10-2013

TODMIS: Mining Communities from Trajectories

Siyuan LIU
Carnegie Mellon University

Shuhui WANG
Chinese Academy of Sciences

Kasthuri JAYARAJAH
Singapore Management University, kasthurij@smu.edu.sg

Archan MISRA
Singapore Management University, archanm@smu.edu.sg

Rammaya KRISHNAN
Carnegie Mellon University

TODMIS: Mining Communities from Trajectories

Siyuan Liu *, Shuhui Wang +, Kasthuri Jayarajah #, Archan Misra #, Ramayya Krishnan *
*Heinz College, Carnegie Mellon University
+Institute of Computing Technology, Chinese Academy of Sciences
#School of Information Systems, Singapore Management University

ABSTRACT

Existing algorithms for trajectory-based clustering usually rely on simplex representation and a single proximity-related distance (or similarity) measure. Consequently, additional information markers (e.g., social interactions or the semantics of the spatial layout) are usually ignored, leading to the inability to fully discover the communities in the trajectory database. This is especially true for human-generated trajectories, where additional fine-grained markers (e.g., movement velocity at certain locations, or the sequence of semantic spaces visited) can help capture latent relationships between cluster members. To address this limitation, we propose TODMIS: a general framework for **T**rajectory c**O**mmunity **D**iscovery using **M**ultiple **I**nformation **S**ources. TODMIS combines additional information with raw trajectory data and creates multiple similarity metrics. In our proposed approach, we first develop a novel approach for computing *semantic level* similarity by constructing a Markov Random Walk model from the semantically-labeled trajectory data, and then measuring similarity at the distribution level. In addition, we also extract and compute pair-wise similarity measures related to three additional markers, namely trajectory level spatial alignment (proximity), temporal patterns and multi-scale velocity statistics. Finally, after creating a single similarity metric from the weighted combination of these multiple measures, we apply dense sub-graph detection to discover the set of distinct communities. We evaluated TODMIS extensively using traces of (i) student movement data in a campus, (ii) customer trajectories in a shopping mall, and (iii) city-scale taxi movement data. Experimental results demonstrate that TODMIS correctly and efficiently discovers the real grouping behaviors in these diverse settings.

Categories and Subject Descriptors: H.2.8 Database applications: Data mining

General Terms: Algorithms; Experimentation.

Keywords: Trajectory community discovery, multiple information, semantic information.

1. INTRODUCTION

The objective of *trajectory clustering* is to identify clusters from a set of trajectory data of moving objects, where the trajectories in a specific cluster exhibit similarity in one or more movement-related features [25, 9]. Examples of trajectory data include vehicle position data, animal movement data and human behavior tracking data. The recent proliferation of location tracking systems, both in outdoor environments (e.g., GPS) and in indoor buildings (e.g., using Wi-Fi-based positioning), has made it much easier to collect detailed trajectory data. Consequently, there is an increasing interest in performing data mining and behavior analysis over such trajectory datasets [31, 4].

Our interest lies in developing an efficient and effective trajectory-based grouping algorithm, that can accommodate the various latent attributes embedded within the raw trajectory data. We are motivated by a couple of important use cases:

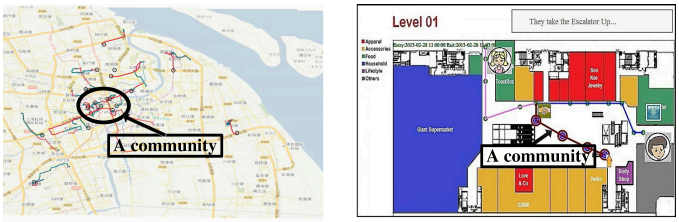
1. Knowing the size of a group of people (e.g., friends, couples, families, etc.) visiting a mall or browsing through a store together will allow merchants to tailor discounts and promotions specifically targeted to the group.
2. By combining knowledge of the real world physical interconnections among people with their online social media data (e.g., from Twitter), computational social scientists may be able to create far richer models of human social interaction (e.g., identify differences in the type of social media content consumed when alone versus when in a group).

In this paper, we focus on **trajectory-based community discovery**, which aims to identify groupings of objects from trajectory data based on additional *behaviorally-driven* markers of individual and collective movement. The difference between *cluster* and *community* (or group) is that a cluster is a set of objects related purely through spatial proximity, whereas a community is a set of objects whose proximity or movement similarity is likely a manifestation of some underlying mutual interaction or shared relationship. As an example illustrated in the left part of Figure 1 (a), taxi drivers in a city may be apportioned in groups, where each group internally shares information for improved profits; drivers of the same group may exhibit a set of distinct trajectory patterns. Similarly, as illustrated in the right part of Figure 1 (a), shoppers in a mall may move around in groups, based on shared social relationships.

Previous studies normally perform trajectory clustering based on only a single information source, such as location data [3, 9, 16]; by viewing shared location as the sole determinant of community relationship, real relationships may be missed or non-existent communities may be falsely identified. In the social graph community detection literature [10, 22, 7], a community is usually defined over a link-based graph capturing direct pair-wise interactions; such explicit interaction markers are obviously hard to directly obtain in many practical environments due to privacy concerns or technological limitations. Hence, we focus on infering groups based only on trajectory-related information (e.g., spatial disperson, temporal

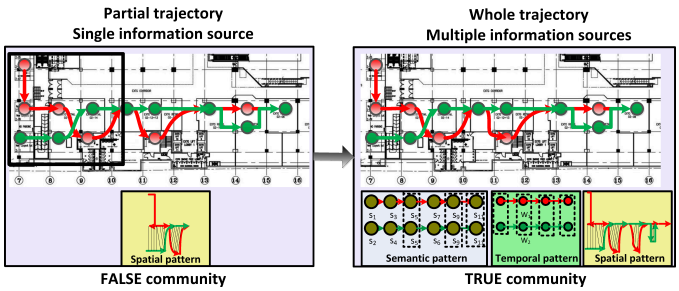
CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2505515.2505552 .



