Information Sciences 538 (2020) 176–192

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Semantic trajectory representation and retrieval via hierarchical embedding

Chongming Gao [a], Zhong Zhang [a], Chen Huang [a], Hongzhi Yin [b], Qinli Yang [a], Junming Shao [a,*]

[a] University of Electronic Science and Technology of China, China
[b] The University of Queensland, Australia

ARTICLE INFO

Article history:
Received 6 September 2019
Received in revised form 19 May 2020
Accepted 26 May 2020
Available online 12 June 2020

Keywords:
Trajectory representation
Clustering
Semantic embedding
Semantic retrieval

ABSTRACT

Trajectory mining has gained growing attention due to its emerging applications, such as location-based services, urban computing, and movement behavior analyses. One critical and fundamental mining task is to retrieve specific locations or trajectories that satisfy particular patterns. However, existing approaches mainly represent the trajectory as a collection of geographic and temporal features, so the latent semantic properties are barely considered. In this paper, we introduce a new semantic trajectory representation method, which considers trajectory structures, temporal information, and domain knowledge to make efficient semantic retrieval possible. Specifically, we first introduce a synchronization-based model to identify multi-resolution regions of interest (ROIs) to extract structures from disordered raw trajectories. Afterward, we proposed a hierarchical embedding model to embed ROIs as well as trajectories on the hierarchical ROI network as continuous vectors by considering multiple kinds of semantic similarity. As a result, users can easily retrieve desirable ROIs or trajectories by computing the similarity among embedded vectors. Experiments show that our approach excels both classical trajectory metric-based models and state-of-the-art deep network embedding models in terms of retrieving interpretable ROIs and trajectories.

1. Introduction

The increasing use of location-aware devices has been boosting the generation of trajectory data in diverse fields. Generally, a trajectory is represented as a sequence of time-stamped geographic locations, and it is used to characterize the movement of objects, such as people, vehicles, animals, hurricanes, and ocean currents [1]. Recently, due to the explosion and increasing use of taxi service software, e.g., DiDi, Uber, and the social media such as Facebook and Foursquare, raw trajectory data has been enriched with various semantic information. Mining the semantic knowledge is a key to understanding human mobility, which makes semantic trajectory mining a prevalent subject [2].

For extracting knowledge from trajectory data, the primary step is to generate a good trajectory representation. However, it remains to be a challenging task because of the disordered nature of trajectory data. For instance, sampling rates and lengths could vary in different trajectories. At present, there are mainly three strategies for trajectory representation: point-based [3,4], segment-based [5,6], and feature-based representation [7,8]. Based on the representative strategies, many distance functions are defined (e.g., DTW [9], LCSS [10], EDR [11], and Fréchet distance [12]).

* Corresponding author.
URL: http://dm.uestc.edu.cn (J. Shao), .

https://doi.org/10.1016/j.ins.2020.05.107
0020-0255/

Unfortunately, the mainstream trajectory representation methods mentioned above focus mainly on geographic information, and they barely consider semantic knowledge. Thus, those trajectory representation methods are limited to some specific and easy mining tasks. Existing methods cannot support complex and multi-faceted queries like (1) Retrieve some dangerous regions in the city that have similar characteristics with the given crime areas; (2) Retrieve the most suspected trajectories from the database by giving an observed trajectory of a criminal. For such complex retrieval tasks, users have to manually dissect the problem into different parts and then construct queries separately.

1.1. Basic Idea

In order to achieve the aforementioned semantic region and trajectory retrieval tasks, we aim to define a similarity metric that is tailored to integrate certain desirable semantic similarity. For illustration, we display in Fig. 1(a) three taxi trajectories (i.e., T_1, T_2 and T_3) in Porto, Portugal. If we only consider the geometric properties of the three trajectories, the similarity between T_1 and T_2 is higher than that between T_2 and T_3. Nevertheless, by taking into account the transport system (i.e., structural information) and region functions (i.e., domain knowledge) of city, we find that T_2 is more similar to T_3 (Fig. 1 (b)). Because both T_2 and T_3 indicate the movement from the rural area to the downtown through the motorways, while T_1 represents the movement from airport to the seashore area through the national roads.

In this work, we propose a semantic trajectory representation, in which we preserve the trajectory structures and incorporate the desirable semantic information. To begin with, we propose a synchronization-based clustering model, CascadeSync, to transform raw GPS points to multi-resolution regions of interest (ROIs) from fine-grained to coarse-grained levels. Each ROI in the tree-shaped hierarchical network is a compact prototype of a large set of GPS points or small ROIs. Therefore, trajectories can be replaced by sequences of ROIs on each level of the network (Fig. 2(a)–(c)). Afterward, we propose a hierarchical embedding model to embed each ROI/trajectory to a continuous vector by leveraging network structures, temporal information, and other domain knowledge. In this way, the semantic information is well preserved in the metric of the embedding space. The whole procedure is shown in Fig. 1(c). We can cast the semantic retrieval task as the problem of searching the most similar vectors to the target vector by utilizing the representation.

1.2. Main contributions

The main contributions of this work are summarized as follows.

1. We propose a synchronization-based clustering algorithm CascadeSync that extracts the trajectory structure as a multi-resolution ROI network. The geographic and temporal information and global statistics are preserved on the hierarchical ROI network.

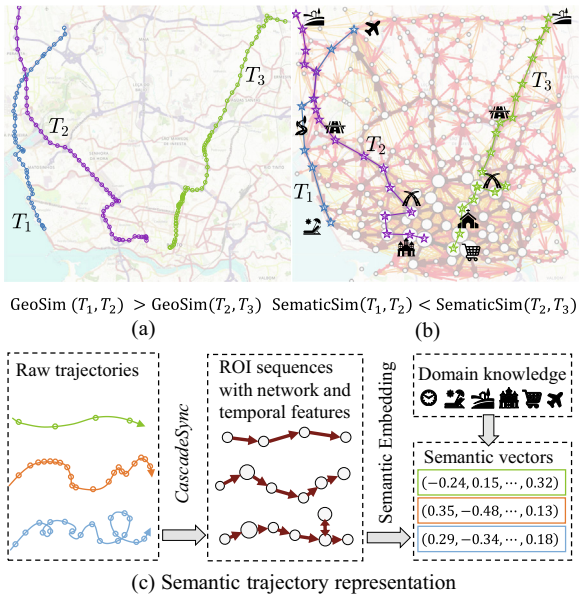

Fig. 1. Illustration of trajectory representation. (a) Three raw trajectories in Porto, Portugal. (b) The same trajectories represented by nodes in the semantic ROI network. (c) Procedures of transformation from raw trajectories to semantic vectors.

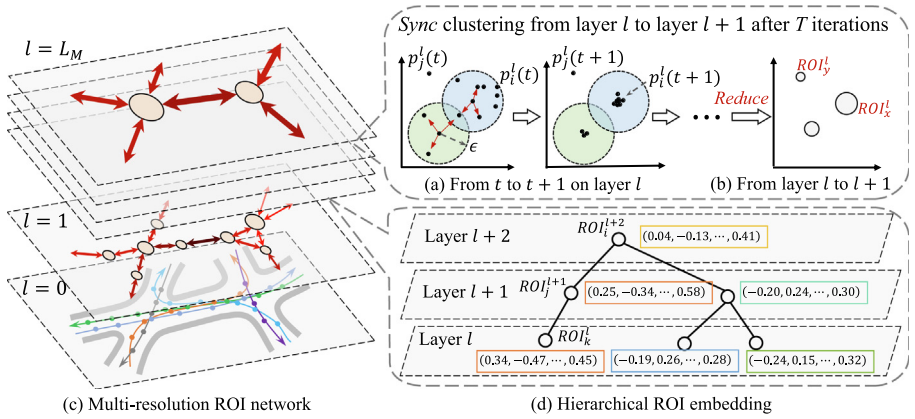

Fig. 2. Illustration of the CascadeSync algorithm and the ROI embedding on the hierarchical ROI network.

2. Utilizing the geometric property and semantic information (e.g., network structures, temporal information, and domain knowledge), we propose a hierarchical embedding model to embed each ROI/trajectory as a continuous vector in a semantic vector space. Thereby, the semantic similarity between two ROIs/trajectories can be measured by computing the Euclidean distance of two vectors directly.
3. The empirical experiments show that the proposed method successfully captures semantic information of trajectory data, and it has superior performance in semantic ROI/trajectory retrieval tasks compared with state-of-the-art methods.

The rest of paper is organized as follows. Section 2 discusses related work. Section 3 elaborates the proposed method. Section 4 presents the experiments. Finally, we conclude our work in Section 5.

2. Related work

2.1. Trajectory representation and retrieval

Trajectory representation, indexing, and retrieval are the foundations of trajectory pattern mining. Here we introduce a summary of state-of-the-art trajectory representation and retrieval strategies.

2.1.1. Trajectory representation

Currently, there are mainly three strategies to represent the trajectory: point-based representation, segment-based representation, and feature-based based representation. Point-based representation is the most common method. The key idea is to detect the most representative points and use them to describe trajectories. Some researchers defined the representative point as *stay points*, which are the small regions where the moving objects stop for a relatively long time [13]. Li et al. [14] detected a few *reference spots* where a moving object visited frequently.

