CP-Link: Exploiting Continuous Spatio-Temporal Check-In Patterns for User Identity Linkage

Xiaoqiang Ma, *Member, IEEE*, Fengxiang Ding, Kai Peng, Yang Yang, *Member, IEEE*, and Chen Wang, *Senior Member, IEEE*

Abstract—Driven by the large amount of spatio-temporal data obtained from location-based social networks, the implementation of cross-domain user linkage, also known as the User Identity Linkage (UIL), has attracted increasing research attentions. While most of the existing UIL works discretize the spatio-temporal sparse data when identifying encountering or co-located events for UIL, user's distinctive behavior patterns implicit in the "check-in" spatio-temporal data with continuous nature pave the way for enhancing UIL performance. In this paper, we propose an approach dubbed *CP-Link* that exploits user behavior patterns in a continuous way. In CP-Link, the continuous space is divided into irregularly shaped stay regions, and a continuous time-based improved dynamic time warping (IDTW) method is proposed to calculate the similarity. To bridge the gap between the ideal scenario with ample records and the reality with sparse data, we adopt the user-associated location frequent pattern (LFP) model to compensate for the sparse deficiency. Extensive experiments conducted on real-world datasets demonstrate the effectiveness and superiority of CP-Link, which outperforms the state of the arts by more than 20% in terms of the AUC.

Index Terms—Continuous check-in pattern, data sparsity, LBSN, spatio-temporal data, user identity linkage

1 Introduction

With the development of network technology and the proliferation of GPS-enabled mobile devices, location-based social network (LBSN) services such as Twitter and Foursquare have generated a considerable amount of spatio-temporal check-in data, which contains users' time and geo-location information [2], [3], [4]. The availability of such spatio-temporal information provides an unprecedented opportunity to explore users' behavior, one of which is the user identity linkage (UIL) technique that associates different user accounts across different LBSN applications of the same person [5], [6].

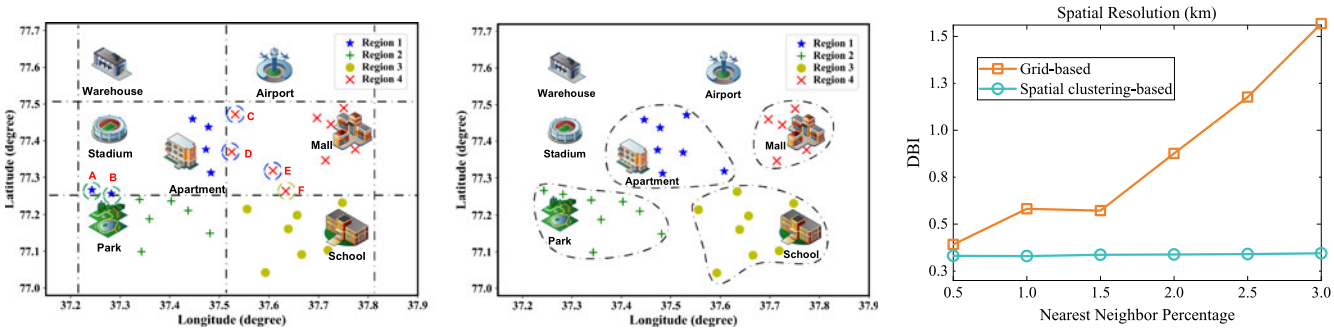
UIL brings huge benefits for both service providers and users by jointly modeling the attributes that belong to the same user. For service providers, UIL offers a more comprehensive view about users by fusing data from a commercial perspective, and thus can provide better services, e.g., personal cross-domain recommendations [7]. For LBSN users, understanding the "linkability" of their accounts helps to raise the awareness of their privacy exposure risks [8], [9]. Hence, the implementation of an efficient UIL has attracted increasing research attentions recently.

Existing studies explore different ways to tackle UIL using spatio-temporal "check-in" data. Most if not all of them link users based on encountering [10] or co-located events [5], [11]. Early works directly compute the distance of all spatial-temporal points in the trajectories to measure the similarity [12], which is confronted with complexity challenges with huge spatial-temporal records. Hence, recent works try to lower the complexity by increasing granularity of the records, in which they divide time and space into bins or grids based on location regions and time intervals. For example, Basık *et al.* [13] develop an algorithm containing two filtering steps for UIL, where the space is partitioned into coarse-grained geographical regions in the spatial filtering step. Chen *et al.* [14] propose a kernel density estimation (KDE) based method and use a grid-based index structure to organize the location data for the sake of high efficiency. Gong *et al.* [15] further propose to partition trajectory segments into 3-dimensional space–time cells. Similar partitioning methods are also utilized in terms of preventing UIL attacks for privacy-preserving LBSNs [16].

Dividing space into bins/grids to satisfy the encountering conditions, however, leads to spatio-temporal *discretization*, which is more likely to result in the boundary effect, i.e., misclassification of locations close-by the division boundaries. As shown in the illustrative example in Fig. 1a, the boundary points "C," "D," "E," and "F" that are grouped to the *Mall* grid are actually closer to the *Apartment* or *School* setting. Such mismatch in grid-based UIL techniques would potentially degrade the linkage success rate, as validated in our experimental results in Section 5.

- *Xiaoqiang Ma, Fengxiang Ding, Kai Peng, and Chen Wang are with the Hubei Key Laboratory of Smart Internet Technology, School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China. E-mail: mxqhust@gmail.com, {dingfx, pkhust}@hust.edu.cn, dr.chen.wang@ieee.org.*
- *Yang Yang is with the School of Computer Science and Information Engineering, Hubei University, Wuhan 430061, China, and also with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. E-mail: yangyang@hubu.edu.cn.*

Manuscript received 1 May 2021; revised 13 Feb. 2022; accepted 3 Mar. 2022. Date of publication 7 Mar. 2022; date of current version 30 June 2023.
This work was supported in part by the National Natural Science Foundation of China under Grants 61872416, 62171189, 62002104, and 62071192, in part by the Fundamental Research Funds for the Central Universities of China under Grant 2019kfyXJJS017, in part by the Key Research and Development Program of Hubei Province under Grant 2020BAB120, and in part by the special fund for Wuhan Yellow Crane Talents (Excellent Young Scholar). Part of this work appeared in Proceedings of 19th IUCC [1].
(Corresponding author: Chen Wang.)
Digital Object Identifier no. 10.1109/TMC.2022.3157292



(a) Grid-based solution. (b) Spatial clustering-based solution. (c) DBI comparison.

Fig. 1. Illustration and comparison of grid-based and spatial clustering-based stay region construction. (a) For grid-based method, the boundary points "C", "D","E", and "F" that are grouped to the Mall grid are actually closer to the Apartment or School setting. (b) Our spatial clustering-based method avoids the mismatch by partitioning the space into arbitrary shapes in a continuous manner. (c) Davies–Bouldin index (DBI) of grid-based vs. spatial clustering-based solution, where a smaller and more stable DBI represents a more compact and effective cluster solution.

Motivated by the *continuous* nature of the space, in this paper, we propose *CP-Link* that exploits continuous spatio-temporal check-in patterns for UIL. We develop a spatial clustering method based on density-peaks (DP), by which the space is partitioned in a continuous manner, and stay regions of arbitrary shapes can be extracted (cf. Fig. 1b). To preliminarily differentiate the merit of our clustering-based method, we measure the clustering quality via a widely-used metric Davies–Bouldin index (DBI) [17], which is the ratio of the sum of within-cluster scatter to between-cluster separation (cf. Fig. 1c). Our superior clustering performance benefits from the consideration of both in-cluster and between-cluster distances, thus avoiding the possible misclassification of locations close-by the division boundary.

Though the basic idea is simple, the design and implementation of CP-Link have two major challenges. The first one is how to deal with the *data sparsity*. On one hand, it is indicated that the majority of the users have very sparse location records in LBSN datasets, say less than 5 records per user on average [20]. On the other hand, for different platforms, existing UIL methods are limited to use the check-in data in a common fixed time interval. According to our observation in the real world datasets, users prefer to share on different platforms in different periods. Therefore, the available records in a common period are rare [21], [22], which is not sufficient for time aligning and identity linkage. To tackle this issue, we adopt the user-associated location frequent pattern (LFP) model to explore the spatial and temporal characteristics hidden in the records, which are supposed to be the same across different services and can be used to predict the associated common locations behind the published records.

The second challenge is how to compute the similarity between users with the extracted stay regions. A straightforward method is to measure the distance of stay regions with irregular shapes, which requires normalization as the premise and has difficulty in dealing with various shapes. Besides, the time information within each stay region should also be taken into account. Therefore, we propose an improved dynamic time warping (IDTW)-based similarity algorithm to transform arbitrarily shaped regions into time series and further calculate the similarity of each stay region in a continuous way. Our proposed algorithm does not require the location records to fall into the same time bin and thus avoids time discretization.

We summarize our major contributions as follows:

- We present CP-Link, the first work that considers check-in records within DP-based stay regions as time series for UIL in LBSNs, and aims to tackle the spatio-temporal discretization problem which has been largely neglected by existing studies.
- CP-Link calculates similarity using IDTW, extracting users' spatio-temporal periodic behavior in particular geographic areas. Based on the extracted features, CP-Link develops a novel similarity measure to match cross-domain users.
- We propose the user-associated LFP model to address the check-in data sparsity issue and predict associated locations for better UIL.
- We conduct extensive experiments using real-world datasets, and the results demonstrate that CP-Link is effective for UIL and outperforms state-of-the-art approaches.

Note that this work is based on and includes solid extensions from our previous work [1]. As for technique contributions, we extend the CP-Link to CP-Link+ to reach a higher precision at the cost of computing complexity; we further analyze the actual application scenarios in view of the speed-accuracy tradeoff. We also conduct extensive experiments from several aspects. First, to enhance the motivation in continuous spatial-based clustering, we test clustering solutions on ground-truth and compare with the traditional grid-based methods. The results verify the outperformance and robustness of our continuous design methodology. Second, we emulate attacks on the user-association location mining parts, where the overall performance is slightly affected with different levels of noises. Besides, the privacy protection effect against the linkage attack is analyzed and evaluated, revealing the risk of privacy leakage.

