $\rightarrow$ *Shop* might be interested in receiving shopping vouchers and promotion information; commuters traveling from *Office* $\rightarrow$ *Residence* might want to know the fastest route to reach home earlier.

Nevertheless, it is challenging to discover the travel purpose and identify the travel patterns of commuters. There are mainly three challenges, as listed below.

- ***Semantic Absence***. Parts of data from mobile devices do not contain any semantic property. The absence of semantic information restrains semantic analysis on massive raw trajectory datasets. A notable example is raw GPS trajectories which do not contain any semantic tags.
- ***Semantic Bias***. Some commuters do share their locations and activities with others (e.g., check-ins on FourSquare, adding locations to Tweets), which provides one source of semantic information. However, it suffers from topic imbalance and selectivity. For example, FourSquare users might be willing to share their dining

- *Zhangqing Shan and Weiwei Sun are with the Department of Computer Science and Technology, Fudan University, Shanghai, China. Weiwei Sun is the corresponding author. E-mail: {shanzhangqing, wwsun}@fudan.edu.cn*
- *Baihua Zheng is with the School of Information Systems, Singapore Management University. E-mail: bhzheng@smu.edu.sg*

at restaurants but not their visits to doctors.

- *Semantic Complexity*. Many users actually perform multiple behaviours or activities in a single movement, which make several semantic properties co-exist in one location. Take shopping centers as an example. People could visit there for shopping and dining.

To the best of our knowledge, the existing works could address one or two challenges listed above, but not all of them [7]. In this paper, we propose a very novel data structure, namely *City Semantic Diagram (CSD)*, and a system namely Pervasive Miner empowered by CSD as a solution to tackle all three challenges.

To be more specific, instead of relying on the semantic information shared by users, CSD fully utilizes *Point of Interests (POIs)* dataset. Mobile Apps that provide *location-based services (LBSs)* allow users to upload and update the description of locations, e.g., Foursquare [8]. With the help of these *User Generated Contents (UGC)*, the number of POIs is growing rapidly. CSD which is built based on POIs is able to resolve the issue of **semantic absence**. Pervasive Miner takes in GPS trajectories that capture ubiquitous movements of commuters as input and it is able to mine the common travel patterns from GPS trajectories to address **semantic bias** issue by data selection and processing. Last but not least, thanks to the dedicated design of a sub-step namely *semantic purification*, Pervasive Miner is also able to address the challenge of **semantic complexity**.

In summary, we have made mainly three contributions in this paper.

- We design a novel data structure namely *City Semantic Diagram* to organize semantic sources and to support common semantic recognition operations.
- We implement a fine-grained semantic pattern extraction system, namely Pervasive Miner, that is applicable to ubiquitous GPS trajectories.
- We perform a comprehensive experimental study using real taxi trajectory dataset collected in Shanghai to evaluate and compare the performance of Pervasive Miner and its competitors.

The rest of this paper is organized as follows. Section 2 reviews the related works. Section 3 defines the city semantic diagram. Section 4 presents the system Pervasive Miner, including the algorithms and data structure design. Section 5 explains the datasets used in experiments and reports the experimental results, and Section 6 showcases some meaningful discoveries based on real data trajectories to demonstrate the impact and practicality of Pervasive Miner. Finally, we conclude our study in Section 7.

## 2 Related Work

Human mobility pattern extraction is a component in trajectory pattern mining [9], [10]. Nevertheless, *Semantic Absence*, *Semantic Bias* and *Semantic Complexity* limit existing approaches to discover fine-grained semantic patterns effectively and accurately.

Mining patterns in spatiotemporal database cannot handle *Semantic Absence*, because semantic information is absent in raw data without any effective annotation. Pioneer studies on this topic [11], [12] adopt space partitioning strategy to perform the mining task, which divides the whole space into many small grids based on a pre-specified granularity. In [13], the authors define *T-pattern* in a collection of GPS trajectories, and use a sequence of Region-of-Interest (ROI) with temporal intervals to describe human mobility regularity. Here, ROI refers to a collection of nearby grids with high popularity density. In [14], [15], [16], clustering-based strategies are proposed to overcome the limitation of grid-based algorithms. Clustering algorithms discover the clusters with similar sub-sequence. Nevertheless, these approaches only focus on spatiotemporal regularity in raw trajectories, and they cannot support semantic related queries or services.

TABLE 1
The top 10 topics in New York and Tokyo of FourSquare check-in data from 2014 January to 2014 October

| **New York** | **Ratio** | **Tokyo** | **Ratio** |
|---|---|---|---|
| Bar | 7.03% | Train Station | 34.93% |
| Home (private) | 6.8% | Subway | 7.26% |
| Office | 5.60% | Noodle House | 3.01% |
| Subway | 4.11% | Convenience Store | 2.93% |
| Fitness Center | 4.03% | Japanese Restaurant | 2.73% |
| Coffee Shop | 3.30% | Bar | 2.60% |
| Food Drink Shop | 2.90% | Food & Drink Shop | 2.44% |
| Train Station | 2.81% | Electronics Store | 1.89% |
| Park | 2.11% | Mall | 1.88% |
| Neighborhood | 2.02% | Coffee Shop | 1.56% |

Mining patterns in semantic-enriched datasets is usually affected by *Semantic Bias*. To discover the semantic patterns from online social media data, check-in logs or twitter messages are introduced to pattern discovery [17], [18], [19], [20]. *Splitter* introduced in [17] is the representative, which is further improved in [19]. The main enhancement includes the utilization of the user generated semantic tags to find patterns, and a topic trajectory pattern mining algorithm namely *TOPTRAC* [18] using messages from online social media. However, people are willing to share daily life like shopping but not more private activities such as visits to doctors.

The topic selectivity is harmful to the analysis of human mobility. On the one hand, many users only share their mobility and behaviour using online social media in leisure time (not regularly); on the other hand, users are very selective in terms of the types of information they are willing to share. Table 1 lists the top 10 most popular topics in New York and Tokyo respectively, extracted from FourSquare check-in data. As expected, sensitive private topics such as hospital or drug store do not top in either list. As a popular destination, New York receives millions of visitors every year. Tourists usually spend short vacations in New York such as several days. Medical activities are rare for temporary vacations and mostly happen in their living cities. Residents regard medical activities as personal privacy and refuse exposure on social network. This fact shows that check-ins data has bias between different topics. Compared with New York users, the users in Tokyo tend to be more reserved and sensitive because most of them keep their home as a secret in Internet. These phenomena not only limit the volume of available online social media datasets, but also introduce unavoidable topic bias into the semantic-enriched trajectories.

*Semantic Complexity* also challenges most pattern extraction approaches. In order to prevent the *Semantic Bias*, semantic pattern discovery based on raw GPS trajectories is presented in [21]. It extracts semantic property from raw GPS trajectories using the semantic annotation and hot region detection hybrid algorithm before semantic pattern mining. Different from traditional semantic annotation based on database query in [22], [23], such a hybrid algorithm adopts clustering algorithms (*e.g.*, DB-Scan and K-means) to detect hot regions, and then uses semantic background information (*e.g.*, POI database, geographic API) to attach the semantic attribute of each region. Based on the spatial overlapping examination, each stay point of raw trajectories can get semantic description from hot regions. However, these algorithms perform poorly in regions with complex semantics, which leads to the weak consistency in pattern discovery.

To the best of our knowledge, *Splitter* [17] is the most typical approach to discover fine-grained semantic patterns in semantic-enriched trajectories. It retrieves a set of spatially coarse semantic patterns using *PrefixSpan* algorithm [24] from semantic-enriched trajectories, then refines each coarse pattern into fine-grained ones with *Mean Shift* strategy [25]. A modified version of *Splitter* is proposed in [19], namely *SDBSCAN*. The key difference between them lies in the distinct strategies taken to break and cluster coarse patterns after *PrefixSpan* step. *SDBSCAN* renders DB-Scan algorithm rather than top down *Mean Shift* strategy to cluster trajectories with high density to extract patterns. However, both *Splitter* and *SDBSCAN* cannot detect semantic patterns from semantic-absent datasets such as raw GPS trajectories.

In order to make above approaches also applicable to raw GPS trajectories, a ROI based algorithm is proposed in [21] to support semantic recognition. Before pattern mining, it generates semantic property for raw GPS trajectories using a hybrid algorithm that combines semantic annotation and hot region detection. Different from traditional semantic annotation based on database query, such a hybrid approach adopts clustering strategy to detect hot regions, and then uses semantic background information to describe each region. Based on the spatial overlapping examination, each stay point of raw trajectories gets semantic description from hot regions. Although *Semantic Absence* is resolved, this algorithm cannot guarantee semantic consistency for nearby stay points in each hot region because of the uncontrolled purity of semantic property in each hot region.

## 3 Preliminaries

In the following, we formally define some important terms that will be used throughout the paper and then introduce CSD. Table 2 summarizes the list of notations frequently used in the rest of the paper.

**Definition 1. GPS Trajectory.** A GPS trajectory $T = \{(p_1, t_1), (p_2, t_2), \ldots, (p_n, t_n)\}$ captures a journey that passes $n$ GPS points sequentially. Here, $p_i$ in the form of $(x_i, y_i)$ refers to the longitude and the latitude of the $i$-th point at time stamp $t_i$ in the trajectory.

**Definition 2. POI.** A Point of Interest (POI) $p^I$ is defined as a single point with spatial location and semantic property, i.e., $p^I = \{id, p, s\}$. Here, $id$ represents $p^I$'s physical title, $p$ in the form of $(x, y)$ refers to the longitude and the latitude of its location, and $s$ indicates its semantic property using keywords, e.g.,business, hospital or entertainment.

