A Deep Spatiotemporal Trajectory Representation Learning Framework for Clustering

Chao Wang, Jiahui Huang, Yongheng Wang, Zhengxuan Lin, Xiongnan Jin, Xing Jin, Di Weng, Yingcai Wu

***Abstract*—Learning trajectory representations is essential in many Location Based Services (LBS) applications. Most traditional methods extract trajectory representations based on manually defined features, while deep learning-based methods can reduce part of the human effort. We propose a Deep Spatiotemporal Trajectory Clustering (DSTC) framework to tackle the Spatiotemporal Trajectory Representation Learning towards the Clustering-friendly space (STRLC) problem. Solving the STRLC problem is not a trivial task because: (1) Defining a uniform token size for datasets with an uneven density of trajectory data is challenging. (2) Measuring the similarity between trajectories spanning time zero in the time dimension is a problem to be solved. (3) It requires first learning a vector that can represent the overall characteristics of spatiotemporal trajectories and then mapping it to a more suitable space for clustering. To tackle these challenges, we first utilize the density-based clustering method to define tokens representing the trajectory points automatically. Then, we use polar coordinates to represent the temporal dimension of trajectories. Additionally, we improve the learned trajectory representations in a clustering-oriented latent space end to end. Experiments conducted on benchmark datasets demonstrate that DSTC achieves better accuracy than existing methods. Moreover, the representations learned from spatiotemporal trajectory data in the real world can be used to identify popular routes during the day.**

***Index Terms*—trajectory clustering, representation learning, trajectory feature, trajectory data mining, hot-routes detection.**

I. INTRODUCTION

WITH the rapid development of location-acquisition technologies like Global Positioning System (GPS) and wireless mobile devices, spatiotemporal trajectory data has become easily collected [1]–[3]. Trajectory data is widely used in many Location Based Services (LBS) applications, such as understanding human trajectory patterns, preference-based route planning, and traffic management [4]–[6]. In recent years, mining travel behavior characteristics of residents from massive trajectory data have become popular, and some related studies in Nature and Science journals have stimulated research enthusiasm in academics and industry [7], [8].

Trajectory clustering is a primary data-mining method that aims to divide a set of trajectory objects into multiple clusters so that objects within a cluster are highly similar to one another but are dissimilar to objects in other clusters [9], [10]. Trajectory clustering is the initial and vital step to reveal the travel behavior patterns of city residents. It proves highly beneficial in comprehending the spatial distribution and temporal features of moving patterns in various LBS application scenarios. For example, clustering algorithms can be used in analyzing hot routes to identify frequently occurring routes as research objects [11]. Suppose there are several alternative routes between two locations, and people tend to select specific routes in the morning and others during noon. In that case, these routes appearing at different times can reveal unique human movement patterns.

Most trajectory clustering methods rely on point-matching similarity measures to compare trajectories. A common practice is using similarity measures such as Longest Common SubSequences (LCSS) [12], Edit Distance on Real sequence (EDR) [13], and Dynamic Time Warping (DTW) [14] followed by clustering algorithms like K-Means [15]. With the recent advances in deep learning, it has become possible to learn feature representations for complex sequence data, making it well-suited for analyzing and learning latent representations of sequential trajectory data. Unfortunately, extracting features from trajectories that incorporate both temporal and spatial dimensions, and optimizing such features to be more suitable for clustering tasks, has received little attention in the literature, leaving room for improvement in accuracy and efficiency in trajectory clustering tasks. This lack of attention is the motivation behind our study on Spatiotemporal Trajectory Representation Learning towards the Clustering-friendly space (STRLC) problem in order to improve the accuracy of trajectory clustering.

In addition, conventional trajectory clustering methods primarily focus on measuring trajectories in two-dimensional space composed of longitude and latitude. Some studies segment trajectory datasets into several groups based on time intervals and mining human trajectory patterns separately. However, a trajectory is not solely a sequence of GPS records with location attributes and timestamps; it also holds an absolute value in the temporal dimension. Obtaining prior knowledge to direct us on partitioning the dataset is challenging. As shown in Fig. 1, there are four trajectories spanning different periods. Dividing them by time intervals may result in some long trajectories being segmented into several short

The work was supported by National Key R&D Program of China (2022YFE0137800) and NSFC (42201510, U22A2032) and Zhejiang Provincial Natural Science Foundation of China (LTGG23F030004). (Corresponding author: Yingcai Wu)

Chao Wang, Yongheng Wang, and Xiongnan Jin are with the Big Data Intelligence Research Center, Zhejiang Lab, Hangzhou, China, 311100 (e-mail: wangc@zhejianglab.com, wangyh@zhejianglab.com, and xiongnan.jin@zhejianglab.com).

Jiahui Huang, Zhengxuan Lin, Di Weng and Yingcai Wu are with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China, 310058 (e-mail:huangjh@zju.edu.cn, 22151123@zju.edu.cn, dweng@zju.edu.cn, ycwu@zju.edu.cn).

Xing Jin is with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China, 310018 (e-mail: jinxing@hdu.edu.cn).

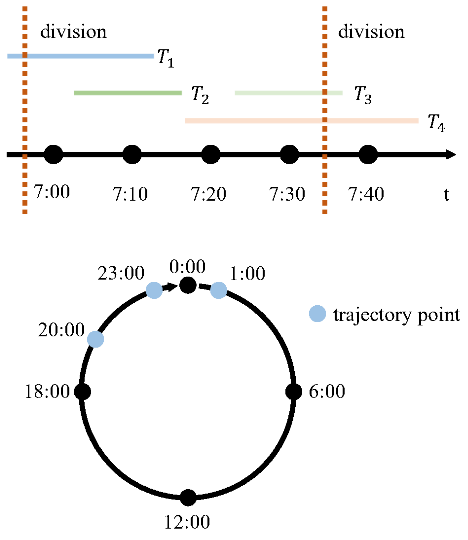


Fig. 1. Mapping trajectories in the time dimension.

trajectories. Also, time dimension is cyclical. Time of 23:00 has a stronger resemblance to 1:00 than 20:00. How to represent time dimension with real numbers to maintain this cyclicality is also a problem to be solved.

Besides, existing deep learning-based trajectory feature extraction methods usually divide the study area into grids and map trajectory points to the grids. However, real-life trajectory datasets often have different densities in different regions, making choosing an appropriate grid size difficult. As shown in Fig. 2(a), areas with a dense distribution of trajectory points may require finer grids, while larger grids can accommodate areas with sparser trajectories.

Overall, solving the STRLC problem is not a trivial task, as there are several challenges involved: (1) Defining token size without prior knowledge is difficult. As trajectory points in real-world data are unevenly distributed in space, it is difficult to define a suitable token size that applies to the entire dataset. (2) It is hard to define spatiotemporal distance between trajectories due to their temporal nature and occurrence at specific times during the day. How to represent the time dimension with real numbers to preserve this periodicity and further measure the similarity of trajectory data that crosses zero time is a problem. (3) Designing a unified framework for learning trajectory representations, mapping them to a space more suitable for clustering, and obtaining clustering results end to end is challenging. It requires ensuring that the trajectory data is distributed evenly around the cluster centers to achieve higher clustering quality.

To overcome these challenges, we propose a novel Deep Spatiotemporal Trajectory Clustering (DSTC) framework for identifying trajectory mobility patterns. We first extract the representation of trajectories in a three-dimensional spatiotemporal space and then improve it by mapping it to a clustering-friendly latent space. DSTC is designed to consider both temporal and spatial information of trajectories. This enables us to learn spatiotemporal trajectory data representations while obtaining end-to-end clustering results. To our knowledge, this is the first deep learning-based solution for addressing the STRLC problem. In summary, this article makes the following key contributions:

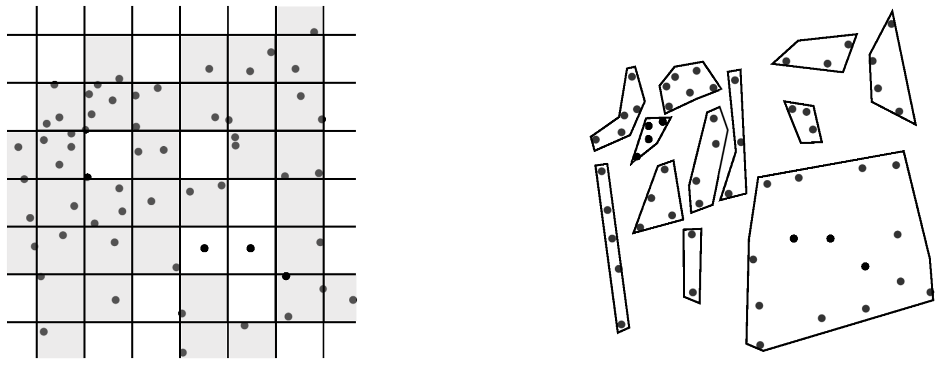
- We present a framework that can learn the representation of trajectories in a three-dimensional spatiotemporal space with two newly proposed auxiliary clustering losses. The weights for representation learning and cluster centroids are updated in the joint training stage, resulting in mapping the input trajectory data to a space suitable for clustering.
- When performing representation learning on trajectories, we consider the non-uniformity of the data density in the spatial dimension and the periodicity in the temporal dimension. Not only do we overcome the challenge of representing trajectories using tokens of a uniform size, but we also address the issue of small cubes being unable to represent the continuity of trajectories in the temporal dimension.
- We introduce two new approaches based on the DSTC framework: G-DSTC and D-DSTC, and the proposed algorithms outperform baseline methods on benchmark datasets. The spatiotemporal trajectory representations learned from real-world data can be utilized to identify popular spatiotemporal trajectory routes in a city. A public spatiotemporal trajectory dataset with ground-truth labels is also proposed, making up for the lack of temporal information in existing public spatial trajectory datasets.