The remainder of this paper is structured as follows. Section 2 reviews some related works. Section 3 describes how to infer the hidden locations. Section 4 discusses the detailed design of CP-Link, and Section 5 presents the performance evaluation of the proposed methods. Finally Section 6 concludes this paper.

TABLE 1
Comparisons of Different UIL Algorithms

Design Principles	POIS [10]	WYCI [18]	SIMP [5]	GKR-KDE [14]	DP-Link [19]	CP-Link
Information used	Content	Content	Content	Content	Content	Content
Flexible time period	✗	✓	✗	✓	✓	✓
Considering data sparsity	✓	✗	✗	✓	✓	✓
Considering spatio-temporal discretization	✗	✓	✗	✗	✗	✓
Considering location relevance	✗	✗	✓	✗	✓	✓
Precision performance	0.68	N/A	0.55	0.5	N/A	0.75

2 Related Work

Based on the types of data used for UIL, existing algorithms can be largely classified into three categories: content-based (user check-in information such as timestamps, location, and posts), user profile-based (users' attributes such as username, address, and age), and network structure-based (e.g., interaction and relationship between users). Spatial-temporal UIL in our scenario mainly utilizes LBSN check-in data and thus falls into the first category.

2.1 Spatial-Temporal UIL

Focusing on the user *content*, a number of UIL algorithms with different manners to represent the spatial-temporal interactions of users have been proposed [5], [10], [11], [13], [14], [18], [19], [23], [24], [25], [26], [27], [28], which largely follow three lines: co-location, co-clustering, and co-occurrence.

Co-location signifies that users' IDs repeatedly appear at the same location (actually the coarse-grained grid). The pioneering work by Rossi *et al.* [18] characterize users by using the frequency of visits to specific locations without using the time information. Wang *et al.* [5] link user IDs across multiple services using a contact graph model to capture the co-location of all users' IDs across multiple services. Chen *et al.* [14] propose a grid-based KDE to organize locations where locations within a common grid are treated as co-location and the similarity is calculated by means of the distance between grids. Unlike previous efforts to link user accounts directly, Wang *et al.* [11] propose a co-location social network "CLSN" which characterizes online interactions between virtual IDs and offline social network encounters, owing to which they achieve an accurate mapping of online accounts. Considering the heterogeneity of mobility data, Feng *et al.* [19] conduct a pre-training strategy to deal with the highly heterogeneous nature and propose an end-to-end deep learning-based framework to extract spatial-temporal locality feature for user linkage. Isaj *et al.* [27] organize the spatial entities into blocks based on spatial proximity and match users with the same attributes of the spatial entities.

Instead of dividing the space equally, some works tend to distinguish the trajectories by clustering locations and considering overlaps of clusters to represent the co-clustering relationship. To adequately utilize user-generated geo-location data, a co-clustering-based framework is proposed [24], in which the clustering in temporal and spatial dimensions is carried out synchronously. Similarly, Qi *et al.* [25] advocate an identification resolution method to find top-N regions whose trajectory points are most frequently distributed and match users between clustering regions.

Some other works are more inclined to regard users as co-occurrence when they encounter in the same time interval. Riederer *et al.* [10] divide space and time into bins, based on which they measure users' similarity by a new type of maximum weight matching combining positive and negative signals from co-occurrence events. With the view of averting negative matches in UIL, Basık *et al.* [13] develop k-l diversity-based ST-Link algorithm and carry out the spatial and temporal filtering steps independently by first distributing entities over coarse-grained geographical regions and building a sliding window-based scan to select candidate co-occurrence entity pairs. They further develop an efficient matching process with a locality-sensitive hashing-based approach that significantly reduces candidate pairs [26], where a tree is established to keep the co-occurrence counts of the dominating grid cell IDs in the time window. To reveal privacy vulnerabilities of spatial cloaking, Li *et al.* [23] capture uncovered actual locations that are periodically visited by the target user, they express spatial and temporal correlations by the proximity of consecutive actual locations in cloaked regions.

We classify the representative algorithms based on their design principles and compare them with CP-Link in Table 1. We can see that the existing algorithms have yet to consider both data sparsity, location relevance and spatio-temporal discretization simultaneously, which motivates the design of CP-Link to achieve better performance. Besides, we list the best precision performance of each algorithm, reported in the original works. From the results we can see that considering the data sparsity is a crucial factor for better linkage in POIS [10], while flexible time period may have less importance. Moreover, compared to other methods, CP-Link shows remarkable precision due to the continuous spatio-temporal design instead of the discretization as well as the location relevance.

2.2 UIL of Profile and Network Data

Since we focus on the identity linkage based on content datasets (spatial-temporal check-ins), we briefly introduce the related study designed for network datasets and profile datasets. Goga *et al.* [29] utilize users' writing styles as captured by language models along with geo-locations and timestamps attached to users' posts, so as to identify accounts on different social network sites. To address the issue that there is no guarantee of stability for pair-wise user linkage, Mu *et al.* [30] propose the "Latent User Space" concept to model the relationship between the underlying

TABLE 2
summary of Notations

Notation	Description
u	A user account
l_u^i	A location of user u
$\rho_{l_u^i}$	Relative density of l_u^i
$\delta_{l_u^i}$	Minimum density ascent distance of l_u^i
$\mathcal{L}_u$	Location set extracted from u's check-ins
$\mathcal{A}_u$	Associated user set of u
$\mathcal{L}_{\mathcal{A}_u}$	Location set extracted from $\mathcal{A}_u$
$F_{\mathcal{L}_{\mathcal{A}_u}}$	Location frequent itemsets of $\mathcal{L}_{\mathcal{A}_u}$
$\mathcal{D}_{l_u}$	Enlarged disk with alterable radius for l_u
$\mathcal{S}_u$	All stay regions of user u
$D(\mathcal{S}_u, \mathcal{S}_v)$	Distance between $\mathcal{S}_u$ and $\mathcal{S}_v$
$Sim(u, v)$	Similarity between u and v

real users and their observed projections onto the varied social platforms. Nie *et al.* [31] discover users' core interests by jointly using user network datasets and interest-based contents datasets (posts content). To overcome the limitation of hand-crafted features, automatic feature learning has been proposed by Yang *et al.* [32]. A hypergraph including user-user friendship and user-time-POI-semantic (check-ins) is formed to predict users' locations and infer users' connections by using the connections between certain users as background information to make their predictions. Li *et al.* [33] focus on the problem of matching user accounts across social networks solely by characterizing information redundancies contained in username and display name and employ the Gale-Shapley algorithm to eliminate the one-to-many or many-to-many relationships in the identification results. These methods cannot be directly applied to our scenario with only time and location information available.

3 Inferring Associated Locations

We formalize our problem as follows: for two different LBSNs, there are two sets of user accounts $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$ and $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$, where each user $u \in \mathcal{U}$ (resp. $v \in \mathcal{V}$) has a set of check-in records $\mathcal{R}_u = \{r_u^1, r_u^2, \ldots, r_u^m\}$ (resp. $\mathcal{R}_v = \{r_v^1, r_v^2, \ldots, r_v^n\}$). Here, a check-in record r_u is in the form of $(u,_u, t_u)$, where l_u and t_u are the location and time when the check-in takes place. The aim of CP-Link is to find all account pairs (u_i, v_j) of the same user between accounts of $\mathcal{U}$ and $\mathcal{V}$. For ease of exposition, we summarize the notations used throughout the paper in Table 2.

Before digging into the CP-Link design, in this section, we first tackle the location sparsity issue. The pipeline of our algorithm is illustrated in Fig. 2. Our idea is simple: since some users have only a few check-in records which is insufficient for mining users' regularity, we have to make up the check-in records for them. Our solution is based on the observation that users with a similar mobility pattern often check in at some common locations [34], [35]. Thus assuming the check-in locations of user u are sparse, we first identify the associated users who have significant overlap of u's locations and further select the highly associated locations from the locations of u's associated users. As such, we can regard these highly associated locations as the supplements of u's locations.

3.1 Identifying Associated Users

We refer to the associated users of user u as those whose check-in locations are of high overlap with u's locations. It is noticed that in practice one's location can rarely cover another's due to its "point" nature. Therefore when we refer to the overlap of two locations, we actually enlarge the location to a disk. Specifically, for u's location l_u, we define its corresponding disk $\mathcal{D}_{l_u} = \{l_x : |l_x - l_u| \leq d_\gamma\}$, where d_γ is the radius of the disk and can be set upon required granularity. We further define the hit-function to indicate whether two locations l_u and $l_{u'}$ overlap as follows

$$Hit(l_u, l_{u'}) = \begin{cases} 1, & \text{if } \mathcal{D}_{l_u} \cap \mathcal{D}_{l'_u} \neq oslash; \\ 0, & \text{if } \mathcal{D}_{l_u} \cap \mathcal{D}_{l'_u} = oslash;. \end{cases} \quad (1)$$

The hit-function here is simplified as a binary value. Actually, there is a second option for the hit-function, representing the actual overlapping proportion of access locations:

$$Hit(l_u, l_{u'}) = \frac{S(\mathcal{D}_{l_u} \cap \mathcal{D}_{l'_u})}{S(\mathcal{D}_{l_u})} \quad (2)$$

where $\mathcal{D}_{l_u}$ is the corresponding disk of location l_u and $S(\mathcal{D}_{l_u})$ is the area of the disk. The numerator measures the