Segment-based representation is also popular. Lee et al. [5] partitioned trajectories to segments by using the concept of Minimal Description Language (MDL). Zheng et al. [15] divided trajectories into alternate *walk segments* and *nonwalk segments* based on locations and velocities of moving objects. Moreover, some work matches trajectories to the road map to get restrained segments [16,17].

Feature-based representation aims to construct a new feature space to represent trajectories. Annoni and Forster [7] transferred trajectories to Fourier coefficients, simplifying a 2D motion representation to a 1D complex sequence representation. Gariel et al. [18] resampled trajectories to have a unified length. Then they used principal component analysis (PCA) to pursue a compact representation. And some work [8,19] fed trajectories to Long-Short Term Memory (LSTM) networks to learn the spatial path features encoded in hidden vectors.

2.1.2. Trajectory retrieval

Usually, the queries of trajectories are straightforward: ask for points of interest (POIs) or trajectories that bear the most resemblance to a given POI/trajectory. The most critical task is to define the similarity metric. Almost all mainstream metrics only focus on geographic similarity. Early researchers [20] used the sum-of-pairs distance, which requires trajectories to have a unified length. This setting is impractical in real-world data. Dynamic time warping (DTW) distance is the first one to overcome the defect. It allows some points repeating multiple times in order to obtain optimal alignment [9]. The longest common subsequence (LCSS) distance [10] and edit distance on real sequence (EDR) distance [11] were proposed to remove the effects caused by the noise points. Fréchet distance was proposed to take into account the locations and ordering of the points along the curves [12].

Building upon the representations and measures, many trajectory indexing structures were proposed. For example, STR-tree [21], TB-tree [21], and HR-tree [22] generalized the R-tree, an effective spatial database, to store spatial and temporal dimensions together. Afterward, SETI [23] was proposed to distinguish temporal information from spacial indexing system to increase the retrieval efficiency.

However, almost all traditional trajectory representation models and indexing systems are built upon the geographic and temporal features of trajectory data. Thus the semantic retrieval and mining tasks are hard to perform.

2.2. Distributed vector embedding

Word embedding becomes ubiquitous in natural language processing (NLP) and information retrieval (IR). It captures the syntactic and semantic similarities among words and exhibits the effectiveness on large datasets [24,25]. The philosophy of word embedding is to project words to a continuous vector space in an unsupervised way where the context relationship of words in documents is exploited. Afterward, the semantic meanings can be exploited by computing on the embedding vectors. Recently, a large body of work has tried to compute the embedding that captures the semantics of word sequences (phrases, sentences, and paragraphs) instead of words, with methods ranging from the simple additional composition of the word vectors [26,27] to sophisticated architectures such as convolutional neural networks [28] and recurrent neural networks [29].

Furthermore, researchers applied the distributed representation strategy to network representation learning. Network representation learning intends to embed network into a low dimensional space while preserving the network structure and property. In consequence, the learned embedding vectors can be applied to the downstream network tasks. In all network embedding methods, random walk-based methods are popular because it can easily captured the semantic network information. DeepWalk [30] is the first work that learns node embeddings by randomly walking on the graph. Then it feeds the generated sequences to a skip-gram model [31]. LINE [32] directly samples node pairs from the graph and preserves nodes' first and second-order proximities to generate representation. Node2Vec [33] develops a generalized bias random walk mechanism that can parameterize Breadth First Search and Depth First Search. Moreover, a lot of work was proposed to handle the heterogeneous information network (HIN). BGEM [34] uses a simple strategy by minimizing the KL-divergence between every pair of bipartite graphs. Metapath-based methods [35,36] define a collection of meta-paths, which are then used to restrict the direction while generating random walks.

In our work, ROIs are analogous to words while trajectories are analogous to sentences. However, the similarity metrics of trajectory data are in distinct contrast to those of languages. For example, if two words are similar in language semantics, they will be contexts of each other in some sentences. Conversely, two similar ROIs will never adjoin each other in any trajectory when the geographic distance between them is too large. Therefore, the semantic contexts (neighborhoods) have to be carefully defined.

2.3. Synchronization-based clustering

Our work is also highly related to synchronization-based clustering [37,38]. Unlike traditional clustering algorithms, synchronization-based clustering makes each data point interact with surrounding neighbors and dynamically changes all points' positions. The model converges when points stop changing positions. Recently, many synchronization-based models and data mining algorithms have been proposed, and they have shown many desirable properties in a wide range of applications [39–44]. For instance, in network mining, each node on the network is pushed or pulled through the local network structure, resulting in a few distinct communities [41]. Seliger et al. [40] discussed mechanisms of learning and plasticity in networks of phase oscillators through a generalized Kuramoto model. In bioinformatics, a strategy was proposed to find groups of genes by analyzing the cell-cycle-specific gene expression [39]. From another perspective, the correlated genes and conditions can interact simultaneously, and a set of co-clusters are yielded [42]. The concept of synchronization is suitable for trajectory data. Because it does not need to determine k explicitly like k-means based models, nor does it render clusters in arbitrary shapes like DBSCAN-based models.

3. Proposed method

Notations. The most prevalent data format of the trajectory is a temporal sequence of latitude/longitude coordinates as follows:

$$T = \{\langle s_1, s_2, \ldots, s_n\rangle | s_i = (p_i, t_i)\}, \tag{1}$$

where $p_i = (x_i, y_i)$ is a (*latitude, longitude*) pair (GPS coordinate) representing a sample point on the map. t_i is the time stamp of the sample s_i, and n is the length of this trajectory.

In our work, as mentioned above, a massive amount of GPS points are reduced to a few multi-resolution regions of interest (ROIs) (Fig. 2(c)). Consequently, the trajectories are reduced to sequences of ROIs on different layers of the hierarchical ROI network. The trajectory represented by ROIs on layer l is written as,

Table 1
Important notations.

Notation	Description
l	$l = \{0, 1, 2, \ldots, L_M\}$ is the layer of ROI networks
$\mathcal{T}^l$	Trajectory set. Each $T_i^l \in \mathcal{T}^l$ is a trajectory written as (2)
$\mathcal{ROI}^l$	ROI set. Each $ROI_i^l \in \mathcal{ROI}^l$ is a node of the network on the l-th layer ($l > 0$)
$\mathcal{P}^l$	Point set. Each $P_i^l \in \mathcal{P}^l$ is a GPS point ($l = 0$) or the center of ROI_i^l($l > 0$)
ϵ	The radius of the interaction range, defined in (3)

$$T^l = \{\langle s_1^l, s_2^l, \ldots, s_m^l\rangle | s_i^l = (p_i^l, t_i^l)\}, \tag{2}$$

where the superscript l denotes the layer of hierarchical ROI network, p_i^l is the center point of the ROI that replaces one or more raw GPS points on raw trajectory T. In addition, some key notations used in this work are listed in Table 1.

3.1. Multi-resolution regions of interest extraction via CascadeSync model

Trajectories are usually represented by a mass of GPS points. It is not a good way to extract patterns and mine the trajectory data on these points directly, as different trajectories often have varying lengths and different sampling rates. One intuitive way is to reduce those raw points to fewer representative prototypes. Here, we employ a synchronization-based clustering to obtain these prototypes.

Typically, a synchronization-based clustering algorithm needs three definitions to simulate a dynamic clustering process: first, a parameter ϵ specifying the interaction range among objects; second, the interaction model for clustering; third, a stopping criterion to terminate dynamic clustering. Our approach follows and extends the synchronization-based clustering algorithm, SYNC, which is presented and discussed in full detail in Ref. [37].

Definition 1 (ϵ-*Range Neighborhood*). Given a GPS dataset $\mathcal{P} \subset \Re^n$, the ϵ-range neighborhood of a GPS point $p \in \mathcal{P}$, denoted as $N_\epsilon(p)$, is defined as:

$$N_\epsilon(p) = \{q | dist(p, q) \leqslant \epsilon\}, \tag{3}$$

where $dist(p, q)$ is a metric distance function. The Euclidean distance is used in this study.

Definition 2 (*Interaction Model*). Let p be a GPS point on the map. With an ϵ-range neighborhood interaction, the dynamics of the value of the point p is defined as:

$$p(t+1) = p(t) + \frac{1}{|N_\epsilon(p)|} \cdot \sum_{q \in N_\epsilon(p)} \sin(q(t) - p(t)), \tag{4}$$

where $sin(x)$ is the coupling function, applied to every dimension of the vector x. $p(t+1)$ is the renewal position of $p(t)$ during the dynamic clustering, $t \in \{0, \ldots, T\}$ denotes the iteration step. Note all dimensions are normalized to $[0, \pi/2]$ to let $sin(\cdot)$ make sense.

Definition 3 (*Cluster Order Parameter*). The cluster order parameter r is used to terminate the dynamic clustering by investigating the degree of local synchronization, which is defined as:

$$r(t) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|N_\epsilon(p(t))|} \sum_{q \in N_\epsilon(p)} \mathbf{e}^{-\|q(t) - p(t)\|}, \tag{5}$$

The dynamic clustering terminates when $r(t)$ converges to 1.0, which indicates the local phase coherence. At this moment, all cluster points have the same position.