Taxi drivers' trajectories Shopping customers' trajectories

(a) Trajectories on map



(b) Multiple information sources

Figure 1: Trajectory community discovery and its challenges. (a) Human behavior traces naturally result in trajectory community which can help us to understand behavior and decision-making process. (b) In a shopping mall, two trajectories in a community are compared locally (left) and globally (right) from different information sources. Only patterns in dashed box are perfectly matched. We can see that only globally considering the multiple information sources, we can detect the true community.

duration, movement velocity) of individual users and the semantic information of the space.

In our approach, we assume that the trajectories are generated and observed in physical spaces, with the individual's movement within the space being encoded by multiple information markers. For example, the trajectories in a mall contain information on what kind of stores the customers visit, how long they would stay, the transition likelihood between stores, and how fast they are walking. The key observation is that it is hard to detect the real community-driven behavior from analysis performed over a single *timescale* or over a single *feature*. Indeed, as seen in Figure 1 (b), apparently unrelated trajectories may turn out to have strong similarity, when viewed at certain time-scales or in terms of certain features (information markers). Therefore, our approach is to develop a unified framework for community discovery, given a set of trajectories and semantic information of sites visited by those trajectories: this unified framework leverages upon different information markers embedded within the basic trajectory data.

Furthermore, when choosing features for trajectory analysis, previous approaches usually leverage upon information from an individual trajectory or from long-term movement trends, but not both. We propose to encode both the global statistical information and individual information into the semantic feature, and then extract similarity measures through the use of kernels that operate on the probability distribution of semantic-level movement. This approach avoids the "curse of dimensionality" and overcomes the difficulty in measuring the similarity among features with unequal length. More importantly, they jointly model the complicated interconnection of trajectories within members of a community.

More specifically, we propose an approach, **T**rajectory c**O**mmunity **D**iscovery using **M**ultiple **I**nformation **S**ources (TODMIS). TODMIS consists of three distinct phases:

- An explicit modeling of trajectory similarities along four distinct dimensions (or information markers): semantic properties of the locations, temporal duration of the trajectory, spatial proximity to other objects and movement velocity at different timescales. The similarities are computed by applying appropriate kernels for each dimension to extract the key relevant features.
- A computation of a weighted similarity measure that linearly combines the similarity measures along each dimension.
- The application of conventional graph clustering techniques (more specifically, dense sub-graph detection algorithms), over a graph with edges corresponding to the computed pairwise similarity score, to then identify the unknown number of different communities (of varying sizes).

Our key contributions are summarized as follows.

1. **Multi-Dimensional Linear Model for Community Detection:** We propose a new unified model for trajectory-based community detection using multiple dimensions. Different dimensions are linearly combined into a single multi-attribute *weighted similarity score* for each pair of objects. Applying the model to different application scenarios requires only the tuning of the relative weights for different dimensions.
2. **Novel Similarity Metrics:** First, we propose a model for extracting the semantic features from the trajectories. This feature measures the stationary distribution of the object's residency probability on different *semantic* sites (e.g., types of stores in a mall), on which a semantic kernel is applied to determine semantic similarity between trajectories. Second, to compute the spatial proximity of trajectory pairs under realistic conditions (such as inaccurate location measurements and highly-crowded indoor spaces), we modify Global Alignment Kernels [2] to incorporate the inverse of crowd density (at a particular location). Third, we design a novel velocity-based similarity measure that computes velocity at multiple temporal resolutions (both coarse and fine-grained) and thus enables us to measure the velocity similarity on both the whole trajectory and different partial trajectories. This approach allows us to go beyond spatial proximity and explicitly incorporate proximity measures under different movement rates (e.g, stationary, moving slow or fast).
3. **Experiments on Multiple Real-life Datasets:** We evaluate TODMIS against other prior trajectory-based clustering and link-based community detection techniques on three distinct datasets: customer behavior in a shopping mall, student behavior in a campus building and taxi driver behavior in a city. Experimental results demonstrate that TODMIS correctly discovers the real grouping behaviors in all three cases, and outperforms existing algorithms.

Section 2 discusses related work. Section 3 presents an overview of our approach. Section 4 and Section 5 propose how to model semantic, spatial, temporal and velocity information. Section 6 proposes how to learn similarity with multiple information sources. Section 7 presents the results of applying TODMIS on different empirical datasets. Finally, Section 8 concludes the paper.

2. RELATED WORK

Trajectory clustering: Trajectory clustering is a diverse research area, with significant diversity in clustering measures and the end

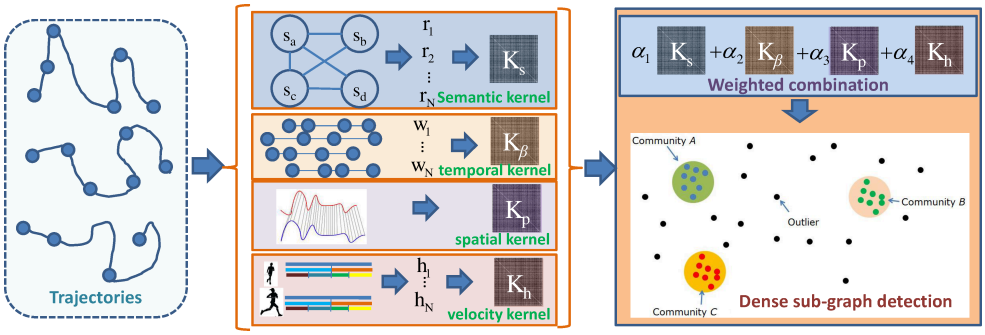


Figure 2: The framework of the proposed approach (TODMIS).

