Wang GW, Zhang JD, Li J. Complete your mobility: Linking trajectories across heterogeneous mobility data sources. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 33(4): 792–806 July 2018. DOI 10.1007/s11390-018-1856-x

Complete Your Mobility: Linking Trajectories Across Heterogeneous Mobility Data Sources

Guo-Wei Wang, Jin-Dou Zhang, and Jing Li, *Member*, *CCF*, *ACM*

School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China

E-mail: {weiking, jindou}@mail.ustc.edu.cn; lj@ustc.edu.cn

Received May 9, 2017; revised January 26, 2018.

Abstract Nowadays, human activities and movements are recorded by a variety of tools, forming different trajectory sets which are usually isolated from one another. Thus, it is very important to link different trajectories of one person in different sets to provide massive information for facilitating trajectory mining tasks. Most prior work took advantages of only one dimensional information to link trajectories and can link trajectories in a one-to-many manner (providing several candidate trajectories to link to one specific trajectory). In this paper, we propose a novel approach called one-to-one constraint trajectory linking with multi-dimensional information (OCTL) that links the corresponding trajectories of one person in different sets in a one-to-one manner. We extract multidimensional features from different trajectory datasets for corresponding relationships prediction, including spatial, temporal and spatio-temporal information, which jointly describe the relationships between trajectories. Using these features, we calculate the corresponding probabilities between trajectories in different datasets. Then, we formulate the link inference problem as a bipartite graph matching problem and employ effective methods to link one trajectory to another. Moreover, the advantages of our approach are empirically verified on two real-world trajectory sets with convincing results.

Keywords trajectory linking, trajectory data mining, trajectory similarity, mobility pattern mining

1 Introduction

The advances in location-acquisition technologies have generated a myriad of trajectories representing the motion of human beings[1]. Mining of such trajectories benefits various industrial and commercial applications, including mobility prediction[2], social computing[3-4], urban computing[5], etc. Nowadays, human beings' activities and movements are recorded by a variety of tools, forming different trajectory sets (WiFi, GPS, check-in, etc.). Thus, each person may have multiple separate trajectories collected by various methods. Here, we call the trajectories generated by different methods heterogeneous trajectories.

Multiple trajectory datasets can provide richer information for trajectory mining tasks, especially to alleviate the difficulties of data sparsity and information limitation faced by traditional single trajectory mining. For example, by combining the information of social relationships and activities in social network trajectories and the information in the dense GPS trajectories, both location recommendation and friend recommendation can achieve better performance. However, it is non-trivial to integrate the information of heterogeneous trajectories together since they are collected from mostly isolated sources and by different methods.

Therefore, it is of great importance to study the trajectory linking problem which aims to discover the correspondences between trajectories of the same person across multiple datasets. Suppose there are two heterogeneous trajectory datasets, for clarity, a source dataset and a target dataset. When there is a source trajectory for one person, there is a target one in the other dataset. The task of trajectory linking is to discover the corresponding pair of heterogeneous trajectories belonging to the same person. Since there is only one trajectory in a dataset of a specific person, the trajectory linking problem is subject to the one-to-one constraint. Also, in order to simplify the problem,

Regular Paper

The work was supported by the National Key Research and Development Program of China under Grant No. 2016YFB0201402.

each person has and only has one pair of trajectories to be identified. At the same time, one target trajectory also can only be linked to one source trajectory.

Intuitively, the problem is closely related to the similar time sequences search[6], as it also investigates the relations between two trajectories. Unfortunately, such a method can hardly be directly applied to our case, since it considers homogeneous trajectories, which are generated by the same method. The reasons are as follows.

• *Sampling Rate Independence.* The variety of the sampling rates among different sources is the most distinctive feature of heterogeneous trajectories. For example, GPS trajectories are usually sampled at a very high rate whereas the social network check-ins are sampled at an extremely low rate. The independent sampling rates make it difficult to find the corresponding relationship among heterogeneous trajectories.

• *Sampling Area Variety.* The spatio-temporal records of the same person in two different databases scarcely intersect. For example, trajectories of WiFi data are sampled by the WiFi access points (APs), and thus the trajectory can only be recorded in the area with WiFi APs. On the other hand, the trajectories of smart card data are limited to the card readers. Thus, it needs more sophisticated methods to solve the trajectory linking problem than prior work on the similar time sequences search.

• *Data Sparsity and Overlap.* Most of the trajectory data mining tasks are facing the data sparsity problem. A sparse trajectory can hardly provide any useful information to the mining tasks. At the same time, the trajectories of persons with close relationship usually have a significant overlap. For example, most of the trajectories of two close friends in a college may lie on their same departments and dormitories. Such an overlap renders it difficult to distinguish them.

Given the above issues, researchers proposed two preliminary methods to solve the heterogeneous trajectory linking problem, called automatic user identification (AUI)[7] and fuzzy trajectory link (FTL)[8], respectively. Both of them aim to solve the sampling rate independence and sampling area variety problems. AUI takes the idea of shared location similarity measurement, which counts the number of trajectory points generated at similar locations, and proposes a new similarity measurement. However, the approach cannot distinguish the overlapped trajectories with only the spatial information. Different from AUI, FTL establishes a model by determining whether two adjacent heterogeneous trajectory points are reachable under a certain speed limit. However, FTL can hardly handle the sparse data because if two trajectories are both sampled at a low frequency, any two adjacent trajectory points are reachable under any speed limit. Furthermore, for a source trajectory, these two methods both come up with several candidate target trajectories with probabilities higher than a given threshold to be generated by the same person. In a word, both AUI and FTL link trajectories in a one-to-many manner and they choose the one candidate with the highest probability to link to the source trajectory. However, this selection strategy may lead to inaccuracy and violations of one-to-one constraint mentioned above. For example, the candidate target trajectory with the highest probability for one source trajectory may also be the first choice of another. As a result, the whole method may be ineffective.

In this paper, we propose a novel approach, called one-to-one constraint trajectory linking with multi-dimensional information (OCTL), to tackle the above issues. Different from the two existing methods that only utilize either spatial features or spatio-temporal features, our approach can extract multidimensional features from multiple heterogeneous trajectory datasets for trajectory linking, including spatial, temporal and spatio-temporal information of trajectories. To be specific, the spatial features describe the scenarios in which the information of a person is acquired in different ways at the same locations. The temporal features represent the scenarios where the information of a person is collected in different ways at the same time. And the spatio-temporal features indicate the similarity of mobility patterns of two heterogeneous trajectories. We explain the features in detail in Section 3. In general, these features can jointly describe the correlations among heterogeneous trajectories more comprehensively even when the trajectories are extremely sparse.

Then we adopt a supervised learning method to calculate the linkable probabilities with these features. The labelled data can be easily constructed using extra identity information, e.g., using users' self-defined records of multiple social accounts to correlate multiple social check-in trajectories. Since users with extra identity information are only part of all users in reality, it is not feasible to use the information to solve the trajectory linking problem for all users. Instead, it is a good way to construct training set using extra identity information for our supervised learning method. With those

features and the labelled heterogeneous trajectories, we train a binary classifier to calculate the probability that two heterogeneous trajectories may be generated from the same person. A well-trained binary classifier can promote the overlapped trajectories to be distinguished more easily and provide accurate linkable probabilities between heterogeneous trajectories.

Besides, unlike previous work that directly outputs all the candidates with probabilities higher than a given threshold, we additionally suggest an inference step to link the heterogeneous trajectories in a one-to-one manner. That is, each trajectory in one dataset can only be exclusively linked with one trajectory in another dataset. For this purpose, we formulate the inference problem as a bipartite graph matching problem based upon the probabilities provided by the binary classifier and solve it accordingly. As a result, OCTL can effectively infer the links of trajectories under the one-to-one constraint. To verify the effectiveness of OCTL, we conduct extensive experiments on real-world heterogeneous trajectory databases, including trajectories in the urban area and trajectories in a local university. The results show that OCTL consistently outperforms other commonly used baselines and the state-of-the-art methods in heterogeneous trajectory linking.

The rest of the paper is organized as follows. We give the preliminary concepts and the problem analysis in detail in Section 2. In Section 3, we propose the OCTL approach for trajectory linking across multiple trajectory datasets. Section 4 reports the experimental results on two real-world trajectory datasets. Finally, we revisit related work in Section 5 and the paper is concluded in Section 6.

2 Problem Details

2.1 Problem Definition

In this paper, we focus on studying the trajectory linking problem on two heterogeneous trajectory datasets, and it can be easily generalized to the scenarios with more than two datasets. We first present several useful definitions.