For synchronization-based dynamic clustering, each point is viewed as a phase oscillator and has its own phase (position) at the beginning. As time elapses, each point interacts with its ϵ-range neighborhood according to the interaction model Eq. (4). Finally, all locally similar points are synchronized together, and prototypes are formed. A prototype also represents the center of an ROI. We illustrate this process in Fig. 2(a) and (b), where p_i^l is a point that would be absorbed into ROI_x^l with other points in this region. Meanwhile, p_j^l has no neighbors to interact. It would, consequently, constitute a solitary region ROI_y^l, which aptly preserves the information of original data. The salient feature of synchronization-based clustering is its dynamic property–the position value of point changes in a non-linear way driven by the local data structure. Hence, the derived ROI well preserves the original data structure.

$$p^l(t+1) = p^l(t) + \frac{1}{\sum_{q^l \in N_\epsilon(p^l)} w_{q^l}} \cdot \sum_{q^l \in N_\epsilon(p^l)} w_{q^l} \sin(q^l(t) - p^l(t)), \tag{6}$$

where w_{q^l} is the number of points that are represented by the prototype q^l on the layer $l-1$. The whole algorithm is illustrated in Algorithm 1.

Finally, bridged by trajectories on every layer, all ROIs compose a hierarchical network. Besides, the speedup of the synchronization process is another desirable property. The reason is that with smaller interaction range, CascadeSync is easier to converge. Although more clusters are generated, they are viewed as new objects and can be further clustered efficiently.

Algorithm 1. Hierarchical ROI Extracting via CascadeSync

Input : Trajectory dataset $\mathcal{T}$;

Output : Hierarchical ROI network $G(V, E)$;

Parameter : Interaction range ϵ_0; Incremental change $\triangle\epsilon$;

$\mathcal{P}^0$ = Raw GPS point set of $\mathcal{T}$;

$\mathcal{P}^0$ = `Norm` $(\mathcal{P}^0)$; ▷ Normalized each dimension to $[0, \pi/2]$.

Layer $l = 0$; Interaction range $\epsilon = \epsilon_0$;

while layer $l \leq \mathcal{L}_M$ **do**

  $t = 0, \mathcal{P}_t^l = \mathcal{P}^l$, *SyncLoopFlag* = *TRUE*;

  while *SyncLoopFlag* == *TRUE* **do**

    foreach point $p_i^l(t) \in \mathcal{P}_t^l$ **do**

      Search its ϵ-range neighbors $N_\epsilon(p_i^l(t))$ with Eq.(3);

      foreach neighbors $q_j^l(t) \in N_\epsilon(p_i^l(t))$ **do**

        Compute $p_i^l(t+1)$ with Eq.(6);

      end

    end

    Compute order parameter $r(t)$ with Eq.(5);

    $t = t + 1; \epsilon = \epsilon + \triangle\epsilon$;

    if $r(t)$ converges **then** *SyncLoopFlag* = *FALSE*;

  end

  $\mathcal{ROI}^l = \{p_1^l(t), \cdots, p_N^l(t)\}$;

  $\mathcal{P}^{l+1} = \mathcal{ROI}^l$; $l = l + 1$;

end

Build a hierarchical network $G(V, E)$ by using $\{\mathcal{ROI}^1, \cdots, \mathcal{ROI}^{L_M}\}$;

3.2. Learning semantic trajectory representation via hierarchical ROI embedding model

After extracting the hierarchical ROI network, we can represent each trajectory as a sequence of ROIs on each layer of the network. Although such a structure provides a more compact way to characterize the trajectory, the semantic information of the whole trajectory is not well exploited. Inspired by the word embedding technique in Natural Language Processing (NLP), we propose an embedding model to further transform ROIs and trajectories to continuous vectors in a semantic space so as to uncover the hidden semantic information in trajectory data.

3.2.1. Semantic ROI embedding on one-layer network

The key to embedding model is to select the neighborhoods or contexts of a given object so that the relationships among objects could be investigated. Therefore, we consider the embedding neighborhoods of an ROI from two aspects: geometric neighborhood and semantic neighborhood.

Geometric Neighborhood. Given an ROI, its geometric neighbors are those surrounding ROIs that can transport to or be transported from this ROI in W-hops on the ROI network. Fig. 3 shows the geometric neighbors with $W = 2$, which are indicated by green and blue circles. For example, ROI_k is one of the geometric neighborhood of ROI_i.

Semantic Neighborhood. An ROI is not just a node on the network; it also represents a region enriched with various semantic information. We define the semantic neighborhoods of a given ROI to encode the rich semantics from the following perspectives:

- *Network properties.* The hierarchical ROI network is a compact representation of trajectory data. In the network, the general statistics of the transportation structure and traffic conditions can be extracted. Specifically, the *node weight* of an ROI, i.e., the number of points the ROI represents, can reflect the population density of this region. The *edge weight* measures the strength of traffic flow between the two regions. The *degree* of an ROI node reflects whether a region is a hub in the city. Moreover, the *neighbor's degree distribution* of an ROI can reflect, from a higher level, the importance of the region. For example, if two ROIs are similar to each other in the above aspects, maybe they represent a hub bus station and a central railway station, or a huge industrial area and a university, respectively.
- *Temporal contexts.* Temporal information is also preserved in the ROI network after applying CascadeSync algorithm. The distribution of *visiting time* and *staying duration* can reflect the temporal patterns of a region. For example, a region that contains many restaurants might be visited mainly at lunchtime and dinner time. The regions representing residential communities have a distinct characteristic that most people stay there all night.
- *Domain Knowledge.* Unlike the network properties and temporal information, domain knowledge cannot be extracted from the trajectory data. Instead, domain knowledge is the auxiliary information incorporated into the embedding model as a kind of supervised label. The domain knowledge includes, for instance, the distance between one ROI and the center point of the city, or whether one ROI represents a road junction, a supermarket, or a dangerous crime area.

An example is illustrated in Fig. 3, where the ROI_j drawn in the purple circle is a semantic neighborhood of target ROI_i. Even though that they are not connected by any trajectory with $W = 2$ hops, the semantics of ROI_j are more similar to those of ROI_i than all other ROI, which results in a strong correlation between them.

Once the geometric and semantic neighborhoods are obtained, we can embed ROIs into a high dimensional space borrowing the idea of the skip-gram model with negative sampling (SGNS) architecture [25].

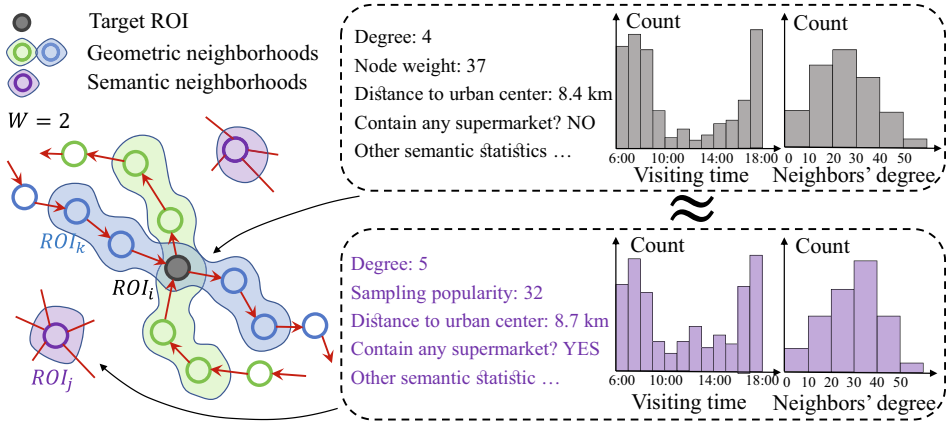

Fig. 3. Illustration of embedding neighborhoods (contexts) of a given target ROI.

Embedding Model. Given a trajectory dataset $\mathcal{T}$, for any layer on the generated hierarchical ROI network, let w be any ROI on this layer, $N(w)$ be the collection of all geometric and semantic neighborhood ROIs of w, and $NEG(w)$ be the negative sampling set of w, which can be drawn using the distribution of network nodes' weight. We define the probability of transportation from a source ROI $u \in N(w)$ to the target ROI v as:

$$p(v|u) = \begin{cases} \sigma(\vec{t_w} \cdot \vec{s_u}) & \text{if } v = w, \\ 1 - \sigma(\vec{t_v} \cdot \vec{s_u}) & \text{otherwise.} \end{cases} \tag{7}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function. $\vec{s_u}$ and $\vec{t_v}$ are the source and target vectors of ROI u and v, respectively.

Accordingly, the likelihood function is written as follows.

$$L_{H^l} = \log \prod_{w \in \mathcal{ROI}^l} \prod_{u \in N(w)} \sigma(\vec{t_w} \cdot \vec{s_u}) \prod_{v \in NEG(w)} [1 - \sigma(\vec{t_v} \cdot \vec{s_u})] = \sum_{w \in \mathcal{ROI}^l} \sum_{u \in N(w)} \left\{ \log \sigma(\vec{t_w} \cdot \vec{s_u}) + \sum_{v \in NEG(w)} \log[1 - \sigma(\vec{t_v} \cdot \vec{s_u})] \right\}, \tag{8}$$

The likelihood function indicates that the target ROI vector in the embedding space should be located as close to all vectors of its neighborhoods as possible, while distinguishing from the vectors of its negative samples.