TABLE 2
Table of Important Notations

| Notation | Description |
|---|---|
| $(p, t)$ | a GPS point where $p$ records the location and $t$ is the time stamp |
| $d(p_i, p_j)$ | Haversine distance between points $p_i$ and $p_j$ |
| $\|S\|$ | the size of a set $S$ |
| $Var(S)$ | spatial variance of location distribution among points in set $S$ |
| $Den(S)$ | spatial density among points in set $S$ |
| $T$ | GPS trajectory, $T = \{(p_1, t_1), \ldots, (p_n, t_n)\}$ |
| $s$ | semantic property (the set of semantic tags) |
| $p^I$ | Point of Interest (POI), $p^I = \{id, p, s\}$ |
| $U$ | Fine-Grained Semantic Unit |
| $sp$ | stay point, indicating ($x$, $y$, $t$, $s$), pick-up or drop-off points of passengers in taxi trajectories |
| $ST$ | semantic trajectory |
| $D$ | database of semantic trajectories |
| $ST.sup(D)$ | support, the number of semantic trajectories which contain or reachable contain pattern ST |
| $Group(sp_i)$ | the collection of corresponding neighbor points from contained or reachable contained semantic trajectories (Definition 10) |
| $\|\|p, p'\|\|$ | the Gaussian distribution coefficient between two points (Equation (2)) |
| $pop(POI)$ | the popularity of POI (Equation (3)) |
| $KL(pr_1, pr_2)$ | The Kullback Leibler divergence between two probability distribution |
| $\rho$ | density threshold, the lower bound of the point density in Algorithm 4 |
| $\sigma$ | support threshold, the lower bound of the trajectory number in Algorithm 4 |
| $\delta_t$ | temporal constraint, the upper bound of time interval in semantic trajectory containment and in Algorithm 4 |
| $Pt^k(ST)$ | the $k$th stay point of semantic trajectory $ST$ |

A raw GPS trajectory, as defined in Definition 1, captures a sequence of points passed by a trip. A POI point refers to one geographic entity, e.g., $p^I_1 = \{1, p_1, \textit{restaurant}\}$ and $p^I_2 = \{2, p_2, \textit{bar}\}$ could refer to two POIs near the Bund of Shanghai. In this work, we make full use of the *rich semantic context* that presents in POIs. In the following, we introduce the concept of *fine-grained semantic unit* as a carrier of semantic information corresponding to a small region in a city. Function $\mathsf{SingleSemantic}(P)$ is to check whether all the POI points $p^I \in P$ share the same semantic property; function $\mathsf{range}(p, \epsilon, P)$ returns all the points $p' \in P$ that are within $\epsilon$ distance to point $p$; and Function $\mathsf{Var}(S)$ is to return the spatial variance of points in set $S$, as defined in Equation (1) where $p_c$ refers to the center point, i.e., $p_c = (x, y) = (\frac{1}{|S|}\sum_{\forall p_i \in S} p_i.x, \frac{1}{|S|}\sum_{\forall p_i \in S} p_i.y)$.

$$\mathsf{Var}(S) = \frac{\sum_{\forall p_i \in S}\left((p_i.x - p_c.x)^2 + (p_i.y - p_c.y)^2\right)}{|S| - 1} \tag{1}$$

**Definition 3. Fine-Grained Semantic Unit.** Given a set of POIs $U = \{p^I_1, p^I_2, \ldots\}$, and parameters $N_{min}$, $\epsilon_p$ and $V_{min}$, $U$ is a fine-grained semantic unit iff $\forall p^I_i \in U$, $\exists V_i \subset U$ satisfies following three conditions

i) $|V_i| \geq N_{min}$; ii) $\forall p_j^I \in V_i$, $d(p_i^I, p_j^I) < \epsilon_p$; and iii) $\mathsf{Var}(V_i) \leq V_{min}$ or $\mathsf{SingleSemantic}(V_i)$ returns true.

As stated in Definition 3, a fine-grained semantic unit refers to a small region in a city where all the bounded POIs are highly homogeneous in terms of either spatial location or semantic property. Spatial homogeneity is motivated by the existence of multi-purpose skyscrapers. For example, the Shanghai Tower is a 128-story skyscraper in Shanghai. It has shops and restaurants in a few lower floors, a subway station in the basement, a conference center in its fifth floor, and offices and hotel rooms in many floors. POIs corresponding to the Shanghai Tower are expected to have mixed semantics properties while their locations are very close to each other or even identical. On the other hand, semantic homogeneity is motivated by the fact that venues are more similar to those located nearby, as compared with other far away. It is consistent with our everyday experience that certain districts are designated for certain purposes. For example, Fifth Avenue in New York is a famous shopping area where there are many prestigious boutiques and flagship stores; Lan Kwai Fong in Hong Kong is a small square of streets with over 100 bars and restaurants; and the Bund in Shanghai houses more than 50 buildings of various architectural styles. Based on the concept of fine-grained semantic unit, we propose to organize POIs in a city in the form of *City Semantic Diagram*, as defined in Definition 4.

**Definition 4. City Semantic Diagram.** A set of fine-grained semantic units within a city forms a City Semantic Diagram, donated as $CSD$.

Next, we are going to introduce a few novel concepts that are proposed to facilitate the discovering of the semantic information related to a trajectory $T$. The first concept is *stay point*, as formally defined in Definition 5. Commuters take taxis to complete a journey for certain purposes, while they need to stop at somewhere to perform activities to fulfill the purposes. The intuition behind the stay point is to locate the locations where commuters stop and perform activities, e.g., the pickup point and drop-off point of a taxi journey. As stated in Definition 5, we adopt two conditions to evaluate whether the center point of a sub-trajectory is a stay point. The first condition is to guarantee that a subtrajectory that is corresponding to a stay point spans a time duration not shorter than the minimum requirement $\theta_t$ to provide sufficient time to perform the activity; and the second condition is to guarantee that all the GPS points passed by this subtrajectory are physically close to each other.

**Definition 5. Stay Point.** Given a GPS trajectory $T = \{(p_1, t_1), \ldots, (p_n, t_n)\}$, and two thresholds $\theta_d$ and $\theta_t$, let $T' = \{(p_i, t_i), \ldots, (p_{i+l}, t_{i+l})\}$ with $i \leq 1$ and $i + l \leq n$ be a sub-trajectory. The center point of $T'$ is defined as a stay point $sp = (p, t, s)$ iff i) $t_{i+l} - t_i \geq \theta_t$; and ii) $\forall k \in [1, l]$, $d(p_i, p_{i+k}) \leq \theta_d$ with $p.x = \frac{1}{l+1}\sum_{\forall p \in T'} p.x$, $p.y = \frac{1}{l+1}\sum_{\forall p \in T'} p.y$, $t = \frac{1}{l+1}\sum_{\forall t \in T'} t$, and $s$ refers to the semantic property of this stay point.

Note, given a GPS trajectory $T$, we can locate the stay points but the semantic property of a stay point is unknown. Identifying its semantic property $s$ for a stay point $sp$ is one of the main tasks we need to complete in this paper.

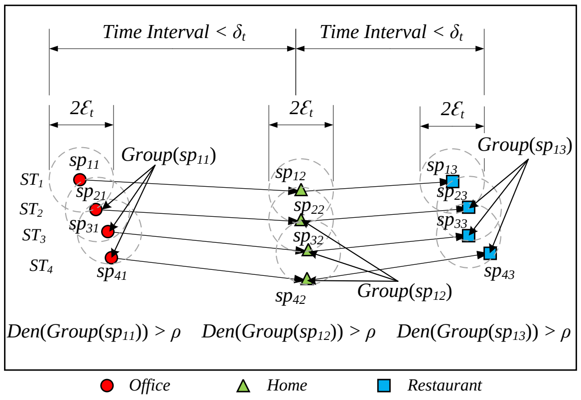


Fig. 1. **Sample fine-grained patterns (*Office* $\rightarrow$ *Home* $\rightarrow$ *Restaurant*), $ST_1$ contains $ST_2$, $ST_2$ contains $ST_3$, $ST_3$ contains $ST_4$, $ST_1$ reachable contains $ST_3$ and $ST_4$, and $ST_2$ reachable contains $ST_4$.**

Based on the concept of stay point, we can convert each GPS trajectory $T$ to a semantic trajectory $ST$ which only involves stay points, as stated in Definition 6. This conversion helps to significantly reduce the size of each trajectory and to effectively facilitate the understanding and identifying of the travel patterns.

**Definition 6. Semantic Trajectory.** Given a GPS trajectory $T$, let $sp_1, sp_2, \cdots$ refer to a sequence of stay points derived from $T$. They form the corresponding semantic trajectory $ST$ for $T$, i.e., $ST = \{sp_1, sp_2, \cdots\}$.

Although semantic trajectory can help effectively reduce the size of each trajectory, the number of trajectories is still huge which increases the complexity of the mining of common patterns from trajectories. Consequently, we introduce the concepts of *containment* and *reachable containment*, as formally presented in Definition 7 and Definition 8 as a solution. The former is to evaluate whether a semantic trajectory $ST'$ has its pattern fully captured by another semantic trajectory $ST$ according to three factors, namely *location proximity*, *temporal similarity*, and *semantic containment*. The latter is to relax the spatial locality condition to recognize temporal similarity and semantic containment among semantic trajectories.

**Definition 7. Containment.** Given two semantic trajectories $ST = \{sp_1, \ldots, sp_m\}$ and $ST' = \{sp'_1, \ldots, sp'_n\}$ with $m \geq n$ and parameters $\delta_t$ and $\epsilon_t$, if $\exists ST'' = \{sp_{i_1}, sp_{i_2}, \cdots, sp_{i_n}\} \subseteq ST$ such that i) $\forall j \in [1, n]$, $d(sp_{i_j}.p, sp'_j.p) \leq \epsilon_t$; ii) $\forall j \in [1, n)$, $|sp_{i_j}.t - sp_{i_{j+1}}.t| \leq \delta_t \wedge |sp'_j.t - sp'_{j+1}.t| \leq \delta_t$; and iii) $\forall j \in [1, n]$, $sp_{i_j}.s \supseteq sp'_j.s$, $ST$ is considered to contain $ST'$, denoted as $ST \Supset ST'$.

For example in Figure 1, $d(sp_{11}.p, sp_{21}.p) \leq \epsilon_t$, $d(sp_{12}.p, sp_{22}.p) \leq \epsilon_t$, $d(sp_{13}.p, sp_{23}.p) \leq \epsilon_t$, $|sp_{11}.t - sp_{12}.t| \leq \delta_t$, $|sp_{12}.t - sp_{13}.t| \leq \delta_t$, $|sp_{21}.t - sp_{22}.t| \leq \delta_t$, $|sp_{22}.t - sp_{23}.t| \leq \delta_t$, $sp_{11}.s = sp_{21}.s = \{\mathit{Office}\}$, $sp_{12}.s = sp_{22}.s = \{\mathit{Home}\}$, $sp_{13}.s = sp_{23}.s = \{\mathit{Restaurant}\}$, thus $ST_1$ contains $ST_2$. Following such conclusion, $ST_2$ contains $ST_3$ and $ST_3$ contains $ST_4$ because their distances between nearby points are lower than $\epsilon_t$, their time intervals between consecutive points are lower than $\delta_t$ and their semantic categories are compatible with each other.

**Definition 8. Reachable Containment.** Given two semantic trajectories $ST$ and $ST'$, $ST$ reachable contains $ST'$ iff

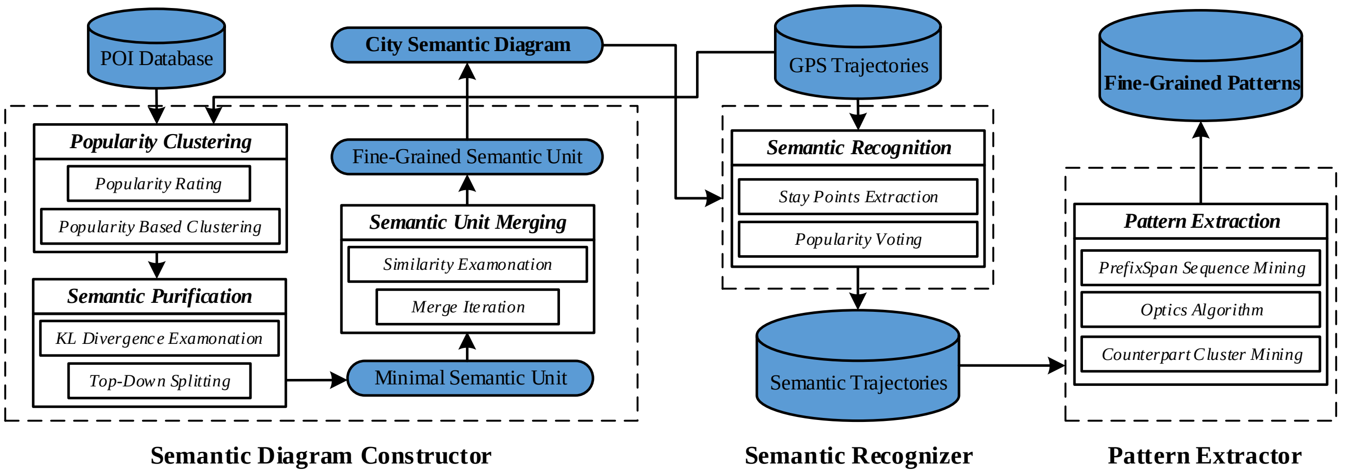


Fig. 2. System architecture of Pervasive Miner

there is a sequence of semantic trajectories $\cup_{i=1}^{j} ST_i$ such that i) $ST$ contains $ST_1$, ii) $ST_j$ contains $ST'$ and iii) $\forall i \in [1, j)$, $ST_i$ contains $ST_{i+1}$, denoted as $ST \Supset ST_1 \wedge ST_1 \Supset ST_2 \wedge \cdots \wedge ST_{j-1} \Supset ST_j \wedge ST_j \Supset ST' \rightarrow ST \sqsupset ST'$.