Definition 1 (Trajectory). *A trajectory $T = \{p_1, p_2, \ldots, p_n\}$ is a temporally ordered sequence of spatio-temporal points. Each point p_i is associated with three attributes x, y, and t where (x, y) indicates the location of p_i and t indicates the timestamp when p_i is recorded. Then, we name the trajectories which are obtained from different sources as heterogeneous trajectories.*

Definition 2 (Stay-Points Trajectory). *A stay-points trajectory $ST = \{sp_1, sp_2, \ldots, sp_n\}$ is also a temporally ordered sequence of spatio-temporal points like trajectory in Definition* 1. *Different from a normal trajectory, each point sp_i of a stay-points trajectory is associated with four attributes x, y, t, and Δt, where (x, y) indicates the location of sp_i, t indicates the timestamp when the user arrives at sp_i and Δt represents the time interval that the user stays at sp_i.*

Definition 3 (Linkable). *Suppose there are two heterogeneous trajectories T_i and T_j. If T_i and T_j are observed from the same person, we call T_j a linkable trajectory of T_i. Thus, the linkable probability is the probability that two trajectories are linkable, denoted by $lp(T_i, T_j)$.*

Our problem is defined as follows. Given a source trajectory dataset D^{S} and a target trajectory dataset D^{T}, which are collected from two different data sources, we assume that different trajectories within one dataset are observed from different persons. For each pair of trajectories $(T_i^{\mathrm{S}}, T_j^{\mathrm{T}})$, where $T_i^{\mathrm{S}} \in D^{\mathrm{S}}, T_j^{\mathrm{T}} \in D^{\mathrm{T}}$, our goal is to identify whether T_i^{S} and T_j^{T} are linkable. In other words, for each T_i^{S}, we aim to find a linkable trajectory T_j^{T} from D^{T}. Moreover, it must be guaranteed that for each T_i^{S}, there exists at most one linkable trajectory T_j^{T}. In a word, we aim to link heterogeneous trajectories under the one-to-one constraint.

2.2 Problem Difficulties

The first difficulty in trajectory linking is how to represent the linkable probability of two heterogeneous trajectories. Intuitively, two heterogeneous trajectories with a higher similarity are more likely to be generated by the same person. However, the sampling rates and sampling areas could be very different between two heterogeneous trajectories, which makes the linkable probability of them hardly be represented by a single trajectory similarity as used in similar time sequences search methods[6]. When two heterogeneous trajectories have little intersection of temporal or spatial space, the trajectory similarity will be extremely noisy. For example, a dense trajectory may have high similarities with several sparse heterogeneous trajectories at the same time.

Another key issue of the trajectory linking problem is the one-to-one constraint matching between two heterogeneous trajectories. For each source trajectory T_i^{S}, previous work[7-8] directly outputs all the candidate target trajectories with probabilities higher than

a given threshold. Thus, both [7] and [8] can efficiently achieve high top k hit rate when the linkable probability is correctly calculated and k is large enough (e.g., $k = 5$ or $k = 10$). However, in the trajectory linking problem, each trajectory is linkable to at most one heterogeneous trajectory, which makes top 1 hit rate (i.e., the accuracy) more important than top k hit rate. Meanwhile, one specific target trajectory may have the highest linkable probability with more than one source trajectory at the same time, which makes the selection strategy of prior work lead to inaccurate results and violation of the one-to-one constraint.

3 Trajectory Linking

We design a two-phase approach to address the two difficulties. The first phase tackles the linkable probability extraction problem, while the second phase takes care of the one-to-one constraint inference. To be specific, in the first phase, we explore the multidimensional information in both trajectory datasets to extract three sets of features for trajectory linking, which correspond to aggregated patterns of persons on spatial, temporal and spatio-temporal distribution. After that, we exploit all the extracted features and the pairs of labeled heterogeneous trajectories to learn a binary classifier which is used to calculate the linkable probabilities for unlabeled heterogeneous trajectories. In the second phase, we propose a one-to-one inference step to infer the pairs of linkable trajectories based upon the probabilities outputted by the binary classifier.

The pseudo-code of this approach is shown as Algorithm 1. Algorithm 1 takes two trajectory datasets D^{S} and D^{T} as input, which contain ns and nt trajectories respectively, and outputs a set LT containing all linkable trajectory pairs. Two training trajectory datasets TD^{S} and TD^{T}, which contain nts and ntt trajectories respectively, are constructed to train the classifier with known labels. The classifier is trained from line 1 to line 6, and the features extraction process for the training set in line 3 and line 4 is the same as it for the testing set in line 8 and line 9. The process from line 7 to line 11 explains how we extract linkable probabilities with the trained classifier. At last, we infer the pairs of linkable trajectories using the linkable probabilities in line 12. We explain each step in detail as follows.

Algorithm 1. One-to-One Constraint Trajectory Linking

Require: $D^{\mathrm{S}} = \{T_i^{\mathrm{S}} | 1 \leqslant i \leqslant ns\}$; $D^{\mathrm{T}} = \{T_j^{\mathrm{T}} | 1 \leqslant j \leqslant nt\}$
Ensure: a set $LT = \{(T_i^{\mathrm{S}}, T_j^{\mathrm{T}}) \mid T_i^{\mathrm{S}}$ and T_j^{T} are linkable, $1 \leqslant i \leqslant ns, 1 \leqslant j \leqslant nt\}$
1: Construct a training set TD of trajectory pairs with known labels, which consists of two heterogeneous trajectory datasets $TD^{\mathrm{S}} = \{TT_i^{\mathrm{S}} | 1 \leqslant i \leqslant nts\}$ and $TD^{\mathrm{T}} = \{TT_j^{\mathrm{T}} | 1 \leqslant j \leqslant ntt\}$
2: **for** each pair $(TT_i^{\mathrm{S}}, TT_j^{\mathrm{T}})$ **do**
3: Transform the original pair into stay-points trajectories $(STT_i^{\mathrm{S}}, STT_j^{\mathrm{T}})$
4: Extract three types of features
5: **end for**
6: Train a classification model C on the training set
7: **for** each pair $(T_i^{\mathrm{S}}, T_j^{\mathrm{T}})$ **do**
8: Transform the original pair into stay-points trajectories $(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})$
9: Extract three types of features
10: Calculate linkable probability $lp(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})$ by C
11: **end for**
12: Infer pairs of linkable trajectories by weighted bipartite graph matching algorithm

3.1 Linkable Probability Extraction

A trajectory is a temporally ordered sequence of spatio-temporal points, which involves abundant information about the spatial, temporal, and spatio-temporal records. In order to take advantage of the multidimensional information of trajectories, we propose to explore the above information about pairwise heterogeneous trajectories to facilitate trajectory linking, which consists of three major steps, i.e., pre-processing, extraction of multidimensional features, and linkable probability calculation.

3.1.1 Pre-Processing

The heterogeneous trajectories are usually recorded in different formats due to different ways of being obtained. To address this, the pre-processing stage transforms each trajectory into a set of stay-points with stay time, i.e., the location points that the user stays for a while rather than passes by. Thus, the heterogeneous trajectories can be uniform as stay-points trajectories and the stay-points usually carry the particular semantic meanings such as the buildings they live or work.

The methods of stay-points detection closely depend on the characteristics of the original trajectory data. For the trajectory formed by location information gained at a fixed time period, e.g., GPS trajectory, the stay-points detection method is well researched in [9]. Through that method, we compress the original trajectories to the stay-points trajectories in Definition 2. Another kind of trajectories are collected when some conditions are triggered, which means they carry more information than the timestamp. For example, a WiFi trajectory point is recorded when a mobile device connects to or disconnects from an access point. The stay-points detection method for this kind of trajectories depends on their triggered conditions. For example, in

the transition from a WiFi trajectory to a stay-points trajectory, we first find two contiguous points which are recorded at the same location while the former trajectory point p_{f} carries the timestamp the mobile device connects to the WiFi access point and the latter p_{l} carries the disconnection timestamp. Then, the four attributes of a stay-point sp can be calculated as follows: $sp.x$, and $sp.y$ are as same as the original location coordinates $p_{\mathrm{f}}.x$, and $p_{\mathrm{f}}.y$ respectively, arrived time $sp.t$ is represented by $p_{\mathrm{f}}.t$, and the stay time $sp.\Delta t$ is then calculated by $p_{\mathrm{l}}.t - p_{\mathrm{f}}.t$. Hence, we can compress WiFi trajectories to the stay-points trajectories in the same form as the GPS trajectories.

In addition to the above two kinds of trajectories, some trajectories are too sparse to be processed as in the above manner. For these trajectories, almost a single spatio-temporal point can represent a stay-point.

3.1.2 Extraction of Multidimensional Features

In this subsection, we introduce the multidimensional features we use in trajectory linking and the method to extract them. Specifically, we first partition the map into grids with different granularities to discrete the locations to reduce computational effort and inaccuracy of location information. In this paper, we partition the map into N grids, denoted as $G_1, G_2, \ldots, G_N$, respectively. By mapping each stay-point into the corresponding grid, namely grids mapping, we can use the distance between grids to replace the distance between stay-points. For example, if two stay-points are located in the same grid, we consider the two stay-points are close enough that the two trajectories share one same location. Similarly, we also partition the whole time interval into M slots, denoted as $S_1, S_2, \ldots, S_M$ respectively, to discrete the time. Then we start to extract the multidimensional features with the aid of grids and slots.