3.2.2. Feature Propagation through the multi-layer networks

The embedding process is conducted on every layer of the hierarchical ROI network. Moreover, considering every ROI has the parent or children (unless it is on the top or bottom layer), we make an intuitive assumption that every ROI should be embedded close to its parent ROI (if it exists). The assumption is justified in urban semantics. For example, the semantic meaning of a business center should contain all the functions and utilities provided by the companies or shops in the region. Therefore, we augment the embedding model with a regularization term as follows:

$$L_{V^l} = \sum_{w \in \mathcal{ROI}^l} \frac{1}{2} \|\vec{t_w} - \vec{t_{\pi(w)}}\|_2^2 + \frac{1}{2} \|\vec{s_w} - \vec{s_{\pi(w)}}\|_2^2, \tag{9}$$

where $\pi(w)$ is its parent of ROI w. Actually, this term regulates the embedding vectors by propagating influences from parent nodes to children nodes on the hierarchical ROI network. An example is shown in Fig. 2(d), where the ROI_j^{l+1} is a node on layer $l+1$ of hierarchical ROI network, and the vector of ROI_j^{l+1} is constrained by vectors of its parent node ROI_i^{l+2} and its child node ROI_k^l.

3.2.3. Hierarchical ROI embedding on multi-layer networks

Eventually, by integrating the embedding model Eq. (8) and the regularization term Eq. (9), the objective function on the whole hierarchical ROI network is reformulated as follows.

$$\max \sum_{l=1:L_M} L_{H^l} - \alpha \sum_{l=1:L_M-1} L_{V^l} \tag{10}$$

where α is a trade-off coefficient. We apply the stochastic gradient descent (SGD) algorithm for the optimization of Eq. (10). The derivatives of parameters are written as lines 7–17 in Algorithm 2.

3.2.4. Trajectory semantic embedding

Once we derive the semantic embedding for each ROI, the semantic vector of a trajectory can be easily obtained by summing up all feature vectors of the corresponding ROIs with weights. Formally, given a trajectory $T^l = \{ROI_1^l, ROI_2^l, \cdots, ROI_n^l\}$ on the layer l, the embedding vector, denoted as $\text{vec}(T)$, is defined as:

$$\text{vec}(T) = \sum_{l=1}^{L_M} \sum_{i=1}^{n} \tau_i^l \cdot \text{vec}(ROI_i^l). \tag{11}$$

The node weight of ROI_i^l is τ_i^l, which is defined as the duration of the stay of the trajectory T at ROI_i^l. Here, some sophisticated strategies (e.g., a time decay function) can be used to capture the detailed temporal information.

It may seem a little straightforward at first glance, compared to those deep models tailored for sentence embedding [28,29]. However, the additive model is prevalent and proved to be very effective for sentence embedding [45]. Wieting et al. [46] even showed that the model, which embeds the sentences by simply averaging the vectors of words, beats almost all neural network-based models. Besides, it is intuitive as the element ROIs in order can completely characterize a trajectory.

Algorithm 2. Semantic Trajectory Embedding

Input : Hierarchical ROI network $G(V, E)$; Geometric and semantic neighborhoods $N(w)$;
Output : Semantic vectors: vec(ROI) and vec(T);
Parameter: Times of the iteration K, Learning rate η; Trade-off coefficient α;

foreach $ROI_i^l \in \mathcal{ROI}^l$ **do** `RandomlyInit` $(\vec{s_i}, \vec{t_i})$;
foreach iteration $k \in \{1, 2, \cdots, K\}$ **do**
  foreach layer $l \in \{1, 2, \cdots, L_M - 1\}$ **do**
    foreach ROI $w \in \mathcal{ROI}^l$ on the ROI network **do**
      Search the geometric and semantic neighborhood $N(w)$;
      Sample its negative samples $NEG(w)$;
      foreach $u \in N(w)$ **do**
        $e = 0$;
        foreach $v \in \{w\} \bigcup NEG(w)$ **do**
          if $v == w$ **then** $I(v) = 1$; **else** $I(v) = 0$;
          $g = \eta(I(v) - \sigma(\vec{t_v} \cdot \vec{s_u}))$; $\vec{e} = \vec{e} + g\vec{t_v}$;
          $\vec{t_v} = \vec{t_v} + g\vec{s_u} + \eta\alpha(\vec{t_{\pi(v)}} - \vec{t_v})$;
          $\vec{t_{\pi(v)}} = \vec{t_{\pi(v)}} + \eta\alpha(\vec{t_v} - \vec{t_{\pi(v)}})$;
        end
        $\vec{s_u} = \vec{s_u} + \vec{e} + \eta\alpha(\vec{s_{\pi(u)}} - \vec{s_u})$;
        $\vec{s_{\pi(u)}} = \vec{s_{\pi(u)}} + \eta\alpha(\vec{s_u} - \vec{s_{\pi(u)}})$;
      end
    end
  end
end
foreach $ROI_i \in \mathcal{ROI}$ **do** vec$(ROI_i) = [\vec{s_i}, \vec{t_i}]$;
foreach trajectory $T \in \mathcal{T}$ **do** Compute $vec(T)$ via Eq.(11);

3.3. Semantic ROI and trajectory retrieval

In reality, many patterns are contained implicitly in trajectories, which means the queries cannot be conducted directly. As an alternative, we can query the most similar places or trajectories of a selected place or trajectory; i.e., we can retrieve all ROIs/trajectories ordered by the similarity to a specific sample, rather than constructing complex queries directly.

Since ROIs/trajectories are represented as vectors in continuous with semantics, we can retrieve the most similar ROIs/-trajectories by computing the distances among the embedding vectors. Without loss of generality, we use Euclidean distance in this work.

3.4. Time complexity analysis

Our semantic trajectory representation mainly includes two parts: ROI extraction and semantic embedding. For ROI extraction via CascadeSync model, the time complexity is $O(L \times T \times (N_l \times Log(N_l)))$, where N_l is the number of points, which reduces exponentially over the layer l; T is the time steps in each round and L is the number of layers in CascadeSync. Usually, L is small with $L \leq 20$ in practice. The time complexity of ROI embedding is $O(K \times N \times N_1 \times N_2)$, where N is the number of points, N_1 is the number of geometric and semantic neighborhoods, N_2 is the number of negative samples, and K is the number of iterations.

4. Experiments

In this section, we evaluate the proposed method on both the synthetic data and real-world datasets.

4.1. Experimental setup

4.1.1. Datasets

We introduce the synthetic data and four real-world trajectory datasets.

Synthetic Data. To prove the concepts, we generate 50 random trajectories, given a random starting point and a random ending point, using the probabilistic path planning algorithm [47]. To define the semantics in the trajectories, we randomly select 10 points as *key points* and ensure that each key point is passed by at least one trajectory. The semantic ROI and trajectory retrieval tasks aim detect the ROIs and trajectories highly related to these key points.

Real-World Data. We evaluate our proposed method on four trajectory data: Geolife[1] [48], T-Drive[2] [49], Kaggle Taxi[3] and Chengdu. Geolife is a trajectory dataset collected from daily human life, which can reveal human mobility. T-Drive, Chengdu, and Kaggle Taxi contain trajectories generated by urban taxis. The statistics of four datasets are listed in Table 2. It should be noted that both Geolife and T-Drive are trajectory datasets in Beijing, and the distinction between them lies in the sampling rates. About 91 percent of trajectories from Geolife are logged in a dense representation, e.g., every 1–5 s or every 5–10 meters between two continuous sample points, while trajectories from T-Drive are sampled in a very low frequency of 2–5 min per point.

4.1.2. Evaluation metrics

In order to measure the accuracy of semantic retrieval results, we need to define the semantic ground-truth of ROI and trajectory. In the synthetic experiment, the semantic ground-truth is implied in the 10 key points. For similar ROI retrieval task, given any key point, the most similar ROI should be those ROIs that contain other nine key points, which are referred to as the ground-truth ROIs, or the key ROIs. The retrieval performance is measured by the ratio of those correctly detected ROIs. For the trajectory retrieval task, we select the target trajectory as the trajectory that passes ground-truth ROIs as many times as possible. The most similar trajectories to the target, in this semantic, should contain as many ground-truth ROIs as possible. Therefore, we can obtain the ground-truth order of retrieved trajectories. The performance can be measured by comparing the ground-truth order and retrieved order via computing the normalized discounted cumulative gain at a particular rank k (NDCG@k), which is defined as follows.

$$\mathrm{NDCG@k} = \frac{\mathrm{DCG@k}}{\mathrm{IDCG@k}}, \tag{12}$$

where IDCG@k is the *ideal* DCG@k value of the ground-truth order, and DCG@k is defined as:

$$\mathrm{DCG@k} = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}, \tag{13}$$

where rel_i is the ordinal relevance of the result at position i. For clarity, we use 5 levels of relevance throughout all experiments and ensure that each level contains objects equally.

4.1.3. Baseline methods

On the semantic trajectory retrieval task, we compare our proposed method with four traditional trajectory metric methods: DTW [9], LCSS [10], EDR [11], and Fréchet distance [12]. All of them are designed to compute the geometric distance between two trajectories. Furthermore, five state-of-the-art network embedding methods: DeepWalk [30], LINE [32], BGEM [34], Metapath2Vec [35], and Node2Vec [33] are used to retrieve trajectories. Specifically, we learn representations of all nodes on the ROI network via these network embedding methods and then obtain the trajectory representation by averaging the embedding vectors of those points or regions which compose a trajectory.