In Figure 1, $ST_1$ reachable contains $ST_3$ and $ST_4$, and $ST_2$ reachable contains $ST_4$. Based on the concepts of containment and reachable containment, we define a function CP($ST$, $ST'$) with $|ST| \geq |ST'|$ that tries to find the counterpart of a semantic trajectory $ST'$ in another semantic trajectory $ST$. It is a recursive function and the output depends on the containment relationship between $ST$ and $ST'$. There are in total three cases. Case i) $ST \Supset ST'$: it returns a subset $ST'' \subseteq ST$ that enables $ST$ to contain $ST'$, i.e., a subset $ST'' \subseteq ST$ that qualifies for all three conditions stated in Definition 7. Case ii) $ST \sqsupset ST'$: there must be a sequence of semantic trajectories $\{ST_1, \cdots, ST_j\}$ such that $ST \Supset ST_1 \wedge ST_1 \Supset ST_2 \wedge \cdots \wedge ST_{j-1} \Supset ST_j \wedge ST_j \Supset ST'$. Let $ST''_j$ be the output returned by CP($ST_j$, $ST'$). CP($ST$, $ST'$) in this case will return CP($ST$, $ST''_j$). Case iii) $ST$ neither contains nor reachable contains $ST'$: it returns an empty set. It is noted that given two semantic trajectories $ST$ and $ST'$, the output of CP($ST$, $ST'$) is either an empty set or a sequence of stay points $\{sp_1, sp_2, \cdots, sp_n\}$ with $|ST'| = n$.

**Definition 9. Counterpart Function.** CP(ST, ST′) is a function to find the counterpart of a semantic trajectory $ST'$ in another semantic trajectory $ST$. The recursive definition is listed below.

$$\mathrm{CP}(ST, ST') = \begin{cases} ST'' = \{sp_{i_1}, \cdots, sp_{i_n}\} & \text{if } ST \Supset ST' \\ \mathrm{CP}(ST, \mathrm{CP}(ST_j, ST')) & \text{if } ST \sqsupset ST' \\ \emptyset & \text{otherwise} \end{cases}$$

To facilitate the understanding of function CP($ST$, $ST'$), an example is depicted in Figure 1 with four semantic trajectories $ST_1$, $ST_2$, $ST_3$ and $ST_4$. Assume $ST_4 \Supset ST_3$, $ST_3 \Supset ST_2$, and $ST_2 \Supset ST_1$. CP($ST_2$, $ST_1$) returns $\{sp_{21}, sp_{22}, sp_{23}\}$, CP($ST_3$, $ST_1$) returns CP($ST_3$, CP($ST_2$, $ST_1$)) = $\{sp_{31}, sp_{32}, sp_{33}\}$, and CP($ST_4$, $ST_1$) = CP($ST_4$, CP($ST_3$, $ST_1$)) = $\{sp_{41}, sp_{42}, sp_{43}\}$.

Then, we are ready to present next concept, *group*, in Definition 10. Let us revisit our example shown in Figure 1. For semantic trajectory $ST_1$, we have $Group(sp_{11}) = \{sp_{11}, sp_{21}, sp_{31}, sp_{41}\}$, $Group(sp_{12}) = \{sp_{12}, sp_{22}, sp_{32}, sp_{42}\}$, and so on. It can be observed that the concept group is to locate the stay points in different semantic trajectories that are physically close to each other and meanwhile own similar semantic properties.

**Definition 10. Group.** Given a semantic trajectory $ST' = \{sp_1, \cdots, sp_n\}$ and a semantic trajectory database $D_{ST}$, let $D' = \{ST_i \in D_{ST} | ST_i \Supset ST' \vee ST_i \sqsupset ST'\}$ and $S = \cup_{\forall ST_i \in D'} S_i$ with $S_i = \mathrm{CP}(ST_i, ST') = \{sp_{i1}, sp_{i2}, \cdots, sp_{in}\}$. Then, for each stay point $sp_j \in ST'$, all the $j^{th}$ stay points returned by CP($ST_i, ST'$) form its group, denoted as $Group(sp_j) = \cup_{\forall S_i \in S} sp_{ij} \cup \{sp_j\}$.

With the help of group and containment, we present *fine-grained pattern* in Definition 11. If a semantic trajectory can represent multiple semantic trajectories as a common pattern, it has to satisfy the three conditions listed in Definition 11. The main task of our work presented in this paper is to mine the fine-grained patterns from a large set of raw GPS trajectories. All the concepts defined in this section are to formalize and facilitate this task.

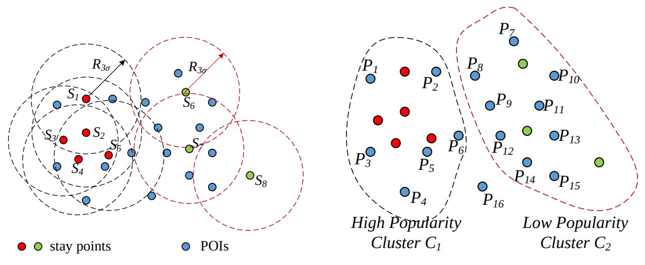
**Definition 11. Fine-grained Pattern.** Given a semantic trajectory $ST = \{sp_1, \ldots, sp_n\}$, a semantic trajectory database $D$, support threshold $\sigma$ and a density threshold $\rho$, if $\exists D_{st} = \{ST_1, \ldots, ST_m\} \subseteq D$ such that i) $\forall ST' \in D_{st}$, $ST'$ contains or reachable contains $ST$, ii) $|D_{st}| \geq \sigma$; and iii) $\frac{1}{n}\sum_{i=1}^{n} Den(Group(sp_i)) \geq \rho$, then $ST$ can be extracted as a fine-grained pattern.

## 4 Pervasive Miner

In this section, we introduce our approach to discover fine-grained patterns, namely Pervasive Miner. It consists of three components, as shown in Figure 2, *Semantic Diagram Constructor*, *Semantic Recognizer* and *Pattern Extractor*, responsible for constructing CSD, recognizing semantic property and extracting patterns respectively.

### 4.1 Semantic Diagram Construction

The constructor of semantic diagram aims at managing and organizing semantic POI datasets via constructing a city semantic diagram. The detailed construction consists



(a) Rating POIs' popularity (b) Clustering POIs

Fig. 3. **Popularity-based clustering**

of three steps, namely *Popularity Based Clustering*, *Semantic Purification* and *Semantic Unit Merging*. To be more specific, it first clusters POIs into coarse clusters based on popularity; next fine-tunes the clusters by considering the semantic property of each POI inside a cluster; and finally merges nearby clusters that share similar semantic property. In the following, we detail these three steps.

**Popularity Based Clustering.** As explained before, stay points capture the locations where commuters perform different tasks such as dining and working to fulfill the purposes of their travels. As POIs close to each stay point could be potential destinations of the corresponding trajectories, we measure the popularity of POIs based on the density of stay points. To be more specific, the popularity of a POI $p^I$, denoted as $pop(p^I)$, is approximated by Gaussian distribution coefficient to nearby stay points, as stated in Equation (3). Note, notation $||p, p'||$ defined in Equation (2) refers to the Gaussian distribution coefficient between two points and notation $D_{sp}$ refers to the complete set of stay points derived from raw trajectories. According to the classical GPS data study [26], the longitude and the latitude from GPS data are often twisted by noises and errors. Researchers usually use the assumption of Gaussian errors to estimate the likelihood distribution of the adjacent region around the real location. Such Gaussian formula avoids the damage from GPS errors and noises. As depicted in Figure 3(a), each stay point $sp$ contributes to the popularity of all the POIs that are located within the circle centered at $sp$ having $R_{3\sigma}$ as the radius.

$$||p, p'|| = \frac{1}{R_{3\sigma}/3 \times \sqrt{2\pi}} \cdot e^{-\frac{d(p,p')^2}{2(R_{3\sigma}/3)^2}} \quad (2)$$

$$pop(p^I) = \sum_{\{sp_i \in D_{sp} | d(sp_i, p^I) < R_{3\sigma}\}} ||p^I.p, sp_i.p|| \quad (3)$$

In our implementation, we adopt *DB-Scan alike algorithm* to perform the clustering, with its pseudo code listed in Algorithm 1. The basic idea is to group nearby POIs with similar popularity (line 5) and similar semantic property (line 6) into clusters. Note not all the POIs in $P$ are clustered into certain clusters, e.g., POI $p_{16}$ in Figure 3(a) is not covered by any cluster. In other words, let $C$ be the set of clusters formed by this step, all the points covered by clusters in $C$ are a subset of $P$. In terms of parameter settings, we set $R_{3\sigma} = 100m$, the vertical overlapping distance threshold $d_v = 15m$, $MinPts_p = 5$, $\epsilon_p = 30m$ and $\alpha = 0.8$ after performing a large number of tests. These values not only achieve an optimal clustering performance (average popularity density of all clusters), but also conform to our daily experience.

**Algorithm 1** Popularity Based Clustering

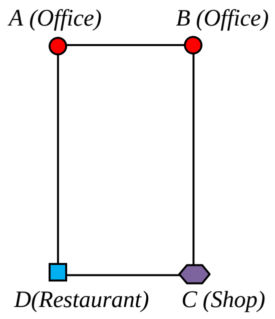
```
Input: support threshold $MinPts_p$, search radius $\epsilon_p$, vertically overlapping distance $d_v$, popularity threshold $\alpha$, trajectory dataset $D_T$ and POI dataset $P$
Output: coarse semantic clusters $C$
1: $C \leftarrow \emptyset$;
2: for each $p^I \in P$ do
3:     $V \leftarrow$ range($p^I, \epsilon_p, P$), $C_l \leftarrow \{p^I\}$, $P \leftarrow P$ - $\{p^I\}$;
4:     for each $p^I_j \in V$ do
5:         if $\frac{pop(p^I_j)}{pop(p^I)} \geq \alpha$ and $\frac{pop(p^I)}{pop(p^I_j)} \geq \alpha$ then
6:             if $d(p^I, p^I_j) \leq d_v$ or $p^I_j.s = p^I.s$ then
7:                 $V \leftarrow V +$ range($p^I_j, \epsilon_p, P$);
8:                 $C_l \leftarrow C_l + \{p^I_j\}$, $P \leftarrow P - \{p^I_j\}$;
9:     if $|C_l| \geq MinPts_p$ then
10:        $C \leftarrow C + \{C_l\}$;
11: return $C$;
```

We also want to highlight that the clustering algorithm mainly packages POIs having similar popularity and semantic property together, but it also groups POIs with different semantic properties but physically very close to each other into one cluster because of condition $d(p^I, p^I_j) \leq d_v$ listed in line 6 of Algorithm 1. This condition is to cater for the multi-purpose skyscrapers where POIs of very different semantic properties are located in one building. Although they might be located at different levels of a skyscraper, their physical locations are very near to each other in the two-dimensional space. Consequently, the clusters generated in this step are coarse in terms of semantic, especially in the case where POIs are close to each other but have different semantic properties. We then perform the next step to further purify the clusters in terms of semantic properties.