Spatial Distribution Features. We notice that the heterogeneous trajectories of the same person are usually gained at similar locations in real life. For example, when a person enters a WiFi access point, the location information gained by GPS sensor is similar to that of the WiFi access point. The spatial features describe the scenarios in which the information of a person is acquired in different ways at the same locations. We use two measurements as the spatial distribution features to evaluate the similarity between the spatial distributions of two heterogeneous trajectories: 1) the inverse document frequency weighted (IDF-weighted) shared locations similarity, 2) the cosine similarity between the two sets of locations.

Algorithm 2 describes the process of calculating the IDF-weighted shared locations similarity of two heterogeneous trajectory datasets. It takes two trajectory datasets D^{S} and D^{T} as input, which contain ns and nt stay-points trajectories respectively, and outputs the IDF-weighted shared locations similarity s_{ij}^{SL} of each pair of heterogeneous trajectories. First, we map a stay-points trajectory into the grids and count the number of stay-points located in each grid as visited frequency $GV[i]$ using Algorithm 3. Thus, a trajectory is converted to a vector of visited grids with the visited frequency, which is the same as term frequency (TF) usually used in document analysis. Then we suppose to sum the visited frequency of each shared grid, where both the two visited frequencies are not zero. However, a straightly summing-up strategy, which considers that each grid has the same contribution to the similarity score, suffers from an obvious limitation. To be specific, when a location is visited by many persons, it means that the location is very popular, and if two persons do not have a particularly close relationship they may also visit this location at the same time. On the

Algorithm 2. IDF-Weighted Shared Locations Similarity

Require: $D^{\mathrm{S}} = \{ST_i^{\mathrm{S}} | 1 \leqslant i \leqslant ns\}$; $D^{\mathrm{T}} = \{ST_j^{\mathrm{T}} | 1 \leqslant j \leqslant nt\}$
Ensure: IDF-weighted shared locations similarities $s^{\mathrm{SL}} = \{(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}}, s_{ij}^{\mathrm{SL}})\}$, where s_{ij}^{SL} is the similarity of ST_i^{S} and ST_j^{T}
for each grid G_k, count the number n_k of stay-points located in the grid
for each grid G_k, calculate IDF weight by $IDF_k = \log(|D^{\mathrm{S}}| + |D^{\mathrm{T}}|)/n_k$
for each pair $(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})$ **do**
    Initialize the grids visited vectors $(GV_i^{\mathrm{S}}, GV_j^{\mathrm{T}})$ by Grid Vector
    $s_{ij}^{\mathrm{SL}} \leftarrow 0$
    for $k \leftarrow 1$ to N **do**
        if $GV_i^{\mathrm{S}}[k] > 0$ and $GV_j^{\mathrm{T}}[k] > 0$ **then**
            $s_{ij}^{\mathrm{SL}} += IDF_k \times (GV_i^{\mathrm{S}}[k] + GV_j^{\mathrm{T}}[k])$
        end if
    end for
    $s_{ij}^{\mathrm{SL}} = s_{ij}^{\mathrm{SL}}/(|ST_i^{\mathrm{S}}| + |ST_j^{\mathrm{T}}|)$
end for

Algorithm 3. Grid Vector

Require: $ST = \{sp_1, sp_2, \ldots, sp_n\}$; $\boldsymbol{G} = (n_1, n_2, \ldots, n_N)$
Ensure: the grid vector $\boldsymbol{GV}$ of ST
$\boldsymbol{GV} \leftarrow (0, 0, \ldots, 0)$
for each stay-point $sp_i \in ST$ **do**
    if sp_i located in G_k **then**
        $GV[k] = GV[k] + 1$
    end if
end for

other hand, when only a few persons have visited the particular location, the trajectories which have records on the location should have a relatively large correlation. Thus, the grids with different visited popularities should not be considered to give the same contribution to the similarity. Considering this, we introduce the inverse document frequency (IDF)[10], which is usually used in document analysis, to differentiate the contribution of each grid to solve the aforementioned limitation. If the number of stay-points n_k that locate in a grid G_k is very large, $IDF_k = \log(|D|)/n_k$ of this grid would be very small. Therefore, this grid will not offer many contributions to the similarity score of these two trajectories. Thus, with the help of IDF, we obtain the similarity by summing up the IDF-weighted visited frequencies of shared grids and normalizing the summation by the total number of trajectory points in two trajectories.

The cosine similarity is represented by the cosine distance of the IDF-weighted grid-vectors of two heterogeneous trajectories which is calculated by (1). In the equation, N denotes the number of grids, and $\boldsymbol{IDF\_GV}_i^{\mathrm{S}}$ and $\boldsymbol{IDF\_GV}_j^{\mathrm{T}}$ are the IDF-weighted visiting frequencies vector of trajectory ST_i^{S} and ST_j^{T} respectively. While IDF-weighted shared locations similarity measures the similarity with only the shared locations, the cosine similarity measures the similarity with the whole trajectories.

$$
\begin{aligned}
&IDF\_GV_i^{\mathrm{S}}[k] = IDF_k \times GV_i^{\mathrm{S}}[k],\\
&IDF\_GV_j^{\mathrm{T}}[k] = IDF_k \times GV_j^{\mathrm{T}}[k],\\
&s_{ij}^{\mathrm{CSL}} = \frac{\sum_{k=1}^{N}(IDF\_GV_i^{\mathrm{S}}[k] \times IDF\_GV_j^{\mathrm{T}}[k])}{\sqrt{\sum_{k=1}^{N} IDF\_GV_i^{\mathrm{S}}[k]^2} \times \sqrt{\sum_{k=1}^{N} IDF\_GV_j^{\mathrm{T}}[k]^2}}.
\end{aligned}
\tag{1}
$$

Temporal Distribution Features. We also notice that the heterogeneous trajectories of the same person are usually gained at similar time slots in real life, such as working hours and shopping time. For example, when a user enters a WiFi access point, the GPS sensor is still working. Thus, the WiFi trajectory and the GPS trajectory will be acquired at the same time. The temporal features represent the scenarios where the information of a person is collected in different ways at the same time. We also propose to use two measurements as the temporal distribution features to evaluate the similarity between the temporal distributions of two heterogeneous trajectories: 1) the IDF-weighted shared time slots similarity, and 2) the cosine similarity between the two vectors of temporal activities.

These two features are calculated using the similar methods for extracting spatial distribution features (Algorithm 2). As we partition the map into grids to simplify the spatial features calculation, we also partition the time into small time slots. We use time slots instead of grids to calculate these two temporal distribution features by Algorithm 2 and (1).

Spatio-Temporal Distribution Features. Along with the features extraction of aforementioned two kinds of distribution, we also combine spatial distribution and temporal distribution together to extract the joint distribution features, which indicate the similarity of mobility patterns of two heterogeneous trajectories. We employ two measurements as the spatio-temporal distribution features to evaluate the mobility pattern similarities of two heterogeneous trajectories: 1) the longest common sub-sequence (LCSS)[11], and 2) the hierarchical-graph-based similarity measurement (HGSM)[9].

These two features are often used to describe the spatio-temporal similarity between trajectories. LCSS treats a trajectory as a sequence with time information, and then uses the longest common sub-sequence (to distinguish it from the method name, we denote the longest common sub-sequence by LC in this paper) of two sequences to represent the similarity of two trajectories. When two heterogeneous stay-points locate in the same grid and have shared time slots, they are called common stay-points. Then the longest common subsequence is built by collecting all common stay-points. At last, the similarity is calculated by applying (2) on the longest common sub-sequence.

$$
s_{ij}^{\mathrm{LCSS}} = \frac{\alpha_{ij} \times \sum_{l \in LC_{ij}} IDF_{l_G}}{|ST_i^{\mathrm{S}}| \times |ST_j^{\mathrm{T}}|},
\tag{2}
$$

where l denotes the common stay-point in the longest common subsequence of these two trajectories, and l_G denotes the index of the grid which the common stay-point located in. LC_{ij} is the longest common subsequence of ST_i^{S} and ST_j^{T}. α_{ij} is a factor related to the length of LC_{ij}, which is involved in distinguishing the importance of LC_{ij} with various lengths. For instance, we use $\alpha_{ij} = |LC_{ij}|$ in our experiment. Thus, the longer the LC_{ij} is, the more related these two trajectories might be. The denominator of (2) is the normalization process, where $|ST_i^{\mathrm{S}}|$ is the number of stay-points in ST_i^{S}, and $|ST_j^{\mathrm{T}}|$ is the number of stay-points in ST_j^{T}.