4.1.4. Data preparation

Real-world Region Ground-truth Acquisition. Real-world trajectory data is enriched with various semantic information; however, the ground-truth labels are usually hard to obtain. Fortunately, benefited from the reverse geocoding function of online map services, MapBox API,[4] and Amap API,[5] we can attain information of regions, which serves as the domain knowledge as well as the ground-truth labels in the experiments. More specifically, the APIs can convert GPS coordinates to function categories, such as banks, schools, or markets. Therefore, we define the semantic ground-truth label of every ROI as its

[1] https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/.
[2] https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/.
[3] https://www.kaggle.com/crailtap/taxi-trajectory.
[4] https://www.mapbox.com/.
[5] https://www.amap.com/.

Table 2
Information of four real-world datasets.

Dataset	#Point	#Traj.	[Longitude, Latitude range]	[Width × Height] (m)
Geolife	24,876,978	18,670	[116.194, 116.553, 39.751, 40.033]	[31,024 × 31,368]
T-Drive	6,969,481	8,768	[116.194, 116.553, 39.751, 40.033]	[31,024 × 31,368]
Kaggle	78,239,735	1,704,769	[−8.702, −8.549, 41.135, 41.246]	[12,613 × 12,365]
Chengdu	9,671,104	2,003	[103.913 ,104.180, 30.538, 30.765]	[25,484 × 25,219]

distribution of function categories, and the ground-truth similarity of two ROIs A and B is defined as the common area of two distributions:

$$\mathrm{Sim}(A,B) = \sum \min(\mathrm{distribution}(A), \mathrm{distribution}(B)). \tag{14}$$

For example, the category distributions of two places are A: {bank, school, market, music} = {0.2, 0.7, 0.1, 0}, B: {bank, school, market, music} = {0.4, 0.3, 0, 0.3}. Then $\mathrm{Sim}(A,B) = 0.2 + 0.3 + 0 + 0 = 0.5$. By the way, the reason why we use Eq. (14) rather than other measures, e.g., KL-divergence, is that it is intuitive and suitable for sparse distribution measuring.

Real-world Trajectory Ground-truth Acquisition. The semantic ground-truth labels and ground-truth similarity of ROIs are easily obtained by utilizing the map APIs and Eq. (14), respectively. The difficulty lies in obtaining the semantic ground-truth of trajectories. Here, we use an intuitive method: A trajectory is composed of a set of sampling points, so the ground-truth label of this trajectory can be computed by the normalized sum of the ground-truth labels of its sampling points. Besides, it is too trivial and expensive to use raw GPS points to call the map APIs. Thus, we use CascadeSync to derive a single-layer ROI network with a small interaction range ($\epsilon = 1/200 * (w + h)$, where w, h are the width and height of map of dataset) to reduce the millions of GPS points to thousands of ROIs. Then the ground-truth labels and similarity (using Eq. (14)) of trajectories can be easily obtained. Later we will use this auxiliary ROI network.

Afterward, we can perform the semantic ROI/trajectory retrieval and compute the NDCG@k between retrieved ROIs/trajectories and ground-truth. Moreover, we can compare the performance with baseline methods. Without loss of generality, the four comparison metrics are computed on the auxiliary ROI network mentioned above.

Trivial Solution Removal. For semantic trajectory retrieval, if retrieved trajectories overlap with querying trajectory, i.e., they share most ROIs, the NDCG@k value will show statistical significance. However, this trivial solution does not have practical significance. Thereby, we stipulate that a retrieved trajectory is valid only when it shares fewer than 50% of ROIs with querying trajectory on the auxiliary ROI network.

4.2. Proof of concept

We start with the experiments on synthetic data to prove the basic properties of our proposed approach. We randomly generate trajectories and key points in a box with the side length being 100 m, illustrated in Fig. 4(a). With varying the interaction ranges $\epsilon = \{2, 4, 6, 8, 10\}(m)$, we apply the CascadeSync model to generate a 5-layer hierarchical ROI network. The bottom layer ROI network is visualized in Fig. 4(b). The key points, which are plotted as grey pentagons on the map, represent important places in reality, e.g., fire stations, crime areas. In the following, we design two experiments of semantic evaluation and trajectory retrieval.

4.2.1. Key point detection on synthetic data

To examine whether our representation embeds information of key points correctly, we experiment by retrieving the most similar region of a given key point. In this task, we use the domain knowledge: whether an ROI contains any key point. To test whether those key ROIs have similar representations, we evaluate the retrieval accuracy of the most similar ROIs for a given key ROI (blue ROI in Fig. 4(b)). The ideal retrieval result is that all key ROIs are more similar than those non-key ROIs.

Building upon the embedding vectors of ROIs and trajectories, we search the most similar POIs. The experiment is repeated 100 times on randomly generated datasets independently. The resulting average accuracy is **99.64**%, demonstrating that the semantics of key points are successfully incorporated into the embedding representation, and the key points can be detected effectively.

4.2.2. Semantic trajectory retrieval on synthetic data

Here, we conduct a similar experiment on the trajectory level. The query is a target trajectory that contains many key ROIs. Thus, the similar trajectories should contain as many key ROIs as possible. We compute and rank the similarity of the rest 49 trajectories to the target, and measure the quality of the ranking result using NDCG@k in Eq. (12). One result is visualized in Fig. 4(c), in which the color shade of green trajectories is proportional to the retrieval order. Moreover, we conduct the same experiment using four state-of-the-art metrics on the first layer ROI network (the finest level that has the key ROI labels). The result is shown in Fig. 5. The experiments are repeated 100 times, and the average NDCG@k value is computed. From the figure, we can observe that our model beats all traditional and deep embedding models, which proves we successfully captured the semantic information in the synthetic data. Also, we observe that all deep embedding models

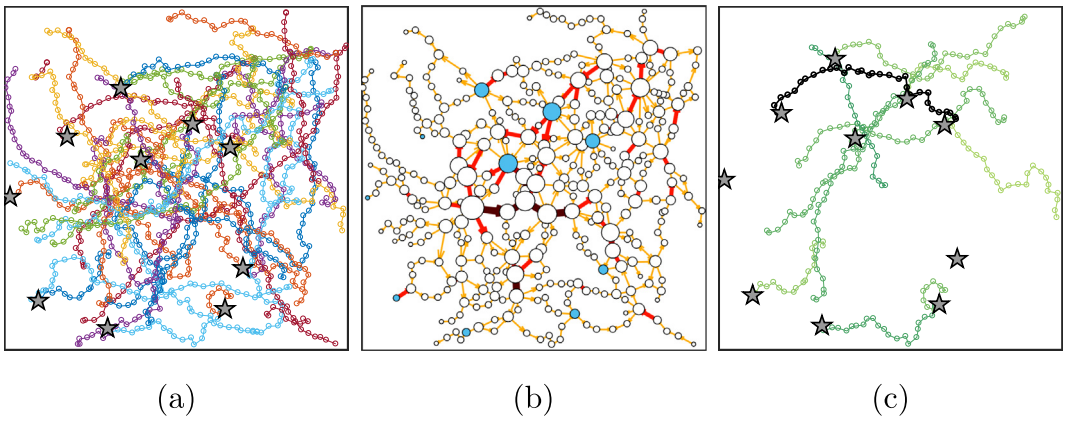

Fig. 4. Illustration of synthetic data and evaluation results. (a) Raw data with key points. The 10 pentagons are randomly generated key points. (b) The bottom layer of the hierarchical ROI network. The blue ROIs are the key ROIs that contain at least one key point. (c) Trajectory retrieval result. The black trajectory is the target. The green trajectories in different shades are retrieved results. The deeper the color is, the more similar they are to the target, i.e., containing more key ROIs.

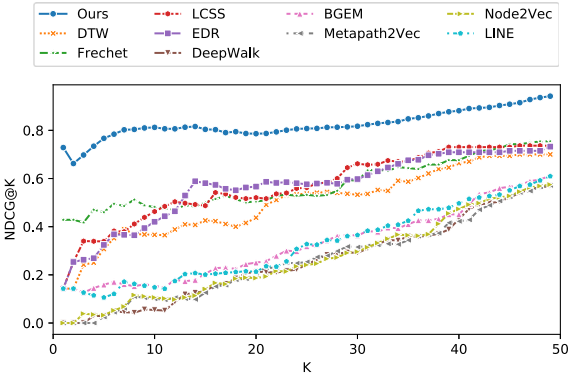

Fig. 5. Semantic trajectory retrieval on the synthetic data.

(DeepWalk, BGEM, Node2Vec, Metapath2Vec, and LINE) have inferior performances compared with the traditional trajectory metric-based models (DTW, LCS, Fréchet distance, and EDR). The reason is that these deep embedding models work by only learning the network structure information, which not only ignores the specific semantics that we defined but also fails to capture the geometric information that traditional metric-based models intend to exploit.

4.3. Evaluation on real-world data

4.3.1. Effect of key parameter

The interaction range ϵ which controls the trade-off of compression degree and the representative accuracy is the most critical parameter in our model. In order to measure the information loss with different values of interaction range ϵ, we use Mean of Squared Error (MSE), defined as:

$$\text{MSE} = \frac{1}{|\mathcal{P}|}\sum_{i=1}^{|\mathcal{P}|}(P_i - ROI_j)^2, \tag{15}$$

where P_i is a GPS point. By setting $\epsilon(l) = l \times 0.005 \times (width + height), l = \{1, \cdots, 10\}$, we generate a 10-layer ROI network with CascadeSync. Fig. 6 shows the number and MSE of ROIs with different ϵ on each layer. The result manifests that the information loss is linearly proportional to ϵ. Besides, the resemblance in the shape of curves manifests the insensitivity of parameter ϵ.