The remainder of this paper is organized as follows. Section II briefly reviews previous deep spatiotemporal trajectory clustering work. Section III defines the STRLC problem. Section IV outlines the proposed framework DSTC followed by two DSTC-based methods. Section V evaluates the effectiveness of the proposed methods. Section VI summarizes the conclusions and directions for future research.

II. Related Work

The similarity comparison between trajectories is the basis of the trajectory clustering methods. In most trajectory clustering algorithms, point matching methods (such as EDR, DTC, and LCSS) are initially used to compare the similarity between trajectories based on artificially designed features (such as combining trajectory speed, direction, and position) [16], [17]. The resulting similarities are then utilized in clustering methods to group similar trajectories. The process heavily depends on prior human knowledge, and accurately measuring the similarity between trajectories can be challenging. The emergence of deep learning provides a promising avenue for enhancing the accuracy of trajectory similarity measurement and thus improving the precision of trajectory clustering.

Typically, trajectories are represented as sequences of multiple grids in two-dimensional space composed of longitude and latitude, which feature learning models can then process. These methods can be categorized into two groups: trajectory representation learning methods based on word embedding and those based on the pixel. Word embedding-based trajectory



(a) Grid-based division. (b) Density-based division.

Fig. 2. Division of trajectory points in the spatial dimension.

representation learning methods first convert the original trajectory data into a sequence of multiple words, followed by vectorizing the words through the word embedding matrix of a neural network language model. Then, deep learning models such as recurrent neural networks or autoencoders are used to learn the complete trajectory representation [18]–[21]. Pixel-based trajectory representation learning methods consider trajectory points as pixels in an image. This involves converting the trajectory data into image data and using convolutional neural networks to learn trajectory features [22], [23]. Word embedding-based trajectory representation learning methods can capture the temporal relationship between trajectory points but may overlook their positional spatial relationship. Pixel-based trajectory representation learning methods can retain the spatial positional relationship between trajectory points to a great extent but may not capture their temporal order. Additionally, determining the appropriate pixel size is challenging.

Some studies on trajectory feature learning consider the temporal dimension of trajectories. For example, some studies extract features from trajectories in space and time dimensions, then compute a weighted sum of these features to represent the trajectories [24], [25]. However, since the relative importance of time and space can differ from one data mining task to another, it can take time to determine the appropriate balance between these factors manually. Some methods partition the study area into multiple spatiotemporal cubic units and then project trajectories onto these units [26]. Nevertheless, these methods still face the challenge of determining the optimal spatiotemporal scale for the cubic units.

Learning representations of trajectories provides an opportunity to gain a deeper understanding of their inherent characteristics and to classify them more effectively. Trajectory clustering is a technique used to group a collection of trajectories into clusters of highly similar objects while dissimilar to objects in other clusters. In order to enhance the accuracy of trajectory clustering applications, it is essential to project trajectory representations into a feature space that is more suitable for clustering. Traditional clustering techniques often need help with high-dimensional data because of inefficient similarity measures and the high computational complexity associated with processing large-scale datasets. Recent advances in deep learning have enabled the optimization of high-dimensional data representations for clustering purposes, often called deep clustering. Deep clustering is an advanced clustering method that leverages the powerful representation capabilities of deep learning to enhance clustering outcomes. It requires the use of neural networks to not only learn low-dimensional representations of data suitable for clustering but also to reflect the information and structural characteristics of the original data. Motivated by this, some studies combine representation learning methods with clustering algorithms. Hierarchical clustering [27] and K-Means [28] clustering are the two most commonly chosen clustering methods because their recurrent procedures are suitable for deep learning frameworks. T2VEC [18] and ST2VEC [26] are two deep learning-based methods for learning trajectory representations. However, they do not include a feature mapping stage that would transform the original representations into a more clustering-friendly space. DTC [28] and E2DTC [19] are two deep learning-based methods that learn more clustering-friendly trajectory representations but do not consider the temporal dimension when learning these representations. Additionally, they need to pay more attention to the uneven spatial distribution of real-life trajectory data.

The studies mentioned above have established a strong foundation for learning spatiotemporal trajectory representations, but a comprehensive clustering framework for spatiotemporal trajectories has yet to be developed. None of these methods address the uneven spatial distribution of trajectory points, and the clustering loss function design needs to be more complex and diverse. Compared with the above existing related methods, the DSTC framework proposed in this article has the following differences.

- Existing methods lack a framework that can simultaneously consider the trajectory representation learning in both temporal and spatial dimensions. DSTC is a comprehensive spatiotemporal trajectory clustering framework not confined to specific trajectory feature extraction or clustering techniques. Examples of methods based on the DSTC framework proposed in this article: G-DSTC and D-DSTC, are able to extract movement patterns from three-dimensional spatiotemporal data.

- The proposed method in this article can overcome trajectory data's spatial inhomogeneity and time periodicity problems. It can use tokens of different sizes to adapt to the trajectory dataset, and also solves the problem that small cubes cannot represent the continuity of the trajectory in the time dimension.
- We propose novel loss functions, inter-cluster distance loss and neighbor loss, extending the possibilities of clustering loss functions.

III. Problem Definition

This section will define spatiotemporal trajectories and present the STRLC problem.

Definition 1. (Spatiotemporal Trajectory) The ideal spatiotemporal trajectory is a continuous curve, but the discrete points are collected in real life. Thus, a spatiotemporal trajectory data can be expressed as a location sequence $T_i = \{\tau_i, i = 1, 2, ..., n\}$. Each record τ_i is a tuple (x_i, y_i, t_i), where: t_i is the timestamp; x_i and y_i is the location of the trajectory T_i at time t_i.

Definition 2. (Spatiotemporal Trajectory Representation Learning) Spatiotemporal trajectory representation learning refers to finding a mapping function f that can convert the trajectory $T_i \in O$ in the original data space into a vector $v \in R^d$ in the d-dimensional space while preserving its spatiotemporal properties.

Definition 3. (Representation Optimization for Clustering-Friendly Spaces) With the learned spatiotemporal trajectory vectors, representation optimization using deep trajectory clustering can train a mapping from the initial representation space R^d to an optimized, clustering-friendly space R'^d. Leveraging the latter space's trajectory features can yield higher-quality clustering results.

Given a spatiotemporal trajectory dataset $D = \{T_i, 1 <= i <= n\}$, the solution to STRLC involves using deep learning techniques to learn both trajectory representations $V = \{V_i, 1 <= i <= n\}$ and cluster centroids $C = \{C_j, 1 <= j <= k\}$ simultaneously. Here, n is the number of trajectories in D, and k is the number of clusters.

Problem Statement. Given a trajectory dataset D, each trajectory comprises a sequence of points. The goal of the STRLC problem is to use a mapping function f to convert trajectories from the original data space O into low-dimensional vectors in an optimized space R' to improve clustering accuracy.

IV. The DSTC Framework

This section overviews the DSTC framework and introduces the detailed methods.

A. Overview

As depicted in Fig. 3, the DSTC framework is based on the sequence-to-sequence architecture, which consists of three core parts:

(1) Definition of the spatiotemporal token. Trajectory data is a type of time series data that contains sequential order attributes, and its features can be extracted using sequence processing methods. Therefore, the representation learning methods used in Natural Language Processing (NLP) for text can also be applied to spatiotemporal trajectory data. The original spatiotemporal trajectory data consists of a series of spatiotemporal points, which need to be converted into discrete units similar to tokens in NLP models. Note that the unit needs to consider both the spatial and temporal dimensions of the trajectory points. Then, these spatiotemporal units form a set similar to the vocabulary in NLP tasks. We describe the method for converting trajectory data points into discrete tokens in Section IV-B.

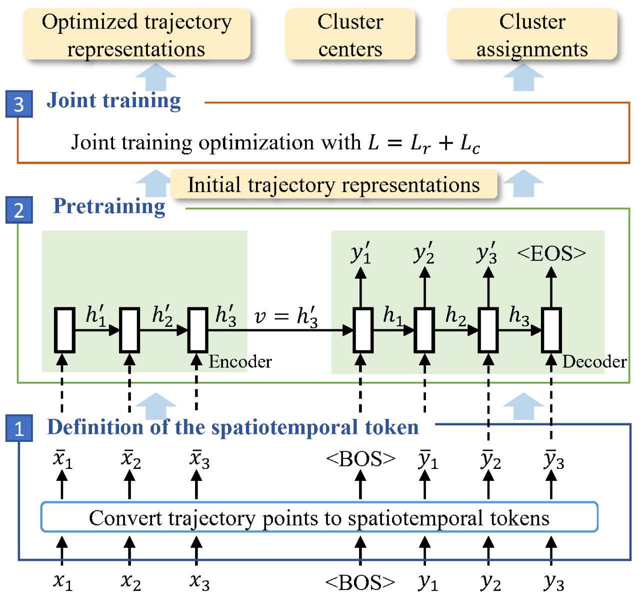

Optimized trajectory representations
Cluster centers
Cluster assignments
3 Joint training
Joint training optimization with $L = L_r + L_c$
Initial trajectory representations
2 Pretraining
y_1' y_2' y_3' <EOS>
h_1' h_2' h_3' $v = h_3'$ h_1 h_2 h_3
Encoder
Decoder
1 Definition of the spatiotemporal token
$\bar{x}_1$ $\bar{x}_2$ $\bar{x}_3$ <BOS> $\bar{y}_1$ $\bar{y}_2$ $\bar{y}_3$
Convert trajectory points to spatiotemporal tokens
x_1 x_2 x_3 <BOS> y_1 y_2 y_3

Fig. 3. Deep trajectory clustering framework. Trajectory points x_t are converted to spatiotemporal tokens $\bar{x}_t$ first and fed into an encoder-decoder sequence to sequence model to learn initial representations v in the pretraining step. Finally clustering results and optimized trajectory representations are generated by minimizing reconstruction loss L_r and clustering loss L_c.

(2) Pretraining to obtain the initial spatiotemporal trajectory representation. The pretraining module applies a deep learning approach to encode each trajectory through a trained sequence-to-sequence (Seq2Seq) [29] model. We employ it to extract an initial spatiotemporal trajectory representation that is robust and suitable for handling trajectories with non-uniform, low sampling rates and noisy points. Section IV-C introduces the details of the pretraining part.