Compared with exactly matching the coincidence of the two trajectories in spatio-temporal dimensions of LCSS, HGSM focuses on the similarity of the transfer patterns of the two trajectories. Different from LCSS that uses the longest common sub-sequence, HGSM investigates the longest similar sub-sequence (LS) to help measure the similarity of two trajectories. Unlike common sub-sequence that requires both stay-points of a common stay-point share the same grid and same time slot, similar sub-sequence just requires the similar transfer time interval between the pair of similar stay-points. Thus, HGSM can measure the similarity of two heterogeneous trajectories even when there is no temporal intersection between them. Suppose there are two sequences of stay-points $seq = \{sp_1, sp_2, \ldots, sp_n\}$ and $seq' = \{sp'_1, sp'_2, \ldots, sp'_n\}$, seq and seq' are similar sequences if and only if they satisfy the following conditions: 1) $\forall 1 \leqslant i \leqslant n$, sp_i shares the same grid with sp'_i; 2) $\forall 1 \leqslant i < n$, $\frac{|\delta t_i - \delta t'_i|}{\max(\delta t_i, \delta t'_i)} \leqslant p$, where p is a predefined ratio threshold, and $\delta t_i = sp_{i+1}.t - (sp_i.t + sp_i.\Delta t)$ ($\delta t'_i = sp'_{i+1}.t - (sp'_i.t + sp'_i.\Delta t)$) is the transfer time between two adjacent stay-points. Due to different calculations, each pair of heterogeneous trajectories will have several longest similar sub-sequences. Then the similarity score s_{ij}^{HGSM} is obtained by summing up all s_{ij}^{LS}s using (3).

$$s_{ij}^{\mathrm{HGSM}} = \sum_{LS_{ij}^k \in LS_{ij}} \frac{\alpha_{ij}^k \times \sum\limits_{l \in LS_{ij}^k} IDF_{l_G}}{|ST_i^{\mathrm{S}}| \times |ST_j^{\mathrm{T}}|}, \tag{3}$$

where LS_{ij}^k denotes one of the longest similar sub-sequences of ST_i^{S} and ST_j^{T}, α_{ij}^k is the factor related to the length of LS_{ij}^k, $LS_{ij} = \{LS_{ij}^1, LS_{ij}^2, \ldots\}$ stores all the longest similar sub-sequences of the two trajectories, and l and l_G are the same as in (2).

3.1.3 Linkable Probability Calculation

After extracting all the three types of features in Subsection 3.1.2, we can train a binary classifier for trajectory linking inference. In general, various binary classifiers can be employed, such as support vector machine (SVM)[12] or logistic regression[13]. However, there exists a difficulty that the labels in the datasets are imbalanced, which makes most of the traditional classifiers (e.g., SVM) invalid. For example, suppose we have N trajectories in each dataset and N pairs of them are labeled linkable, then the number of unlinkable pairs is as large as $N(N-1)$. As N increases, the data becomes more and more imbalanced. As a result, it is almost impossible to get a linkable prediction from traditional classifiers practically.

There are lots of studies trying to solve the imbalanced learning problem, such as re-sampling techniques[14], cost-sensitive methods[15], and ensemble methods[16]. Re-sampling techniques, such as over-sampling[17] and under-sampling[18], change the number of training data in order to balance the positive samples and the negative samples. However, these techniques may cause data redundancy (over-sampling) or data loss (under-sampling) depending on different methods. Cost-sensitive methods treat different misclassifications differently to solve the classification problem for imbalanced data[15], while traditional classifiers do not take misclassification costs into consideration. However, it is non-trivial to quantify the misclassification costs in practice. In contrast, ensemble methods combine multiple learners to improve the performance, which can conceptually avoid the above drawbacks[16]. Moreover, the generalization of the ensemble methods can be guaranteed so that it performs well in both training set and test set. For instance, we choose the random forests[19], a representative of the ensemble classifiers, which is widely-acknowledged for its efficiency and easy implementing, to train and classify the imbalanced data.

After obtaining scores $\{Scores(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}}) | ST_i^{\mathrm{S}} \in D^{\mathrm{S}}, ST_j^{\mathrm{T}} \in D^{\mathrm{T}}\}$ from the random forests classifier, we calculate the linkable probabilities $lp(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})$ of ST_i^{S} and ST_j^{T} by (4). Hence, $lp(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})$ may be not equal to $lp(ST_j^{\mathrm{T}}, ST_i^{\mathrm{S}})$.

$$lp(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}}) = \frac{Scores(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})}{\sum_{1 \leqslant j \leqslant |D^{\mathrm{T}}|} Scores(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})}. \tag{4}$$

3.1.4 One-to-One Constraint Inference

After the linkable probabilities being extracted, we are able to select the linkable target heterogeneous trajectory with the largest linkable probability for each source trajectory. Unfortunately, one target heterogeneous trajectory may be the most linkable one for several source trajectories at the same time. As shown in Fig.1(a), T_1^{S} and T_2^{S} will both choose T_1^{T} as the linkable trajectory, which leads to the violation of the one-to-one constraint in trajectory linking and inaccuracy of linking results. Thus, we propose an inference step to link the heterogeneous trajectories in a one-to-one manner.

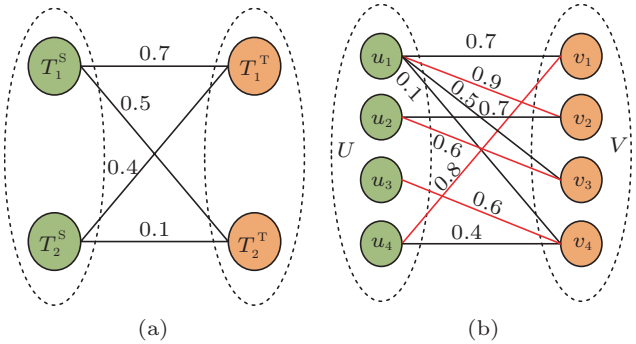

Fig.1. (a) Example that one target trajectory is the most linkable one for two source trajectories. (b) Example of weighted bipartite graph matching.

We formulate the inference problem as a weighted bipartite graph matching problem based on the probabilities provided by the binary classifier. As shown in Fig.1(b), in this problem, we have two disjoint vertex sets: U in which each vertex u_i represents a source trajectory ST_i^{S} and V in which each vertex v_j represents a target trajectory ST_j^{T}. e_{ij} in edge set E connects u_i and v_j, and the weight w_{ij} represents the linkable probability $lp(ST_i^{\mathrm{S}}, ST_j^{\mathrm{T}})$. A maximum weighted bipartite matching is a subset $M \in E$ without common vertices, where the sum of the weights of the edges in the matching has a maximal value. For example, $M = \{e_{12}, e_{23}, e_{34}, e_{41}\}$ is the maximum weighted bipartite matching of the graph in Fig.1(b). Thus, the one-to-one constraint inference problem is transferred to find the maximum weighted bipartite matching of the graph $G = (U, V, E)$. To complete the bipartite graph, we insert edges with weight zero for those unconnected vertices.

Since finding such a matching has already been well solved, we employ two efficient algorithms to find the maximum matching in this paper. The first is the classic Kuhn-Munkres algorithm (KM)[20] which aims to find the maximum weighted matching, while the second is the recently proposed multi-network anchoring (MNA)[21] which uses the ranking of the weights rather than the scores to reduce the influence of uncalibrated weights. These two algorithms both can efficiently solve the one-to-one constraint inference problem and improve the ultimate accuracy of the trajectory linking problem. We compare these two algorithms with the linkable probabilities calculated by our approach to demonstrate whether the linkable probabilities are calibrated or not.

4 Experiments

In this section, we conduct extensive experiments on real-world trajectory databases to demonstrate the performance of our approach. We first describe our experimental settings and data preparations in the following subsection. Then we compare our algorithm with several existing methods including traditional trajectory similarity measurements and trajectory linking methods. Finally, we give a discussion of our experimental results.

4.1 Experimental Settings and Data Preparations

We use two trajectory datasets in our experiments. The first one is a synthetic dataset, which has been used in [8], denoted as T-Drive dataset and the second is a real-world trajectory dataset, denoted as University dataset.

• *T-Drive Dataset*①. This dataset consists of trajectories of about 10 000 taxies in Beijing within one week in Feb. 2008. The data is obtained by GPS sensors on taxies, and the average sampling interval is around 177 seconds. Each record of a trajectory consists of three attributes: longitude, latitude and time stamp, and different trajectories are distinguished by different numeral IDs. We randomly select 1 000 trajectories in this dataset for experiments. As the same as the dataset used in [8], we manually split one original trajectory into two trajectories by randomly dropping each individual trajectory point into one of the two trajectories with the same probability, since the dataset contains only one category of data. Thus, the two artificial trajectories have the same length of timespan, a week, as the original trajectory, and there is no common trajectory point within them. This splitting method simulates the fact that a taxi is traced by two different methods. Then we randomly drop the two trajectories into two different datasets separately to construct the two heterogeneous trajectory datasets. Hence, we get two heterogeneous trajectory datasets, T^{S} as the source trajectory dataset and T^{T} as the target trajectory dataset, and each contains 1 000 trajectories and the average sampling interval is around 354 seconds. Again, we down-sample trajectories to study the performance under different sparsity.