**Semantic Purification.** In this step, we progressively decompose each coarse cluster into smaller ones to improve semantic consistency of POIs within one cluster. Our objective is to make sure POIs in each cluster correspond to a fine-grained semantic unit.

In our implementation, we adopt *Kullback Leibler divergence* (notated as $KL$) to check the inner semantic distribution homogeneity in each coarse cluster. In statistics, Kullback Leibler divergenc is a measure of how one probability distribution diverges from another. In our context, let $C$ denote the set of clusters produced by Algorithm 1. For a given cluster $C_o \in C$, $\forall p^I_i, p^I_j \in C_o$, Equation (5) defines their $KL$ value. Note if all POIs in $C_o$ share the same semantic property, the corresponding $KL$ is zero. On the contrary, two POIs with high $KL$ must have very different semantic properties and they suggest the current cluster should be decomposed as it currently contains POIs with different semantic properties. Figure 4 shows an intuitive example of $KL$ computation. Given four POIs located in a coarse cluster, the right chart lists the inner semantic distribution of $A$ and $B$, and the last line lists the $KL$ between $A$ and $B$.

$$\Pr{}_{p^I_i}(s) = \frac{\sum_{p^I_j \in C_o \wedge p^I_j.s = s} ||p^I_j, p^I_i||}{\sum_{p^I_k \in C_o} ||p^I_k, p^I_i||} \quad (4)$$



| | |
|---|---|
| $\Pr_A(office) = \frac{\lVert A,A \rVert + \lVert B,A \rVert}{sum_A}$ | $\Pr_B(office) = \frac{\lVert A,B \rVert + \lVert B,B \rVert}{sum_B}$ |
| $\Pr_A(restaurant) = \frac{\lVert D,A \rVert}{sum_A}$ | $\Pr_B(restaurant) = \frac{\lVert D,B \rVert}{sum_B}$ |
| $\Pr_A(shop) = \frac{\lVert C,A \rVert}{sum_A}$ | $\Pr_B(shop) = \frac{\lVert C,B \rVert}{sum_B}$ |
| $KL(\Pr_A, \Pr_B) = \sum_{s \in \{office, restaurant, shop\}} \Pr_A(s) \log \frac{\Pr_A(s)}{\Pr_B(s)}$ | |

Fig. 4. ***Kullback Leibler divergence* computation (ps,** $sum_A = ||A,A|| + ||B,A|| + ||C,A|| + ||D,A||$ **and** $sum_B = ||B,A|| + ||B,B|| + ||C,B|| + ||D,B||$**)**

**Algorithm 2** Semantic Purification

```
Input: spatial variance threshold $V_{min}$, mixed coarse semantic clusters $C$
Output: minimal semantic unit set $U$
 1: $U \leftarrow \emptyset$;
 2: while $C$ is not empty do
 3:   $C_i \leftarrow$ RandomCluster($C$);
 4:   if SingleSemantic($C_i$) or Var($C_i$) $< V_{min}$ then
 5:     $C \leftarrow C - \{C_i\}; U \leftarrow U + \{C_i\}$;
 6:   else
 7:     $p^I \leftarrow$ CenterPoint($C_i$); $DistList \leftarrow \emptyset$;
 8:     for each $p_k^I \in C_i$ do
 9:       $DistList[k] \leftarrow KL(Pr_{p^I}, Pr_{p_k^I})$;
10:     $Median \leftarrow$ GetMedian($DistList$); $C_{new} \leftarrow \emptyset$;
11:     for each $p_k^I \in C_i$ do
12:       if $DistList[k] > Median$ then
13:         $C_{new} \leftarrow C_{new} + \{p_k^I\}; C_i \leftarrow C_i - \{p_k^I\}$;
14:     $C \leftarrow C + \{C_{new}\}$
15: return $U$;
```

$$KL(\Pr{}_{p_i^I}, \Pr{}_{p_j^I}) = \sum_{s \in \cup_{\forall p^I \in C_o} p^I.s} \Pr{}_{p_i^I}(s) \log \frac{\Pr{}_{p_i^I}(s)}{\Pr{}_{p_j^I}(s)} \quad (5)$$

Algorithm 2 lists the pseudo code of the step of semantic purification. It randomly picks one cluster $C_i \in C$ for evaluation. If $C_i$ qualifies for a fine-grained semantic unit, $C_i$ is removed from $C$ and no further purification is required (lines 4-5). Otherwise, we try to purify the semantic property of POIs in $C_i$ by decomposing the current cluster into two smaller clusters. The decomposition is guided by the measure $KL$. We pick the center POI $p^I \in C_i$ (the point closest to the cluster center) as the reference, and derive the $KL$ values between $p^I$ and another POI $p_k^I \in C_i$ (lines 7-9). Guided by the median $KL$ value, POIs with higher $KL$ values (and hence more different from $p^I$) are removed from $C_i$ and form a new cluster $C_{new}$ which will be reinserted back to $C$ for further purification if required (lines 10-14). On the other hand, the original $C_i$ only keeps those POIs with lower $KL$ values (and hence more similar to $p^I$ in terms of semantic properties). We still keep $C_i$ in $C$ for further purification. This step terminates only when all the clusters in $C$ qualify for fine-grained semantic units. An example is plotted in Figure 5(a), where the right cluster is decomposed into two smaller clusters with more consistent semantic properties.

In brief, the main objective of this step is to detect seman-

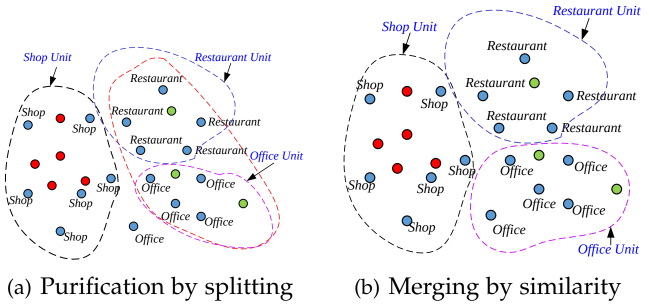


Fig. 5. **Purification and merging**

tic complexity. With the help of *Kullback Leibler divergence*, coarse clusters with POIs having different semantic properties are decomposed into smaller ones for better semantic consistency

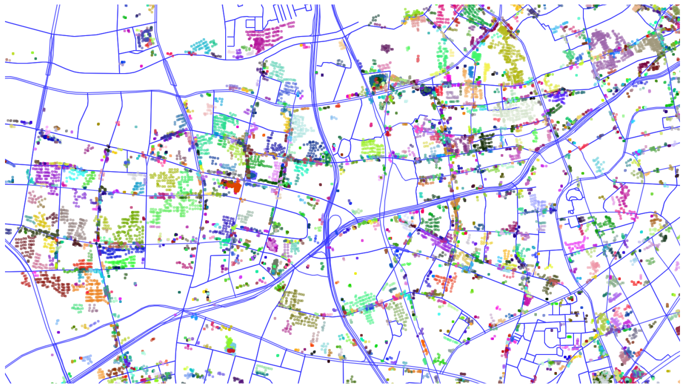
**Semantic Unit Merging.** After previous steps, a collection of minimal semantic units is generated. Because of the randomness and complexity of POI datasets, some parts of a single semantic cluster are sometimes unlinked in previous steps (separated by pedestrian streets or public squares). Thus, the distribution of units might be fragmental. In order to combine the fragments corresponding to similar semantic properties, we adopt cosine similarity examination to check whether two nearby units are mutually semantic consistent, as stated in Equation (8). Different from the Kullback Leibler divergence which specializes in reflecting the inner distribution heterogeneity of a single unit, cosine similarity is generally applied to measure the external similarity between items. Given a set of semantic units $U = \{u_1, u_2, \dots, u_n\}$ (output of Algorithm 2), $u_i \in U$ in the form of $\{p_{i1}^I, p_{i2}^I, \dots, p_{im}^I\}$ is a collection of POIs, and $s$ indicates the semantic property. $\forall u_i, u_j \in U$,

$$\Pr{}_{u_i}(s) = \frac{\sum_{\forall p^I \in u_i \wedge p^I.s = s} pop(p^I)}{\sum_{p^I \in u_i} pop(p^I)} \quad (6)$$

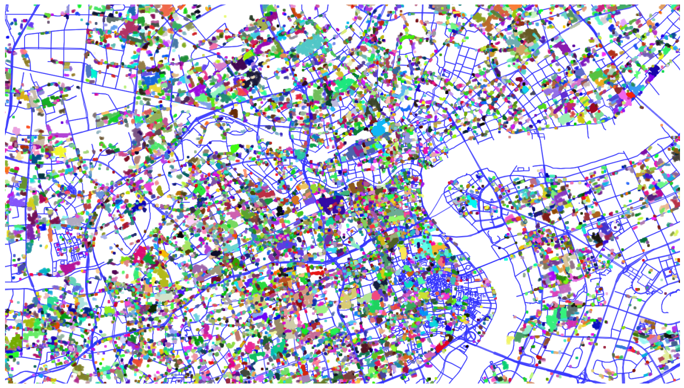
$$Prod(u_i, u_j) = \sum_{s \in \{p^I.s | p^I \in u_i \cup u_j\}} (\Pr{}_{u_i}(s) \cdot \Pr{}_{u_j}(s)) \quad (7)$$

$$Cos(u_i, u_j) = \frac{Prod(u_i, u_j)}{\sqrt{Prod(u_i, u_i) \cdot Prod(u_j, u_j)}} \quad (8)$$

We illustrate the process of this merging step via an example shown in Figure 5(b). For each pair of nearby semantic units, we derive their similarity according to Equation (8). If the similarity exceeds a given threshold (e.g., the



(a) Detail view



(b) Overall view

Fig. 6. ***City Semantic Diagram* in Shanghai**

threshold is set to 0.9 in our experiments after lots of testing), these two semantic units are considered semantically similar and hence are merged into a bigger one. Note that the un-clustered POIs in the first popularity based clustering step (i.e., those left over POIs such as $p_{16}$ in Figure 3(b)) are also considered in this merging step, and they might be merged with other similar units nearby. As shown in Figure 5(b), a single POI whose semantic property is office is merged with the office unit.