4.3.2. Semantic retrieval with different embedding schemes

Given an ROI, we retrieve the top-5 most similar ROIs and compute the NCDG@5. Besides, given a trajectory, we retrieve the top-10 most similar trajectories and compare the performance with the comparison methods. Here we report

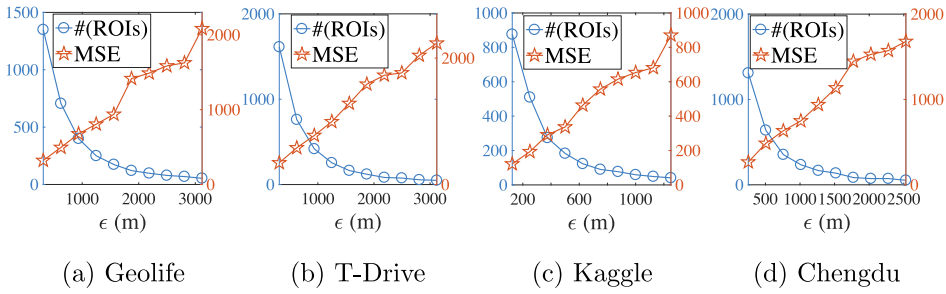

Fig. 6. Exploring ϵ's effects on four real-world datasets.

NCDG@3,5,10, respectively. Both the semantic ROI retrieval and semantic trajectory retrieval tasks are repeated 10 times for all methods, and the average value and standard deviation of the result are reported. Moreover, we set the number of each neighborhood as 3 and the negative sampling number as 5, and $\epsilon(l) = l \times 1/200 \times (width + height), l = \{1, \ldots, 5\}$ to derive a 5-layer hierarchical ROI network.

To explore the effects of different semantic information on real-world data, we vary the types of neighborhoods incorporated into embedding model in three ways: 1) Only geometric information (i.e., basic trajectory structure); 2) Geometric and basic semantics (i.e., network features and temporal information); 3) All kinds of information and domain knowledge are modeled. Moreover, we investigate the effect of feature propagation by constructing two embedding models: Embedding model trained only on the bottom layer ($\epsilon(t) = 1/200 \times (width + height)$) of hierarchical ROI network via Eq. (8) and the whole model on all five layers via Eq. (10).

For better illustration, we select an ROI that represents a hub junction in Kaggle Taxi data as a target ROI. We apply the embedding models with/without semantic neighborhoods on the aforementioned 5-layer network. The retrieval results are shown in Fig. 7. From the results, we can obtain that the model with semantic neighborhoods captures the semantic information and thus easily identifies the urban roles of regions, and retrieves hub junctions far away from querying target junction. The reason is that the semantics of traffic is reflected by the degree and the distribution of neighbors' degree of ROIs on the network. Therefore, similar regions are not necessary to be near to each other.

The results of semantic ROI retrieval and semantic trajectory retrieval are summarized in Tables 3 and 4, respectively, from which we can gain several insights as follows:

First, both of the tables show that the retrieval performances on the hierarchical ROI embedding network are superior to the one-layer embedding network on the two datasets Geolife and Kaggle; however, the retrieval results on the other two datasets, i.e., T-Drive and Chengdu, show that one-layer network is better than the hierarchical network. It can be explained by the fact that the POI points in T-Drive and Chengdu are far more sparse than those in Geolife and Kaggle, so more errors exist in the hierarchical ROI network, which adversely affects the feature propagation on different layers (Eq. (9)). The retrieval performances, moreover, are degenerated. Even so, the performances of the proposed method are superior to all the other state-of-the-art similarity-based methods.

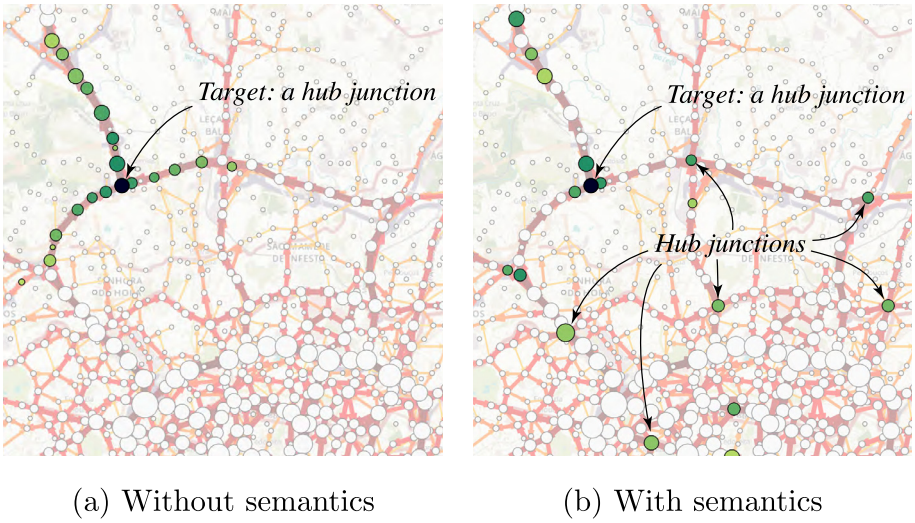

Fig. 7. Illustration of ROI retrieval on the Kaggle Taxi data.

Table 3
NDCG@5 results of ROI retrieval. Model 1 only considers geometric neighborhoods. Model 2 considers geometric and basic semantic neighborhoods. Model 3 is trained with domain knowledge. (Bold: Best; underline: runner-up).

DataSets	Structure	Random	Model 1	Model 2	Model 3
Geolife	One-layer	0.429 ± 0.24	0.526 ± 0.18	0.560 ± 0.15	0.713 ± 0.18
	Hierarchical		0.581 ± 0.15	0.615 ± 0.13	**0.724 ± 0.17**
T-Drive	One-layer	0.382 ± 0.23	0.632 ± 0.21	0.661 ± 0.18	**0.713 ± 0.21**
	Hierarchical		0.650 ± 0.24	0.655 ± 0.23	0.693 ± 0.18
Kaggle	One-layer	0.410 ± 0.28	0.524 ± 0.20	0.542 ± 0.22	0.679 ± 0.19
	Hierarchical		0.524 ± 0.22	0.567 ± 0.21	**0.691 ± 0.16**
Chengdu	One-layer	0.415 ± 0.25	0.666 ± 0.20	0.677 ± 0.21	**0.817 ± 0.17**
	Hierarchical		0.657 ± 0.23	0.665 ± 0.20	0.802 ± 0.16

Second, in terms of the semantic embedding models, the more semantic neighborhoods are considered, the better the performances of semantic retrieval are. Therefore, in Table 3, the performances are increasing on Model 1, 2, and 3; in Table 4, the retrieval performances of embedding model with domain knowledge are better than those without domain knowledge, regardless of whether it is the one-layer or multi-layer embedding model. When embedding the model trained with domain

Table 4
Evaluation results of semantic trajectory retrieval on four datasets. The abbreviation **D.K.** denotes domain knowledge. (Bold: Best; underline: runner-up).