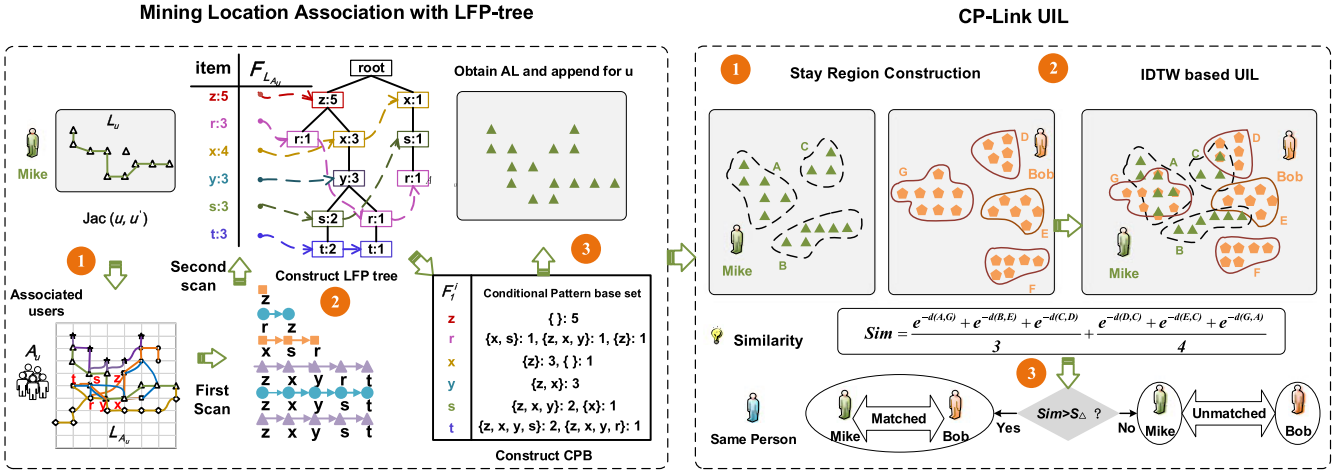


Fig. 2. Workflow of our algorithm, which contains two main parts: (a) Inferring associated locations to tackle the location sparsity issue; and (b) user identity linkage with CP-Link.

area of the location overlap. In this way, the calculation can capture the overlap more accurately; yet requiring more computing cost. This design is integrated into an efficacy improved CP-Link+ described in Section 4.3.

To identify u's associated users, we use the Jaccard coefficient as the measurement, which takes both the commonality and the difference of the visited locations into account [36] and also enables us to measure the proportion of common locations accessed by different users. The Jaccard coefficient between users u and u' can be computed as follows

$$Jac(u,u') = \frac{\sum_{i=1}^{|\mathcal{L}_u|}\sum_{j=1}^{|\mathcal{L}_{u'}|} Hit(l_u^i, l_{u'}^j)}{|\mathcal{L}_u \cup \mathcal{L}_{u'}| - \sum_{i=1}^{|\mathcal{L}_u|}\sum_{j=1}^{|\mathcal{L}_{u'}|} Hit(l_u^i, l_{u'}^j)}, \tag{3}$$

where $\mathcal{L}_u$ (resp. $\mathcal{L}_{u'}$) is the location set extracted from their check-in record set R_u (resp. $R_{u'}$); $\sum_{i=1}^{|\mathcal{L}_u|}\sum_{j=1}^{|\mathcal{L}_{u'}|} Hit(l_u^i, l_{u'}^j)$ evaluates the number of common locations accessed by the two users, while $|\mathcal{L}_u \cup \mathcal{L}_{u'}|$ denotes the number of all locations visited by the two users.

After obtaining all the Jaccard coefficients between u and each of the other users in $\mathcal{U}$, we set a threshold Jac_{thr} to determine the number of associated users for each individual by means of filtering out users with lower association, i.e., users whose Jaccard coefficient values are above Jac_{thr} are chosen as u's associated users. Through experiments, we find that a too high or too low threshold Jac_{thr} will affect the performance of the algorithm, and the coincidence degree between user access locations can achieve an ideal effect at around 10%. In addition, considering that some LBSN platforms contain vast amounts of check-in records, and the number of one's associated users may be redundant, yielding unnecessary computing costs. Therefore, we also adopt a simple top-k selection method which chooses users with the top-k Jaccard coefficient values from the ranked list as u's associated users ($k = 10$ in our experiments). The final selected associated users of u are denoted by $\mathcal{A}_u$.

3.2 Mining Location Association With LFP-Tree

After obtaining $\mathcal{A}_u$, we next seek for highly associated locations as the hidden locations of u (denoted by $\mathcal{L}_u^{hdn}$), which are picked out from the locations of $\mathcal{A}_u$ (denoted by $\mathcal{L}_{\mathcal{A}_u}$) by adapting the frequent pattern (FP) growth technique [37], [38] (c.f. Fig. 2). We are inspired to settle location items $\mathcal{L}_{\mathcal{A}_u}$ in virtue of the FP-tree structure, which we refer to as LFP-tree. In particular, we first transform $\mathcal{L}_{\mathcal{A}_u}$ into an LFP-tree, which is a compact data structure to compress and store the location frequent items of $\mathcal{L}_{\mathcal{A}_u}$. Then we develop a pattern-fragment growth mining method based on FP-tree to mine location frequent patterns.

Definition 1 (FP-Tree [37], [38]). *The* FP-tree *is an extended prefix-tree structure, which mines the access frequency of each item (i.e., location in our setting). The root of an FP-tree is labeled as "null" with a set of item-prefix sub-trees as children. Each node in the item-prefix sub-tree stores three elements: item-name, occurrences on the path, and node-link where nodes with the same item-names are linked in support-descending order via such node-links.*

To construct the LFP-tree, we first gather the location datasets of the user and its corresponding associated users, which will be scanned to filter and sort location items according to the support (i.e., the frequency of occurrence in $\mathcal{L}_{\mathcal{A}_u}$). An LFP-tree is then constructed by inserting locations into the tree nodes. We demonstrate this via an example of an arbitrary user u, whose associated users $\mathcal{A}_u$ have been extracted. We display part of the filtered trajectories and process of associated location inference in Fig. 2. In the first scan, we conduct trajectories of users $\mathcal{A}_u$ by traversing their location datasets $\mathcal{L}_{\mathcal{A}_u}$. To filter out the most representative points of interest and reduce the number of the candidates, locations with less than 2 supports are omitted in the trajectories. Meanwhile, we collect the sets of location frequent items $F_{\mathcal{L}_{\mathcal{A}_u}}$ and sort the items according to their corresponding supports, e.g. the most visited location z is arranged at the top. Then for the second scan, with the sorted trajectories, we create an LFP-tree with a root labeled with "null" and put the locations into the tree nodes according to the order of $F_{\mathcal{L}_{\mathcal{A}_u}}$, where each node stores a location and its number of occurrences on the path. In this way, an LFP-tree is generated recursively after completing the two scans. Therefore, for user u, the size of its LFP-tree is bounded by $\sum_{\mathcal{L}_{\mathcal{A}_u}^i \in \mathcal{L}_{\mathcal{A}_u}} |F_{\mathcal{L}_{\mathcal{A}_u}^i}|$, and the height is bounded by $\max_{\mathcal{L}_{\mathcal{A}_u}^i \in \mathcal{L}_{\mathcal{A}_u}} |F_{\mathcal{L}_{\mathcal{A}_u}^i}|$ without considering the root.

Some important properties and theorems from [38] are presented in the following, which will be used to validate the rationality and compactness of the tree structure when applying frequent mining.

Property 1 (Location-link Property). *For any location frequent item $l_{\mathcal{A}_u}^i \in \mathcal{L}_{\mathcal{A}_u}$, all the possible patterns containing only frequent items and $l_{\mathcal{A}_u}^i$ can be obtained by following location-links of $l_{\mathcal{A}_u}^i$, starting from $l_{\mathcal{A}_u^i}$'s head in the LFP-tree header.*

Property 1 ensures that after the LFP-tree construction, we can access all the related frequent-pattern information of $l_{\mathcal{A}_u}^i$ including locations and occurrences in the path by traversing the LFP-tree once following its location-links.

Property 2 (Prefix Path Property). *When calculating frequent patterns for location item $l_{\mathcal{A}_u}^i$ in a path P of the tree, we only need to get the nodes in the prefix path of $l_{\mathcal{A}_u}^i$ and adjust the number of accesses of these nodes to be the same as $l_{\mathcal{A}_u}^i$. Such information is then transformed into a count-adjusted prefix path of $l_{\mathcal{A}_u}^i$ for path P, which is a database of patterns occurring with $l_{\mathcal{A}_u}^i$ and is referred to as the conditional pattern-base (CPB) of $l_{\mathcal{A}_u}^i$.*

Property 2 indicates that after creating the LFP-tree, we can construct a small-scale CPB for any location item.

Theorem 1 (Pattern Growth [38]). *Let α be a frequent itemset in $\mathcal{L}_{\mathcal{A}_u}$, $B(\alpha)$ be CPB of α, and β be an itemset in $B(\alpha)$. Then $\alpha \cup \beta$ is frequent in $\mathcal{L}_{\mathcal{A}_u}$ if and only if β is frequent in $\mathcal{L}_{\mathcal{A}_u}$.*

Based on Theorem 1, we can first mine the frequent 1-location-itemsets α in $\mathcal{L}_{\mathcal{A}_u}$ and identify the location frequent itemsets in the CPB we build for α. In this way, we can simplify the process to a k frequent 1-location-itemset mining problems, and the location frequent items we extract from the CPB on the LFP-tree is exactly what we need for the target location set $\mathcal{L}_{\mathcal{A}_u}$ of user u.

Now we apply pattern-fragment growth to conduct the mining on the corresponding path of the LFP-tree, which takes full advantage of the CPB. We extract CPB by looking for all prefix paths for each frequent 1-location-itemsets α in $\mathcal{L}_{\mathcal{A}_u}$, starting with the header. Next, a conditional LFP-tree is thus formed using each prefix path as a sample, which is then mined recursively. Eventually, we get all the location frequent patterns that meet the minimum support, from which we can obtain the associated locations. For instance, in Fig. 2, the associated locations for y are $\{x, z\}$. Similarly, we can get all the associated locations for other location items. Algorithm 1 illustrates the process of finding the associated user of user u and extrapolating hidden locations using LFP growth.