goals. Trajectory has been studied using a variety of measures, ranging from probability function of time [3], behavior correlation representation [29], density-based distance function [16, 5] and uncertainty measurement of trajectories [20]. Different similarity measures (time and location distances) and clustering methodologies have their strengths and weaknesses [15]. In contrast to most prior work, our method is able to handle multiple information sources (not just movement trajectories but also the semantics of the underlying space) and apply a general metric-based learning framework to the clustering problem. Trajectory-based clustering has been used for different broad objectives, such as discovering common sub-trajectories [9], identifying spatial structures [18], estimating common patterns of behaviors and isolating outliers [1], semantic region modeling [27], as well as application-specific objectives such as video surveillance and monitoring systems [21], vehicle motion analysis [11] and discovering object groups that travel together (the objects spatially close at a snapshot). But such work is based purely on spatial locations, making it hard to to extend it to incorporate semantic, velocity or other information that may contain distinctive markers of real community interaction.

Semantic trajectory analysis: There are three common types of approaches for addressing semantic analysis of trajectories: 1) segmentation-based method which segments a scene into semantic regions [28] or maps a trajectory into a multidimensional space [19]; 2) conceptual model which structures movement data into countable semantic units [23]; 3) segmentation and annotation-based approach which transforms the raw mobility data into stops and moves [30]. But the previous analysis do not consider multiple information sources and are difficult to generalize to incorporate additional similarity metrics. Our approach is different from not only the previous techniques, but also the problem of finding semantically similar individuals, who may or may not be visiting the same semantic sites within the same time period (for purposes such as *profile-building* of individuals). Furthermore, on the feature design for trajectory analysis, previous approaches usually extract either individual information from the trajectory data or long-term movement trends, but not both. For example, GAK [2] measure the similarity between trajectories by applying the dynamic time warping kernels to trajectories. On the other hand, the global statistical information measures the trend and probability how a user may act, and is more robust to noise and missing information in individual trajectories. Thus both the global statistical information and individual information are needed jointly to model the complicated inter-connection of trajectories.

Community detection: Communities in networks are groups of vertices within which connections are dense, but between which connections are sparser. There are mainly four types of methods [17, 10, 22, 7]: hierarchical clustering, similarity on edge betweenness scores, counts of short loops and voltage differences in resistor networks. These methods focus on detection given a network structure and social-link distance between nodes which are hard to be captured from trajectories. If we construct such input based on spatial and temporal information, the detection results are not promising (Figure 5 (d) and Figure 6 (d)).

3. OVERVIEW

First, we define a trajectory database $\mathbf{X} = [x_1, ..., x_k, ..., x_N]^T$, where each trajectory x_k is a structured element containing information from multiple aspects, namely, the two dimensional spatial coordinate sequence c_k, the site annotation of the sequence s_k, the time marker e_k and velocity v_k associated with each spatial coordinate. Second, we have the following problem definition.

Problem definition: Given a set of trajectories, we aim to discover a set of trajectory groups, which we name as the *community* in this paper. The trajectories in a certain community demonstrate similar behaviors, when evaluated under different measures (both spatial and semantic).

DEFINITION 1. *Given N trajectories, we define a trajectory affinity graph $G(\alpha)$ with N vertices, denoted by $x_i, i = 1, ...N$. The weight of the edge between two vertices represent the similarity of the two trajectories.*

The similarity between the vertexes in the graph is a weighted combination of a set of similarities calculated from multiple information sources, *i.e.*, the semantic kernel K_s, the temporal kernel K_β, the spatial kernel K_p and the velocity kernel K_h. We denote the weighted combined similarity $\mathbf{K}$ which is defined as:

$$K(k,k') = \sum_{m=1}^{M} \alpha_m K_m(k,k'), \alpha_m \geq 0, \sum_{m=1}^{M} \alpha_m = 1 \\ K(k,k) = 0, k, k' = 1, ..., N \quad (1)$$

where K_m denotes the element of one type of kernel (similarity) from the kernel set. The α_m denotes the pre-assigned weights reflecting the specific interests of the problem domain.

For example, when one prefers to find the trajectories with similar semantic relation, temporal relation and similar spatial relation, and emphasize the semantic relation, the weight can be: $\alpha_1 = 1/2$, $\alpha_2 = 1/4$, $\alpha_3 = 1/4$, $\alpha_4 = 0$. To acquire the best performance of community discovery, we can fine-tune the kernel weight α on a given validation set.

Trajectory community discovery framework: Our algorithm consists of three phases - 1) modeling the similarity of trajectories by first applying kernels along four dimensions: semantic-level movement, temporal, spatial (proximity) and velocity; and then 2) deriving an overall similarity measure between every pair of trajectories

Table 1: The notations used in this paper

Notation	Description
$\mathbf{X}$	Trajectory dataset
N	The number of trajectories
x_k	The k-th trajectories
S	The set of semantic sites
M_s	The number of sites in S
s_i	The i-th sites
r_k	The semantic feature of k-th trajectory
$\mathbf{K}_s$	The semantic kernel
$w_k \in \mathbb{R}^4$	The time stamp of k-th trajectory
$\mathbf{K}_\beta$	Temporal kernel
$\mathbf{K}_p$	The spatial kernel
h_k	The velocity feature of k-th trajectory
$\mathbf{K}_h$	The velocity kernel
α	The kernel weights
$\mathbf{K}(\alpha)$	The combined kernel given α
$z \in \mathbb{R}^N$	The probabilistic cluster
t	The iteration number

in $G(\alpha)$ through a weighted combination of multiple information sources [26]; finally followed by 3) detecting trajectory communities from these similarity values. The framework is illustrated in Figure 2. Based on this framework, the problem of trajectory-based community discovery is equivalent to the computation of cliques on $G(\alpha)$. Note that unlike the traditional trajectory clustering approach, each trajectory is not necessarily assigned to a community ID, as some trajectories in real-life data show a unique motion pattern that are totally distinct from the rest. We denote these trajectories without any community ID as the "outliers" in our study. We provide Table 1 to describe notations in this paper for convenience.