Methods	D.K.	Metrics	Geolife	T-Drive	Kaggle	Chengdu
DTW		NDCG@3	0.391 ± 0.198	0.416 ± 0.201	0.311 ± 0.223	0.417 ± 0.168
		NDCG@5	0.410 ± 0.185	0.430 ± 0.168	0.321 ± 0.215	0.442 ± 0.153
		NDCG@10	0.521 ± 0.134	0.443 ± 0.149	0.342 ± 0.207	0.457 ± 0.145
Fréchet		NDCG@3	0.328 ± 0.179	0.304 ± 0.191	0.250 ± 0.175	0.360 ± 0.239
		NDCG@5	0.393 ± 0.178	0.346 ± 0.156	0.288 ± 0.151	0.388 ± 0.229
		NDCG@10	0.439 ± 0.165	0.371 ± 0.141	0.300 ± 0.148	0.417 ± 0.211
LCSS		NDCG@3	0.360 ± 0.258	0.396 ± 0.241	0.349 ± 0.227	0.314 ± 0.241
		NDCG@5	0.368 ± 0.224	0.402 ± 0.211	0.375 ± 0.218	0.346 ± 0.221
		NDCG@10	0.379 ± 0.184	0.402 ± 0.180	0.382 ± 0.214	0.376 ± 0.209
EDR		NDCG@3	0.340 ± 0.189	0.358 ± 0.195	0.290 ± 0.182	0.416 ± 0.196
		NDCG@5	0.411 ± 0.171	0.363 ± 0.184	0.304 ± 0.177	0.424 ± 0.171
		NDCG@10	0.467 ± 0.171	0.379 ± 0.161	0.308 ± 0.162	0.449 ± 0.179
DeepWalk		NDCG@3	0.244 ± 0.173	0.309 ± 0.159	0.263 ± 0.168	0.340 ± 0.156
		NDCG@5	0.278 ± 0.190	0.330 ± 0.183	0.257 ± 0.157	0.377 ± 0.167
		NDCG@10	0.301 ± 0.187	0.351 ± 0.173	0.281 ± 0.175	0.385 ± 0.164
BGEM		NDCG@3	0.302 ± 0.121	0.376 ± 0.155	0.249 ± 0.196	0.343 ± 0.182
		NDCG@5	0.324 ± 0.134	0.374 ± 0.132	0.286 ± 0.128	0.367 ± 0.167
		NDCG@10	0.330 ± 0.129	0.393 ± 0.137	0.298 ± 0.170	0.371 ± 0.172
Metapath2Vec		NDCG@3	0.248 ± 0.168	0.294 ± 0.203	0.252 ± 0.185	0.361 ± 0.215
		NDCG@5	0.263 ± 0.185	0.328 ± 0.192	0.262 ± 0.191	0.365 ± 0.218
		NDCG@10	0.309 ± 0.192	0.326 ± 0.198	0.312 ± 0.216	0.392 ± 0.206
Node2Vec		NDCG@3	0.267 ± 0.184	0.301 ± 0.191	0.259 ± 0.196	0.382 ± 0.186
		NDCG@5	0.282 ± 0.191	0.295 ± 0.161	0.263 ± 0.162	0.409 ± 0.194
		NDCG@10	0.289 ± 0.188	0.312 ± 0.178	0.287 ± 0.180	0.417 ± 0.185
LINE		NDCG@3	0.377 ± 0.209	0.411 ± 0.183	0.302 ± 0.187	0.404 ± 0.195
		NDCG@5	0.382 ± 0.203	0.423 ± 0.172	0.334 ± 0.207	0.419 ± 0.201
		NDCG@10	0.394 ± 0.225	0.419 ± 0.185	0.338 ± 0.193	0.421 ± 0.198
One-layer model		NDCG@3	0.468 ± 0.177	0.529 ± 0.169	0.406 ± 0.185	0.588 ± 0.159
		NDCG@5	0.483 ± 0.165	0.545 ± 0.144	0.413 ± 0.160	0.622 ± 0.124
		NDCG@10	0.520 ± 0.156	0.544 ± 0.139	0.430 ± 0.150	0.651 ± 0.112
	✓	NDCG@3	0.607 ± 0.152	**0.707 ± 0.179**	0.593 ± 0.164	**0.834 ± 0.143**
		NDCG@5	0.639 ± 0.120	**0.711 ± 0.172**	0.604 ± 0.150	**0.859 ± 0.139**
		NDCG@10	0.657 ± 0.109	**0.734 ± 0.168**	0.607 ± 0.157	**0.866 ± 0.113**
Hierarchical model		NDCG@3	0.545 ± 0.171	0.547 ± 0.182	0.438 ± 0.183	0.611 ± 0.160
		NDCG@5	0.552 ± 0.162	0.564 ± 0.169	0.445 ± 0.151	0.635 ± 0.139
		NDCG@10	0.558 ± 0.148	0.599 ± 0.155	0.452 ± 0.145	0.666 ± 0.114
	✓	NDCG@3	**0.663 ± 0.148**	0.697 ± 0.184	**0.600 ± 0.209**	0.833 ± 0.120
		NDCG@5	**0.676 ± 0.104**	0.702 ± 0.183	**0.612 ± 0.183**	0.849 ± 0.112
		NDCG@10	**0.689 ± 0.094**	0.721 ± 0.175	**0.626 ± 0.188**	0.862 ± 0.096

knowledge, we found that the performance relies mainly on label quality and are generally better than those without domain knowledge. This insight is straightforward and suggests that one should pay more attention to collecting more domain knowledge and thus integrating it into the representation features. Besides, because our experiments are evaluated in term of semantic similarity, the classical models (DTW, LCS, Fréchet distance, and EDR) which focus on computing the geometric distance and the state-of-the-art deep network embedding methods (DeepWalk, BGEM, Node2Vec, Metapath2Vec, and LINE) which are designed for exploiting the network properties cannot perform as well as our model.

5. Conclusion

In this paper, we propose a new semantic trajectory representation. To start with, we extract the trajectory structure globally and represent the trajectory dataset as a multi-resolution ROI network via the synchronization-based clustering model: CascadeSync. Relying on the network, by extracting the geometric, semantic, temporal information as well as additional domain knowledge as context information, we embed ROIs and trajectories as continuous vectors in the semantic space via distributed vector representation, whose metric is tailored for measuring the semantic similarity. Building upon the derived vectors, semantic retrieval tasks can be effectively and efficiently performed by computing the Euclidean distance of embedding vectors directly. Empirical experiments on both synthetic and real-world datasets indicate that our proposed approach allows extracting semantic information effectively, and it outperforms classical metrics methods and state-of-the-art deep network embedding models on the semantic trajectory retrieval tasks.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Chongming Gao: Writing - original draft, Conceptualization, Methodology, Investigation, Software. **Zhong Zhang:** Visualization, Software. **Chen Huang:** Visualization, Software. **Hongzhi Yin:** Writing - review & editing. **Qinli Yang:** Writing - review & editing. **Junming Shao:** Writing - review & editing, Conceptualization, Methodology, Supervision.

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (ZYGX2019Z014), National Natural Science Foundation of China (61976044), Fok Ying-Tong Education Foundation for Young Teachers in the Higher Education Institutions of China (161062), National Key Research and Development Program (2016YFB0502300), The Belt and Road Fund on Water and Sustainability of the State Key Laboratory of Hydrology-Water Resources and Hydraulic Engineering (2019nkzd02), State Key Laboratory of Simulation and Regulation of Water Cycle in River Basin (IWHR-SKL-201911).

References

[1] J.D. Mazimpaka, S. Timpf, Trajectory data mining: a review of methods and applications, vol. 13, 2016, pp. 61–99, doi: 10.5311/JOSIS.2016.13.263, URL: https://doi.org/10.5311/JOSIS.2016.13.263.
[2] V. Bogorny, B. Kuijpers, L.O. Alvares, ST-DMQL: a semantic trajectory data mining query language, Int. J. Geogr. Inf. Sci. 23 (10) (2009) 1245–1276, URL: http://www.informaworld.com/smpp/content%7Edb=all%7Econtent=a915093101%7Efrm=abslink.
[3] N.J. Yuan, Y. Zheng, L. Zhang, X. Xie, T-finder: a recommender system for finding passengers and vacant taxis, IEEE Trans. Knowl. Data Eng. (25(10), 2013,) 2390–2403, https://doi.org/10.1109/TKDE.2012.153, URL: https://doi.org/10.1109/TKDE.2012.153.
[4] A. Anagnostopoulos, R. Atassi, L. Becchetti, A. Fazzone, F. Silvestri, Tour recommendation for groups, Data Min. Knowl. Discov. 31 (5) (2017) 1157–1188, https://doi.org/10.1007/s10618-016-0477-7, URL: https://doi.org/10.1007/s10618-016-0477-7.
[5] J.-G. Lee, J. Han, K.-Y. Whang, Trajectory clustering: A partition-and-group framework, in: Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, SIGMOD '07, ACM, New York, NY, USA, 2007, pp. 593–604, doi: 10.1145/1247480.1247546. URL: http://doi.acm.org/10.1145/1247480.1247546.
[6] W.-C. Lee, J. Krumm, in: Computing with Spatial Trajectories, 2011, pp. 3–33, doi: 10.1007/978-1-4614-1629-6_1, URL:https://link.springer.com/chapter/10.1007%2F978-1-4614-1629-6_1.
[7] R. Annoni, C.H.Q. Forster, Analysis of aircraft trajectories using fourier descriptors and kernel density estimation, in: 2012 15th International IEEE Conference on Intelligent Transportation Systems, 2012, pp. 1441–1446, https://doi.org/10.1109/ITSC.2012.6338863, URL:https://ieeexplore.ieee.org/document/6338863.
[8] I.S. Ardakani, K. Hashimoto, Encoding bird's trajectory using recurrent neural networks, 2017 IEEE International Conference on Mechatronics and Automation (ICMA) (2017) 1644–1649, https://doi.org/10.1109/ICMA.2017.8016063.
[9] B.-K. Yi, H.V. Jagadish, C. Faloutsos, Efficient retrieval of similar time sequences under time warping, in: Proceedings of the Fourteenth International Conference on Data Engineering, ICDE '98, IEEE Computer Society, Washington, DC, USA, 1998, pp. 201–208 URL:http://dl.acm.org/citation.cfm?id=645483.653609.
[10] M. Vlachos, D. Gunopoulos, G. Kollios, Discovering similar multidimensional trajectories, in: Proceedings of the 18th International Conference on Data Engineering, ICDE '02, IEEE Computer Society, Washington, DC, USA, 2002, pp. 673–684, URL:http://dl.acm.org/citation.cfm?id=876875.878994.
[11] L. Chen, M.T. Özsu, V. Oria, Robust and fast similarity search for moving object trajectories, in: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD '05, ACM, New York, NY, USA, 2005, pp. 491–502, doi: 10.1145/1066157.1066213, URL: http://doi.acm.org/10.1145/1066157.1066213.