Algorithm 1. Inferring Associated Locations

for each $u \in \mathcal{U}$ **do**
  Compute the Jaccard coefficient of u and all other users in $\mathcal{U}$ and find the top-k associated users $\mathcal{A}_u = \{u_1, u_2, \ldots, u_k\}$ with $Jac_i > Jac_{thr}$;
  Scan $\mathcal{L}_{\mathcal{A}_u}$, the location set of $\mathcal{A}_u$, twice to collect the support of each location frequent items and construct LFP-tree in the support-descending order, which stores all location frequent itemsets $F_{\mathcal{L}_{\mathcal{A}_u}}$;
  for Each frequent 1-location-itemset F_1^i in $F_{\mathcal{L}_{\mathcal{A}_u}}$ **do**
    Generate pattern F_1^i with support, and let $freq(P(F_1^i)$ be the set of patterns;
    Construct CPB and the corresponding conditional LFP-tree for F_1^i;
    Mine LFP in the conditional LFP-tree $freq(Q(F_1^i))$;
  end for
  Return the LFP set for $\mathcal{L}_{\mathcal{A}_u}$: $freq(\mathcal{L}_{\mathcal{A}_u}) = freq(P(T)) \cup freq(Q(T)) \cup (freq(P(T)) \times freq(Q(T)))$;
  Let AL be the relevant locations of user u inferred from $freq(\mathcal{L}_{\mathcal{A}_u})$ and obtain the associated locations in AL;
  Append these associated location records to the check-in records of user u.
end for

Given the corresponding highly associated locations $\mathcal{L}_u^{hdn}$ containing locations which are most likely visited by u can be found, they are further appended to $\mathcal{L}_u$ to make up for u's sparse location data. In this way, most users' check-in locations can be significantly supplemented, which are further utilized for UIL in CP-Link.

4 CP-Link Design

After replenishing the sparse data by supplementing the inferred hidden locations, we can next focus on UIL in LBSNs. In reality, different from other sequential data, trajectory with spatial-temporal properties involves distinct user mobility patterns, e.g., the frequency of location check-ins follow a power-law distribution [39]. Besides, records of a common user or even different users are not independent. In our algorithm, we extract users' spatial and temporal patterns adequately. In general, CP-Link involves two steps (c. f. the right part of Fig. 2):

(1) *Stay Region Construction*. Users' stay localities distributes at significant clusters according to the characteristics of social activities. In order to mine the representative clusters as well as to extract users' mobility patterns, we first construct individual stay regions for each user with a DP-based clustering method.

(2) *IDWT based UIL*. We use a time series similarity matching model IDTW to calculate the similarity between cross-domain users' stay regions, followed by completing UIL where the user pair with the highest similarity is selected as the output linked pair.

4.1 Stay Region Construction

A common practice of UIL adopts a set of stay regions to mine users' spatio-temporal behaviors, where a stay region stands for a geographical area contains a set of users' check-in records. A stay region can be of various shapes — rectangle, circular, or irregular, depending on the construction method. One typical strategy to construct stay regions is to divide the space of check-in records into grids with different scales, which may suffer from spatial discretization.

In CP-Link, we use a DP clustering-based stay region construction method to avoid spatial discritization and consider the following two crucial factors.

Definition 2 (Relative Density). *The* relative density *of a location $l_u^i \in \mathcal{L}_u$ is defined as its Radial Basis Function approximation to all other locations in $\mathcal{L}_u$:*

$$\rho_{l_u^i} = \sum_{l_u^j \in \mathcal{L}_u \setminus \{l_u^i\}} \exp(-|l_u^i - l_u^j|^2). \tag{4}$$

From the definition we can see that $\rho_{l_u^i}$ reflects l_u^i's local density among all locations in $\mathcal{L}_u$. It is obvious that the more number of locations in $\mathcal{L}_u$ that are closer to l_u^i, the higher value $\rho_{l_u^i}$ is.

Definition 3 (Minimum Density-Ascent Distance). *Assume $\rho_{l_u^i} \neq \max_{\mathcal{L}_u} \rho_{l_u^j}$ and denote the density-ascent set of l_u^i as $\mathcal{L}_{da|l_u^i} = \{l_u^j : \rho_{l_u^j} > \rho_{l_u^i}\}$. The* minimum density-ascent distance *$\delta_{l_u^i}$ is then defined as:*

$$\delta_{l_u^i} = \min_{l_u^j \in \mathcal{L}_{da|l_u^i}} |l_u^i - l_u^j|. \tag{5}$$

It can be observed that $\mathcal{L}_{da|l_u^i}$ contains all locations of l_u^i whose relative density is higher than $\rho_{l_u^i}$, and $\delta_{l_u^i}$ measures the minimum distance from l_u^i to locations in $\mathcal{L}_{da|l_u^i}$.

When constructing stay regions, we first determine the clustering center of each region and then assign the rest locations to the nearest stay region. Notice that the larger the value of $\rho_{l_u^i}\delta_{l_u^i}$ product is, the more likely the location l_u^i being the cluster center. Therefore, we have the following definition.

Definition 4 (Location DP). *The* location density *of l_u^i is defined as $\gamma_{l_u^i} = \rho_{l_u^i}\delta_{l_u^i}$, and the* location DP *are locations of u with the top-K largest location density, denoted as $\mathcal{L}_{uc} = \{l_{uc}^1, l_{uc}^2, \ldots, l_{uc}^K\}$.*

The above definitions actually provide the approach to obtain $\mathcal{L}_{uc}$ of a specific user u, and locations in $\mathcal{L}_{uc}$ are then treated as the candidate cluster centers of u. The final cluster centers of u can be then determined iteratively from $i = 1$ to $i = K - 1$ calculating the following distances in $\mathcal{L}_{uc}$:

$$l_{uc}^{j} \in \begin{cases} C_u, & |l_{uc}^{i} - l_{uc}^{j}| \geq d_c \\ \mathcal{L}_u \setminus C_u, & |l_{uc}^{i} - l_{uc}^{j}| < d_c \end{cases} \quad j = i+1, \ldots, K; \tag{6}$$

where C_u is the cluster center set of u, and d_c is the cut-off distance that ascertains the minimum distance between two cluster centers. In this way, we can obtain C_u by filtering out the candidate centers in $\mathcal{L}_{uc}$ with a shorter distance (i.e., less than d_c) which may belong to the same geographic area.

Given C_u, we further assign each of the remaining locations of u (i.e., $\mathcal{L}_u \setminus C_u$) to its nearest cluster center. In this way, we can finally obtain the spatial clusters considered as stay regions for each user. The locations within each stay region are arranged according to the generating order of the check-ins. For user u, $\mathcal{S}_u = \{s_u^1, s_u^2, \ldots, s_u^p\}$, for stay region s_u^p, which consists of location records of length x, $\{l_u^1, l_u^2, \ldots, l_u^x\}$ sorted by generation time, it is considered as a time series.

4.2 IDTW Based UIL

After obtaining the stay regions for users in $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$ and $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$, we accomplish the UIL work in three steps. First, an improved DTW estimation-based solution is proposed, which uses the distance of cross-domain users' stay regions and well reflects the user activity characteristics in each area. The distance of stay regions is then converted into similarities between users. Finally, the cross-domain users with the highest similarity will be returned as the same users.

One of the key steps is to calculate the distance between the corresponding cross-domain users' stay regions on the basis of DTW, and we consider the location records within each stay region as a time series. DTW [40] can measure the similarity or distance between two discrete time series and extend or compress two sequences with certain adaptability at the same time [41]. Compared with Euclidean distance which is very sensitive to even small mismatches and requires the two time series to be of equal lengths, DTW is more suitable for our scenario and elegantly overcomes the above concerns.

Definition 5 (DTW). *Given $u \in \mathcal{U}$ and $v \in \mathcal{V}$, along with their corresponding stay region sets $\mathcal{S}_u$ and $\mathcal{S}_v$, DTW aims to calculate the minimum distance between each stay region $s_u \in \mathcal{S}_u : \{l_u^1, l_u^2, \ldots, l_u^x\}$ and $s_v \in \mathcal{S}_v : \{l_v^1, l_v^2, \ldots, l_v^y\}$, which requires constructing the Distance Matrix and finding a path from the upper left corner to the lower right corner such that the path through the elements minimizes the distance of s_u and s_v.*

For time series $s_u : \{l_u^1, l_u^2, \ldots, l_u^x\}$, $s_v : \{l_v^1, l_v^2, \ldots, l_v^y\}$, an y-by-x distance matrix is constructed and the path $\Phi = \{\phi(1), \phi(2), \ldots, \phi(T)\}$ is found to map or align s_u and s_v. We represent each element $\phi(k) \in \Phi$ as:

$$\phi(k) = (\phi_{s_u}(k), \phi_{s_v}(k)) \qquad k = 1, 2, \ldots, T, \tag{7}$$

which indicates T correspondences from the locations of s_u to the locations of s_v, with the possible value of $\phi_{s_u}(k)$ and $\phi_{s_v}(k)$ as $1, 2, \ldots, x$ and $1, 2, \ldots, y$, respectively.

Obtaining the optimal warping curve Φ, the distance between s_u and s_v is minimized by:

$$DTW(s_u, s_v) = \min_{\Phi} d_{\Phi}(s_u, s_v), \tag{8}$$

where $d_{\Phi}(s_u, s_v) = \sum_{k=1}^{T} d(\phi_{s_u}(k), \phi_{s_v}(k))$ denotes the accumulated distortion of corresponding value in the distance matrix on the path elements $\phi_{s_u}(k)$ and $\phi_{s_v}(k)$.

Since DTW has an $O(N^2)$ time and space complexity, which limits the efficiency of performing similarity calculations on large data, we use FastDTW [42], an approximation of DTW which combines constraint and data abstraction on the basis of DTW and is able to find an accurate warped path with a linear time and space complexity.

FastDTW involves three steps. First, the original time series is abstracted into a smaller time series that represents the same curve as accurately as possible. Then the DTW algorithm is applied to time series at a coarse-grained level. The path obtained at a coarser granularity is finally fine-grained through local adjustments of the warped path.