(3) Joint training to optimize representations for improving clustering quality. The joint training module utilizes the spatiotemporal trajectory representation obtained from the pretraining module to map the initial representations into a new space that is more appropriate for the clustering task. As a result, we can obtain deep spatiotemporal trajectory representations suitable for clustering tasks, updated cluster centers, and trajectory cluster assignments. Section IV-D describes various loss functions related to clustering.

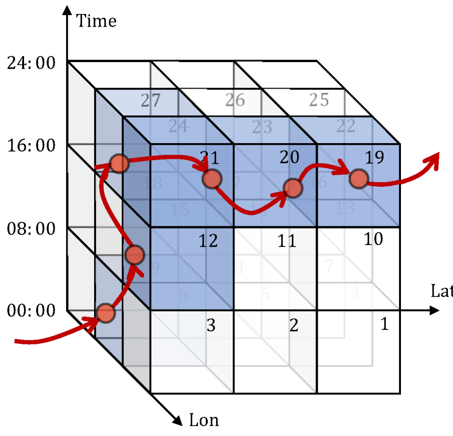


Fig. 4. Grid-based divided token representation.

B. Definition of spatiotemporal token

We aim to apply extending from "words" to "sentences" in natural language feature learning models to spatiotemporal trajectory data. Since trajectories can be considered a sequence of spatiotemporal points, treating them as tokens in the Seq2Seq model is a natural choice.

Grid-based divided token. The first method for defining tokens in spatiotemporal trajectory data is the grid-based divided token representation method, which treats the study area as a sizeable spatiotemporal cube. It divides the study area into multiple small spatiotemporal cubes called $token_{st}$. As shown in Fig. 4, the study area is divided into 27 small cubes of equal size, each denoted as $token_{st}^{i}$, where i is a unique identifier for each cube. The collection of all these small cubes can be considered a vocabulary. The three coordinate axes of each small cube represent the longitude, latitude, and time dimensions, respectively. For example, the trajectory point sequence s in Fig. 4 passes through six small cubic units, which can be expressed as $s = \{token_{st}^{6}, token_{st}^{12}, token_{st}^{24}, token_{st}^{21}, token_{st}^{20}, token_{st}^{19}\}$.

Although this method is easy to understand and simple to implement, it requires determining the size of the small cube units in advance. If the size of units is too large, the trajectory will be represented as a sequence of tokens with lower spatial resolution. On the other hand, if the size of units is too small, the study area will be divided into too many tokens, resulting in high training complexity. In real life, the spatial distribution density of trajectory data is often uneven, making it difficult to determine an appropriate size for the small cube units. For instance, human activities are typically more frequent in commercial and residential areas of a city, resulting in denser distribution of trajectory points in these areas. In contrast, the distribution of trajectory points in suburbs is relatively sparse. To address this issue, we introduce density-based divided tokens.

Density-based divided token. The second approach to defining tokens in spatiotemporal trajectory data considers the inhomogeneity of the spatial distribution of trajectory data density. We utilize STME [9], a density-based clustering method, to group trajectory points in spatial dimensions. This allows us to obtain clusters of varying densities and arbitrary shapes. As shown in Fig. 2(a), the distribution of trajectory points in the business district in the upper left corner is relatively dense, while the distribution of trajectory points in the lower right corner is comparatively sparse. It indicates that it may be challenging to determine the optimal grid size when using a grid-based divided token representation method. Fig. 2(b) displays the clustering results of trajectory points in the spatial dimension using the density-based divided token definition method. The clusters are recorded as $C = \{C_K, i = 1, 2, ..., K\}$, where K is the number of clusters. The minimum convex hull of points in a cluster C_i represents the spatial information of $token_{st}$. This approach divides regions with dense trajectory points into smaller and refined tokens, while areas with sparse trajectory points can be divided into larger tokens.

The periodic representation of the time dimension. Considering that adjacent time intervals are more likely to exhibit similar traffic patterns in the transportation field, grid-based methods may have difficulty accurately measuring the similarity of trajectory points across time zero. Therefore, we use polar coordinates to represent the temporal dimension of trajectories. As depicted in Fig. 5(a), we construct a circle C centered on $(0, 0)$, where $C \subset R^2$. Next, $(Rcos\theta, Rsin\theta)$ can be used to represent the time dimension of the trajectory, where R takes a constant of 1, and θ can be expressed as:

$$\theta = \frac{\text{floor}(t/segment)}{T/segment} 2\pi \tag{1}$$

where t represents the time when a trajectory point appears, $segment$ indicates the time interval into which the overall period is divided, and T denotes the total duration of the period. Let us assume the day is divided into 12 parts, with T being 24 hours and $segment$ being 2 hours. When t is 5 hours, the radian θ of the trajectory point is $\frac{\pi}{3}$. As shown in Fig. 5(b), this approach can also be extended to other periods, such as weeks.

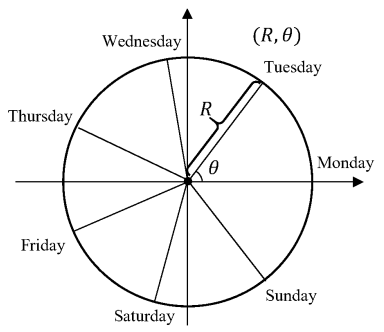
C. Pretraining

Using the defined spatiotemporal tokens, the DSTC employs an encoder-decoder Seq2Seq model to learn a feature representation of the entire spatiotemporal trajectory data.

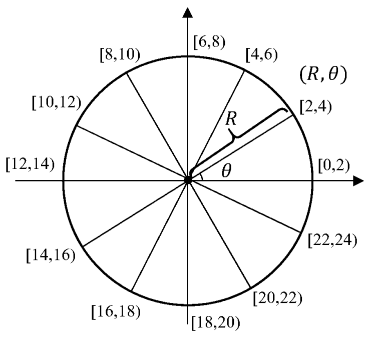
Seq2Seq model. Our Seq2Seq model is composed of an encoder and a decoder. The encoder transforms trajectories into low-dimensional trajectory features while the decoder reconstructs them back into the original trajectory data space. As illustrated in part 2 of Fig. 3, the encoder reads an input sequence sequentially and updates the hidden state accordingly. This can be formulated as follows:

$$h_t^{'} = \begin{cases} \text{f}_{encoder}(0, \bar{x}_t) & t = 1 \\ \text{f}_{encoder}(h_{t-1}^{'}, \bar{x}_t) & t \geq 2 \end{cases} \tag{2}$$

where the hidden state $h_t^{'}$ at each time is jointly determined by the hidden state $h_{t-1}^{'}$ at the previous time and the input $\bar{x}_t$ at current time. The final hidden state v produced by the encoder is a fixed-dimensional vector that can represent the original data.



(a) Time period division in weeks.



(b) Time period division in days.

Fig. 5. Time period division.

The goal of the decoder is to predict the next symbol $y_t^{'}$, and the hidden state h_t can be formulated as

$$h_t = \begin{cases} \mathrm{f}_{decoder}(v, \mathrm{BOS}) & t = 1 \\ \mathrm{f}_{decoder}(h_{t-1}, \bar{y}_{t-1}) & t \geq 2 \end{cases} \tag{3}$$

where the initial hidden state is mainly determined by the output vector v, and an BOS is a signal of the begin of a sequence. Then, the hidden state h_t is jointly determined by the hidden state h_{t-1} and output $\bar{y}_{t-1}$ at time $t-1$.

Learning trajectory deep representations. We utilize the Seq2Seq model and maximize the probability of learning trajectory representations in the latent space based on the T2VEC [18] method. To generate a distorted trajectory x with a lower sampling rate, we take the original trajectory in the dataset as y and apply certain distortions, such as adding noise or randomly discarding some sampling points. As we aim to recover high-sampling trajectories from the learned representations of low-sampling trajectories, the objective of the pretraining stage is to maximize the conditional probability $P(y|x)$. That is, to make x as close as possible to the original trajectory y after encoding and decoding by the autoencoder. Negative log likelihood loss [30] is used to train the network, as shown in Eq. 4.

$$L_r = -\log P(y|x) \tag{4}$$

Teacher forcing technique [31] is also utilized to speed up the training, thus the input of the decoder is y_t, not the last output of RNN $y_{t-1}^{'}$, as shown in Fig. 3.

By pretraining the network in the above manner, we can obtain the initial feature representation vector of the spatiotemporal trajectory data and the weight of the Seq2Seq model.

D. Joint training with clustering

The quality of data representation significantly impacts the performance of clustering. We adopt a joint training approach that combines the clustering process and trajectory representation learning to achieve higher-quality clustering results. This is done to transform the data into more clustering-friendly representations instead of directly using the neural network-learned representations for clustering.

When combining representation learning and clustering processes, their respective loss functions are combined as follows:

$$L = \lambda L_r + (1-\lambda) L_c \tag{5}$$

where L_r refers to the reconstruction loss function of the neural network that learns trajectory representations, L_c refers to the clustering loss and $\lambda \in [0, 1]$ is a hyper-parameter that balances L_r and L_c.

Generally, clustering loss can be summarized into two types: principal clustering loss and auxiliary clustering loss [32]. The principal clustering loss involves cluster centroids and cluster assignments, allowing for direct results following training [33]–[35]. In contrast, the auxiliary clustering loss does not directly output clustering results, that is, clustering methods must be run after network training to obtain clustering results [36], [37]. Auxiliary clustering loss is mainly used for guiding the network to learn a more feasible representation for clustering.

We use two main clustering losses, namely K-Means loss and soft cluster assignment loss, and propose two auxiliary clustering losses, namely inter-cluster distance loss and neighbor loss.