4. MODELING SEMANTIC INFORMATION OF LOCATIONS

Given the trajectory database $\mathbf{X}$, we can extract the following description for the sequential record of the sites visited by an individual user considering that the number of spatially distinct sites is M. For example, the site visiting record of trajectory k: $s_1 \rightarrow s_3 \rightarrow s_8 \rightarrow s_4$...

Our concern is to measure the traverse statistic on the sites from $\mathbf{X}$, and use this statistic to measure the semantic correlation of user trajectories. To this end, we propose the solution outlined below.

Markov state transition: We construct the Markov state transition matrix $\mathbf{A} \in \mathbb{R}^{M_s \times M_s}$, where $A(s_a, s_b)$ represents the transition probability from site s_a to site s_b. To calculate $\mathbf{A}$, firstly we collect all the pairs from the whole trajectory database. Then we count the number of occurrence of each transition pair. Finally, we do column normalization of $\mathbf{A}$, satisfying $\sum_{s_a} A(s_a, s_b) = 1$.

Temporal intervals: For many applications, the time spent at each site and the time taken to transit from site s_a to site s_b can be acquired. Let $t_k(s_a)$ denotes the time spent of k-th trajectory at site s_a and $t_k(s_a, s_b)$ be the time taken to move from site s_a to site s_b. Then, the temporal interval of k-th trajectory is: $[t_k(1), t_k(1,3), t_k(3), t_k(3,8), t_k(8), t_k(8,4)]$.

The interval vector may indicate the corresponding *level of interest* shown by the objects (for example, when a shop is very "interesting", the shoppers may choose to stay longer), or illustrates the convenience of moving from site s_a to site s_b (showing the semantic relation of two sites). This resembles a Markov Random Walk with self-loop.

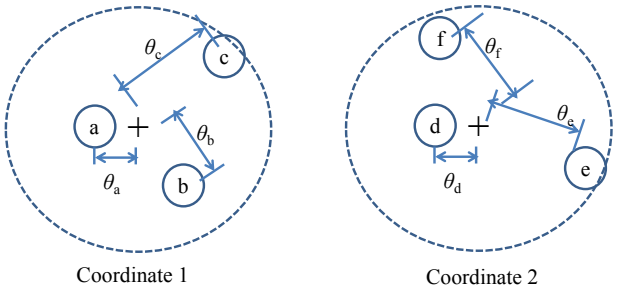


Figure 3: Location uncertainty. Due to the error of the signal sensing and processing, the spatial coordinates may be inaccurate. Therefore, a soft assignment of the coordinates to the sites can be used to alleviate the negative effect of hard-assignment to the nearest sites. The weights of soft assignment is determined as described in Eq.3.

To calculate $\mathbf{A}$ by considering temporal interval, firstly, we normalize the time interval vectors for all user trajectories to ensure the total time taken for each trajectory is 1. For example, we normalize the vector of k-th trajectory so that $t_k(1) + t_k(1,3) + t_k(3) + t_k(3,8) + t_k(8) + t_k(8,4) = 1$. Secondly, accumulation is done by using the time interval vector by $A(s_a, s_b) = A(s_a, s_b) + t_k(s_a, s_b)$. We proceed similarly for the time spent at other locations. For example, if we have $t_k(s_a)$, then $A(s_a, s_a) = A(s_a, s_a) + t_k(s_a)$. Finally, we normalize each column of $\mathbf{A}$ to ensure $\sum_{s_a} A(s_a, s_b) = 1$.

Dealing with location inaccuracy: In many cases, the location information may be inaccurate due to the error of the sensors. Therefore, probabilistic weighted accumulation is preferred over determinant accumulation. The probabilistic weighting scheme is shown in Figure 3, where the crosses denote the detected location center, and the in-dashed circle represents the range of uncertainty of the signals (localization error bounds).

In this case, the *degree of site membership* should be constructed. A reasonable way is to use the Nadaraya-Watson kernel regression on the Gaussian kernel to approximate the probabilistic site membership, which we denote by o. For example, the probabilistic site membership of Coordinate 1 for site a, b and c in Figure 3 is:

$$o_a = \frac{exp(\frac{-\theta_a^2}{\sigma})}{\sum exp(\frac{-\theta_i^2}{\sigma})}, o_b = \frac{exp(\frac{-\theta_b^2}{\sigma})}{\sum exp(\frac{-\theta_i^2}{\sigma})}, o_c = \frac{exp(\frac{-\theta_c^2}{\sigma})}{\sum exp(\frac{-\theta_i^2}{\sigma})}, \quad o_a + o_b + o_c = 1. \tag{2}$$

where σ denotes the uncertainty range of the sensor. Similar calculations can be done on Coordinate 2. Therefore, if there is a pair from Coordinate 1 to Coordinate 2 for User k, and the transition time is denoted by $t_k(c_1, c_2)$, we conduct probabilistic weighted accumulation as follows:

$$A(s_i, s_j) = A(s_i, s_j) + t_k(c_1, c_2) * o_i * o_j \quad i = [a, b, c], j = [d, e, f] \tag{3}$$

Representative distribution of a user's site visits: Given a user trajectory, we calculate the stable distribution which represents the probability a user appearing at one site. We denote such stable distribution for k-th trajectory as $r_k \in \mathbb{R}^{M_s}$. To this end, we firstly construct the Markov state transition matrix $\mathbf{A}$ by the above-mentioned method. Then we collect the set of sites where the users appears, and enumerate the number of visits. For example, by con-

sidering the temporal interval information, we calculate the personalized appearance vector of k-trajectory:

ρ_k^o=[$t_k(1)$, 0, $t_k(3)$,$t_k(4)$,0,0,0,$t_k(8)$,0,0]^T.