As mentioned above, a user's activities in a stay region may follow a periodic pattern and we improve the conventional FastDTW based on this. In the process of finding the optimal path, in order to better match the periodic time series, we introduce the longest common subsequence factor on the basis of the FastDTW. Specifically, for stay regions $s_u : \{l_u^1, l_u^2, \ldots, l_u^x\}$ and $s_v : \{l_v^1, l_v^2, \ldots, l_v^y\}$, the attenuation coefficient of the longest common subsequence can be defined as:

$$\alpha = 1 - \frac{a_{x,y}{}^2}{x \cdot y}, \tag{9}$$

where x and y are the number of elements in s_u and s_v, respectively, and $a_{x,y}$ is the length of the longest common subsequence, which calculated according to the dynamic programming method as:

$$a_{i,j} = \begin{cases} 0, & \text{i=0 or j=0} \\ a_{i-1,j-1} + 1, & \text{Hit(lui-luj)=1} \\ \max(a_{i-1,j}, a_{i,j-1}), & \text{Hit(lui-luj)=0} \end{cases}, \tag{10}$$

where $i \in (0, 1, \ldots, x), j \in (0, 1, \ldots, y)$, and $a_{i,j}$ represents the length of the common subsequence of $\{l_u^1, l_u^2, \ldots, l_u^i\}$ and $\{l_v^1, l_v^2, \ldots, l_v^j\}$. We consider two locations as overlapped common location as long as $Hit(l_u^i - l_u^j) = 1$ (c.f. Eq. (1)).

Since stay regions can largely reflect users' activity and behavior characteristics, we assign each stay region with specific weight determined by the number of locations within the region to highlight individual frequently visited spots. For user u with locations $\mathcal{L}_u = \{l_u^1, l_u^2, \ldots, l_u^n\}$ partitioned into stay region set $\mathcal{S}_u = \{s_u^1, s_u^2, \ldots, s_u^p\}$, we define the weight of the stay region $s_u : \{l_u^1, l_u^2, \ldots, l_u^x\}$ as:

$$\omega(s_u) = \frac{x}{n}, \tag{11}$$

where x is the number of locations within the current stay region s_u, and n is the total location records generated by user u. Each user's weight allocation method of stay regions is unique compared to that of grid-based method, which leverages the uniqueness of individual users and improves the performance of differentiating users.

For stay region s_u and s_v, we modify the IDTW by:

$$d(s_u, s_v) = \min_{\phi} \sum_{k=1}^{T} \alpha \omega(s_u) d(\phi_{s_u}(k), \phi_{s_v}(k)). \tag{12}$$

In other words, the distance between s_u and s_v largely depends on the length of longest common subsequence of the two sequences. The larger overlap of the stay regions between two users, the higher possibility that the two accounts belong to the same user. Having processed the clustering and calculating distance for all users' stay regions, for user u and v from social network platform $\mathcal{U}$ and $\mathcal{V}$ with corresponding $\mathcal{S}_u = \{s_u^1, s_u^2, \ldots, s_u^p\}$ and $\mathcal{S}_v = \{s_v^1, s_v^2, \ldots, s_v^q\}$, the distances between $\mathcal{S}_u$ and $\mathcal{S}_v$ are obtained, respectively:

$$D(\mathcal{S}_u, \mathcal{S}_v) = \{\mathrm{mind}(s_u^1, s_v^j), \cdots, \mathrm{mind}(s_u^p, s_v^j)\}, j = 1, \ldots, q,$$
$$D(\mathcal{S}_v, \mathcal{S}_u) = \{\mathrm{mind}(s_v^1, s_u^1), \cdots, \mathrm{mind}(s_v^q, s_u^i)\}, i = 1, \ldots, p. \quad (13)$$

We calculate the similarity between u and v as follows:

$$Sim(u, v) = \frac{\sum_{d_i \in D(\mathcal{S}_u, \mathcal{S}_v)} e^{-d_i}}{p} + \frac{\sum_{d_j \in D(\mathcal{S}_v, \mathcal{S}_u)} e^{-d_j}}{q}, \quad (14)$$

where p and q represent the number of stay regions of user u and v, respectively.

For each user u from $\mathcal{U}$, we calculate the similarity between u and all the users from $\mathcal{V}$ and return the linked pair with the highest similarity $\max_{v \in \mathcal{V}} Sim(u, v)$ on the condition that $\max_{v \in \mathcal{V}} Sim(u, v) > S_\Delta$. We summarize CP-Link in Algorithm 2, which consists of the processes of stay regions construction, similarity measure, and user linkage.

4.3 efficacy Improved CP-Link+ Design

CP-Link adopts relatively coarse-grained designs to achieve efficient UIL. To improve the efficacy, we propose CP-Link + by modifying CP-Link from two aspects.

First, recall that in Section 3.2, we define the hit-function in CP-Link to decide whether two locations overlap using a binary value function. In CP-Link+, we replace the binary value function to real-valued function, i.e. to compute the area of the overlap relation. The calculation of hit-function is upgraded to Eq. (2), and the corresponding time complexity increases from $O(n)$ to $O(n^2)$.

Second, in Section 4.2, we employ FastDTW to accelerate the computing speed. This method achieves linear time and space complexity with coarsening, projection and refinement operations on the basis of DTW, thus only outputting an approximation of the optimal warp path between two stay region sets. In CP-Link+, we simply adopt the original DTW instead of FastDTW to reach the optimal distance, though the original DTW takes longer computing time.

5 Performance Evaluation

5.1 Experiment Setup

5.1.1 Datasets

We use the real-world check-in datasets from Foursquare and Twitter in our experiment, which are originally used in [43], and were crawled with a shell script from websites https://foursquare.com/ and https://twitter.com/ in November, 2012. The datasets are commonly adopted by recent works [5], [10], [14], [30], [44], [45]. To evaluate the performance of CP-Link, we only use the location information and timestamps in the check-in records while removing other information such as post contents. The detail of the datasets is presented in Table 3, with 1644 linked user pairs as the ground truth.

TABLE 3
Description of Our Datasets

Domain	Users	Check-ins	Date Range
Foursquare	2970	44915	2008.10-2012.11
Twitter	3518	516787	2010.01-2012.11

Algorithm 2. CP-Link

Input: User account sets $\mathcal{U}$ and $\mathcal{V}$
Output: Linked user pairs LP.
for u_i in $\mathcal{U}$ and v_j in $\mathcal{V}$ **do**
  Extract stay regions of u_i and v_j with DP clustering method as $\mathcal{S}_{u_i} = \{s_{u_i}^1, s_{u_i}^2, \cdots, s_{u_i}^p\}$ and $\mathcal{S}_{v_j} = \{s_{v_j}^1, s_{v_j}^2, \cdots, s_{u_j}^q\}$, respectively;
end for
for u_i in $\mathcal{U}$ **do**
  for v_j in $\mathcal{V}$ **do**
    Calculate $D(\mathcal{S}_{u_i}, \mathcal{S}_{v_j})$ based on Eq. (13);
    Calculate $Sim(u_i, v_j)$ based on Eq. (14);
  end for
  if $\max_{v \in \mathcal{V}} Sim(u_i, v)) > S_\Delta$ **then**
    Add user pair (u_i, v) into LP;
  end if
end for

5.1.2 Compared Algorithms

We compare CP-Link with the following three state-of-the-art algorithms.

POIS [10] uses "encountering" events to match users. The number of visits of each user to a location during a time period is assumed to follow the Poisson distribution, and an action on each service is assumed to occur independently with the Bernoulli distribution. Based on this mobility model, the similarity is calculated as:

$$Sim_{u,v} = \sum_{t \in T} \sum_{l \in L} \phi_{l,t}(\mathcal{U}(t)\mathcal{V}(t)),$$

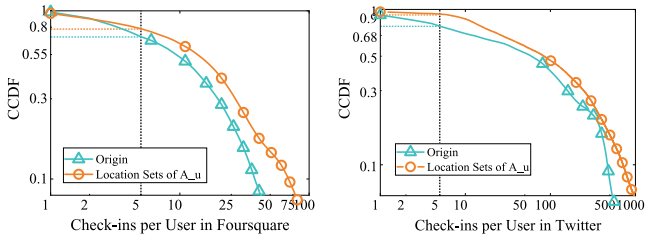
where $\phi_{l,t}$ measures the probability of an "encountering" event in location l at time slot t.

SIMP [5] uses a contact model to map all IDs into a big graph where an edge represents the same location and the weight depends on the time information. It calculates users' similarity by weighting users' "co-locations," and the similarity function is defined as:

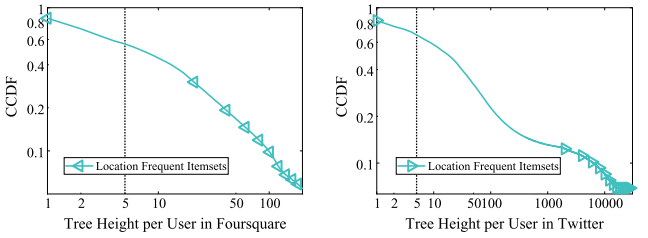
$$Sim_{u,v} = P(X(u, v) = 1 | r(\mathcal{V})) = \frac{Q(u, v)}{\sum_{u \in N^{s_1}(v)} Q(u, v) + \beta(v) r(\mathcal{V})},$$

where $N^{s_1}(v)$ represents a collection of ID accounts in $\mathcal{V}$, $Q(u, v)$ is defined as the joint probability of the observation $r(\mathcal{V})$ and $X(u, v) = 1$, and $\beta(v)$ is the probability that v is not in $N^{s_1}(v)$.