K-Means loss. The purpose of K-Means loss is to evenly distribute data samples around the cluster centers [38], defined as :

$$L_k(\theta) = \sum_{i=1}^{N} \sum_{k=1}^{K} s_{ik} \parallel z_i - \mu_k \parallel^2 \tag{6}$$

where z_i is the embedded data sample, μ_k is the cluster center, and s_{ik} is a Boolean variable that assigns z_i to μ_k. N represents the size of the dataset, and K refers to the number of clusters. The K-Means loss helps to improve the cluster quality of representations by minimizing the distance between each data point and its assigned cluster center.

Soft cluster assignment loss. Soft cluster assignment loss uses the Student's t-distribution as a kernel to measure the similarity between sample data and cluster centers [19], defined as:

$$q_{ij} = \frac{(1 + \parallel z_i - \mu_j \parallel^2 / v)^{-\frac{v+1}{2}}}{\sum_{j'} = (1 + \parallel z_i - \mu_{j'} \parallel^2 / v)^{-\frac{v+1}{2}}} \tag{7}$$

where z_i is the trajectory representation, μ_j is the j^{th} cluster centroid, and v is a constant. q_{ij} can be interpreted as the probability of assigning trajectory T_i to cluster C_j.

Then, we calculate the target distribution P to make the probability of cluster distribution more strict. The probabilities p_{ij} in P are calculated as:

$$p_{ij} = \frac{q_{ij}^2 / \sum_{i'} q_{i'j}}{\sum_{j'} (q_{ij'}^2 / \sum_{i'} q_{i'j'})} \tag{8}$$

The distribution P forces the assigned values closer to 0 and 1 by squaring the original distribution and then normalizing it. On the one hand, it improves the purity of clustering by focusing on sample data with high-confidence assignments. On the other hand, it prevents large clusters from distorting the hidden feature space.

Then, the KL divergence can be used to measure the difference between the distributions P and Q, defined as:

$$L_s(\theta) = \mathrm{KL}(P\|Q) = \sum_i \sum_j = p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{9}$$

Inter-cluster distance loss. Taking into account that the cluster centroids should not be too close to each other, we also innovatively introduce the inter-cluster distance loss as follows:

$$L_d(\theta) = \sum_i \sum_{j \neq i} e^{-(\mu_i - \mu_j)^2} \tag{10}$$

where μ_i and μ_j represent the representation of cluster centers c_i and c_j. This loss aims to ensure that the differences between clusters are as large as possible.

Neighbor loss. It is assumed that the underlying distribution of the data follows the assumptions of continuity and clustering. The continuity assumption posits that data points close to each other are likelier to belong to the same class [39], [40]. The clustering assumption suggests that data points tend to aggregate into discrete clusters, with the points in the same cluster, sharing similar labels [41], [42]. We assume that a data point has the same label as its k-nearest neighbors within the same cluster, and we devise a neighbor loss function as follows:

$$L_n(\theta) = \sum_i \sum_{j \in N'_k(i)} c_{ij} (z_i - z_j)^2 \tag{11}$$

where $N'_k(i)$ is the k neighbors of the ith trajectory data, z_i and z_j are the representations of the ith and jth trajectories, and $c_{ij} \in \{0, 1\}$ indicates whether two trajectories belong to the same cluster.

E. Methodology

After identifying a framework of deep spatiotemporal trajectory clustering, creating new and improved methods became more straightforward. Based on the DSTC framework, we propose two clustering-oriented spatiotemporal trajectory-to-vector methods: the G-DSTC (Grid-based Deep Spatiotemporal Trajectory Clustering) method and the D-DSTC (Density-based Deep Spatiotemporal Trajectory Clustering) method. These two DSTC-based methods can learn representation vectors with better clustering results, and cluster assignments can be obtained at the end of training. The network training for these methods consists of two phases. The first phase involves pretraining using a Seq2Seq model with a reconstruction loss. In the second phase, the reconstruction and clustering losses are optimized jointly.

V. Experiments

In this section, we evaluate DSTC-based methods on four benchmark datasets with six baseline methods and identify popular routes in a real dataset during a day. The source code for DSTC is publicly available here[1].

A. Data

We use four benchmark datasets to evaluate our model, including a simulated dataset and three public spatial datasets.

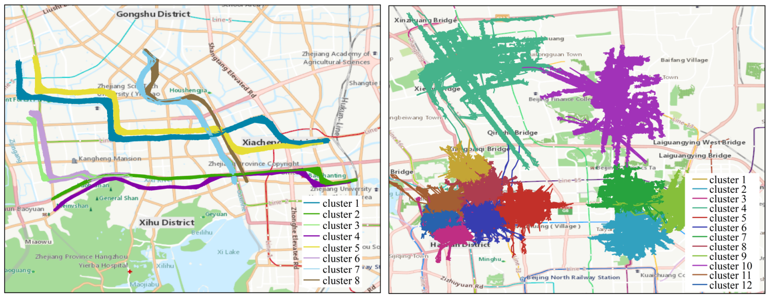
- **Hangzhou_Sim**[2]. Since existing public spatial trajectory datasets with ground-truth labels lack temporal information, we created a simulated three-dimensional spatiotemporal trajectory dataset. Fig. 6(a) illustrates the spatial distribution of data within the time range of 23:00-00:30, which consists of 8,000 trajectories that overlap in both space and time.
- **Geolife** [3] dataset is acquired from the paper [19], which contains 86,113 spatial trajectories within 12 clusters, as shown in Fig. 6(b).
- **Cross** [4] dataset [43] has 1,900 spatial trajectories in 19 clusters, simulating a four-way traffic intersection with multiple through and turn patterns, as shown in Fig. 6(c).
-
- **Lab** [5] dataset [43] consists of trajectories of humans walking through a laboratory captured by an omnidirectional camera. This dataset includes 209 spatial trajectories distributed across 15 clusters, as depicted in Fig. 6(d).

Additionally, we collect a taxi-trajectory dataset of May 2020 in Hangzhou, Zhejiang Province, China, denoted as the dataset **Hangzhou**. It is obtained from the official site of the Hangzhou Transportation Bureau. It contains 2,318 taxi trajectories from Hangzhou East Railway Station to Zhejiang University Yuquan Campus in Hangzhou City. The geographical boundaries of this dataset are depicted in Fig. 7(a). We cluster trajectories and further identify frequently traveled routes during the day in **Hangzhou** dataset.

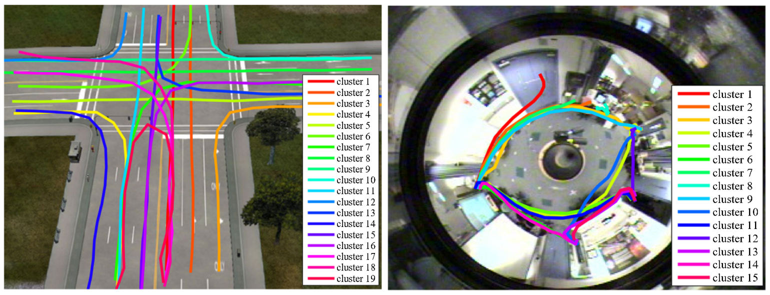
B. Metrics

We evaluate the performance of our algorithms and baseline algorithms using two widely used clustering metrics: the Normalized Mutual Information (NMI) [44] and the Adjusted Rand Index (ARI) [45].

[1]https://github.com/wangchao5032/DSTC
[2]https://figshare.com/articles/dataset/dataset_D_1/22644973
[3]https://figshare.com/articles/dataset/Geolife/24798993
[4]https://figshare.com/articles/dataset/Cross/24798939
[5]https://figshare.com/articles/dataset/Lab/24798978

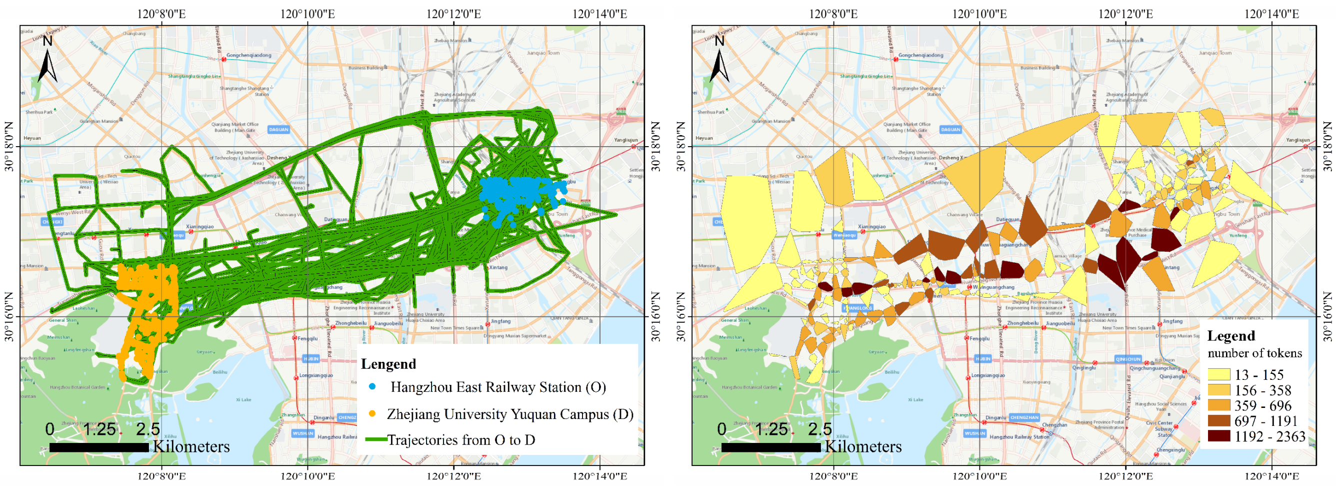


(a) **Hangzhou_Sim** (b) **Geolife**



(c) **Cross** (d) **Lab**

Fig. 6. Benchmark trajectory datasets of **Hangzhou_Sim**, **Geolife**, **Cross**, and **Lab**.



(a) Spatial distribution of **Hangzhou** dataset. (b) Token distribution of **Hangzhou** dataset by the D-DSTC method.