After this step, Pervasive Miner extracts many fine-grained semantic units to form *City Semantic Diagram* (CSD). CSD of Shanghai is depicted in Figure 6 for illustration purpose. The colourful regions located in Shanghai's road network are fine-grained semantic units. Each unit owns different color from nearby ones. In the detail view, most units distribute regularly and orderly. Many units share boundary between roads. Some discrete units in same large block separate from each other. In summary, after the step of semantic diagram construction, the semantic properties of POI datasets are managed and organized in CSD properly.

### 4.2 Semantic Recognition

In this step of semantic recognition, Pervasive Miner tries to identify the semantic properties associated with each stay point of a semantic trajectory. As mentioned before, we propose to make full use of the rich semantic information associated with POI dataset to address the challenge of **semantic absence**. In the following, we explain how to recognize the semantic property of a stay point based on CSD.

**Algorithm 3** Semantic Recognition

```
Input: search radius $R_{3\sigma}$, city semantic diagram $CSD$, GPS trajectory dataset $T$
Output: semantic trajectory dataset $ST$
1:  $ST \leftarrow \emptyset$;
2:  for each $T_i \in T$ do
3:      $ST_i \leftarrow$ SemanticTrajectory($T_i$);
4:      for each stay point $sp_j \in ST_i$ do
5:          $P \leftarrow$ range($sp_j.p, R_{3\sigma}, CSD$);
6:          $List_v \leftarrow \emptyset$; $List_s \leftarrow \emptyset$;
7:          for each $p^I \in P$ do
8:              $id \leftarrow$ FindSemanticUnit($p^I, CSD$);
9:              $List_v[id] \leftarrow List_v[id] + pop(p^I) \times ||p^I, sp_j.p||$;
10:             $List_s[id] \leftarrow List_s[id] \cup p^I.s$;
11:         $hv \leftarrow \arg\max_{id}(List_v[id])$; $sp_j.s \leftarrow List_s[hv]$;
12:     $ST \leftarrow ST + \{ST_i\}$;
13: return $ST$;
```

As listed in Algorithm 3, we adopt a voting strategy to find the semantic property for each stay point. To be more specific, given a raw GPS trajectory $T_i$, we first invoke function SemanticTrajectory to identify all the stay points of $T_i$ and to form the corresponding semantic trajectory $ST_i$ with the semantic property of each stay point not yet identified (line 3). Thereafter, we need to identify the semantic property of each stay point (lines 4-11). For each stay point $sp_j \in ST_i$, a range search is executed to locate all the POIs, denoted as $P$, that are within the circular range centered at the stay point $sp_j$ with $R_{3\sigma}$ being the radius. For each located POI $p^I \in P$, its popularity and the distribution coefficient to the stay point $sp_j$ jointly determine the visited possibility of $p^I$ by $sp_j$.

Although we can find the POI with highest visited possibility and assume that POI is able to explain the reason behind the trips stopped at the stay point $sp_j$, Pervasive Miner adopts a different approach. Its voting strategy is based on fine-grained semantic units. To be more specific, it considers all the POIs that are within the circular range and meanwhile belong to the same fine-grained semantic unit as one group. As listed in lines 8-10, we invoke function FindSemanticUnit($p^I, CSD$) to find the fine-grained semantic unit $u_{id}$ that $p^I$ belongs to, and $p^I$'s visited possibility contributes to the weight vote of $u_{id}$ (maintained by $List_v[id]$). In line 11, index $hv$ represents the highest vote unit by searching the candidate unit list $List_v[id]$. Eventually, the unit $U_{hv}$ with the highest weighted vote becomes the best representative of the stay point $sp_j$ in terms of semantic (line 11). Consequently, we assign the semantic property associated with the POIs in the unit $U_{hv}$ that are within the initial circular search range to the stay point $sp_j$. After all the stay points related to semantic trajectory $ST_i$ have been evaluated, we finish the formation of a semantic trajectory (line 12) and we continue the process until all the semantic trajectories have been formed.

Rendering voting strategy between fine-grained semantic units on CSD reduces the semantic ambiguity under GPS noise and errors. For instance, different semantic units distribute on both sides of the river are located closely in Shanghai. Some stay points may locate on the river region because of the GPS noise and errors. If we invoke range search to select the POIs that are located close to a stay

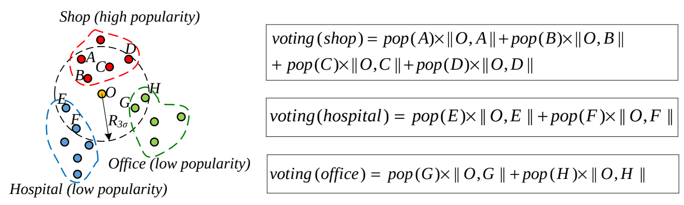

Shop (high popularity)
A
B
C
D
O
E
F
G
H
$R_{3\sigma}$
Office (low popularity)
Hospital (low popularity)
$voting(shop) = pop(A)\times\|O,A\| + pop(B)\times\|O,B\| + pop(C)\times\|O,C\| + pop(D)\times\|O,D\|$
$voting(hospital) = pop(E)\times\|O,E\| + pop(F)\times\|O,F\|$
$voting(office) = pop(G)\times\|O,G\| + pop(H)\times\|O,H\|$

Fig. 7. **Semantic recognition**

point, and find the POI with largest visited probability as a way to determine the semantic property of a stay point, it might suffer from semantic ambiguity as multiple POIs might share similar voting scores or raw GPS locations have errors. In Pervasive Miner, we strategically employ fine grained semantic units to bind POIs into compact clusters to evaluate their integral scores with historical popularity factors. This integral voting strategy enhances the robustness to GPS noise and errors.

An example is depicted in Figure 7 for illustration. Let $O$ be the location of a stay point $sp$ whose semantic information is unknown. In order to determine its semantic property, Pervasive Miner launches a range search centred at $O$ with radius $R_{3\sigma}$. Totally, eight points ($A - H$ in dash circle) from three semantic units are discovered. In order to pick up the semantic unit that is visited by $sp$ with the highest possibility, a voting competition starts. All eight POIs vote for their ownership of semantic unit, with voting power weighted by their popularity and coefficients from Gaussian distribution. The detailed computation is listed in Figure 7. Finally, the semantic unit with the highest vote is selected and the stay point retrieves the semantic property from that unit. In Figure 7, shop unit scores the highest because of its higher popularity, shorter distance to the stay point $sp$, and more selected POIs. Thus, the semantic property of stay point $sp$ is defined as shop.

## 4.3 Pattern Extraction

In *Pattern Extraction*, we use the counterpart conception in Definition 9 to collect the nearby trajectories from coarse semantic patterns. All collected trajectories are divided into several groups described in Definition 10. They are scanned group by group to form the fine-grained semantic patterns.

We first leverage classic *PrefixSpan* algorithm [24] to detect frequent semantic patterns from semantic trajectories. The basic idea of *PrefixSpan* is to use short patterns as prefixes to project the database and to progressively grow the short patterns by searching for local frequent items. Reference [24] describes more details about *PrefixSpan*. The output of *PrefixSpan* is a collection of coarse semantic patterns with same length denoted as $PA_{coarse} = \{pa\}$, where $pa$ is a coarse semantic pattern which contains $n$ semantic trajectories with same length $m$. $\forall pa = \{ST_1, ST_2, \ldots, ST_n\} \in PA_{coarse}$, $\exists O = \{o_1, o_2, \ldots, o_m\}$(as a list of semantic property), let $1 \leq i \leq n$, $1 \leq j \leq m$, we can make sure that $sp_{ij} = Pt^j(ST_i)$(as the $j$th point of $ST_i$) and $sp_{ij}.s = o_j$.

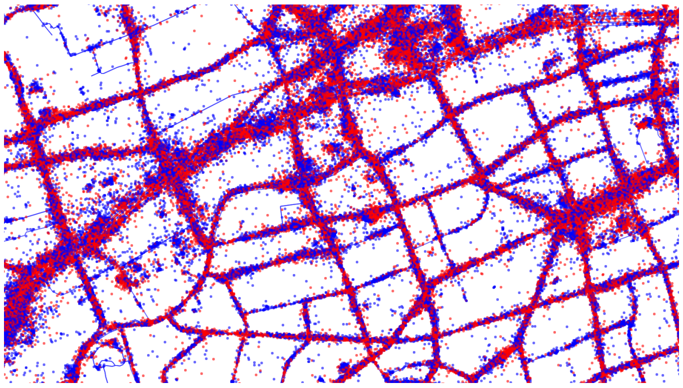
For each coarse pattern detected by *PrefixSpan*, Pervasive Miner renders the *CounterpartCluster* algorithm to detect its spatio-temporal regularity. Algorithm 4 describes the details of *CounterpartCluster*.

**Algorithm 4** CounterpartCluster

```
Input:
    semantic trajectory set $D$, support threshold $\sigma$, temporal
    constraint $\delta_t$ and density threshold $\rho$
Output:
    collection fine-grained patterns $PA_{finegrained}$
 1: $PA_{finegrained} \leftarrow \emptyset$;
 2: $PA_{coarse} \leftarrow PrefixSpan(D)$;
 3: for each $pa \in PA_{coarse}$ do
 4:     $m \leftarrow$ the length of each trajectories in $pa$;
 5:     for $1 \leq k \leq m$ do
 6:         $C_k \leftarrow Optics(\{Pt^k(ST)|ST \in pa\}, \sigma)$;
 7:     for each $ST_i \in pa$ do
 8:         $C^0_{CP} \leftarrow pa$; $C^m_{CP} \leftarrow \emptyset$; valid $\leftarrow$ true;
 9:         for $1 \leq k \leq m$ do
10:             $C^k_{CP} \leftarrow \{ST_j|ST_j \in C^{k-1}_{CP},$
                $\exists c \in C_k, Pt^k(ST_i) \in c \; and \; Pt^k(ST_j) \in c\}$;
11:             if $(k > 1) \; and \; (\exists ST_j \in C^k_{CP},$
                $Pt^k(ST_j).t - Pt^{k-1}(ST_j).t) \geq \delta_t)$ then
12:                 delete all such $ST_j$ from $C^k_{CP}$;
13:             if $Den(\{Pt^k(ST)|ST \in C^k_{CP}\}) < \rho$ then
14:                 $pa \leftarrow pa - C^k_{CP}$; valid $\leftarrow$ false; break;
15:         $pa \leftarrow pa - C^m_{CP}$;
16:         if $|C^m_{CP}| < \sigma \; or$ not valid then
17:             continue;
18:         for $1 \leq k \leq m$ do
19:             $pa_{finegrained} \leftarrow pa_{finegrained} \cup$
                $\{CenterPoint(\{Pt^k(ST)|ST \in C^m_{CP}\})\}$;
20:         $PA_{finegrained} \leftarrow PA_{finegrained} \cup \{pa_{finegrained}\}$;
21: return $PA_{finegrained}$;
```

For each coarse semantic pattern, we can calculate its length $m$ (line 4). All $k$th points in each semantic trajectory of the coarse pattern are collected to generate clusters by *Optics* [27], with the results stored in $C_k$ (line 6). We render *Optics* to finish clustering tasks without the configuration of distance threshold. It initiates with a default maximum distance threshold and cluster size threshold $\sigma$ (support threshold) to mark all core points. After that, *Optics* calculates the core distance for each point and generates a queue of points with their smallest reachable distance. It chooses an optimal distance threshold with sufficiently high density for each cluster.