Furthermore, the values of ρ_k^o should be represented in a probabilistic form based on the probabilistic site membership o to deal with location inaccuracies. (This extension is straightforward and omitted here). We normalize ρ_k^o, to make $\sum \rho_k^o = 1$. Finally, for each trajectory, we apply an iterative process to calculate the stationary distribution r_k for each trajectory,where:

1. FOR $t = 1$ **to** T_A

   $r_k = \eta \mathbf{A} \cdot r_k + (1-\eta)\rho_k^o$

2. Normalize r_k so that $\sum_{m=1}^{M_s} r_k(m) = 1$.

The process is inspired by the famous personalized Pagerank method in [6]. To analyze the property of r_k, we have the following lemma.

LEMMA 1. *Each semantic feature r_k converges to a unique analytical solution when $T_A \to \infty$*[1].

PROOF. At T_A-th iteration, we have:

$$r_k = (\eta \mathbf{A})^{T_A} r_k^0 + (1-\eta)\,\rho_k^o \left(I + \eta\mathbf{A} ... + (\eta\mathbf{A})^{(T_A-1)}\right) \quad (4)$$

where r_k^0 denotes the initialized value of r_k. Note that the maximum eigenvalue of $\mathbf{A}$ is 1. When $0 < \eta < 1$ and $T_A \to \inf$, the first term converges to 0, which means that the final solution is irrelevant with the initial value r_k^0. For the second term, we have the following analytical form:

$$r_k = (1-\eta)\,(I - \eta\mathbf{A})^{-1}\rho_k^o \quad (5)$$

From the analytical solution, we see that r_k is determined by ρ_k^o and η. When $\eta = 0$, $r_k = \rho_k^o$. When $\eta = 1$, r_k corresponds to the eigenvector of $\mathbf{A}$ with eigenvalue 1. □

Remark: If rank$(\mathbf{A}) = M_s - 1$ and $0 \le \eta < 1$, for any ε satisfying $||\rho_k^o - \rho_{k'}^o||^2 \le \varepsilon$, then there exists δ so that $||r_k - r_{k'}||^2 \le \delta$.

PROOF. If rank$(\mathbf{A}) = M_s - 1$, consider that the maximal eigenvalue of $\mathbf{A}$ is 1, when $0 \le \eta < 1$, $I - \eta\mathbf{A}$ is invertible and the maximal eigenvalue is bounded. Therefore, for any ε:

$$\begin{aligned} &||r_k - r_{k'}||^2 = (1-\eta)^2 ||(I-\eta\mathbf{A})^{-1}(\rho_k^o - \rho_{k'}^o)||^2 \\ &\le \lambda_{\max}^2 ||r_k - r_{k'}||^2 \le \lambda_{\max}^2 \varepsilon^2 \end{aligned} \quad (6)$$

where $\lambda_{\max}$ denotes the maximal eigenvalue of $(I - \eta\mathbf{A})^{-1}$. □

From the remark we see that, if the personalized pattern ρ_k^o and $\rho_{k'}^o$ is similar, their semantic features will be similar as well. By using r_k as the stable distribution of the sites, the semantic divergence between k-th and $k^{'}$-th trajectory can be easily calculated. The non-zero dimensions in r_k not only tells that how long a user has stayed on the certain site, but also provide a probability that how likely a user would be visiting the site if it is not visited by user k in the initial site occurrence vector ρ_k^o. The proposed stationary feature encodes more abundant semantic information by using the global statistical information, and thus it helps to better discriminate among the *semantic behavior* of different people.

Another issue to be considered is the semantic annotation of the site. While each element of the probability vector r_k denotes a site, it is possible for multiple sites to be *semantically equivalent*: e.g., there may be three coffee shops in a shopping mall. To accommodate this equivalence, we apply a simple post-processing step on each stationary distribution r_k. We merge and accumulate the dimensions of the probabilities of those sites with the same semantic meaning into one dimension, and then the M_s-dimensional stationary distribution feature r_k is reduced to M', where $M' < M$. Note that, after the post-processing step, r_k is still guaranteed to be a stable probability distribution and $\sum_{i=1}^{M_s} r_k(i) = 1$.

Given semantic residency distributions, r_k and $r_{k'}$, their semantic similarity can be computed by *Histogram Intersection* kernel: $K_s(k,k') = \sum_{m=1}^{M_s} min(r_k(m), r_{k'}(m))$ or *Radial Basis Function* (RBF) kernel: $K_s(k,k') = \exp(-\lambda \left\|r_k - r_{k'}\right\|^2)$, where λ denotes the bandwidth parameter of RBF kernel.

For parameter tuning of the semantic feature using RBF, we can experiment with some reasonable choices of α (such as α=0.8, 0.7 or 0.6). When similarity is computed using the histogram intersection kernel, no parameters are required. When we use the RBF kernel with parameter λ, we calculate the pairwise distance matrix D with $N \times N$, where $D(i,j) = ||r_i - r_j||^2$. A heuristic setting of λ can be the inverse of the average of all the elements in D where $\lambda_0 = \frac{1}{mean(D)}$. Given λ_0, we select $\lambda = [0.8\lambda_0, 0.9\lambda_0, \lambda_0, 1.1\lambda_0, 1.2\lambda_0]$ to obtain a set of kernel matrices K_s, and finally the kernel with the best performance is selected. In our studies and presented results, we shall focus solely on the RBF kernel, as it consistently provides superior results.