• *University Dataset.* The University dataset consists of two heterogeneous trajectory datasets collected

①http://research.microsoft.com/apps/pubs/?id=152883, Dec. 2017.

from 500 students of a local university within one month in 2016. One of them consists of the connecting records of WiFi access points (APs) in which the trajectory ID is the MAC address of mobile devices, and the other is built by collecting the smart card check-in records in which the trajectory ID is the student identification number. Since both WiFi APs and smart card machines are mounted in the building, the locations of the trajectory records are the building locations in those two datasets. We pre-process these raw data to obtain the ground-truth of linkable heterogeneous trajectories using extra identity information. If a user has logged in the network account with his or her identity information through a specific mobile device, we treat that the mobile device belongs to the user, and the trajectory of the mobile device and the trajectory of the user's smart card are linkable heterogeneous trajectories. In order to protect the privacy of users, all IDs are anonymized with unique numbers in our experiments. These two datasets are unique, sparse and overlapping. We experiment with these two datasets to illustrate how our approach can handle the trajectory linking problem even with these features.

We summarize the experimental data in Table 1. The first experiment on the T-Drive dataset aims to evaluate the performance of approaches on a general scene, namely the urban area. The trajectories of taxies in the T-Drive dataset show the trajectories of the taxi drivers. To preprocess the trajectories, we transform the original trajectories to stay-points trajectories using stay-points detection method which is described in Subsection 3.1.1. This method can not only uniform the heterogeneous trajectories, but also filter the impact of passengers on the drivers' trajectories, since the stay regions mainly indicate the home, the familiar roads of drivers, etc. Thus, the experiment on the T-Drive dataset shows the performance of our approach to link the trajectories of the drivers. By tuning the sampling rate of trajectories in T^{S} and T^{T}, e.g., to sample 20% trajectory points in T^{S} and T^{T} to form $T^{\mathrm{S}}_{\mathrm{A}}$ and $T^{\mathrm{T}}_{\mathrm{A}}$ respectively, we form six different trajectory datasets from the T-Drive dataset ($T^{\mathrm{S}}_{\mathrm{A}}$, $T^{\mathrm{T}}_{\mathrm{A}}$, $T^{\mathrm{S}}_{\mathrm{B}}$, $T^{\mathrm{T}}_{\mathrm{B}}$, $T^{\mathrm{S}}_{\mathrm{C}}$ and $T^{\mathrm{T}}_{\mathrm{C}}$). We summarize the source dataset and target dataset with same sampling rate as one in Table 1 (e.g., $T^{\mathrm{S}}_{\mathrm{A}}$ and $T^{\mathrm{T}}_{\mathrm{A}}$ are summarized as T_{A} in Table 1, so as T_{B} and T_{C}). In particular, the number of locations in Table 1 is calculated after the stay-points detection and grids mapping of the trajectories. Thus, it is actually the number of different grids that have appeared throughout the whole trajectory dataset. The greater the average visited times vt of a location, which is the ratio between the total number of all stay-points and the number of locations, the greater the degree of overlap between two different trajectories. In T_{A}, the number of locations is 128 929, the total number of stay-points is $79.56 \times 1\,000 \times 2 = 159\,120$, and thus the vt of T_{A} is 1.23. The second experiment on the University dataset demonstrates the effectiveness of our approach on the special scene with high sparsity and high overlap. U_{S} consists of smart card check-in records and U_{W} is collected from the WiFi access information.

Even if we do not down-sample the University data, the time interval between two stay-points is larger than that in the T-Drive dataset, which means the University data is sparser. Meanwhile, there are only 24 different locations (buildings in the University dataset) in smart card check-in trajectories ($vt = 1\,445.42$) and 59 different locations in WiFi trajectories ($vt = 471.02$), which means the trajectories are highly overlapped.

In our experiments, we partition the trajectories in each dataset into groups using 5-fold cross-validation: one fold is used as testing data, and the remaining folds are used as training data. We report the average results and standard deviations of 5-fold cross-validation on both datasets. For experiments on the T-Drive dataset, each fold contains 200 trajectories. And each fold contains 100 trajectories for experiments on the University dataset.

Table 1. Experiments Data from the T-Drive Dataset and the University Dataset

	Sampling Rate	Records	Stay-Points	Timediff (h)	Location	Visited Times
T_{A}	0.20	136.758	79.56	1.13	128 929	1.23
T_{B}	0.10	69.190	45.34	2.19	66 765	1.63
T_{C}	0.05	34.890	25.10	4.17	34 166	1.47
U_{S}	1.00	69.380	69.38	11.19	24	1 445.50
U_{W}	1.00	55.580	55.58	14.47	59	471.00

Note: Records: the mean number of trajectory points in each trajectory. Stay-points: the mean number of stay-points of each trajectory after stay-points detection. Timediff: the mean time interval (hours) between two stay-points. Location: the different locations in the whole dataset. Visited times: the average visited times of a location.

4.2 Methods Comparison

In order to evaluate the performance of trajectory linking, we evaluate different approaches in terms of precision (PRE), recall (REC), $F1$-measure ($F1$), accuracy (ACC) and area under receiver operating characteristic (AUC). The first four measures can evaluate the trajectory linking performance, while AUC evaluates the ranking performance.

For the sake of demonstrating the effectiveness of our approach, we compare the following methods by the aforementioned evaluation measures.

• *One-to-One Constraint Trajectory Linking* (*OCTL*). It is the proposed approach in this paper. OCTL exploits multidimensional information from heterogeneous trajectory datasets. In addition, OCTL introduces the one-to-one constraint inference to facilitate the trajectory linking. We use OCTL_KM to denote the inference with the KM algorithm[20] and OCTL_MNA to denote it with the MNA algorithm[21].

• *Similar Time Sequences Search Methods.* They are the similarity measurements used for searching similar homogeneous time sequences, including shared locations similarity (SL) which counts the number of similar locations and LCSS[11]. In this paper, we use the performance of these commonly used methods as the baselines. Different from the original methods, we also use grids and time slots in the calculation of these two similarities.

• *State-of-the-Art Methods in Trajectory Linking.* They are proposed in [7] and [8]. These studies aim to solve the trajectory linking problem and both have achieved great performance on their own datasets. However, AUI[7] can hardly distinguish the overlapped data, while FTL[8] performs badly on sparse data.

Besides, we also conduct experiments using the following methods to illustrate the performance of one-to-one inference.

• *OCTL Without One-to-One Inference* (*OCTL_NO*). It is our approach without one-to-one constraint inference. We choose the target trajectory with the largest linkable probability to link with a source trajectory.

• *State-of-the-Art Methods in Trajectory Linking with One-to-One Constraint Inference.* They are the approaches that use a one-to-one constraint inference algorithm to infer the linkable pairs with the linkable probabilities calculated by AUI[7] and FTL[8], denoted by AUI_IN and FTL_IN respectively in the experiments.

For fair comparisons, the parameters are set as the same for all the compared methods. To be specific, the length of the grid is set to 200 meters in the experiments on the T-Drive dataset, and we treat each building as a grid in the experiments on the University dataset since the smart card readers and WiFi APs are mounted in the buildings and the locations are represented by the locations of buildings. Also, the length of the time slot is set to six minutes in the experiments on both datasets.

Moreover, we also evaluate different approaches in terms of the top k hit rate, which is the probability that the linkable target trajectory is within the k candidates of a specific source trajectory. In our method, the k candidates are selected according to the top k highest linkable probabilities for a source trajectory, while both AUI and FTL have their own way to generate candidates. This evaluation illustrates the efficiency of our approach without the link inference step.

In the real-world trajectory linking problem, the data is sampled at different rates. Thus, we first study the performance of the proposed OCTL method on different trajectory datasets with different sampling rates compared with the aforementioned comparative methods. The results of all compared methods are reported in Table 2. The best performance on each of the measurements is listed in bold.

4.3 Performance of Trajectory Linking

Table 2 shows that our approach outperforms the common-used baselines and the state-of-the-art trajectory linking methods, especially on the sparse data. The precision, recall and $F1$ scores measure the minor label, i.e., the linkable pair, in our classification problem, and the accuracy scores measure both minor and major labels. Thus, the accuracy scores are larger than 0.99 in almost all of our experiments due to the data imbalance problem. Since we aim to find the linkable pairs of heterogeneous trajectories, we care the performance of minor label (e.g., precision, recall and $F1$) more than the accuracy scores.