Then, the algorithm picks each semantic trajectory $ST_i$ to gather all counterpart trajectories point by point (line 9-14). First, the procedure is divided into $m$ loops according to the trajectory length. The candidate set $C^k_{CP}$ stores all trajectories that may share counterpart relationship with trajectory $ST_i$ by scanning the leading $k$ points. In $k$th loop, we check whether the $k$th points from trajectories $ST_i$ and $ST_j$ are located in the same spatial cluster $c$ from the *Optics* result $C_k$. If not, the function CP($ST_j$, $ST_i$) in Definition 9 returns empty set. $ST_j$ does not contain or reachable contain $ST_i$ because the points distribute dispersively. Otherwise (i.e., points are located in same cluster and $ST_j$ exists in previous candidate set $C^{k-1}_{CP}$), $ST_j$ will be preserved into current candidate set $C^k_{CP}$ (line 10), that means leading $k$ points in $ST_i$ and $ST_j$ are close enough to make the leading parts of two trajectories share counterpart relationship. This counterpart pending considers the temporal constraint


(a) Detail view

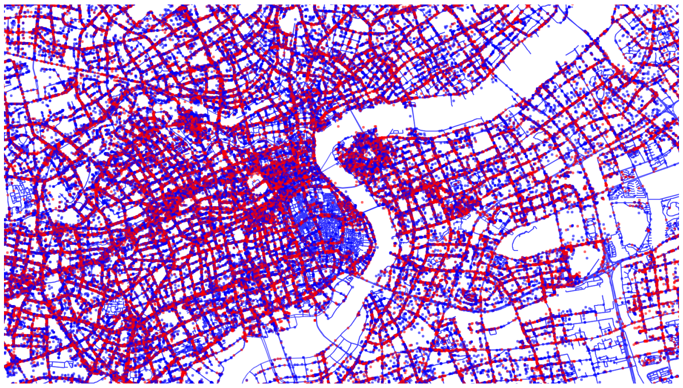

(b) Overall view

Fig. 8. **Taxi stay points in Shanghai, the pick-up/drop-off points are selected to be stay points in our experiments directly**

between two consecutive points in $ST_j$. If the time interval between $k$th and $(k-1)$th points exceeds $\delta_t$, $ST_j$ will not be included in candidate set $C^k_{CP}$ (line 11-12). The pending also examines the spatial density around all $k$th points from trajectories in $C^k_{CP}$. If the density of points is not higher than the threshold $\rho$, we break the pending loop and all failed trajectories in candidate set will be deleted from the coarse pattern $pa$ (line 13-14). The group around all $k$th points is denoted as $\{Pt^k(ST)|ST \in C^k_{CP}\}$ according to Definition 10. High spatial density of each group maintains the low spatial sparsity in the fine-grained semantic patterns.

After these $m$ loops, we gather the counterpart set $C^m_{CP}$, that will be separated from the coarse pattern $pa$ (line 15). If the size of set outnumbers the support threshold $\sigma$ (line 16-17), we push the counterpart trajectories into next extraction stage (line 18-20). For all $k$th points in the counterpart trajectories, they are collected as one group to calculate the center location and the average timestamp. The closest point to the center location will be selected as the representative point, and the new timestamp will be recorded as the average value (line 19). The new trajectories of these representative points form the fine-grained semantic patterns.

## 5 Experiment

In this section, we conduct extensive experiments to evaluate the performance of Pervasive Miner and its competitors under disparate parameter configuration. Experiments are based on a real taxi trajectory dataset collected in Shanghai in the month of April, 2015. The raw GPS trajectory dataset consists of $2.2 \times 10^7$ taxi journeys, with each journey recording a taxi trip from a pick-up location to a drop-off location, as depicted in Figure 8. The red points refer to pick-up locations and the blue points stand for drop-off locations. They are selected to be stay points in our experiments directly. Besides the pick-up and drop-off records, payment card information of 20% passengers (around 184,000 passengers) are also stored in the logs. By linking the consecutive journey trajectories for each passenger in a day, we recover many long movement trajectories with at least three stay points. In order to retrieve semantic property for stay points in raw trajectories, we form the POI dataset from AMAP (http://ditu.amap.com/) to support semantic recognition. This POI dataset contains $1.2 \times 10^6$ POIs with exact semantic property, while all the POIs are categorized into 15 major semantic types and 98 minor semantic types (described in Table 3). The POI dataset covers an area of $6,120$ square kilometer in Shanghai.

TABLE 3
Statistic table of POI category in Shanghai

| **Category** | **Count** | **Percentage** |
|---|---|---|
| Residence | 218,327 | 18.09% |
| Shop & Market | 197,411 | 16.36% |
| Business & Office | 180,962 | 15.00% |
| Restaurant | 136,322 | 11.30% |
| Entertainment | 120,986 | 10.03% |
| Public Service | 113,446 | 9.40% |
| Traffic Stations | 91,079 | 7.55% |
| Technology & Education | 32,190 | 2.67% |
| Sports | 23,418 | 1.94% |
| Government Agency | 22,670 | 1.88% |
| Industry | 17,732 | 1.47% |
| Financial Service | 17,251 | 1.43% |
| Medical Service | 15,894 | 1.32% |
| Accommodation & Hotel | 12,795 | 1.06% |
| Tourism | 6,166 | 0.51% |

In our experiments, we implement six approaches, including Pervasive Miner proposed in this paper and five competitors. Pervasive Miner adopts *City Semantic Diagram* to implement semantic recognition, and we abbreviate this standard Pervasive Miner with name *CSD-PM* (*City Semantic Diagram based Pervasive Miner*). Then, in order to evaluate the strength of *City Semantic Diagram*, we replace CSD with ROI based algorithm [21] in standard Pervasive Miner and rename it as *ROI-PM* (*Region of Interest based Pervasive Miner*). In order to apply *Splitter* and *SDBSCAN* in raw GPS trajectories, we integrate *City Semantic Diagram* or ROI based algorithm with *SDBSCAN* or *Splitter*, so we have another four implementations totally, named as *CSD-SDBSCAN*, *ROI-SDBSCAN*, *CSD-Splitter* and *ROI-Splitter* respectively.

**Parameter Setting.** For all above six approaches, support threshold $\sigma$, temporal constraint $\delta_t$, and density threshold $\rho$ influence their performance in pattern extraction. Support threshold $\sigma$ is the size lower bound of included trajectory set for each pattern as stated in Definition 11. Temporal constraint $\delta_t$ sets the upper bound for time interval between adjacent stay points in semantic trajectories, which contain or reachable contain the semantic pattern as we mentioned in Definition 7 and 8. Density threshold $\rho$ defines the lower bound of spatial density in clusters of semantic trajectories, which contain or reachable contain the semantic pattern in Definition 11. These three parameters are universal factors in all six approaches and declared in Algorithm 4 and Table 2. In normal condition, we set $\sigma = 50$, $\delta_t = 60mins$ and $\rho = 0.002m^{-2}$, because this configuration is generally suitable for all above six approaches.

**Metrics of evaluation.** According to the detail study in previous related works [17], [18], [19], [21], quantity of patterns, semantic and spatial quality of extracted patterns are the pivotal and classical indices for performance evaluation. Same as previous works and study, four common performance metrics are introduced in our experiments as benchmarks, including *number of patterns* detected (notated as ***#patterns***), *coverage*, *spatial sparsity* and *semantic consistency*. ***Coverage*** metric represents the sum of support for each pattern. As we mentioned in Table 2, a pattern's support is the number of semantic trajectories which contain or reachable contain itself. The larger the value, the better the performance. ***Spatial sparsity*** (notated as $ss$) for a pattern indicates the average point distance in each group (Definition 10) of points from trajectories which contain or reachable contain the pattern. Its math expressions are defined in Equation 9 and 10, and a smaller $ss$ is more preferable, because it means higher density. ***Semantic consistency*** (notated as $sc$) is defined as the average semantic cosine similarity between trajectories, and a bigger value is considered to be more preferable.

Let pattern $pa = \{sp_1, \cdots, sp_n\}$, $\forall 1 \leq k \leq n$, there exists a list of groups $\{Group(sp1), \cdots, Group(sp_n)\}$ including corresponding neighbor stay points (see in Definition 10). If $Group(sp_k) = \{sp'_1, \cdots, sp'_m\}$ with $sp'_i.s$ and $sp'_j.s$ referring to the semantic property queried by semantic recognition from $CSD$, we can get:

$$ss(Group(sp_k)) = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} d(sp'_i, sp'_j) \tag{9}$$

$$spatial\ sparsity(pa) = \frac{1}{n} \sum_{k=1}^{n} ss(Group(sp_k)) \tag{10}$$

$$sc(Group(sp_k)) = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} Cos(sp'_i.s, sp'_j.s)) \tag{11}$$

$$semantic\ consistency(pa) = \frac{1}{n} \sum_{k=1}^{n} sc(Group(sp_k)) \tag{12}$$

**Performance Evaluations.** In Figure 9, we plot the frequency curves to demonstrate the spatial sparsity performance of each approach. The x-axis is divided into 20 segments uniformly from 0 to 100 with each segment representing a spatial sparsity bin with the same width 5. Each pattern is assigned to a bin when the sparsity range of this bin covers the pattern's sparsity value. Then the count value of each bin is plotted as a frequency point. By linking the consecutive frequency points, we get the frequency curve to represent the spatial sparsity distribution. After analyzing six sets of patterns under the configuration $\sigma = 50$, $\delta_t = 60mins$ and $\rho = 0.002m^{-2}$, three curves are drawn in the upper part

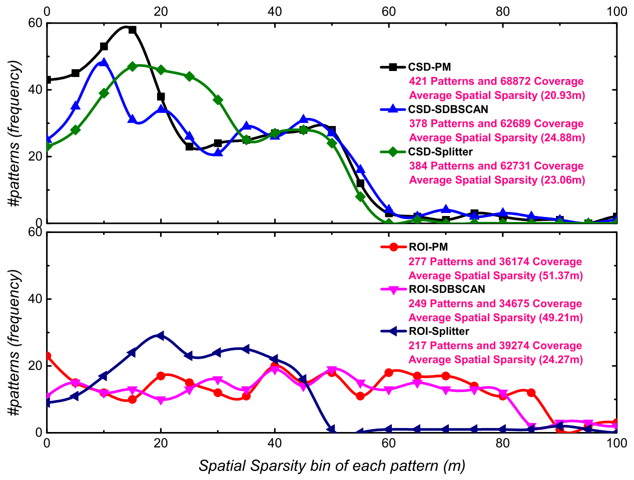


Fig. 9. **Frequency distribution of patterns' spatial sparsity for each approach. A higher point locating on a curve means higher density or frequency.**

of Figure 9 to indicate the CSD based algorithms and other three curves are depicted in the lower part to represent the ROI based ones.

As we mentioned above, *spatial sparsity* indicates the average point distance in each group of points. The higher the curve is located in the low sparsity range, the more the patterns with high density are extracted. On one hand, the upper three curves score higher than the lower three curves in the low sparsity range ($\leq$ 20), which means that more patterns with high density are extracted by CSD based algorithms. On the other hand, the lower three curves maintain relatively large frequency values in the high sparsity range ($\geq$ 60), meaning that ROI based algorithms may output some extremely sparse patterns which are actually harmful to the collection of the fine-grained patterns. Besides, the *CSD-PM* curve owns obvious advantages in low sparsity range ($\leq$ 15), as compared with other two CSD based algorithms. After aggregated statistical calculating, the average spatial sparsity, total number of patterns and total coverage of each curve are exposed in legend. *CSD-PM* owns the minimal average spatial sparsity (20.93m) with the maximal #patterns (421) and coverage (68872). In summary, our approach is good at detecting more dense patterns, in the meanwhile, rejecting the extremely sparse patterns.