Fig. 7. Dataset **Hangzhou** composed of trajectories from Hangzhou East Railway Station to Zhejiang University Yuquan Campus in Hangzhou City.

NMI. The NMI metric evaluates the information shared between the predicted clustering labels Ω and ground truth labels C, defined as:

$$\mathrm{NMI}(\Omega, C) = \frac{\mathrm{I}(\Omega, C)}{\sqrt{\mathrm{H}(\Omega)\mathrm{H}(C)}} \tag{12}$$

where $\mathrm{I}(\Omega, C)$ denotes the mutual information between predicted labels and ground truth labels, and $\mathrm{H}(\cdot)$ denotes their entropy. The denominator normalizes the mutual information to the $[0, 1]$ range.

ARI. The ARI metric is defined as:

$$\mathrm{ARI} = \frac{\mathrm{RI} - E[\mathrm{RI}]}{\max(\mathrm{RI}) - E[\mathrm{RI}]} \tag{13}$$

where RI is the Rand Index [19] and represents the percentage of correct predictions, defined as:

$$\mathrm{RI} = \frac{\mathrm{TP} + \mathrm{TN}}{N(N-1)/2} \tag{14}$$

where N is the conditional cardinality of the trajectory dataset, TP is the number of trajectory pairs correctly placed in the same cluster, and TN is the number of trajectory pairs correctly placed in different clusters.

C. Baseline Methods

We compare the G-DSTC and D-DSTC methods with six baseline methods: LCSS, EDR, DTW, T2VEC, ST2VEC, and

DTC.

- The LCSS algorithm computes the similarity between trajectories by comparing the distance between two trajectories based on their longest common subsequence [12], [46].
- The EDR method calculates the similarity between two trajectories by determining the minimum number of editing operations necessary to transform one trajectory into another [13].
- The DTW method dynamically aligns trajectories of different lengths and then calculates distances, allowing a sequence to stretch or shrink to match another sequence better [14], [47].
- T2VEC is the first algorithm to learn trajectory representation using deep learning methods and achieves excellent performance [18].
- ST2VEC is a similar trajectory search method for spatiotemporal deep representation learning based on T2VEC [26].
- DTC is a deep spatial trajectory clustering approach that utilizes the K-Means algorithm to cluster trajectories in the embedded latent feature space [28].

In addition, we also evaluated the impact of G-DSTC and D-DSTC on experiment outcomes during a pretraining phase without joint training. We denoted these methods as G-DSTC$_{\text{pre}}$ and D-DSTC$_{\text{pre}}$, respectively.

D. Experiment results

We conduct three sets of experiments to evaluate the efficacy of the algorithms using the DSTC framework. Firstly, we compare the performance of the methods proposed in this paper with other baseline approaches. Secondly, we analyze the influence of various clustering losses on the outcomes. Subsequently, we assess the effect of different token definition methods on the clustering outcomes. We run each algorithm ten times and calculate the average. We discuss the experiment results in performance evaluation, clustering loss function evaluation, and token definition evaluation.

1) Performance evaluation: We first quantitatively evaluate the G-DSTC and D-DSTC methods and six baseline algorithms using two well-known clustering metrics: NMI and ARI. Table I displays the NMI and ARI values of the clustering outcomes for dataset **Hangzhou_Sim** obtained via different algorithms. The best results are highlighted in bold. The experiment results indicate that deep learning-based methods generally outperform traditional point-matching-based methods. One exception is that T2VEC and DTC methods in **Hangzhou_Sim** dataset because they only consider the spatial dimension of trajectory points. The D-DSTC method obtains the highest NMI and ARI values in three datasets, making it the most effective method because the learned trajectory features are more suitable for the clustering task. The superior clustering results of deep learning-based methods confirm the effectiveness of utilizing neural networks to extract trajectory features. The D-DSTC method with density-based divided tokens outperformed G-DSTC in three out of four datasets. We believe the main reason is that D-DSTC can set the grid range automatically and avoid the accuracy loss caused by fixed grid division and inappropriate grid size. Moreover, D-DSTC and G-DSTC with joint training stage perform better than G-DSTC$_{\text{pre}}$ and D-DSTC$_{\text{pre}}$ with only the pretraining stage in most datasets, which means the joint training stage can make the representation of similar trajectories closer to each other.

To visually compare the clustering effects of different methods, we use the t-SNE (t-distributed stochastic neighbor embedding) method [48] to visualize the learned high-dimensional trajectory representations in a two-dimensional map. As shown in Fig. 8, it can be seen that the trajectory representation vectors in the same cluster obtained by the D-DSTC method are more tightly clustered together; that is, the D-DSTC method can accurately distinguish between the trajectories in clusters.

2) Ablation study on clustering loss functions: In this experiment, we choose the D-DSTC method as an example to assess the influence of different clustering loss functions, as displayed in Table II. We first compare the influence of using different principal clustering losses on the clustering results. The first three rows of Table II indicate that combining the reconstruction loss L_r with the soft cluster assignment loss L_s yields better results than using only L_r or combining L_r with the K-Means loss L_k. The comparison result demonstrates that models trained by aligning the soft assignment with the target distribution will achieve better clustering outcomes. Furthermore, we compare the impact of auxiliary clustering losses on the clustering outcomes. Simultaneously combining the inter-cluster distance loss L_d and the neighbor loss L_n produces the best experiment outcome in most datasets. The DSTC-based methods used in the experiments later in this paper combine these four clustering loss functions.

3) Evaluation on the token definitions: As the grid size is a crucial parameter for grid-based trajectory representation learning methods, we evaluate the clustering outcomes by G-DSTC on various datasets using diverse spatial division sizes. In contrast, the D-DSTC method does not require specifying the token size since it uses the density-based clustering method to cluster the trajectory points to create varying-sized tokens. As illustrated in Table III, the D-DSTC method achieves superior outcomes with the highest NMI and ARI on datasets **Hangzhou_Sim** and **Lab**. Although the G-DSTC method achieves better results on datasets **Geolife** and **Cross**, a token size of 300 meters performs best on dataset **Geolife**, whereas a token size of 100 meters is more suitable for dataset **Cross**. Besides, dataset **Lab** comprises trajectory data of indoor human walking. Given the small area of the research region and the limited camera recording range, the G-DSTC method with sizes of 300 meters and 500 meters only generates one token, which cannot differentiate between different trajectories. It indicates that trajectory datasets with distinct data distributions need tokens with varying spatial sizes to obtain optimal outcomes, implying that tokens of the same size cannot accommodate all data distributions. Even though the D-DSTC method does not achieve the best results on datasets **Geolife** and **Cross**, the ARI and NMI metrics are only marginally inferior to the optimal outcomes of the G-DSTC model.

TABLE I
COMPARISON OF CLUSTERING RESULTS BETWEEN DIFFERENT METHODS.

Type	Method	**Hangzhou_Sim**		**Geolife**		**Cross**		**Lab**	
		ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI
Point-matching-based methods	EDR	0.6319	0.8355	0.4312	0.6632	0.2730	0.6399	0.4210	0.6690
	LCSS	0.6496	0.8452	0.5662	0.7846	0.4943	0.7374	0.6797	0.8270
	DTW	0.6814	0.8774	0.6037	0.8184	0.7016	0.8961	0.6578	0.8126
Deep learning-based methods	T2VEC	0.4732	0.7074	0.7447	0.8218	0.8566	0.9391	0.6904	0.8366
	ST2VEC	0.9027	0.9546	0.4868	0.6872	0.7549	0.9065	0.4935	0.7312
	DTC	0.4798	0.7088	0.8859	0.9268	0.8758	0.9457	0.6858	0.8405
DTSC-based methods	G-DSTC$_{\text{pre}}$	0.9641	0.9859	0.8641	**0.9805**	0.8621	0.9406	0.6817	0.8314
	G-DSTC	0.9494	0.9799	**0.8917**	0.9248	0.8729	0.9422	0.6853	0.8357
	D-DSTC$_{\text{pre}}$	0.9690	0.9867	0.8492	0.8992	0.8549	0.9298	0.6628	0.8412
	D-DSTC	**0.9787**	**0.9910**	0.8730	0.9160	**0.8899**	**0.9471**	**0.6956**	**0.8506**

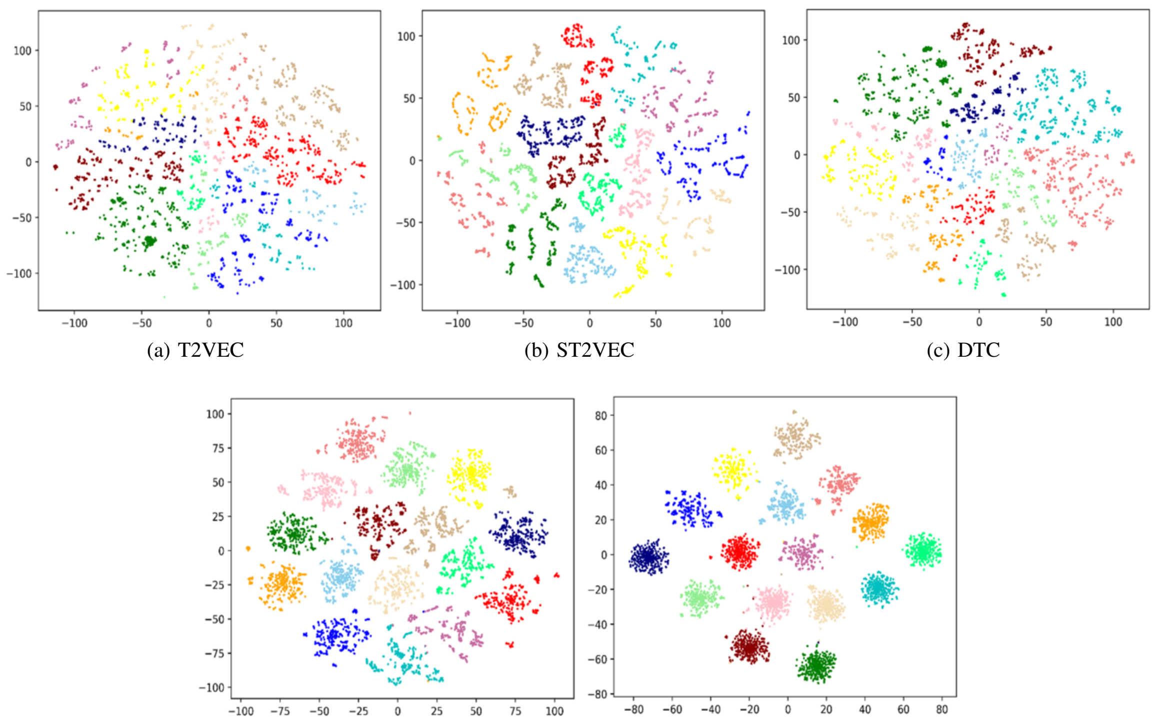

(a) T2VEC (b) ST2VEC (c) DTC
(d) G-DSTC (e) D-DSTC

Fig. 8. t-distributed stochastic neighbor embedding visualization of clustering results of T2VEC, ST2VEC, DTC, G-DSTC and D-DSTC algorithms on **Hangzhou_Sim**.