GKR-KDE [14] designs a grid-based kernel density estimation (KDE) method. Each user is represented by a sequence of $k \times k$ grid cells with corresponding confidences, using the Renyi entropy to calculate the weight. The similarity function is as follows:



(a) Number of records in $\mathcal{L}_u$ vs. in $\mathcal{L}_{\mathcal{A}_u}$ in Foursquare (b) Number of records in $\mathcal{L}_v$ vs. in $\mathcal{L}_{\mathcal{A}_v}$ in Twitter



(c) Number of itemsets in $F_{\mathcal{L}_{\mathcal{A}_u}}$ in Foursquare (d) Number of itemsets in $F_{\mathcal{L}_{\mathcal{A}_v}}$ in Twitter

Fig. 3. Complementary cumulative distribution function (CCDF) of the number of check-in records/location frequent itemsets per user. (a) and (b) compare the number of individual data $\mathcal{L}_u$ and the associated users $\mathcal{L}_{\mathcal{A}_u}$, while (c) and (d) show the location frequent itemsets $F_{\mathcal{A}_u}$ which represents the maximum tree height per user.

$$Sim_{u,v} = \sum_{i=1}^{k} f(g_{u_i}|G(v), h),$$

where $G(v)$ is the grid representation of user v; g_{u_i} is the grid cell ID of user u; $f(g_{u_i}|G(v), h)$ is the probability density function, which measures the distance between g_{u_i} and $G(v)$, and its radial range is controlled by a bandwidth parameter h.

Simplified CP-Link. For baseline comparison, we also propose a simplified version of CP-Link which uses the sparse location data without inferring the associated locations with LFP growth method. The similarity function is the same as in Eq. (14).

CP-Link+. We implement the same experiment by CP-Link+ with advanced hit-function, the similarity function is the same as in Eq. (14).

5.1.3 Metrics

We evaluate the performance using the following metrics.

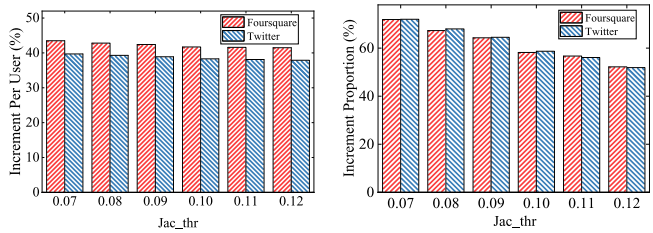
Precision: the ratio of correctly linked user pairs over all the returned user pairs.

Recall: the ratio of correctly linked user pairs over actually linked user pairs in the ground truth.

F1 Score : the harmonic mean of *Recall* and *Precision:*

$$F1 = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

AUC: the area under the ROC curve, where the ROC curve plots the true positive rate (TPR) against the false positive rate (FPR). We use AUC to evaluate the quality of similarity rankings, which is calculated as the probability that the randomly selected positive cases rank higher than the negative. Here, "positive" means the returned user account pairs belong to the same user [5].



(a) Increment per user vs. Jac_{thr} (b) Overall increment vs. Jac_{thr}

Fig. 4. Data Increment w.r.t. varied Jac_{thr}. We evaluate the extent to which Jac_{thr} compensates for (a) a single user and (b) the whole dataset.

Hitting Ratio: the ratio of the number of u's highly associated locations $\mathcal{L}_u^{hdn}$ over $\mathcal{L}_u^{hdn}$:

$$HR(u) = \frac{\sum_{i=1}^{|\mathcal{L}_u|} \sum_{j=1}^{|\mathcal{L}_u^{hdn}|} Hit(l_u^i, l_{u'}^j)}{|\mathcal{L}_u^{hdn}|}, u' \in \mathcal{L}_u^{hdn}.$$

5.2 Performance of Associated Location Inference

As mentioned that the individual records are sparse, we tackle this issue by the associated location inference where location frequent itemsets are explored utilizing $\mathcal{L}_{\mathcal{A}_u}$ instead of $\mathcal{L}_u$. To examine the quality of the inference, we plot the comparison of the amount of $\mathcal{L}_u$ and $\mathcal{L}_{\mathcal{A}_u}$ in Figs. 3a and 3b. We can see that in Foursquare, users with more than 5 records account for 60%, while the proportion is about 80% for $\mathcal{L}_{\mathcal{A}_u}$; similar results can be observed in Twitter, indicating that the amount of records in $\mathcal{L}_{\mathcal{A}_u}$ is relatively more sufficient. Further, after the LFP-tree construction, we acquire the LFP-tree with the maximum height of $F_{\mathcal{L}_{\mathcal{A}_u}}$, based on which we mine and obtain the associated locations. In Figs. 3c and 3d, we plot CCDF of the tree height per user. We can find that in Foursquare, users with more than 5 location frequent itemsets account for 60% and the amount is approximate in Twitter, which is enough to guarantee the associated location mining.

Next, to investigate the effect of our proposed associated location inference method, we vary the user association threshold Jac_{thr} to study the impact on the data increment effect and determine a proper value setting in the following comparison experiments.

Data Increment Varying $\boldsymbol{Jac_{thr}}$. As shown in Figs. 4a and 4b, the implement of associated location inference brings about at least 40% increase of valid trajectories per user where over 50% of the users can benefit from the gain. As for the trend, the amount of data increment decreases with the value of Jac_{thr} increasing. Still, the increment is considerable from the view of quantity as Jac_{thr} varies. In terms of quality, we evaluate the effectiveness of hidden locations generated by the associated location inference, and we vary Jac_{thr} to examine the hitting ratio on Foursquare and Twitter. The results in Fig. 5 show that nearly 80% of the hidden locations we inferred are highly associated with users' original locations (i.e., the locations are highly overlapping). Only a small difference of the hitting ratio can be observed between different settings, indicating that by utilizing the hidden locations, nearly 80% relative performance gain will be achieved, which demonstrates its effectiveness and robustness over the sparsity issue.

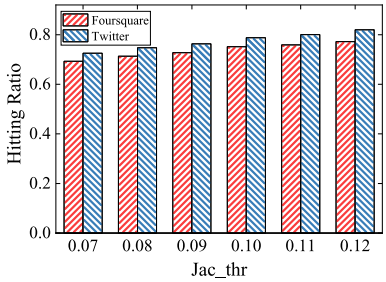


Fig. 5. Hitting ratio w.r.t. varied Jac_{thr}. The results validate the efficacy of the hidden locations generated by the associated location inference.

Performance Varying $\boldsymbol{Jac_{thr}}$. According to the previous description, the increase of Jac_{thr} will reduce the proportion of increment data. It can be observed in Fig. 6a that an increasing Jac_{thr} leads to the decrease of all the AUC, precision, recall and F1. In other words, a smaller Jac_{thr} can better compensate for the data sparsity. This observation can be confirmed in Figs. 4 and 5. According to Eq. (3), the Jaccard coefficient measures the commonality and the difference of the visited locations between users. So, as Jac_{thr} increases, fewer associated users would be involved, leading to less $\mathcal{A}_u$, $\mathcal{L}_{\mathcal{A}_u}$ and $\mathcal{L}_{hdn}$. On the other hand, the higher correlation of $\mathcal{L}_{hdn}$ will result in the increase of the hitting ratio. Nevertheless, precision and recall measure the proportion of correctly returned user pairs and AUC reflects the rankings of the positive case, which are affected by both the quantity (regarding the data increment) and the quality (regarding the hitting ratio) of the augmented data. As shown in Fig. 6a, the overall performance is more affected by the quantity of the data increment, and there are two turning points $Jac_{thr} = 0.08$ and $Jac_{thr} = 0.09$, indicating significant decrease of data increment around the turning points where the rankings are less affected. As a consequence, Jac_{thr} is set to be 0.07 in the following experiments.

5.3 Impact of Parameters in CP-Link

To examine how parameters in our algorithm affect the UIL results and obtain most suitable parameters, we select the nearest neighbor percentage p of DP clustering and the similarity threshold S_Δ to study the impact.

Varying Nearest neighbor percentage $\boldsymbol{p}$. The value of parameter p determines the cutoff distance d_c, which directly affects the calculation of local density ρ and relative distance δ_i. A larger cut-off distance d_c allows more locations in the stay region and lowers limit when dividing regions. Therefore, the quality of clustering and the similarity between stay regions depend on the selection of d_c. Fig. 6b shows the performance under different Nearest neighbor percentage p. Our method achieves higher recall and precision as p increases for the reason that a higher p leads to a higher d_c thus users may have larger common area and similarity due to a larger extracted stay region, where more user pairs and actual linked user pairs may be returned. Besides, AUC is less affected as it mainly measures the ranking values of the positive case which are more stable than Boolean values. As a whole, the result is consistent with our envision that the performance of the algorithm is not greatly affected by the change of p, indicating that our algorithm is robust. To this end, we set $p = 1.5$ in our experiments.

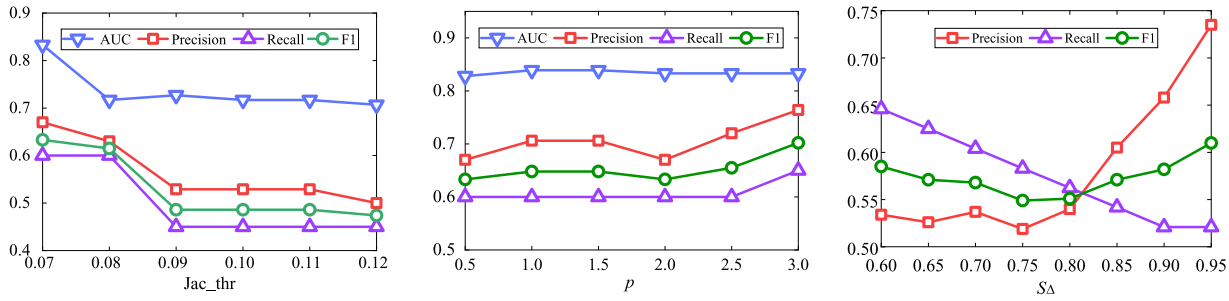
Varying similarity $\boldsymbol{S_\Delta}$. Since the varying S_Δ has no effect on the rankings, resulting in a constant AUC, we only explore the impact on precision, recall and F1 here. As presented in Fig. 2, user account pair with the similarity $S(u, v) > S_\Delta$ could be returned as the same user. Apparently, a too large S_Δ will filter more actual linked user pairs but a too small S_Δ will include more unmatched user pairs in the returned result. To this end, we apply CP-Link under different similarity S_Δ to determine an appropriate value where better performance is achieved at the equilibrium between precision and recall. Fig. 6c plots the performance curves in terms of precision, recall and F1 with varying S_Δ. With S_Δ increasing, a larger S_Δ will filter out more negative user pairs; hence the number of correctly linked user pairs and returned user pairs all decrease, and the latter is expected to drop even more. As a result, the precision increases more steeply than the decrease of the recall. It is noticed that the value of F1 is determined by the combination of precision and recall, both of which are relatively close when $S_\Delta = 0.8$. To balance the performance and determine a proper threshold S_Δ, we set $S_\Delta = 0.9$.