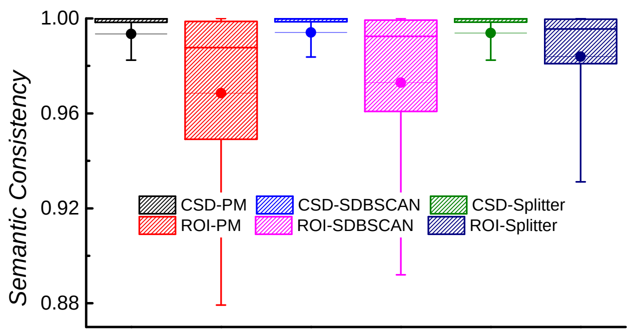


Fig. 10. **Box plots of patterns' semantic consistency for each approach**

Figure 10 shows the box plots of patterns' semantic consistency for each approach. Semantic consistency is defined as the average semantic cosine similarity between trajectories in Equation 11 and Equation 12. The middle point with a crossing horizontal line of each box represents the average value and other horizontal lines indicate the $minimum$, $Q1$, $Q2$, $Q3$ and $maximum$. For CSD based algorithms, their average lines are all higher than 0.99 and none of their minimum values is less than 0.98. Such a stable distribution demonstrates CSD's superior performance in semantic consistency, thanks for the step of semantic purification (presented in Section 4.1) that is able to recognize and handle multiple semantic regions to minimize the damage of *Semantic Complexity*. For ROI based approaches, the *Semantic Complexity* leads to some confusions in pattern extraction. Consequently, all three boxes of ROI based algorithms occupy a large scale.

**Parameter Experiments.** Next, we study the influence of support threshold $\sigma$, with the results shown in Figure 11. We observe that, under different support values, *CSD-PM* consistently outperforms others in terms of number of patterns detected and coverage, as reported in Figure 11(a) and Figure 11(b) respectively. The advantages of *CSD-PM* over other approaches are contributed mainly by *Optics* in pattern extraction. To be more specific, *Optics* optimizes the configuration of distance threshold automatically, thus *CSD-PM* is empowered to discover more fine-grained patterns. As shown in Figure 11(c) and Figure 11(d), CSD-based algorithms perform better than ROI-based ones in general. The trend of curves in Figure 11 shows that if support threshold $\sigma$ is increased, the quality of patterns is improved but the quantity falls.

In addition, we study the impact of density threshold $\rho$, as reported in Figure 12. We observe that density threshold $\rho$ seems to have the similar influence as $\sigma$, as Figure 11(a) demonstrates a similar trend as Figure 12. *CSD-PM* shows its advantages on number of patterns detected and coverage again under varying density, as shown in Figure 12(a) and Figure 12(b). As for the spatial sparsity and semantic consistency, we observe the algorithms based on CSD always perform better than those based on ROI.

Last but not least, we study the impact of temporal constraint $\delta_t$, with the results shown in Figure 13. Temporal constraint $\delta_t$ sets the upper bound for time interval between adjacent stay points in semantic trajectories which contain or reachable contain the semantic pattern as we declared in Algorithm 4, Definition 7, and Definition 8. A shorter temporal constraint wound filter more patterns with a large time interval. For example, the pattern transiting from residences to airports usually costs more than half an hour in its travel. If we set temporal constraint shorter than 30 minutes, considerable time-consuming patterns like residences to airports wound disappear from the results. We observe that there is almost no fluctuation when $\delta_t \geq 30mins$. This phenomenon is caused by some unique characteristics of Shanghai taxi datasets. We find that the average duration of most taxi trips is around 30 minutes, according to the pick up and drop off records, thus the coverage and pattern number drop when $\delta_t = 15mins$. However, *CSD-PM* and CSD based algorithms stand out again when we vary the setting of the temporal constraint.

## 6 Demonstration

In addition to quantitative evaluation reported previously, we visualize the patterns discovered by CSD-PM in Shanghai downtown region in Figure 14. Here, all the patterns are categorized into six specific time intervals in a week, based on the time of a day and the day within a week, including *weekday morning*, *weekday afternoon*, *weekday night*, *weekend morning*, *weekend afternoon*, and *weekend night*. In the morning of a weekday, a large number of people move from the faraway residential areas to the work places of the city, as shown in Figure 14(a). Most of the patterns are recognized as *Residence* $\rightarrow$ *Office* or *Residence* $\rightarrow$ *Airport*. In the afternoon of a weekday, both the number of patterns detected and the average length of patterns decrease, because majority of people work in their offices with relatively low demand for taxi, as visualized in Figure 14(b). In the evening of a weekday, the demand for taxi becomes high again, and lots of patterns re-appear. Consistent with our expectation, patterns such as *Office* $\rightarrow$ *Supermarket* and *Restaurant* $\rightarrow$ *Residence* are detected, as shown in Figure 14(c).

Similarly, we also plot the patterns detected in weekends in Figure 14(d), Figure 14(e) and Figure 14(f), corresponding to the morning, the afternoon, and the evening of weekends respectively. Compared with the findings from weekdays, weekend's patterns are sparse and irregular; and many patterns observed during weekday do not appear in weekend. This is because our life in weekend is not constrained by our regular jobs and hence we have more freedom to arrange our activities with more variety.

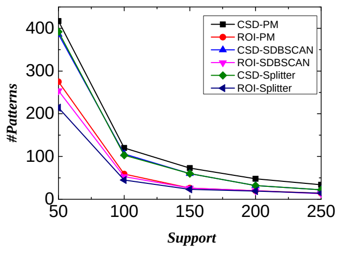
Figure 14(g) shows a group of patterns detected by Pervasive Miner around Shanghai Hongqiao International Airport. This group of patterns covers $20\%$ of total taxi pick-up and drop-off records in our dataset. As observed, many patterns direct to the airport and the movement trend is obvious. This dominant phenomenon reveals human's enormous demand on taxis. In addition, we report some patterns detected near Children's Hospital of Fudan University in Figure 14(h). It is observed that our approach is able to detect many trips to/from hospitals, which are hardly discovered from online social media datasets such as check-in records or twitter messages. Most people consider their visits to hospitals are private and sensitive. Consequently, the number of check-in logs or text messages that are related to medical visits is extremely small. This example shows that our solution excludes the deficiency of *Semantic Bias* by data selection and processing.

## 7 Conclusion

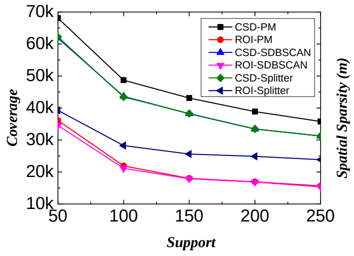
In summary, our approach addresses the challenge of *Semantic Absence* by recognizing semantic behaviours from raw trajectories; it prevents *Semantic Bias* via using ubiquitous GPS trajectory datasets; and it is capable of detecting *Semantic Complexity* from human mobility, thanks to the dedicated design of *semantic purification* for *City Semantic Diagram*. Comprehensive and massive experiments are conducted based on trajectories from more than 10,000 taxis, 920,000 passengers and 1,200,000 POIs in Shanghai. *City Semantic Diagram* is proved to maintain high semantic consistency and low spatial sparsity in extracted patterns.

## Acknowledgments

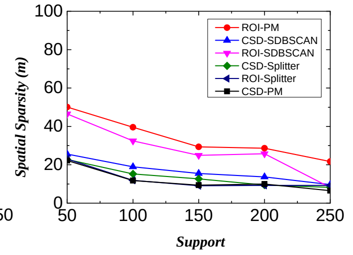
This research is supported by the National Key Research and Development Program of China under grant



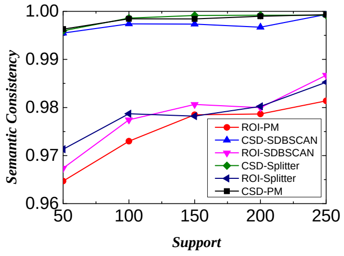
(a) $\#Patterns\ w.r.t.\ \sigma$



(b) $Coverage\ w.r.t.\ \sigma$

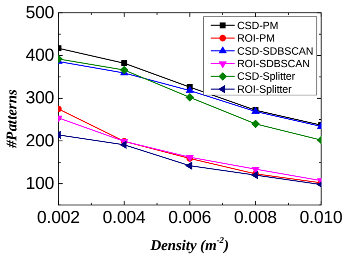


(c) $Spatial\ sparsity\ w.r.t.\ \sigma$

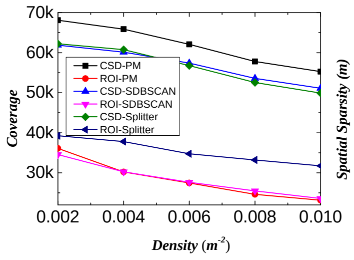


(d) $Semantic\ consistency\ w.r.t.\ \sigma$

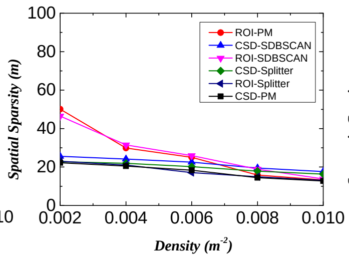
Fig. 11. **Performance comparison on pattern number, coverage, average spatial sparsity and average semantic consistency with various support $\sigma$**



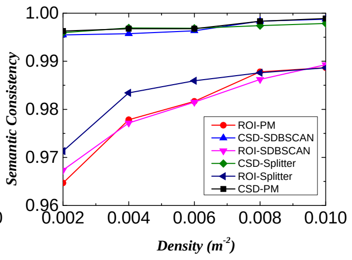
(a) $\#Patterns\ w.r.t.\ \rho$



(b) $Coverage\ w.r.t.\ \rho$

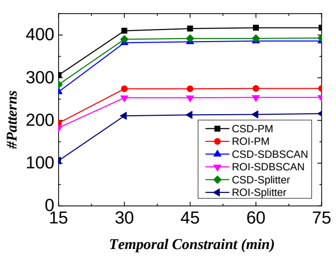


(c) $Spatial\ sparsity\ w.r.t.\ \rho$

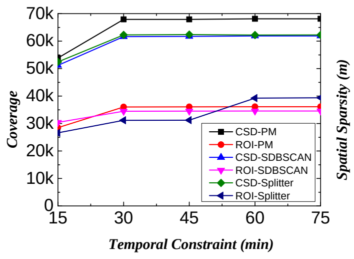


(d) $Semantic\ consistency\ w.r.t.\ \rho$

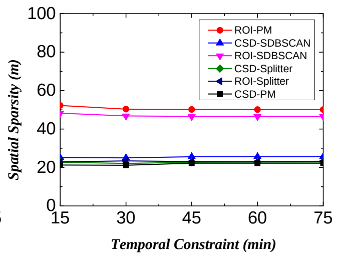
Fig. 12. **Performance comparison on pattern number, coverage, average spatial sparsity and average semantic consistency with various density $\rho$**