According to the characteristics of the test data, it is much easier to link heterogeneous trajectories on the urban areas (e.g., the T-Drive dataset) than on the university or company areas (e.g., the University dataset). Since the trajectories of different persons are little overlapped in the urban areas, it is much easier to distinguish the linkable pairs and unlinkable pairs in the university areas. Thus, all methods that use only spatial information achieve good performance such as AUI and SL. The experimental results also show that

Table 2. Performance Comparison of Different Methods for Trajectory Linking

		OCTL_KM	OCTL_MNA	OCTL_NO	AUI	FTL	SL	LCSS
T_A^S vs T_A^T	PRE	**0.992 50±0.005 00**	**0.992 50±0.005 00**	0.990 00±0.000 00	0.907 50±0.020 60	0.973 40±0.007 60	0.870 00±0	0.949 20±0
	REC	**0.992 50±0.005 00**	**0.992 50±0.005 00**	0.990 00±0.000 00	0.907 50±0.020 60	0.958 80±0.007 50	0.870 00±0	0.840 00±0
	$F1$	**0.992 50±0.005 00**	**0.992 50±0.005 00**	0.990 00±0.000 00	0.907 50±0.020 60	0.966 00±0.007 60	0.870 00±0	0.891 20±0
	ACC	**0.999 93±0.000 05**	**0.999 93±0.000 05**	0.999 90±0.000 00	0.999 08±0.000 21	0.999 66±0.000 08	0.998 70±0	0.998 98±0
	AUC	**0.999 97±0.000 01**	**0.999 97±0.000 01**	**0.999 97±0.000 01**	0.998 16±0.000 87	0.970 21±0.005 77	0.998 40±0	0.924 42±0
T_B^S vs T_B^T	PRE	**1.000 00±0.000 00**	0.997 50±0.005 00	0.977 50±0.002 90	0.806 30±0.021 70	0.652 70±0.012 60	0.860 00±0	0.906 70±0
	REC	**0.993 80±0.002 50**	0.991 30±0.004 80	0.977 50±0.002 90	0.806 30±0.021 70	0.638 80±0.010 30	0.860 00±0	0.680 00±0
	$F1$	**0.996 90±0.001 30**	0.994 40±0.004 70	0.977 50±0.002 90	0.806 30±0.021 70	0.645 60±0.011 10	0.860 00±0	0.777 10±0
	ACC	**0.999 97±0.000 01**	0.999 89±0.000 10	0.999 78±0.000 03	0.998 06±0.000 21	0.996 49±0.000 12	0.998 60±0	0.998 05±0
	AUC	**0.996 69±0.001 25**	**0.996 69±0.001 25**	**0.996 69±0.001 25**	0.992 97±0.003 36	0.983 64±0.004 57	0.992 44±0	0.847 13±0
T_C^S vs T_C^T	PRE	**0.910 00±0.017 20**	0.893 10±0.033 40	0.862 50±0.026 30	0.690 00±0.035 40	0.245 30±0.033 40	0.820 00±0	0.966 10±0
	REC	**0.885 00±0.017 80**	0.867 50±0.033 80	0.862 50±0.026 30	0.690 00±0.035 40	0.195 00±0.023 80	0.820 00±0	0.570 00±0
	$F1$	**0.897 30±0.017 50**	0.880 70±0.032 40	0.862 50±0.026 30	0.690 00±0.035 40	0.217 30±0.027 80	0.820 00±0	0.717 00±0
	ACC	**0.998 99±0.000 17**	0.998 82±0.000 33	0.998 63±0.000 26	0.996 90±0.000 35	0.992 97±0.000 31	0.998 20±0	0.997 75±0
	AUC	**0.972 62±0.001 51**	**0.972 62±0.001 51**	**0.972 62±0.001 51**	0.968 92±0.008 35	0.942 65±0.011 56	0.970 80±0	0.784 83±0
T_A^S vs T_B^T	PRE	**0.997 50±0.005 00**	0.992 50±0.005 00	0.985 00±0.009 90	0.856 30±0.032 20	0.879 60±0.022 60	0.860 00±0	0.906 80±0
	REC	**0.991 30±0.006 30**	0.986 30±0.004 80	0.981 30±0.002 50	0.856 30±0.032 20	0.867 50±0.023 30	0.860 00±0	0.760 00±0
	$F1$	**0.994 40±0.005 20**	0.989 30±0.004 30	0.981 30±0.002 50	0.856 30±0.032 20	0.873 00±0.022 20	0.860 00±0	0.808 90±0
	ACC	**0.999 94±0.000 05**	0.999 89±0.000 04	0.996 10±0.007 40	0.998 56±0.000 32	0.998 74±0.000 23	0.998 60±0	0.998 28±0
	AUC	0.996 78±0.002 39	0.996 78±0.002 39	0.996 78±0.002 39	**0.997 75±0.000 97**	0.977 54±0.002 20	0.995 13±0	0.881 99±0
T_A^S vs T_C^T	PRE	**0.997 50±0.005 10**	0.983 70±0.007 50	0.966 30±0.006 30	0.767 50±0.019 40	0.575 80±0.010 30	0.845 00±0	0.926 20±0
	REC	**0.992 50±0.008 70**	0.977 50±0.006 50	0.966 30±0.006 30	0.767 50±0.019 40	0.560 00±0.012 20	0.845 00±0	0.690 00±0
	$F1$	**0.995 00±0.006 80**	0.980 60±0.006 90	0.966 30±0.006 30	0.767 50±0.019 40	0.567 80±0.011 30	0.845 00±0	0.790 80±0
	ACC	**0.999 95±0.000 07**	0.999 81±0.000 07	0.999 66±0.000 06	0.997 68±0.000 19	0.995 74±0.000 10	0.998 45±0	0.998 18±0
	AUC	**0.997 26±0.002 01**	**0.997 26±0.002 01**	**0.997 26±0.002 01**	0.991 97±0.004 07	0.971 88±0.004 45	0.996 79±0	0.859 63±0
T_B^S vs T_C^T	PRE	**0.949 90±0.005 00**	0.949 80±0.011 60	0.892 50±0.005 00	0.762 50±0.011 90	0.253 30±0.021 40	0.810 00±0	0.930 80±0
	REC	**0.925 00±0.007 10**	0.922 50±0.015 50	0.892 50±0.005 00	0.762 50±0.011 90	0.233 80±0.015 50	0.810 00±0	0.605 00±0
	$F1$	**0.937 30±0.006 10**	0.935 90±0.013 60	0.892 50±0.005 00	0.762 50±0.011 90	0.243 10±0.018 20	0.810 00±0	0.733 30±0
	ACC	**0.999 38±0.000 06**	0.999 37±0.000 13	0.998 93±0.000 05	0.997 63±0.000 12	0.992 64±0.000 33	0.998 10±0	0.997 80±0
	AUC	0.982 32±0.002 37	0.982 32±0.002 37	0.982 32±0.002 37	**0.987 18±0.002 71**	0.891 51±0.007 13	0.977 08±0	0.807 27±0
U_S vs U_W	PRE	0.604 10±0.044 40	**0.606 40±0.043 40**	0.422 50±0.018 90	0.057 50±0.012 60	0.338 50±0.029 00	0.040 00±0	0.160 00±0
	REC	**0.572 50±0.043 50**	0.570 00±0.040 80	0.422 50±0.018 90	0.057 50±0.012 60	0.305 00±0.031 10	0.040 00±0	0.160 00±0
	$F1$	**0.587 90±0.043 90**	0.587 60±0.042 10	0.422 50±0.018 90	0.057 50±0.012 60	0.320 90±0.030 20	0.040 00±0	0.160 00±0
	ACC	0.991 98±0.000 85	**0.992 00±0.000 82**	0.988 45±0.000 38	0.981 15±0.000 25	0.987 10±0.000 48	0.980 80±0	0.983 20±0
	AUC	**0.870 79±0.006 94**	**0.870 79±0.006 94**	**0.870 79±0.006 94**	0.679 66±0.013 03	0.729 89±0.017 43	0.624 46±0	0.799 63±0

Note: T_A^S vs T_A^T means to find the linkable target trajectory in T_A^T for each source trajectory in T_A^S.

the stay-points detection successfully filters the impact of passengers on the drivers' trajectories. Especially, SL outperforms AUI on the T-Drive dataset when the data is sparse (e.g., T_{C}). The reason is that AUI uses a threshold to filter the pairs of heterogeneous trajectories with small linkable probabilities, which makes it ignore some truly linkable trajectories when there are insufficient shared locations between them, while SL just selects the pair with the largest linkable probability.

Influence of Sparsity. On the dense data, such as T_{A} and T_{B}, all methods perform well, while our approach finds out more than 99% pairs of linkable trajectories on both of the datasets. However, when the data is sparse as in T_{C}, the performance of AUI and FTL quickly drops, while the $F1$ score of our approach is still 20% larger than AUI and 67% larger than that of FTL. The large time duration between two records makes FTL hard to find the unreachable points, which makes it perform the worst on this dataset among the experimented methods.