5.4 Performance Comparison

The performance comparison is reported in Fig. 7a in terms of *Precision*, *Recall*, *F1*, and *AUC*.[1] As we can observe, our proposed CP-Link substantially outperforms the state-of-the-art under all evaluation metrics. Remarkably, our proposed baseline simplified CP-Link also performs very well, which benefits from jointly considering spatial-temporal discretization, and the further improvement of CP-Link over the baseline demonstrates the effectiveness of addressing data sparsity. We further show the precision-recall plot in Fig. 7b, which again verifies that our proposed method outperforms other techniques since CP-Link achieves a higher *precision* and *recall* simultaneously. Compared to simplified CP-Link, the user association LFP model in CP-Link can deal with data sparsity better, which is more desirable for real-world applications where trajectory is sparse in general.

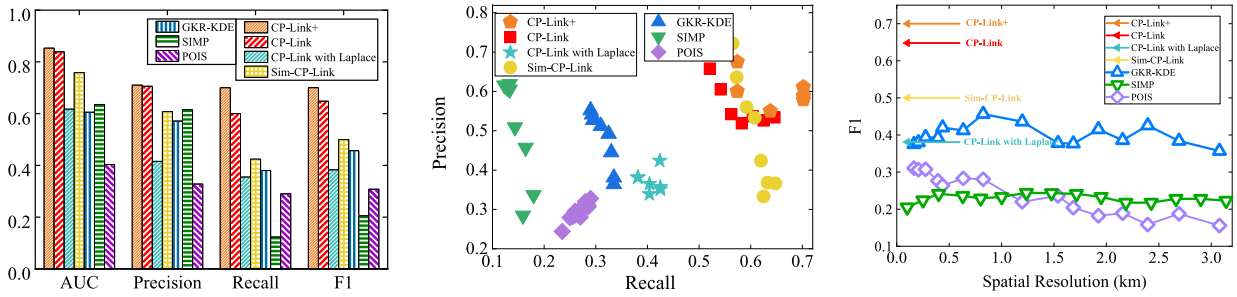
To emphasize the significance of prioritizing spatial-temporal discretization when designing algorithms and set forth the preponderance of CP-Link, we explore the influence of spatial-temporal resolution on the performance of each algorithm. As has been noted, POIS is based on "encountering" events which is sensitive to both spatial and temporal resolution while SIMP and GKR-KDE are susceptible to spatial resolution since they distinguish users based on "co-locations" and grid respectively. For the sake of simplicity, we just calculate the performance of these algorithms under varying spatial resolution. Fig. 7c plots this relationship. As expected, the performance of SIMP, POIS, and GKR-KDE fluctuate when increasing spatial resolution where POIS and GKR-KDE achieve their best performance under the spatial resolution of 0.8 KM, SIMP achieves its best performance under the spatial resolution of 1.2 KM. On account of the consideration of spatio-temporal discretization, the performance of CP-Link is not affected by spatial resolution, the F1 of which remains at a high constant value.

1. Some of the algorithms are not as good as demonstrated in [5], [10] since we set a threshold S_Δ as a constraint on returned pairs.



(a) Performance w.r.t. varied Jac_{thr} (b) Performance w.r.t. varied p (c) Performance w.r.t. varied S_Δ

Fig. 6. Performance w.r.t. different parameters in CP-Link. We evaluate the effect of (a) the user association threshold Jac_{thr}, (b) the nearest neighbor percentage of DP clustering p, and (c) the similarity threshold S_Δ.



(a) Performance in FS-TW (b) Precision-recall plots. (c) Performance w.r.t. varied spatial resolution

Fig. 7. Performance under different parameters with different methods. We first compare the overall performance in (a), followed by plotting the precision-recall curve in (b); and explore the spatial resolution influence in (c).

We further evaluate the performance of CP-Link and CP-Link+ on the same datasets and the results are reported in Fig. 7 and Table 4. Not surprisingly, compared to CP-Link, CP-Link+ achieves reasonably higher precision and recall but is tremendously more time-consuming especially in the hit-function computing process, where the consumed time grows from 0.03 s to 85.65 s. Apparently, with a large quantity of locations, the exponential increase of the time cost does not worth the fractional growth of precision. The coarse-grained design in CP-Link+ may fit better the scenarios with large number of locations, and the speed-accuracy tradeoff is adjustable by specific parameters in practical applications.

5.5 discussion on User Privacy

The implementation of UIL reveals the risk of user privacy disclosure, as discussed in [16]. In our datasets, all the users are anonymous and the trajectories are relatively sparse. For example, as displayed in Fig. 8, the amount of the returned user-pairs by CP-Link which generate less than 5 records is quite close to that of users in the origin datasets, indicating that the linkage performance is not quite influenced by the data sparsity issue. Actually our UIL method can achieve de-anonymization with a precision of up to 0.68, which is quite effective mainly due to our associated location inference.

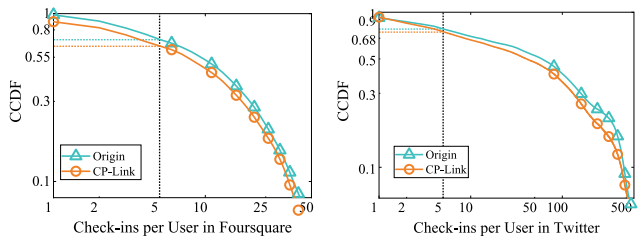
To enhance the privacy protection against our CP-Link, we can introduce Laplace noise into the original location records, as done in [46], and the performance is shown in Fig. 7a. As observed, the linkage performance degrades significantly compared to CP-Link and thus adding noise can protect privacy well. Still, the precision is up to 0.4, which indicates that CP-Link may still work even involving Laplace noise protection. Therefore, we may call for better anonymization techniques, especially for the associated users, in the future.

TABLE 4
comparison of CP-Link and CP-Link+

Algorithm	AUC	Precision	Recall	Time Cost (per user)	
				Hit Func.	(I)DTW
CP-Link	0.82	0.63	0.60	0.03s	147s
CP-Link+	0.85	0.71	0.70	85.65s	311s

5.6 performance of CP-Link With Attacked Associated Location Inference

As mentioned before, the associated location inference is critical to the performance of CP-Link. Here, we consider the situation when the inference is attacked by replacing the



(a) Foursquare (b) Twitter

Fig. 8. Complementary cumulative distribution function (CCDF) of the number of check-in records per user. The green curve plots the CCDF of the number of each user's check-in records in the original dataset, while the orange curve reports the CCDF of the number of linked user-pairs generated by CP-Link.

TABLE 5
performance of CP-Link With Attacked Associated Location Inference

Noise Proportion	0	10%	20%	30%	40%
AUC	0.82	0.82	0.82	0.762	0.762
Precision	0.63	0.63	0.63	0.585	0.585
Recall	0.6	0.6	0.6	0.5	0.5
F1	0.615	0.615	0.615	0.539	0.539

inferred locations with noises (i.e., randomized locations). We introduce varying proportion of noises into the predicted hidden locations and evaluate corresponding performance of UIL under this simulated attack. The result is presented in Table 5. With the noises increasing from 0 to 40%, the overall performance is slightly affected and the precision decreases less than 0.1. The robust performance depends on the noise management design in our DP clustering which screens out all the introduced noises and can thus achieve satisfactory and robust clustering results.

6 Conclusion

In this paper, we have revealed the spatio-temporal discretization, a fundamental issue of existing UIL techniques which divides space and time into independent units when calculating the user similarity. We thus proposed CP-Link that exploits continuous spatio-temporal check-in patterns based on real-world sparse check-in records and calculates the similarity in a continuous manner. To further tackle the data sparsity issue, we designed a user-associated LFP model where individual users' sparse data is compensated by location records of associated users within the same platform. We also discuss a more effecitve version of CP-LINK at the price of computing cost. Extensive experiments on real-world datasets demonstrate the effectiveness and superiority of CP-Link, which outperforms the state of the arts by more than 20% in terms of the AUC.

References

[1] F. Ding, X. Ma, Y. Yang, and C. Wang, "User identity linkage across location-based social networks with spatio-temporal check-in patterns," in *Proc. 19th Int. Conf. Ubiquitous Comput. Commun.*, 2020, pp. 1278–1285.

[2] Z. Yu, F. Yi, Q. Lv, and B. Guo, "Identifying on-site users for social events: Mobility, content, and social relationship," *IEEE Trans. Mobile Comput.*, vol. 17, no. 9, pp. 2055–2068, Sep. 2018.

[3] P. Zhao, J. Li, F. Zeng, F. Xiao, C. Wang, and H. Jiang, "ILLIA: Enabling *k*-anonymity-based privacy preserving against location injection attacks in continuous LBS queries," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1033–1042, Apr. 2018.

[4] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, "Location privacy-preserving mechanisms in location-based services: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 1, pp. 1–36, 2021.

[5] H. Wang, Y. Li, G. Wang, and D. Jin, "You are how you move: Linking multiple user identities from massive mobility traces," in *Proc. SIAM Int. Conf. Data Mining*, 2018, pp. 189–197.