Next, we visually compare the size and distribution of tokens generated by the G-DSTC and D-DSTC methods on dataset **Hangzhou_Sim**. Fig. 9(a) depicts the spatial distribution of the dataset where distinct colors differentiate trajectories of diverse shapes. We visualize the token distribution of **Hangzhou_Sim** using the D-DSTC and G-DSTC methods with token sizes set to 100 meters and 500 meters, respectively. Areas A and B, marked by red and blue rectangles, denote specific regions of interest for subsequent critical research. Fig. 9(b)-(d) display partially enlarged views of area A, where the irregular block represents the minimum circumscribed convex hull of all trajectory points within the original square token. The darker the color of the distinct blocks, the greater the number of trajectory points within the token. Due to the relatively close distance between the two trajectory routes in area A, it is challenging for the G-DSTC method with a token size of 500 meters to differentiate them. In contrast, the D-DSTC method and the G-DSTC approach with a token size of 100 meters exhibit superior differentiation ability. Fig. 9(e)-(g) display partially enlarged views of area B, where all three methods can differentiate between the two trajectory routes. The observations above suggest that the grid-based approach is not robust. In addition, while the G-DSTC method with a token size of 100 meters can effectively differentiate between the trajectory routes in areas A and B, the fine-grained division significantly increases the number of tokens, which decreases the model's training speed. In contrast, the D-DSTC method can readily adapt to trajectory data with various distributions and obtain an optimal number of tokens.

4) Case study: Fig. 7(a) shows the spatial distribution of dataset **Hangzhou** composed of trajectories from Hangzhou East Railway Station to Zhejiang University Yuquan Campus

TABLE II
ABLATION STUDY OF D-DSTC WITH DIFFERENT KINDS OF CLUSTERING LOSS.

Ablation Setting					**Hangzhou_Sim**		**Geolife**		**Cross**		**Lab**	
L_r	L_k	L_s	L_d	L_n	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI
✓	×	×	×	×	0.9690	0.9867	0.7026	0.7997	0.8549	0.9298	0.6628	0.8412
✓	✓	×	×	×	0.9619	0.9885	0.8079	0.8800	0.8587	0.9330	0.6784	0.8421
✓	×	✓	×	×	0.9751	0.9901	0.8299	0.8953	0.8688	0.9355	0.6809	0.8503
✓	×	✓	✓	×	0.9700	**0.9910**	0.8496	**0.9086**	0.8722	0.9379	0.6849	**0.8514**
✓	×	✓	×	✓	0.9706	0.9862	0.8059	0.8656	0.8706	0.9379	0.6884	0.8480
✓	×	✓	✓	✓	**0.9787**	**0.9910**	**0.8503**	0.9010	**0.8899**	**0.9471**	**0.6956**	0.8506

TABLE III
IMPACT OF VARYING GRID SIZES ON THE DSTC-BASED METHODS.

Method	Grid Size(meter)	**Hangzhou_Sim**			**Geolife**			**Cross**			**Lab**		
		#Token	ARI	NMI	#Token	ARI	NMI	#Token	ARI	NMI	#Token	ARI	NMI
G-DSTC	4	208,697	0.0716	0.2487	400003	0.2729	0.3936	8232	0.7049	0.8575	847	0.6853	0.8357
	100	20,355	0.9134	0.9650	15195	0.6685	0.8011	32	0.8729	0.9422	11	0.6399	0.8265
	300	4,575	0.9494	0.9799	2763	0.9557	0.9625	9	0.6066	0.8449	1	-	-
	500	2,380	0.9136	0.9656	1309	0.8917	0.9248	7	0.1357	0.5995	1	-	-
D-DSTC	-	17,430	0.9787	0.9910	1993	0.9138	0.9331	25	0.8245	0.9274	44	0.7506	0.8767

TABLE IV
HOT ROUTE STATISTICS OF DATASET **HANGZHOU**.

Route	#Trajectories on weekends	#Trajectories on weekdays	Average time duration (min)	Average length (km)
1	28	73	22.4721	11.4304
2	5	13	28.8167	12.6278
3	10	17	24.2875	12.0569

in Hangzhou City. As it is challenging to identify hot routes directly from the raw data, we utilize the D-DSTC method to find the most popular driving routes at different time intervals. Fig. 7(b) shows the spatial distribution of the tokens generated by the D-DSTC method. To simplify the analysis, we select the 10 percent of data closest to the cluster center in each cluster as the research object. Fig. 10 displays the three most frequent routes: Route 1, Route 2, and Route 3. As shown in Table IV, Route 1 is the most frequently selected route from Hangzhou East Railway Station to Zhejiang University's Yuquan Campus, as it has a shorter driving distance and takes less time to reach the destination. Compared to Route 2, Route 3 is chosen more often due to its shorter travel distance and time.

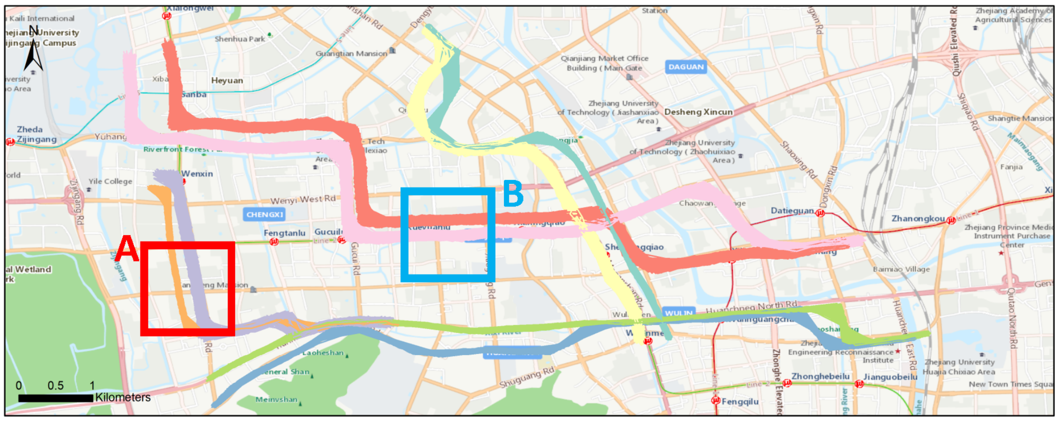
Fig. 11 displays the trajectories of the three routes during different time intervals. As illustrated in Fig. 11(a)-(c), during the time interval of 19:30-20:00, drivers tend to choose the trajectories of Route 1 and Route 2, while they lean towards selecting Route 3 between 20:00-20:30. As shown in Fig. 11(d)-(f), during 17:20-18:30, drivers tend to choose Route 1 and Route 3. It may be because this period coincides with the peak of rush hour when drivers prefer shorter driving distances and times.

VI. CONCLUSIONS AND FUTURE STUDIES

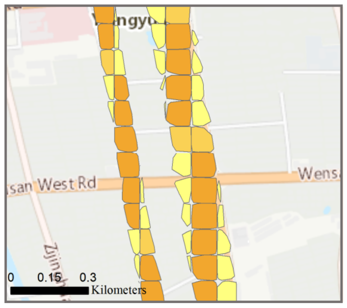
This paper presented a Deep Spatiotemporal Trajectory Clustering framework (DSTC) to address the Spatiotemporal Trajectory Representation Learning towards the Clustering-friendly space (STRLC) problem, which involves learning representations of spatiotemporal trajectories and utilizing them for clustering tasks. Subsequently, we developed two specific methods, D-DSTC and G-DSTC, based on the DSTC framework. In particular, the D-DSTC method considers the non-uniformity in the spatial dimension and periodicity in the temporal dimension of the trajectory data. Moreover, the hot routes discovered in the real-world dataset can reveal human travel habits. In the future, we plan to implement more method instances based on the DSTC framework to discover more exciting patterns from large-scale real-world spatiotemporal trajectory datasets.