5. MODELING TEMPORAL SPATIAL AND VELOCITY INFORMATION

5.1 Modeling temporal information

For each trajectory k, we extract a vector with four dimensional time stamps w from the time marker e_k: $w_k(1)$ records the first showing time; $w_k(2)$ (the first meeting time stamp for two visitors in a group; $w_k(3)$ denotes the ending time: $w_k(4)$ is the day index. The time stamp trajectory features describe the temporal activity pattern for real-life communities. For example, young people may prefer to go shopping or play together in the afternoon and evening, while the elderly may prefer to get up earlier and collectively go for exercise. Moreover, the teenagers may be seen to spend an appreciable amount of time waiting for the arrival of their friends, whereas the elderly may be more punctual. To appropriately measure the temporal similarity, the temporal kernel can be formulated as:

$$K_\beta\left(k,k'\right) = e^{-\beta_1 \sum_{i=1}^{3}\left(w_k(i) - w_{k'}(i)\right)^2} e^{-\beta_2\left(w_k(4) - w_{k'}(4)\right)^2} \quad (7)$$

where β_1 and β_2 denote the bandwidth factors. While these need to be adjusted for the specific trajectory mining scenario, the adjustment process is not very complicated. One feasible approach for setting β_1 and β_2 is as follows:

1. Given N trajectories, calculate the average temporal pattern distances as:

$$d_{\beta_1} = \frac{1}{N^2}\sum_{k'=1}^{N}\sum_{k=1}^{N}\sum_{i=1}^{3}\left(w_k(i) - w_{k'}(i)\right)^2 \quad (8)$$

$$d_{\beta 2} = \frac{1}{N^2}\sum_{k'=1}^{N}\sum_{k=1}^{N}\left(w_k(4) - w_{k'}(4)\right)^2 \quad (9)$$

2. To acquire the best performance, one can conduct parameter tuning near the initial setting. For example:

$$\beta_1 = [\tfrac{0.8}{d_{\beta_1}}, \tfrac{0.9}{d_{\beta_1}}, \tfrac{1}{d_{\beta_1}}, \tfrac{1.1}{d_{\beta_1}}, \tfrac{1.2}{d_{\beta_1}}], \beta_2 = \tfrac{1}{d_{\beta_2}} \quad (10)$$

We then select the best parameters that maximize the performance of community discovery.

[1]Typically, setting T_A to 20 comfortably guarantees convergence.

5.2 Modeling spatial information

We can directly use the Global Alignment Kernel (GAK) [2] to measure the spatial similarity between two trajectories. We denote the similarity of spatial information as K_p. According to the analysis in [2], the GAK is positive definite.

However, GAK only measures the spatial closeness of individual trajectories, which may be incapable of discovering the true communities if a large mass of additional trajectories exhibit similar proximity. For example, if Alice, Bob and 100 other people are waiting in a concourse area, then the spatial similarity between the trajectories of Alice and Bob should not be too significant, because this concourse may be the only way for people to traverse through the space. However, if Alice and Bob are the only two people in the group study room on a college campus, and everyone else is in a different classroom, then these two trajectories should be viewed as significantly similar. Based on this intuitive observation, we propose a new Global Alignment Kernel with Inverse Proportion (GAK-IP) (inspired by TF-IDF in IR research), that intuitively weighs the spatial similarity in inverse proportion to how many other people are located within the similar distance range.

To describe this in detail, we first explain how an unmodified GAK works. Consider two time series (trajectories):

$$c_x = (c_x(1), ..., c_x(n)), c_y = (c_y(1), ..., c_y(m)) \tag{11}$$

where each $c_x(i)$ or $c_y(j)$ can be the two dimensional spatial coordinates or the indicator of a certain site. An alignment is $\pi = (\pi_1, \pi_2)$ where a pair of increasing integral vectors of length $p \leq n + m - 1$. Note that such alignment may not be unique, we write $A(n, m)$ for the set of all alignments between two time series of length n and m. The Global Alignment Kernel is defined as the exponential soft-minimum of all alignment distances.

$$K_p^0(c_x, c_y) \stackrel{def}{=} \sum_{\pi \in A(n,m)} e^{-D_\pi(c_x, c_y)} \tag{12}$$

where $D_\pi(c_x, c_y) \stackrel{def}{=} \sum_{i=1}^{|\pi|} \varphi(c_x(\pi_1(i)), c_y(\pi_2(i))), \varphi(c_x, c_y) = ||c_x - c_y||^2$.

It is equivalent with the following formulation:

$$K_p^0(c_x, c_y) \stackrel{def}{=} \sum_{\pi \in A(n,m)} \prod_{i=1}^{|\pi|} \kappa(c_x(\pi_1(i)), c_y(\pi_2(i))), \tag{13}$$
$$\kappa(c_x(\pi_1(i)), c_y(\pi_2(i))) = e^{-||c_x(\pi_1(i)) - c_y(\pi_2(i))||^2}$$

In our modified version of GAK-IP, we revise the definition of distance as follows:

$$D_\pi(c_x, c_y) \stackrel{def}{=} \sum_{i=1}^{|\pi|} w_{\pi(i)} \varphi(c_x(\pi_1(i)), c_y(\pi_2(i))), \tag{14}$$

where $w_{\pi(i)}$ denotes the weight calculated by considering the number of people within a range or on the same site. We denote the number of people on the site $\pi(i)$ at the time when c_x and c_y appear together as $n_{\pi(i)} \geq 1$, $w_{\pi(i)}$ can be calculated as:

$$w_{\pi(i)} = \frac{|\pi| \ln(1 + n_{\pi(i)})}{\sum_{i=1}^{|\pi|} \ln(1 + n_{\pi(i)})}. \tag{15}$$

Note that using $\log_e$ reduces the disproportionate impact of a site where the pair are co-resident with a very large number of objects (a high $n_{\pi(i)}$), which may cause that K_p will be highly dependent at a certain location with crowded objects.