[12] T. Eiter, H. Mannila, Computing discrete fréchet distance, Tech. rep., 1994.
[13] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, W.-Y. Ma, Mining user similarity based on location history, in: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '08, ACM, New York, NY, USA, 2008, pp. 34:1–34:10, doi: 10.1145/1463434.1463477, URL: http://doi.acm.org/10.1145/1463434.1463477.
[14] Z. Li, B. Ding, J. Han, R. Kays, P. Nye, Mining periodic behaviors for moving objects, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '10, ACM, New York, NY, USA, 2010, pp. 1099–1108, doi: 10.1145/1835804.1835942. URL: http://doi.acm.org/10.1145/1835804.1835942.
[15] Y. Zheng, L. Liu, L. Wang, X. Xie, Learning transportation mode from raw gps data for geographic applications on the web, in: Proceedings of the 17th International Conference on World Wide Web, WWW '08, ACM, New York, NY, USA, 2008, pp. 247–256, doi: 10.1145/1367497.1367532, URL: http://doi.acm.org/10.1145/1367497.1367532.
[16] G. Kellaris, N. Pelekis, Y. Theodoridis, Trajectory compression under network constraints, in: Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases, SSTD '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 392–398, doi: 10.1007/978-3-642-02982-0_27, URL: https://doi.org/10.1007/978-3-642-02982-0_27.
[17] R. Song, W. Sun, B. Zheng, Y. Zheng, Press: A novel framework of trajectory compression in road networks, Proc. VLDB Endow. 7 (9) (2014) 661–672, https://doi.org/10.14778/2732939.2732940, URL https://doi.org/10.14778/2732939.2732940.
[18] M. Gariel, A.N. Srivastava, E. Feron, Trajectory clustering and an application to airspace monitoring, in: Transactions on Intelligent Transportation Systems, vol. 12, 2011, pp. 1511–1524, doi: 10.1109/TITS.2011.2160628, URL:https://ieeexplore.ieee.org/abstract/document/5959983/.
[19] Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, F. Zhang, Identifying human mobility via trajectory embeddings, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI'17, AAAI Press, 2017, pp. 1689–1695, URL:http://dl.acm.org/citation.cfm?id=3172077.3172122.
[20] R. Agrawal, C. Faloutsos, A.N. Swami, Efficient similarity search in sequence databases, in: Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms, FODO '93, Springer-Verlag, Berlin, Heidelberg, 1993, pp. 69–84, URL:http://dl.acm.org/citation.cfm?id=645415.652239.
[21] D. Pfoser, C.S. Jensen, Y. Theodoridis, Novel approaches in query processing for moving object trajectories, in: Proceedings of the 26th International Conference on Very Large Data Bases, VLDB '00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000, pp. 395–406.
[22] M.A. Nascimento, J.R.O. Silva, Towards historical r-trees, in: Proceedings of the 1998 ACM Symposium on Applied Computing, SAC '98, ACM, New York, NY, USA, 1998, pp. 235–240, doi: 10.1145/330560.330692, URL: http://doi.acm.org/10.1145/330560.330692.
[23] V.P. Chakka, A. Everspaugh, J.M. Patel, Indexing large trajectory data sets with SETI, in: CIDR 2003, First Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 5–8, 2003, Online Proceedings, 2003, URL:http://www-db.cs.wisc.edu/cidr/cidr2003/program/p15.pdf.
[24] Y. Bengio, R. Ducharme, P. Vincent, C. Janvin, A neural probabilistic language model, J. Mach. Learn. Res. 3 (2003) 1137–1155, URL:http://dl.acm.org/citation.cfm?id=944919.944966.
[25] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Proceedings of the 26th International Conference on Neural Information Processing Systems, vol. 2, NIPS'13, Curran Associates Inc., USA, 2013, pp. 3111–3119, URL: http://dl.acm.org/citation.cfm?id=2999792.2999959.
[26] J. Mitchell, M. Lapata, Composition in distributional models of semantics, Cogn. Sci. 34 (8) (2010) 1388–1429, https://doi.org/10.1111/j.1551-6709.2010.01106.x, URL: https://doi.org/10.1111/j.1551-6709.2010.01106.x.
[27] M. Iyyer, V. Manjunatha, J.L. Boyd-Graber, H.D. III, Deep unordered composition rivals syntactic methods for text classification, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26–31, 2015, Beijing, China, Long Papers, vol. 1, 2015, pp. 1681–1691, URL:https://www.aclweb.org/anthology/P15-1162/.
[28] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network for modelling sentences, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22–27, 2014, Baltimore, MD, USA, Long Papers, vol. 1, 2014, pp. 655–665, URL:https://www.aclweb.org/anthology/P14-1062/.
[29] K.S. Tai, R. Socher, C.D. Manning, Improved semantic representations from tree-structured long short-term memory networks, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26–31, 2015, Beijing, China, Long Papers, vol. 1, 2015, pp. 1556–1566, URL:https://www.aclweb.org/anthology/P15-1150/.
[30] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, Association for Computing Machinery, New York, NY, USA, 2014, pp. 701–710, https://doi.org/10.1145/2623330.2623732.
[31] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, in: arXiv preprint arXiv:1301.3781, 2013.
[32] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, WWW '15, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2015, p. 1067–1077, doi: 10.1145/2736277.2741093.
[33] A. Grover, J. Leskovec, 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, NY, USA, 2016, pp. 855–864, https://doi.org/10.1145/2939672.2939754, URL: https://doi.org/10.1145/2939672.2939754.
[34] H. Yin, L. Zou, Q.V.H. Nguyen, Z. Huang, X. Zhou, Joint event-partner recommendation in event-based social networks, in: 2018 IEEE 34th International Conference on Data Engineering (ICDE), 2018, pp. 929–940, https://doi.org/10.1109/ICDE.2018.00088.
[35] Y. Dong, N.V. Chawla, A. Swami, Metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17, ACM, New York, NY, USA, 2017, pp. 135–144, doi: 10.1145/3097983.3098036, URL: http://doi.acm.org/10.1145/3097983.3098036.
[36] J. Yu, M. Gao, J. Li, H. Yin, H. Liu, Adaptive implicit friends identification over heterogeneous network for social recommendation, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM '18, ACM, New York, NY, USA, 2018, pp. 357–366, doi: 10.1145/3269206.3271725, URL: http://doi.acm.org/10.1145/3269206.3271725.
[37] C. Böhm, C. Plant, J. Shao, Q. Yang, Clustering by synchronization, in: KDD'10, ACM, New York, NY, USA, 2010, pp. 583–592, doi: 10.1145/1835804.1835879, URL: http://doi.acm.org/10.1145/1835804.1835879.
[38] J. Shao, X. He, C. Bohm, Q. Yang, C. Plant, Synchronization-inspired partitioning and hierarchical clustering, IEEE Trans. Knowl. Data Eng. 25 (4) (2013) 893–905, doi: 10.1109/TKDE.2012.32, URL: https://doi.org/10.1109/TKDE.2012.32.
[39] C.S. Kim, C.S. Bae, H.J. Tcha, A phase synchronization clustering algorithm for identifying interesting groups of genes from cell cycle expression data, vol. 9, 2008, doi: 10.1186/1471-2105-9-56.
[40] P. Seliger, S.C. Young, L.S. Tsimring, Plasticity and learning in a network of coupled phase oscillators, Vol. 65, 2002, p. 041906.
[41] J. Shao, Z. Han, Q. Yang, T. Zhou, Community detection based on distance dynamics, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '15, ACM, New York, NY, USA, 2015, pp. 1075–1084, doi: 10.1145/2783258.2783301, URL: http://doi.acm.org/10.1145/2783258.2783301.
[42] J. Shao, C. Gao, W. Zeng, J. Song, Q. Yang, Synchronization-inspired co-clustering and its application to gene expression data, in: 2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18–21, 2017, 2017, pp. 1075–1080, doi: 10.1109/ICDM.2017.141.
[43] Z. Zhang, D. Kang, C. Gao, J. Shao, Semisync: Semi-supervised clustering by synchronization, in: International Conference on Database Systems for Advanced Applications, Springer, 2019, pp. 358–362, doi: 10.1007/978-3-030-18590-9_45, URL: https://doi.org/10.1007/978-3-030-18590-9_45.

[44] C. Gao, Y. Zhao, R. Wu, Q. Yang, J. Shao, Semantic trajectory compression via multi-resolution synchronization-based clustering, Knowl.-Based Syst. 174 (2019) 177–193, https://doi.org/10.1016/j.knosys.2019.03.006, URL: https://doi.org/10.1016/j.knosys.2019.03.006.
[45] W. Blacoe, M. Lapata, A comparison of vector-based representations for semantic composition, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '12, Association for Computational Linguistics, Stroudsburg, PA, USA, 2012, pp. 546–556, URL: http://dl.acm.org/citation.cfm?id=2390948.2391011.
[46] J. Wieting, M. Bansal, K. Gimpel, K. Livescu, Towards universal paraphrastic sentence embeddings, in: 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, 2016, URL: http://arxiv.org/abs/1511.08198.
[47] L.E. Kavraki, P. Svestka, J. Latombe, M.H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robot. Automat. 12 (4) (1996) 566–580, https://doi.org/10.1109/70.508439, URL: https://doi.org/10.1109/70.508439.
[48] Y. Zheng, L. Zhang, X. Xie, W.-Y. Ma, Mining interesting locations and travel sequences from gps trajectories, in: Proceedings of the 18th International Conference on World Wide Web, WWW '09, ACM, New York, NY, USA, 2009, pp. 791–800, doi: 10.1145/1526709.1526816, URL: http://doi.acm.org/10.1145/1526709.1526816.
[49] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, Y. Huang, T-drive: Driving directions based on taxi trajectories, in: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '10, ACM, New York, NY, USA, 2010, pp. 99–108, doi: 10.1145/1869790.1869807, URL: http://doi.acm.org/10.1145/1869790.1869807.