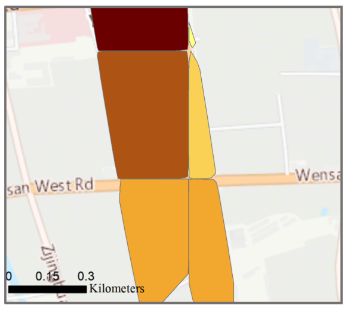
REFERENCES

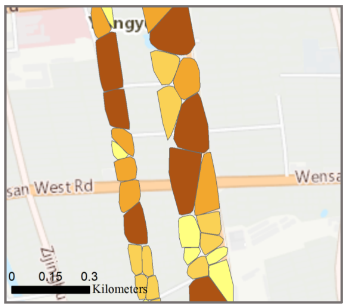
[1] J. Han, Z. Li, and L. A. Tang, "Mining moving object, trajectory and traffic data," in *International Conference on Database Systems for Advanced Applications*. Springer, 2010, pp. 485–486.
[2] Y. Zheng, X. Xie, W.-Y. Ma *et al.*, "Geolife: A collaborative social networking service among user, location and trajectory." *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010.
[3] C. Li, S. Ling, H. Zhang, H. Zhao, L. Liu, and N. Jia, "A sequence and network embedding method for bus arrival time prediction using gps trajectory data only," *IEEE Transactions on Intelligent Transportation Systems*, 2023.
[4] Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, and F. Zhang, "Identifying human mobility via trajectory embeddings." in *IJCAI*, vol. 17, 2017, pp. 1689–1695.
[5] B. Jiang and X. Yao, "Location-based services and gis in perspective," *Computers, Environment and Urban Systems*, vol. 30, no. 6, pp. 712–725, 2006.
[6] P. Zhang, J. Zheng, H. Lin, C. Liu, Z. Zhao, and C. Li, "Vehicle trajectory data mining for artificial intelligence and real-time traffic information extraction," *IEEE Transactions on Intelligent Transportation Systems*, 2023.
[7] A.-L. Barabasi, "The origin of bursts and heavy tails in human dynamics," *Nature*, vol. 435, no. 7039, pp. 207–211, 2005.
[8] D. Brockmann, L. Hufnagel, and T. Geisel, "The scaling laws of human travel," *Nature*, vol. 439, no. 7075, pp. 462–465, 2006.

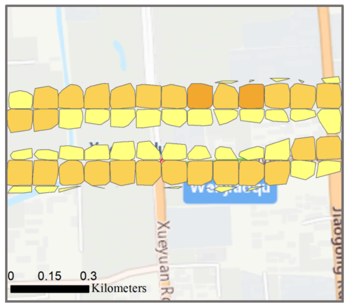


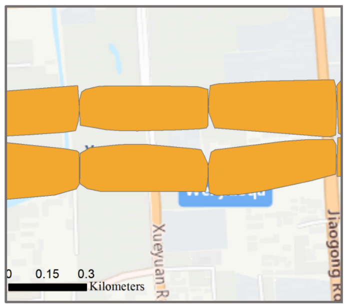
(a) The spatial distribution of the dataset **Hangzhou_Sim**.


(b) Area A is divided into tokens of 100 meters by G-DSTC.


(c) Area A is divided into tokens of 500 meters by G-DSTC.


(d) Area A is divided into tokens by D-DSTC.


(e) Area B is divided into tokens of 100 meters by G-DSTC.


(f) Area B is divided into tokens of 500 meters by G-DSTC.

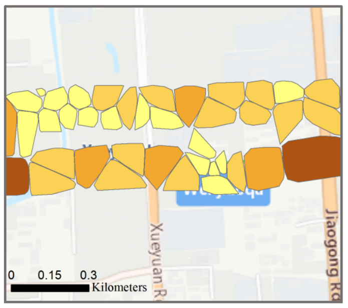
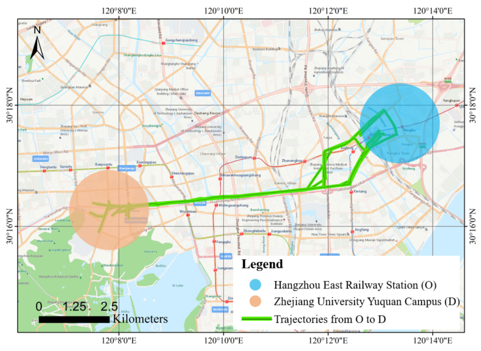
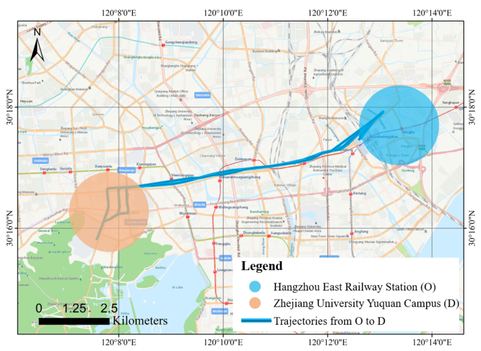

(g) Area B is divided into tokens by D-DSTC.

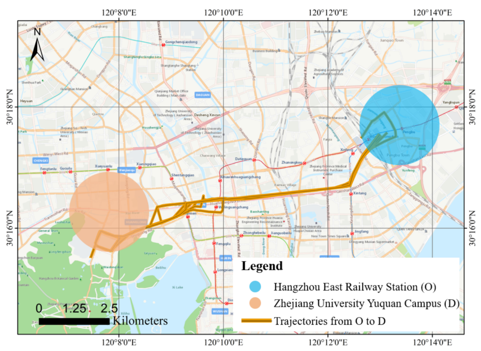
Fig. 9. Illustration of the spatial division of the dataset **Hangzhou_Sim**.


Legend
Hangzhou East Railway Station (O)
Zhejiang University Yuquan Campus (D)
Trajectories from O to D
0 1.25 2.5 Kilometers

(a) Route 1


Legend
Hangzhou East Railway Station (O)
Zhejiang University Yuquan Campus (D)
Trajectories from O to D
0 1.25 2.5 Kilometers

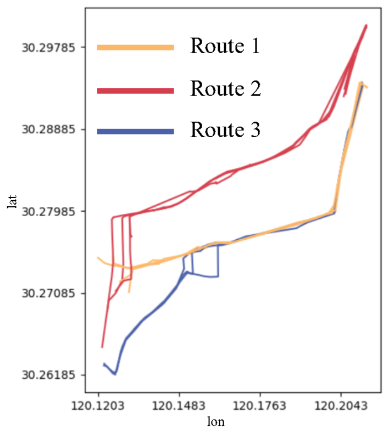
(b) Route 2


Legend
Hangzhou East Railway Station (O)
Zhejiang University Yuquan Campus (D)
Trajectories from O to D
0 1.25 2.5 Kilometers

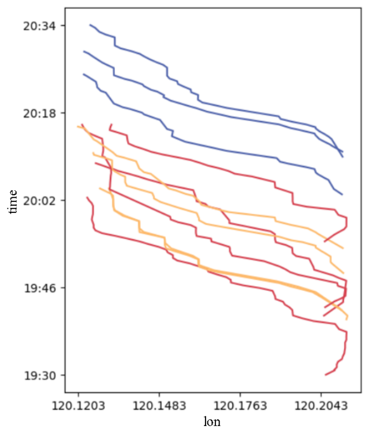
(c) Route 3

Fig. 10. Hot routes of dataset **Hangzhou**.

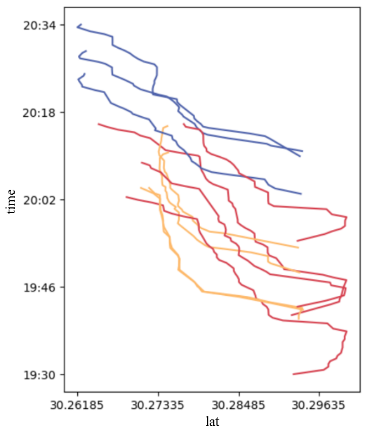
[9] C. Wang, Z. Du, Y. Gu, F. Zhang, and R. Liu, "Stme: An effective method for discovering spatiotemporal multi-type clusters containing



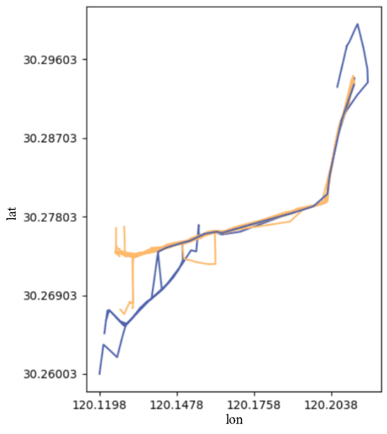
(a) Hot routes of latitude-longitude distribution in 19:30-20:30.



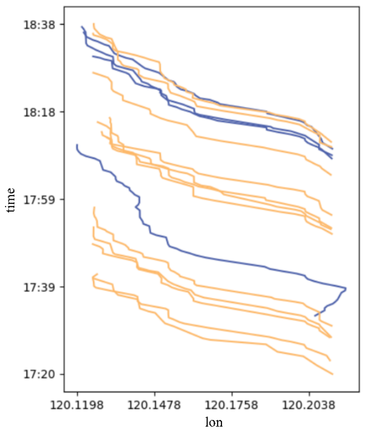
(b) Hot routes of longitude-time distribution in 19:30-20:30.



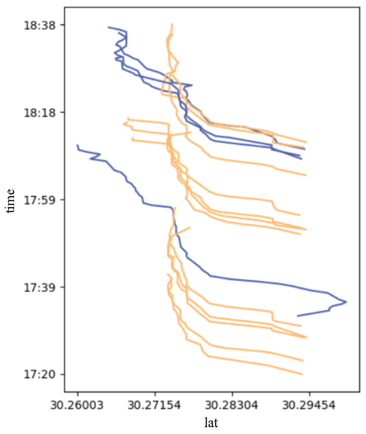
(c) Hot routes of latitude-time distribution in 19:30-20:30.



(d) Hot routes of latitude-longitude distribution in 17:20-18:30.



(e) Hot routes of longitude-time distribution in 17:20-18:30.



(f) Hot routes of latitude-time distribution in 17:20-18:30.

Fig. 11. The latitude-longitude, longitude-time distribution, and latitude-time distribution for the three routes during different periods are presented in (a)-(c) for 19:30 to 20:30 and (d)-(f) for 17:20 to 18:30.

events with different densities," *Transactions in GIS*, vol. 24, no. 6, pp. 1559–1577, 2020.

[10] W. Wang, F. Xia, H. Nie, Z. Chen, Z. Gong, X. Kong, and W. Wei, "Vehicle trajectory clustering based on dynamic representation learning of internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3567–3576, 2020.

[11] Z. Cheng, L. Jiang, D. Liu, and Z. Zheng, "Density based spatio-temporal trajectory clustering algorithm," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 3358–3361.