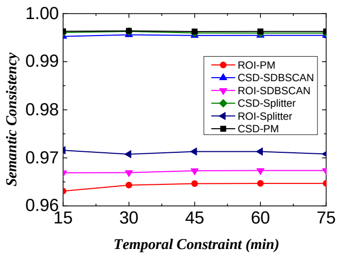
(a) $\#Patterns\ w.r.t.\ \delta_t$



(b) $Coverage\ w.r.t.\ \delta_t$

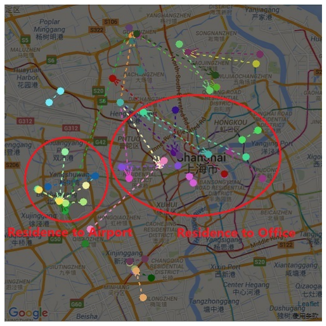


(c) $Spatial\ sparsity\ w.r.t.\ \delta_t$

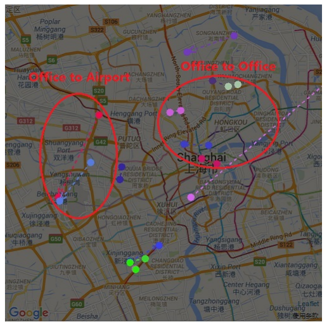


(d) $Semantic\ consistency\ w.r.t.\ \delta_t$

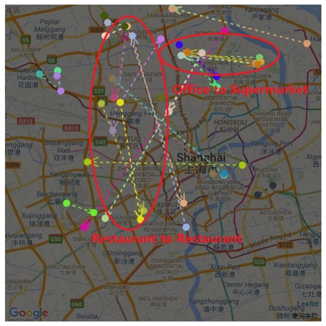
Fig. 13. **Performance comparison on pattern number, coverage, average spatial sparsity and average semantic consistency with various temporal constraint $\delta_t$**



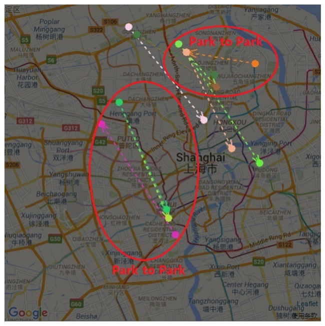
(a) weekday morning



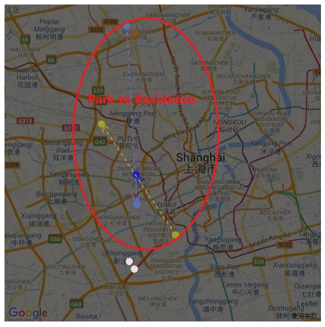
(b) weekday afternoon



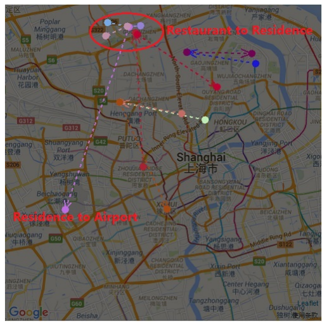
(c) weekday night



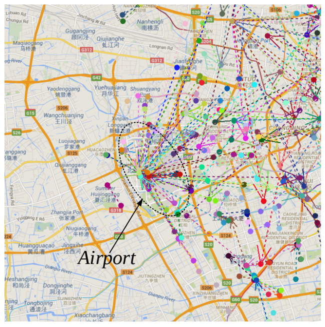
(d) weekend morning



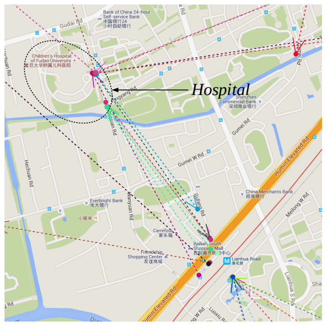
(e) weekend afternoon



(f) weekend night



(g) Patterns around airport



(h) Patterns around hospital

Fig. 14. **(a)-(f) show patterns discovered by Pervasive Miner in Shanghai downtown region from one day taxi records of weekday or weekend. (g) and (h) show dominant and typical patterns**

2018YFB0505000, the National Natural Science Foundation of China under grant 61772138, and National Research Foundation, Singapore under its International Research Centres in Singapore Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

## References

[1] Xin Cao, Gao Cong, and Christian S. Jensen. Mining significant semantic locations from GPS data. *Proceedings of the VLDB Endowment*, 3(1-2):1009–1020, 2010.

[2] Nicholas Jing Yuan, Fuzheng Zhang, Defu Lian, Kai Zheng, Siyu Yu, and Xing Xie. We know how you live: exploring the spectrum of urban lifestyles. In *Conference on Online Social Networks, COSN'13, Boston, MA, USA, October 7-8, 2013*, pages 3–14, 2013.

[3] Andrey Tietbohl Palma, Vania Bogorny, Bart Kuijpers, and Luis Otávio Alvares. A clustering-based approach for discovering interesting places in trajectories. In *Proceedings of the 2008 ACM Symposium on Applied Computing (SAC), Fortaleza, Ceara, Brazil, March 16-20, 2008*, pages 863–868, 2008.

[4] Zhixian Yan, Dipanjan Chakraborty, Christine Parent, Stefano Spaccapietra, and Karl Aberer. Semantic trajectories: Mobility data computation and annotation. *ACM Transactions on Intelligent Systems and Technology*, 4(3):49, 2013.

[5] Miriam Baglioni, José Antônio Fernandes de Macêdo, Chiara Renso, Roberto Trasarti, and Monica Wachowicz. Towards semantic interpretation of movement behavior. In *Advances in GIScience, Proceedings of the 12th AGILE Conference, Hannover, Germany, 2-5 June 2009*, pages 271–288, 2009.

[6] Yingjie Hu, Krzysztof Janowicz, David Carral, Simon Scheider, Werner Kuhn, Gary Berg-Cross, Pascal Hitzler, Mike Dean, and Dave Kolas. A geo-ontology design pattern for semantic trajectories. In *Spatial Information Theory - 11th International Conference, COSIT 2013, Scarborough, UK, September 2-6, 2013. Proceedings*, pages 438–456, 2013.

[7] Christine Parent, Stefano Spaccapietra, Chiara Renso, Gennady L. Andrienko, Natalia V. Andrienko, Vania Bogorny, Maria Luisa Damiani, Aris Gkoulalas-Divanis, José Antônio Fernandes de Macêdo, Nikos Pelekis, Yannis Theodoridis, and Zhixian Yan. Semantic trajectories modeling and analysis. *ACM Comput. Surv.*, 45(4):42:1–42:32, 2013.

[8] https://foursquare.com/.

[9] Y. Zheng. Trajectory data mining: an overview. *TIST*, 6(3):29, 2015.

[10] C. Chen and M. Chiang. Trajectory pattern mining: exploring semantic and time information. In *TAAI*, pages 130–137, 2016.

[11] Ilias Tsoukatos and Dimitrios Gunopulos. Efficient mining of spatiotemporal patterns. In *7th International Symposium on Advances in Spatial and Temporal Databases*, pages 425–442, 2001.

[12] Junmei Wang, Wynne Hsu, Mong-Li Lee, and Jason Tsong-Li Wang. Flowminer: Finding flow patterns in spatio-temporal databases. In *16th IEEE International Conference on Tools with Artificial Intelligence*, pages 14–21, 2004.

[13] Fosca Giannotti, Mirco Nanni, Fabio Pinelli, and Dino Pedreschi. Trajectory pattern mining. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 330–339, 2007.

[14] Hoyoung Jeung, Man Lung Yiu, Xiaofang Zhou, Christian S. Jensen, and Heng Tao Shen. Discovery of convoys in trajectory databases. *The Proceedings of the VLDB Endowment*, 1(1):1068–1080, 2008.

[15] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 593–604, 2007.

[16] Zhenhui Li, Bolin Ding, Jiawei Han, and Roland Kays. Swarm: Mining relaxed temporal moving object clusters. *The Proceedings of the VLDB Endowment*, 3(1):723–734, 2010.

[17] Chao Zhang, Jiawei Han, Lidan Shou, Jiajun Lu, and Thomas F. La Porta. Splitter: Mining fine-grained sequential patterns in semantic trajectories. *The Proceedings of the VLDB Endowment*, 7(9):769–780, 2014.

[18] Younghoon Kim, Jiawei Han, and Cangzhou Yuan. TOPTRAC: topical trajectory pattern mining. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 587–596, 2015.

[19] Renhe Jiang, Jing Zhao, Tingting Dong, Yoshiharu Ishikawa, Chuan Xiao, and Yuya Sasaki. A density-based approach for mining movement patterns from semantic trajectories. In *TENCON 2015 IEEE Region 10 Conference*, pages 1–6, 2015.

[20] Zhijun Yin, Liangliang Cao, Jiawei Han, Jiebo Luo, and Thomas Huang. Diversified trajectory pattern ranking in geo-tagged social media. In *Proceedings of the 11th SIAM International Conference on Data Mining*, pages 980–991, 2011.

[21] Chien-Cheng Chen, Chia-Hsiang Kuo, and Wen-Chih Peng. Mining spatial-temporal semantic trajectory patterns from raw trajectories. In *2015 IEEE International Conference on Data Mining Workshop*, pages 1019–1024, 2015.

[22] Luis Otavio Alvares, Vania Bogorny, Bart Kuijpers, Jose Antonio Fernandes de Macedo, Bart Moelans, and Alejandro Vaisman. A model for enriching trajectories with semantic geographical information. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*, page 22, 2007.

[23] Zhixian Yan, Dipanjan Chakraborty, Christine Parent, Stefano Spaccapietra, and Karl Aberer. Semitri: a framework for semantic annotation of heterogeneous trajectories. In *Proceedings of the 14th international conference on extending database technology*, pages 259–270, 2011.

[24] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Meichun Hsu. Prefixspan: Mining sequential patterns by prefix-projected growth. In *Proceedings of the 17th International Conference on Data Engineering*, pages 215–224, 2001.

[25] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[26] James Chaffee, Jonathan Abel, and Barbara K McQuiston. GPS positioning, filtering, and integration. In *Proceedings of the IEEE 1993 National Aerospace and Electronics Conference-NAECON 1993*, pages 327–332. IEEE, 1993.

[27] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: ordering points to identify the clustering structure. In *ACM Sigmod Record*, volume 28, pages 49–60. ACM, 1999.

**Zhangqing Shan** is a phd candidate student in the School of Computer Science and Technology, Fudan University, China. He received his Bachelor degree in computer science from Fudan University, China, in 2015. His research interests include spatio-temporal data mining, urban computing and pervasive computing.

**Weiwei Sun** received his Ph.D. degrees in computer software and theory in 2002 from Fudan University, Shanghai. He is currently an professor in the School of Computer Science and the director of Mobile Data Management Laboratory, Fudan University, Shanghai. His research interests include spatial database and spatiotemporal data mining.

**Baihua Zheng** is a professor in the School of Information Systems, Singapore Management University. She received her PhD degree in computer science from Hong Kong University of Science and Technology in 2003. Her research interests include mobile data management, mobile/pervasive computing, and big data analytics. She has published more than 100 technical papers in these areas.