Influence of Overlap. When the data is highly overlapped as in U_{S} and U_{W}, shared locations based methods, such as AUI, SL, and LCSS, almost cannot find any pair of linkable trajectories. Nevertheless, FTL achieves better performance on U_{S} and U_{W} than on T_{C} even with the longer time duration. The reason is that the speed limitation in the University dataset (15 km/h) is smaller than that in the T-Drive dataset (120 km/h) according to the different transportation (e.g., bicycles and cars). Even the University dataset is extremely sparse and overlapped, we still find out nearly 60% of all pairs of linkable trajectories, which is impossible for the compared work.

Besides, OCTL_KM outperforms OCTL_MNA on all datasets, which means our approach can extract the calibrated linkable probabilities. Thus, in the following experiments, we use OCTL_KM to illustrate the performance of our approach.

Despite the fact that OCTL achieves the best performance in finding linkable trajectories, the computational complexity of our approach is much higher than that of the compared methods. The most time-cost stage of our approach, i.e., the first stage, in which we extracted six different kinds of features, is at least six times more complex than that of AUI and FTL.

4.3.1 *Effectiveness of Linkable Probability*

Our approach consists of two main stages: linkable probability extraction and link inference. We firstly evaluate the performance of linkable probability extraction of our method.

The performance of linkable probability can be measured with the top k hit rate, which is frequently employed in prior studies since they are lack of the one-to-one inference ability. Also, AUC is the measurement commonly used in ranking problems to evaluate the ranking performance.

In Table 3, we compare the top k ($k = 1, 3, 5$) hit rate among different approaches on T_{B} and the University dataset, and the best performance on each of the measurements is listed in bold. These two experiments are representative since datasets $T_{\mathrm{B}}^{\mathrm{S}}$ and $T_{\mathrm{B}}^{\mathrm{T}}$ have the proper density to illustrate the different performance of different approaches and the datasets U_{S} and U_{W} have the characteristics of sparseness and overlap. The results show that our linkable probability is more precise to express the linkable relationship between two heterogeneous trajectories.

Table 3. Top k Hit Rate of Linkable Trajectories

	$T_{\mathrm{B}}^{\mathrm{S}}$ vs $T_{\mathrm{B}}^{\mathrm{T}}$			U_{S} vs U_{W}		
	$k=1$	$k=3$	$k=5$	$k=1$	$k=3$	$k=5$
OCTL	**0.977 4**	**0.988 8**	**0.991 3**	**0.422 5**	**0.625 0**	**0.680 0**
AUI	0.806 3	0.901 3	0.932 5	0.057 5	0.142 5	0.212 5
FTL	0.638 8	0.842 5	0.850 0	0.305 0	0.390 0	0.430 0
SL	0.860 0	0.925 0	0.950 0	0.040 0	0.130 0	0.150 0
LCSS	0.680 0	0.695 0	0.695 0	0.160 0	0.300 0	0.420 0

4.3.2 *Effectiveness of One-to-One Constraint Inference*

In this experiment, we conduct the one-to-one constraint inference with both our linkable probabilities and the linkable probabilities of prior work. Since AUI uses two indicators to link the heterogeneous trajectories, which cannot be directly used as the input of an one-to-one inference algorithm, we normalize the two indicators separately, add the two normalized indicators together to form the linkable probability, and apply KM to infer linkable pairs with the linkable probabilities. The results in Fig.2 show the $F1$ scores of each approach on different datasets. The $F1$ score is a representative measurement to demonstrate the performance of a classification problem, which is the combination of precision and recall. For example in Table 2, the precision scores of LCSS are quite high while its recall scores are very low. Thus, we use $F1$ scores to illustrate the effectiveness of the one-to-one constraint inference. Fig.2 shows that the one-to-one constraint inference can effectively improve the selection of linkable target trajectory

for each source trajectory rather than simply choose the one with the largest linkable probability. The one-to-one inference works well in most cases when the linkable probabilities are correctly calculated. And the more difficult to distinguish the candidates, as experiments on the University dataset, the better performance the one-to-one inference can achieve.

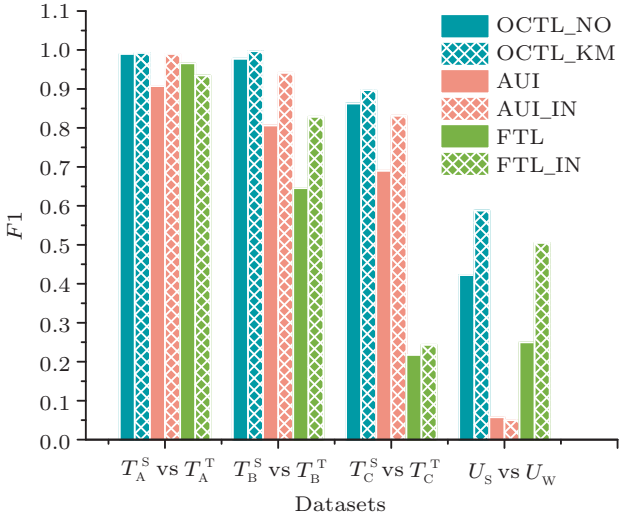

Fig.2. Effectiveness of one-to-one constraint inference.

4.3.3 *Effectiveness of Different Features*

Fig.3 and Fig.4 show importance of different distribution features to link heterogeneous trajectories, including spatial, temporal and spatio-temporal, denoted by OCTL_SP, OCTL_TM, and OCTL_ST respectively. These results are calculated in the same way as OCTL, except that only features of a certain distribution (spatial, temporal, or spatio-temporal distribution) are used instead of all.

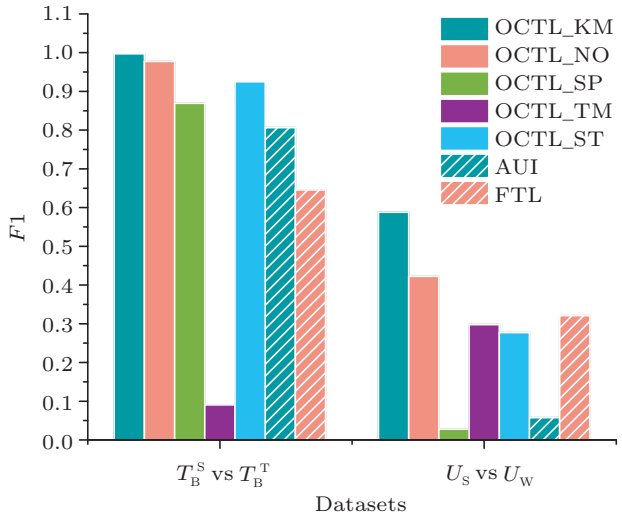

Fig.3. $F1$ scores of different methods.

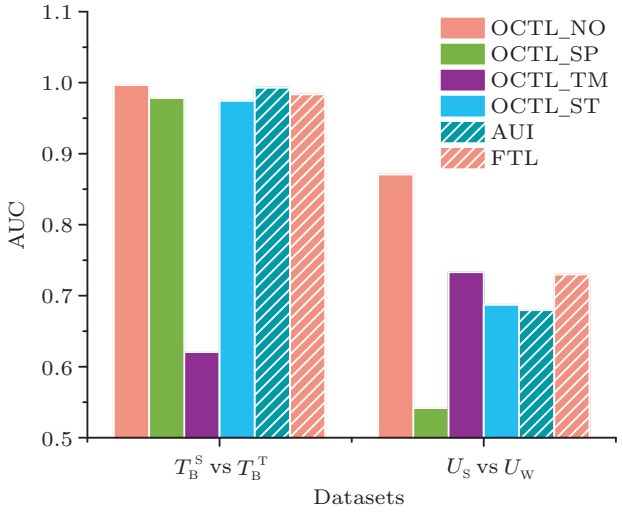

Fig.4. AUC scores of different methods.

On the T-Drive dataset, the same as the spatial information based methods (AUI, SL, LCSS), the spatial distribution features and spatio-temporal distribution features make the most of the contributions to our performance and the temporal distribution features contribute little on the linking of the T-Drive dataset due to the construction method of the experimental dataset.

We randomly select the GPS records into two different datasets; thus the two linkable heterogeneous trajectories have few common time slots, which leads to the inaccuracy of the temporal distribution features. On the University dataset, the two heterogeneous trajectories are generated at the similar time, which makes the temporal distribution features contribute a lot to the final results. On the contrary, spatial distribution features can hardly handle the overlapped trajectories while they perform well in the urban area. In both datasets, the spatio-temporal distribution features play an important role in facilitating the performance of OCTL. Better performance is achieved by combining the three distributions of features than by using only one distribution of features in the OCTL algorithm.