[12] R. A. K.-l. Lin and H. S. S. K. Shim, "Fast similarity search in the presence of noise, scaling, and translation in time-series databases," in *Proceeding of the 21th International Conference on Very Large Data Bases*. Citeseer, 1995, pp. 490–501.

[13] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, 2005, pp. 491–502.

[14] D. Sankoff, "Time warps, string edits, and macromolecules," *The Theory and Practice of Sequence Comparison, Reading*, 1983.

[15] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297.

[16] Q. Yu, Y. Luo, C. Chen, and S. Chen, "Trajectory similarity clustering based on multi-feature distance measurement," *Applied Intelligence*, vol. 49, pp. 2315–2338, 2019.

[17] X. Niu, T. Chen, C. Q. Wu, J. Niu, and Y. Li, "Label-based trajectory clustering in complex road networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 10, pp. 4098–4110, 2019.

[18] X. Li, K. Zhao, G. Cong, C. S. Jensen, and W. Wei, "Deep representation learning for trajectory similarity computation," in *2018 IEEE 34th international conference on data engineering (ICDE)*. IEEE, 2018, pp. 617–628.

[19] Z. Fang, Y. Du, L. Chen, Y. Hu, Y. Gao, and G. Chen, "E2dtc: an end to end deep trajectory clustering framework via self-training," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 696–707.

[20] P. Yang, H. Wang, Y. Zhang, L. Qin, W. Zhang, and X. Lin, "T3s: Effective representation learning for trajectory similarity computation," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 2183–2188.

[21] Y. Endo, K. Nishida, H. Toda, and H. Sawada, "Predicting destinations from partial trajectories using recurrent neural network," in *Advances in Knowledge Discovery and Data Mining: 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I 21*. Springer, 2017, pp. 160–172.

[22] J. Lv, Q. Li, Q. Sun, and X. Wang, "T-conv: A convolutional neural network for multi-scale taxi trajectory prediction," in *2018 IEEE international conference on big data and smart computing (bigcomp)*. IEEE, 2018, pp. 82–89.

[23] Y. Endo, H. Toda, K. Nishida, and J. Ikedo, "Classifying spatial trajectories using representation learning," *International Journal of Data Science and Analytics*, vol. 2, no. 3-4, pp. 107–117, 2016.

[24] J. Tang, L. Liu, J. Wu, J. Zhou, and Y. Xiang, "Trajectory clustering method based on spatial-temporal properties for mobile social networks," *Journal of Intelligent Information Systems*, vol. 56, pp. 73–95, 2021.

[25] T. Dan, C. Luo, Y. Li, B. Zheng, and G. Li, "Spatial temporal trajectory similarity join," in *Web and Big Data: Third International*

Joint Conference, APWeb-WAIM 2019, Chengdu, China, August 1–3, 2019, Proceedings, Part II 3. Springer, 2019, pp. 251–259.

[26] D. A. Tedjopurnomo, X. Li, Z. Bao, G. Cong, F. Choudhury, and A. K. Qin, “Similar trajectory search with spatio-temporal deep representation learning,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 12, no. 6, pp. 1–26, 2021.

[27] J. Yang, D. Parikh, and D. Batra, “Joint unsupervised learning of deep representations and image clusters,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5147–5156.

[28] C. Wang, F. Lyu, S. Wu, Y. Wang, L. Xu, F. Zhang, S. Wang, Y. Wang, and Z. Du, “A deep trajectory clustering method based on sequence-to-sequence autoencoder model,” *Transactions in GIS*, vol. 26, no. 4, pp. 1801–1820, 2022.

[29] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, “Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture,” in *2018 IEEE intelligent vehicles symposium (IV)*. IEEE, 2018, pp. 1672–1678.

[30] H. Yao, D.-l. Zhu, B. Jiang, and P. Yu, “Negative log likelihood ratio loss for deep neural network classification,” in *Proceedings of the Future Technologies Conference*. Springer, 2019, pp. 276–282.

[31] R. J. Williams and D. Zipser, “A learning algorithm for continually running fully recurrent neural networks,” *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[32] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, “A survey of clustering with deep learning: From the perspective of network architecture,” *IEEE Access*, vol. 6, pp. 39 501–39 514, 2018.

[33] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, “Towards k-means-friendly spaces: Simultaneous deep learning and clustering,” in *international conference on machine learning*. PMLR, 2017, pp. 3861–3870.

[34] D. Beeferman and A. Berger, “Agglomerative clustering of a search engine query log,” in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2000, pp. 407–416.

[35] J. Xie, R. Girshick, and A. Farhadi, “Unsupervised deep embedding for clustering analysis,” in *International conference on machine learning*. PMLR, 2016, pp. 478–487.

[36] P. Huang, Y. Huang, W. Wang, and L. Wang, “Deep embedding network for clustering,” in *2014 22nd International conference on pattern recognition*. IEEE, 2014, pp. 1532–1537.

[37] X. Peng, J. Feng, J. T. Zhou, Y. Lei, and S. Yan, “Deep subspace clustering,” *IEEE transactions on neural networks and learning systems*, vol. 31, no. 12, pp. 5509–5521, 2020.

[38] E. Aljalbout, V. Golkov, Y. Siddiqui, M. Strobel, and D. Cremers, “Clustering with deep learning: Taxonomy and new methods,” *arXiv preprint arXiv:1801.07648*, 2018.

[39] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, “Learning with local and global consistency,” *Advances in neural information processing systems*, vol. 16, 2003.

[40] B. Athiwaratkun, M. Finzi, P. Izmailov, and A. G. Wilson, “There are many consistent explanations of unlabeled data: Why you should average,” *arXiv preprint arXiv:1806.05594*, 2018.

[41] Y. Jia, S. Kwong, and J. Hou, “Semi-supervised spectral clustering with structured sparsity regularization,” *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 403–407, 2018.

[42] F. Taherkhani, A. Dabouei, S. Soleymani, J. Dawson, and N. M. Nasrabadi, “Transporting labels via hierarchical optimal transport for semi-supervised learning,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*. Springer, 2020, pp. 509–526.

[43] B. Morris and M. Trivedi, “Learning trajectory patterns by clustering: Experimental studies and comparative evaluation,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 312–319.

[44] S. Fogel, H. Averbuch-Elor, D. Cohen-Or, and J. Goldberger, “Clustering-driven deep embedding with pairwise constraints,” *IEEE computer graphics and applications*, vol. 39, no. 4, pp. 16–27, 2019.

[45] Q. Feng, L. Chen, C. P. Chen, and L. Guo, “Deep fuzzy clustering—a representation learning approach,” *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 7, pp. 1420–1433, 2020.

[46] M. Vlachos, G. Kollios, and D. Gunopulos, “Discovering similar multidimensional trajectories,” in *Proceedings 18th international conference on data engineering*. IEEE, 2002, pp. 673–684.

[47] N. Magdy, M. A. Sakr, T. Mostafa, and K. El-Bahnasy, “Review on trajectory similarity measures,” in *2015 IEEE seventh international conference on Intelligent Computing and Information Systems (ICICIS)*. IEEE, 2015, pp. 613–619.

[48] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.” *Journal of machine learning research*, vol. 9, no. 11, 2008.

Chao Wang received the B.S. degree from Lanzhou University in 2016 and the Ph.D. degree from Zhejiang University in 2021. She is currently a senior researcher with the Big data intelligence research center, Zhejiang Lab. Her research interests include spatial data mining and geographic information science.

Jiahui Huang received her B.S degree in Computer Science and Technology from Zhejiang A&F University, Hangzhou, China, in 2020. She is currently working toward the M.S degree in Software Engineering at Zhejinag University. Her main research interests include deep learning and data mining.

Yongheng Wang received the Ph.D degree in computer science and technology from National University of Defense Technology, Changsha, China, in 2006. He is currently a research specialist in Big data intelligence research center of Zhejiang Lab. His research interests include Big data analytics, machine learning and intelligent decision making. His current research is on intelligent interactive data analysis and simulation-based intelligent decision making in the economic field.

Zhengxuan Lin received his B.S degree in Computer Science and Technology from Fujian Agriculture and Forestry University, Fuzhou, China, in 2021. He is currently working toward the M.S degree in Software Engineering at Zhejinag University. His research interests include data visualization and trajectory data mining.

Xiongnan Jin received his B.S. degree in software engineering from Zhejiang University in 2012, and the Ph.D. degree in computer science from Yonsei University in 2019. Then he worked as a postdoctoral researcher at National Institute of Standards and Technology, Maryland, US. Now he is a senior researcher in Big data intelligence research center of Zhejiang Lab, Hangzhou, China. His main research interests include knowledge graphs, deep learning, and data mining.

Xing Jin received the bachelor's degree in computer science from Lanzhou University, Lanzhou, China, in 2012, and the Ph.D. degree from the School of Software Technology, Dalian University of Technology (DLUT), Dalian, China, in 2019. Since 2019, he has been a Lecturer with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His research interests include multi-agent systems, cooperation theory, evolutionary game theory, and reinforcement learning..

Di Weng is a ZJU100 Young Professor at School of Software Technology, Zhejiang University. His main research interest lies in information visualization and visual analytics, focusing on interactive data transformation and spatiotemporal data analysis. He received his Ph.D. degree in Computer Science from State Key Lab of CAD&CG, Zhejiang University. Prior to his current position, Dr. Weng was a researcher at Microsoft Research Asia from 2022 to 2023. For more information, please visit https://dwe.ng..

Yingcai Wu is a Professor at the State Key Lab of CAD&CG, Zhejiang University. His main research interests are in visual analytics and information visualization, with focuses on sports analytics and big data intelligence. He received his Ph.D. degree in Computer Science from the Hong Kong University of Science and Technology. Prior to his current position, Dr. Wu was a postdoctoral researcher in the University of California, Davis from 2010 to 2012, and a researcher in Microsoft Research Asia from 2012 to 2015. For more information, please visit http://www.ycwu.org.