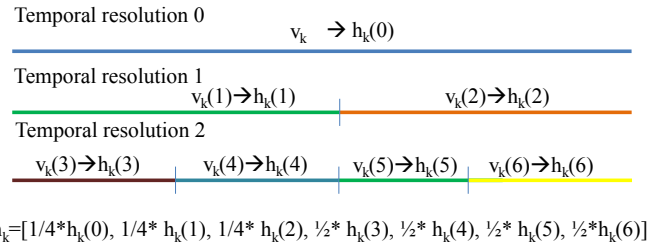


Figure 4: Temporal pyramid matching. $\sum h_k(0) = 1$, $\sum h_k(1) + \sum h_k(2) = 1$, **and** $\sum h_k(3) + \sum h_k(4) + \sum h_k(5) + \sum h_k(6) = 1$.

LEMMA 2. *GAK-IP is positive definite.*

PROOF. Since
$K_p(c_x, c_y) \stackrel{def}{=} \sum_{\pi \in A(n,m)} \prod_{i=1}^{|\pi|} (\kappa(c_x(\pi_1(i)), c_y(\pi_2(i))))^{w_{\pi(i)}}$.
Obviously, by the enclosure property of kernel, the weighted combination of kernel is still guaranteed to be positive definite. □

5.3 Modeling velocity information

The information encoded in velocity pattern of moving objects is also critical for real-life trajectory analysis. For example, the younger taxi drivers tend to drive fast, while the experienced driver will keep a safe speed. However, we face two challenges when modeling the velocity pattern. The first is that trajectories are with non-uniform lengths, which brings about difficulty in directly measuring their pairwise similarity from velocity aspect. The second challenge is that velocity characteristics are diversified on different object, time and location, therefore, it is hard to directly construct their velocity pattern correlation. To analyze the velocity consistency of the trajectories by addressing the two challenges, we design a temporal pyramid kernel by considering different temporal resolutions, which is inspired by [8] in image classification domain on calculating the image level similarity.

We suppose each trajectory is attached with a velocity vector v_k with unequal lengths. We uniformly quantize the velocity into L levels[2]. Given v_k with length l_k, we calculate the normalized histogram $h_k(0)$ on v_k. Then we equally divide v_k into two parts $v_k \rightarrow [v_k(1), v_k(2)]$, where both $v_k(1)$ and $v_k(2)$ are also velocity vectors with $\frac{l_k}{2}$. We calculate the normalized histogram $h_k(1)$ and $h_k(2)$ on $v_k(1)$ and $v_k(2)$, respectively, and normalize them so that $\sum h_k(1) + \sum h_k(2) = 1$. Consequently, we further equally divide $v_k(1)$ or $v_k(2)$ into two parts again and calculate the histograms in the same way.

Such process can be conducted until a predefined level is achieved. We concatenate all the histograms with predefined weights. The detailed process is demonstrated in Figure 4. Note that Figure 4 corresponds to a three-level temporal pyramid, therefore, the weight of each level is assigned with [1/4, 1/4, 1/2], where the bottom level is assigned with the highest weight [8]. For a four-level pyramid, the weight vector is [1/8,1/8,1/4,1/2].

Based on this, we can extract a velocity histogram h_k of equal length D_h with coarse-to-fine temporal resolution. The similarity between user trajectory k and k' can be either calculated with histogram intersection or Chi-Square kernel. The histogram inter-

[2]In fact, the quantization scheme can be application dependent

section and the Chi-Square kernel are:

$$K_h(k,k') = \sum_{i=1}^{D_h} \min(h_k(i), h_{k'}(i))$$
$$K_h(k,k') = \exp\left(-\frac{1}{2\varpi}\sum_{i=1}^{D_h}\frac{\left(h_k(i)-h_{k'}(i)\right)^2}{\left(h_k(i)+h_{k'}(i)\right)}\right) \quad (16)$$

where ϖ denotes the bandwidth parameter for the Chi-Square kernel. ϖ may be tuned as follows:

1. Given N trajectories, calculate the average velocity pattern distances as:

$$d_h = \frac{1}{2N^2}\sum_{k'=1}^{N}\sum_{k=1}^{N}\sum_{i=1}^{D_h}\frac{\left(h_k(i)-h_{k'}(i)\right)^2}{\left(h_k(i)+h_{k'}(i)\right)}. \quad (17)$$

2. Set the initial value ϖ_0 as $1/d_h$.
3. To acquire the best performance, we can conduct parameter tuning near the initial setting (e.g., setting candidate parameter set as: $\varpi = [0.8\varpi_0, 0.9\varpi_0, \varpi_0, 1.1\varpi_0, 1.2\varpi_0]$).

6. TRAJECTORY COMMUNITY DISCOVERY FROM MULTI-SOURCE SIMILARITY MEASUREMENT

Now we have several types of kernels representing the similarity from different information sources. The kernel set includes semantic kernel K_s, temporal kernel K_β, spatial kernel K_p and velocity kernel K_h. Based on the similarity definition in Eq. 1, we detect the trajectory communities from the trajectory database. Note that the trajectory data is collected from open environment with diversified customer behaviors, so that each trajectory does not have to be assigned with a community label. To this end, we use the dense sub-graph detection method proposed by [13] which robustly detects a set of highly connected sub-graphs (cliques) from the graph, where the nodes represent the trajectories and the weights of the vertices represent the pair-wise similarity among trajectories.