5 Related Work

A closely related topic is similar time sequences search[6], with a wide range of applications from research to business. As the distance-based methods in entity resolution, they first define a distance function, and then use the distance function to measure the similarity between two time sequences and search the similar time sequences for a certain sequence. Representative distance functions include Euclidean Distance[22],

Edit Distance on Real sequence (EDR)[23], and Longest Common Sub-Sequence (LCSS)[11]. Agrawal *et al.*[22] used discrete Fourier transform (DFT) to map time sequences to the frequency domain, and used the Euclidean distance of the Fourier coefficients to represent the similarity of two time sequences. Chen and Ng[23] took the idea of edit distance in textual strings similarity, which searches for the minimum number of edit operations (insertion, deletion, and replacement) to transform one time sequence to another and uses the number to measure the similarity. LCSS[11] also treats the time sequences as strings and calculates the similarity only by the longest common sub-sequence of the two time sequences. Thus, it is robust to the noise and variation of the sampling rate. Besides the above distance functions, others include but are not limited to dynamic time wrapping[22], locality in-between polylines[24], edit distance with real penalty[25] and synchronous euclidean distance[26]. Wang *et al.*[27] made an experimental comparison of various distance functions. Those methods are well reviewed in [1] and [6]. However, those methods were designed for measuring the similarity between homogeneous trajectories; thus they cannot simply be adopted to solve the trajectory linking problem.

Few studies have been proposed to solve the trajectory linking problem[7-8]. Fuzzy trajectory link (FTL)[8] and automatic user identification (AUI)[7] are two representative studies. FTL aligns trajectory points from two trajectories by increasing time order to form a new fusion trajectory. Then a filter model is trained by statistically analyzing the reachability of two trajectory points under a certain speed limit. And finally, FTL uses the filter model to determine whether the two trajectories can be linked. AUI takes the idea of shared location similarity measurement and proposes a grid-based share location similarity algorithm. Similar to the other work[1] which calculates trajectory similarity in urban areas, AUI also divides the map area into grids. Instead of just counting the location points in a grid, AUI presents a new similarity measurement called signal based similarity (SIG). The linkage probability of two trajectories is then represented by the SIG and the weighted Jaccard similarity (WJS) of their shared grids. Both FTL and AUI have achieved considerable hit rates on their experimental datasets. However, both of them are designed depending on their own data characteristics and do not adapt well to the general situation. For example, FTL cannot handle sparse data because any two locations are reachable at any speed limit for a sufficiently long time span. In addition, both the two studies are lack of the ability to accurately find out the target trajectory which is truly linkable for the source trajectory, and instead they only generate several candidate linkable target trajectories for a source trajectory.

6 Conclusions

In this paper, we investigated and studied the problem of trajectory linking across multiple heterogeneous mobility data sources. Different from prior work, we conducted the multi-dimensional information, i.e., spatial, temporal and spatio-temporal information, to describe the corresponding relationship between two heterogeneous trajectories in a more comprehensive manner. Thus, our method can outperform the existing work in different scenarios, especially on the scenarios with sparse or overlapped data. Besides, we also proposed a one-to-one constraint inference step to facilitate the linking performance and to fit the one-to-one constraint of the trajectory linking problem, which is absent in prior work. The extensive experiments on two real-world heterogeneous trajectory databases showed that our approach can effectively predict the pairs of linkable heterogeneous trajectories w.r.t. one-to-one constraint across multiple heterogeneous mobility data sources.

In the future, we would like to study the trajectory linking problem on the scenario where two heterogeneous trajectory datasets have no intersection on both spatial and temporal space, which cannot be handled by existing work. Besides, the privacy problem caused by trajectory linking will be studied in future work.

References

[1] Zheng Y. Trajectory data mining: An overview. *ACM Trans. Intelligent Systems and Technology* (*TIST*), 2015, 6(3): Article No. 29.

[2] Wang Y Z, Yuan N J, Lian D F, Xu L L, Xie X, Chen E H, Rui Y. Regularity and conformity: Location prediction using heterogeneous mobility data. In *Proc. the 21st ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, August 2015, pp.1275-1284.

[3] Zheng Y, Zhang L Z, Ma Z X, Xie X, Ma W Y. Recommending friends and locations based on individual location history. *ACM Trans. the Web* (*TWEB*), 2011, 5(1): Article No. 5.

[4] Xiao X Y, Zheng Y, Luo Q, Xie X. Inferring social ties between users with human location history. *Journal of Ambient Intelligence and Humanized Computing*, 2014, 5(1): 3-19.

[5] Zheng Y, Capra L, Wolfson O, Yang H. Urban computing: Concepts, methodologies, and applications. *ACM Trans. Intelligent Systems and Technology* (*TIST*) 2014, 5(3): Article No. 38.

[6] Esling P, Agon C. Time-series data mining. *ACM Computing Surveys* (*CSUR*), 2012, 45(1): Article No. 12.

[7] Cao W, Wu Z W, Wang D, Li J, Wu H S. Automatic user identification method across heterogeneous mobility data sources. In *Proc. the 32nd Int. Conf. Data Engineering*, May 2016, pp.978-989.

[8] Wu H Y, Xue M Q, Cao J N, Karras P, Ng W S, Koo K K. Fuzzy trajectory linking. In *Proc. the 32nd Int. Conf. Data Engineering*, May 2016, pp.859-870.

[9] Li Q N, Zheng Y, Xie X, Chen Y K, Liu W Y, Ma W Y. Mining user similarity based on location history. In *Proc. the 16th ACM SIGSPATIAL Int. Conf. Advances in Geographic Information Systems*, November 2008.

[10] Jones K S. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 1972, 28(1): 11-21.

[11] Das G, Gunopulos D, Mannila H. Finding similar time series. In *Proc. the 1st European Symp. Principles of Data Mining and Knowledge Discovery*, June 1997, pp.88-100.

[12] Cortes C, Vapnik V. Support-vector networks. *Machine Learning*, 1995, 20(3): 273-297.

[13] Walker S H, Duncan D B. Estimation of the probability of an event as a function of several independent variables. *Biometrika*, 1967, 54(1/2): 167-179.

[14] Chawla N V. Data mining for imbalanced datasets: An overview. In *Data Mining and Knowledge Discovery Handbook*, Maimon O, Rokach L (eds.), Springer, 2009, pp.875-886.

[15] Chai X Y, Deng L, Yang Q, Ling C X. Test-cost sensitive naive Bayes classification. In *Proc. the 4th IEEE Int. Conf. Data Mining*, November 2004, pp.51-58.

[16] Dietterich T G. Ensemble methods in machine learning. In *Proc. the 1st Int. Workshop on Multiple Classifier Systems*, June 2000.

[17] Chawla N V, Bowyer K W, Hall L O, Kegelmeyer W P. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002, 16(1): 321-357.

[18] Raskutti B, Kowalczyk A. Extreme re-balancing for SVMs: A case study. *ACM SIGKDD Explorations Newsletter*, 2004, 6(1): 60-69.

[19] Ho T K. Random decision forests. In *Proc. the 3rd Int. Conf. Document Analysis and Recognition*, August 1995, pp.278-282.

[20] Munkres J. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 1957, 5(1): 32-38.

[21] Kong X N, Zhang J W, Yu P S. Inferring anchor links across multiple heterogeneous social networks. In *Proc. the 22nd ACM Int. Conf. Information & Knowledge Management*, October 2013, pp.179-188.

[22] Agrawal R, Faloutsos C, Swami A. Efficient similarity search in sequence databases. In *Proc. the 4th Int. Conf. Foundations of Data Organization and Algorithms*, October 1993, pp.69-84.

[23] Chen L, Ng R. On the marriage of Lp-norms and edit distance. In *Proc. the 30th Int. Conf. Very Large Data Bases-Volume 30*, August 2004, pp.792-803.

[24] Nanni M, Pedreschi D. Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, 2006, 27(3): 267-289.

[25] Chen L, Özsu M T, Oria V. Robust and fast similarity search for moving object trajectories. In *Proc. ACM SIGMOD Int. Conf. Management of Data*, June 2005, pp.491-502.

[26] Pelekis N, Kopanakis I, Marketos G, Ntoutsi I, Andrienko G, Theodoridis Y. Similarity search in trajectory databases. In *Proc. the 14th Int. Symp. Temporal Representation and Reasoning*, June 2007, pp.129-140.

[27] Wang X Y, Mueen A, Ding H, Trajcevski G, Trajcevski P, Keogh E. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 2013, 26(2): 275-309.

Guo-Wei Wang is a Ph.D. student in School of Computer Science and Technology at University of Science and Technology of China (USTC), Hefei. He received his Bachelor's degree at USTC, Hefei, in 2012. His current research interests include trajectory data mining and cloud computing.

Jin-Dou Zhang is a Ph.D. student in School of Computer Science and Technology at the University of Science and Technology of China, Hefei. He received his Bachelor's degree at Anhui University, Hefei, in 2010. His research interests include big data processing, rule-based computing, and distributed computing.

Jing Li received his B.E. degree in computer science from the University of Science and Technology of China (USTC), Hefei, in 1987, and Ph.D. degree in computer science from USTC, Hefei, in 1993. Now he is a professor at the School of Computer Science and Technology in USTC, Hefei. His research interests include distributed systems, cloud computing, big data processing, and mobile cloud computing.