[6] F. Xu, Y. Li, Z. Tu, S. Chang, and H. Huang, "No more than what I post: Preventing linkage attacks on check-in services," *IEEE Trans. Mobile Comput.*, vol. 20, no. 2, pp. 620–633, Feb. 2021.

[7] H. Chen, H. Yin, X. Sun, T. Chen, B. Gabrys, and K. Musial, "Multi-level graph convolutional networks for cross-platform anchor link prediction," in *Proc. 26th ACM KDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 1–12.

[8] K. Shu, S. Wang, J. Tang, R. Zafarani, and H. Liu, "User identity linkage across online social networks: A review," *ACM SIGKDD Explorations Newslett.*, vol. 18, no. 2, pp. 5–17, 2017.

[9] J. Zhang and S. Y. Philip, *Broad Learning Through Fusions*. Berlin, Germany: Springer, 2019.

[10] C. Riederer, Y. Kim, A. Chaintreau, N. Korula, and S. Lattanzi, "Linking users across domains with location data: Theory and validation," in *Proc. 25th Int. Conf. World Wide Web*, 2016, pp. 707–719.

[11] H. Wang, Y. Li, Y. Chen, and D. Jin, "Co-location social networks: Linking the physical world and cyberspace," *IEEE Trans. Mobile Comput.*, vol. 18, no. 5, pp. 1028–1041, May 2019.

[12] L. Rossi, J. Walker, and M. Musolesi, "Spatio-temporal techniques for user identification by means of GPS mobility data," *EPJ Data Sci.*, vol. 4, no. 11, pp. 1–16, 2015.

[13] F. Basık, B. Gedik, Ç. Etemoğlu, and H. Ferhatosmanoğlu, "Spatio-temporal linkage over location-enhanced services," *IEEE Trans. Mobile Comput.*, vol. 17, no. 2, pp. 447–460, Feb. 2018.

[14] W. Chen, H. Yin, W. Wang, L. Zhao, and X. Zhou, "Effective and efficient user account linkage across location based social networks," in *Proc. IEEE 34th Int. Conf. Data Eng.*, 2018, pp. 1085–1096.

[15] X. Gong, Z. Huang, Y. Wang, L. Wu, and Y. Liu, "High-performance spatiotemporal trajectory matching across heterogeneous data sources," *Future Gener. Comput. Syst.*, vol. 105, pp. 148–161, 2020.

[16] F. Xu *et al.*, "No more than what I post: Preventing linkage attacks on check-in services," in *Proc. World Wide Web*, 2019, pp. 3405–3412.

[17] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 12, pp. 1650–1654, Dec. 2002.

[18] L. Rossi and M. Musolesi, "It's the way you check-in: Identifying users in location-based social networks," in *Proc. ACM Conf. Online Social Netw.*, 2014, pp. 215–226.

[19] J. Feng *et al.*, "DPLink: User identity linkage via deep neural network from heterogeneous mobility data," in *Proc. World Wide Web*, 2019, pp. 459–469.

[20] H. Gao and H. Liu, "Data analysis on location-based social networks," in *Proc. Mobile Social Netw.*, 2014, pp. 165–194.

[21] G. Liao, S. Jiang, Z. Zhou, C. Wan, and X. Liu, "POI recommendation of location-based social networks using tensor factorization," in *Proc. IEEE Int. Conf. Mobile Data Manage.*, 2018, pp. 116–124.

[22] H. Wang, Y. Li, C. Gao, G. Wang, X. Tao, and D. Jin, "Anonymization and de-anonymization of mobility trajectories: Dissecting the gaps between theory and practice," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 796–815, Mar. 2021, doi: 10.1109/TMC.2019.2952774.

[23] B. Li, H. Zhu, and M. Xie, "LISC: Location inference attack enhanced by spatial-temporal-social correlations," in *Proc. IEEE Int. Conf. Ubiquitous Intell. Comput.*, 2019, pp. 1083–1092.

[24] X. Han, L. Wang, L. Xu, and S. Zhang, "Social media account linkage using user-generated geo-location data," in *Proc. IEEE Conf. Intell. Secur. Informat.*, 2016, pp. 157–162.

[25] Z. Shao, "User identification across asynchronous mobility trajectories," *Sensors*, vol. 19, no. 9, 2019, Art. no. 2102.

[26] F. Basık, H. Ferhatosmanoğlu, and B. Gedik, "SLIM: Scalable linkage of mobility data," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2020, pp. 16–32.

[27] S. Isaj, E. Zimányi, and T. B. Pedersen, "Multi-source spatial entity linkage," in *Proc. Int. Symp. Spatial Temporal Databases*, 2019, pp. 1–10.

[28] J. Li *et al.*, "Drive2friends: Inferring social relationships from individual vehicle mobility data," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5116–5127, Jun. 2020.

[29] O. Goga, H. Lei, S. H. K. Parthasarathi, G. Friedland, R. Sommer, and R. Teixeira, "Exploiting innocuous activity for correlating users across sites," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 447–458.

[30] X. Mu, F. Zhu, E.-P. Lim, J. Xiao, J. Wang, and Z.-H. Zhou, "User identity linkage by latent user space modelling," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1775–1784.

[31] Y. Nie, Y. Jia, S. Li, X. Zhu, A. Li, and B. Zhou, "Identifying users across social networks based on dynamic core interests," *Neurocomputing*, vol. 210, pp. 107–115, 2016.

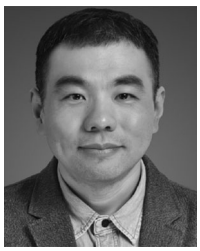
[32] D. Yang, B. Qu, J. Yang, and P. Cudre-Mauroux, "Revisiting user mobility and social relationships in LBSNs: A hypergraph embedding approach," in *Proc. World Wide Web*, 2019, pp. 2147–2157.

[33] Y. Li, Y. Peng, Z. Zhang, H. Yin, and Q. Xu, "Matching user accounts across social networks based on username and display name," *World Wide Web*, vol. 22, no. 3, pp. 1075–1097, 2019.

[34] M. C. González, C. A. Hidalgo, and A. L. Barabási, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008.
[35] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: User movement in location-based social networks," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2011, pp. 1082–1090.
[36] A. S. Shirkhorshidi, S. Aghabozorgi, and T. Y. Wah, "A comparison study on similarity and dissimilarity measures in clustering continuous data," *PLoS One*, vol. 10, no. 12, 2015, Art. no. e0144059.
[37] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *Proc. ACM SIGMOD*, vol. 29, no. 2, pp. 1–12, 2000.
[38] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data Mining Knowl. Discov.*, vol. 8, no. 1, pp. 53–87, 2004.
[39] Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, and F. Zhang, "Identifying human mobility via trajectory embeddings," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 1689–1695.
[40] R. J. Kate, "Using dynamic time warping distances as features for improved time series classification," *Data Mining Knowl. Discov.*, vol. 30, no. 2, pp. 283–312, 2016.
[41] F. Petitjean, G. Forestier, G. I. Webb, A. E. Nicholson, Y. Chen, and E. Keogh, "Dynamic time warping averaging of time series allows faster and more accurate classification," in *Proc. IEEE Int. Conf. Data Mining*, 2014, pp. 470–479.
[42] J. Lohrer and M. Lienkamp, "Building representative velocity profiles using FastDTW and spectral clustering," in *Proc. 14th Int. Conf. ITS Telecommun.*, 2015, pp. 45–49.
[43] J. Zhang, X. Kong, and P. S. Yu, "Transferring heterogeneous links across location-based social networks," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2014, pp. 303–312.
[44] J. Zhang, J. Chen, S. Zhi, Y. Chang, S. Y. Philip, and J. Han, "Link prediction across aligned networks with sparse and low rank matrix estimation," in *Proc. IEEE 33rd Int. Conf. Data Eng.*, 2017, pp. 971–982.
[45] Y. Wang, C. Feng, L. Chen, H. Yin, C. Guo, and Y. Chu, "User identity linkage across social networks via linked heterogeneous network embedding," *World Wide Web*, vol. 22, no. 6, pp. 2611–2632, 2019.
[46] K. Chatzikokolakis, C. Palamidessi, and M. Stronati, "Constructing elastic distinguishability metrics for location privacy," *Proc. Privacy Enhancing Technol.*, vol. 2015, no. 2, pp. 156–170, 2015.

Xiaoqiang Ma (Member, IEEE) received the BE degree from the Huazhong University of Science and Technology, China, in 2010, and the MSc and PhD degrees from Simon Fraser University, Canada, in 2012 and 2015, respectively. He is currently an assistant professor with the Huazhong University of Science and Technology. His research interests include wireless networks, social networks, and cloud computing.

Fengxiang Ding received the BE degree from Central South University, China, in 2019. She is currently working toward the MS degree in electronics and information engineering with the Huazhong University of Science and Technology, China. Her research interests include spatio-temporal data mining and social computing.



Kai Peng received the BS, MS, and PhD degrees from the Huazhong University of Science and Technology, China, in 1999, 2002, and 2006, respectively. He is currently a full professor with the Huazhong University of Science and Technology. His research interests include wireless networking and Big Data processing.

Yang Yang (Member, IEEE) received the BE and MS degrees from the Wuhan University of Technology, China, in 2009 and 2012, respectively, and the PhD degree from the Huazhong University of Science and Technology, China, in 2017. Since 2020, he has been a visiting research fellow with the Department of Computing, The Hong Kong Polytechnic University. He is currently an associate professor with the School of Computer Science and Information Engineering, Hubei University, China. His research interests include wireless networks, mobile computing, and edge computing.

Chen Wang (Senior Member, IEEE) received the BS and PhD degrees from the Department of Automation, Wuhan University, China, in 2008 and 2013, respectively. From 2013 to 2017, he was a postdoctoral research fellow with the Networked and Communication Systems Research Lab, Huazhong University of Science and Technology, China. Thereafter, he joined the faculty of the Huazhong University of Science and Technology, where he is currently an associate professor. His research interests include wireless networking, Internet of Things, and mobile computing, with a recent focus on privacy issues in wireless and mobile systems. He is a senior member of the ACM.