The method represents a set of vertices by a probabilistic cluster, which is a unit vector in the space of standard simplex. Then, a quadratic function is introduced to measure the average edge weight among them and the dominant set is defined as the sub-graph with the largest average edge weight. In this paper, we refer to such average edge weight as the dominance of a subgraph. More detailed as:

1. The probabilistic cluster is defined as $z \in \Delta^N$, where $\Delta^N = \{z | z \in R^N, z \geq 0, ||z||_1 = 1\}$ is the space of standard simplex and N is the total number of vertices. In fact, z is a unit mapping vector; the value of z_i, which is the i-th dimension of z, is the probability that the probabilistic cluster z contains the i-th vertex . Particularly, if $z = I_i$, whose i-th dimensional value is 1, then it represents a probabilistic cluster that contains only the i-th vertex with probability $z_i = 1$. Any i-th vertex with $z_i = 0$ is not included by the cluster.
2. The dominance of the probabilistic cluster z is defined in Eq. 18, where K is the symmetric connection matrix (i.e., the weighted combination of kernels where the diagonal element is set to 0) of the consistency graph G.

$$g(z) = z^T K z \quad (18)$$

The dense sub-graph seeking problem can be formulated as a standard quadratic optimization problem [13]. It can be solved by the replicator dynamics method, where z is the probabilistic cluster and t indicates the iteration time.

$$\max_z g(z) = z^T K z, s.t. z \in \Delta^N$$
$$z_i(t+1) = z_i(t)\frac{(Kz(t))_i}{z(t)^T K z(t)}, i = 1, ..., N \quad (19)$$

For the graph $G(\alpha)$ with N vertices, the probabilistic cluster is initialized as: $z(0) = \left\{z_i(0) = \frac{1}{N} | z(0) \in \Delta^N\right\}$. Such iteration is easy to implement and easy to calculate and according to the experiment results, it converges in 4 iterations on average for a graph with less than 100 vertices. After detecting one dense sub-graph using the above process (*i. e.*, find an optimal solution z^* by using the method in Eq. 19 [13]), we exclude the vertexes with non-zero value in z^*, and start the optimization process again on the reduced graph, where the new affinity matrix K' is the sub matrix of K. This process is repeated until the vertex set becomes empty. For each z^*, we define the average similarity of the currently detected dense sub-graph as $(z^*)^T K z^*$. The whole procedure of TODMIS is demonstrated in Algorithm 1.

Algorithm 1 TODMIS algorithm

Input: **Data:** $\mathbf{X}$, **Parameters:** α, η, λ, β_1, β_2, ϖ
Output: $z^*_{c'}, c' = 1, ..., C$

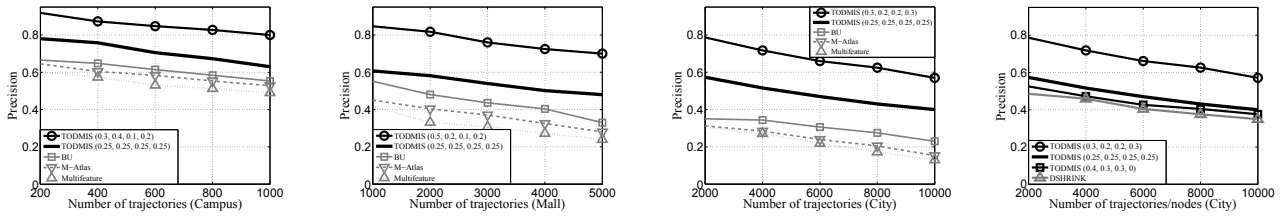
```
Construct K_s on X with parameters η and λ.
Construct K_β on X with parameters β_1 and β_2.
Construct K_p on X with GAK-IP.
Construct K_h on X with parameter κ.
Construct K_0 based on [K_s, K_β, K_p, K_h] and α.
c' = 1.
while The vertex set in G(α) is not empty do
    Get z*_{c'} by solving Eqn. 19 on K_{c'-1}.
    Exclude the vertexes with non-zero elements in z*_{c'} from G(α).
    Form K_{c'} by extracting the sub-matrix from K_{c'-1}.
    c' = c' + 1
end while
```

According to the analysis in [13], the time complexity depends on the number of edges, which is $O(N^2)$ for fully connected graph. However, for processing large trajectory database, a sparse nearest neighbor graph can be constructed instead of a fully connected graph. Therefore, the time complexity can be reduced to $O(qN)$, which is linear with respect to the number of trajectories N and the number of nearest neighbors q.

7. EXPERIMENTAL EVALUATION

7.1 Experiment setup

Real-life data sets: We conduct our experiments on three real life data sets: 1) Campus student tracking data (**Campus**): We collected 1,000 students' trajectories with time duration as long as one week in one level of a university campus building. Based on the students' real activities (ground truth data), we are able to label 40 real groups (a.k.a. trajectory community). 2) Customer shopping behavior data (**Mall**): We collected 5,000 customers' trajectories (over an observation duration of two days) in one level of a big shopping mall in Singapore. In this data set, we introduced 16 controlled groups (ground truth) to test if TODMIS can identify these groups. 3) Taxi driver tracking data (**City**)[14]: We collected 10,000 taxi drivers' trajectories (over an observation duration of one week) from a big city in China. In this data set, there exist 650 groups (ground truth from taxi company), such that each group is assigned a region to traverse by the company). The whole city has more than 20,000 road segments, around 10,000 taxis belonging to more than 100 taxi companies; each company partitions its drivers



(a) Precision in campus test (b) Precision in mall test (c) Precision in city test (d) Precision in social test

Figure 5: Precision evaluation (TODMIS performs the best in trajectory and social connection based community detection).

into several taxi groups (10 to 50 groups). The tracking records including Taxi ID, instant speed, driving direction, location, company ID, taxi type ID and group ID (associated with company ID). From the data, we observe that taxi drivers do exhibit a grouping pattern (community): given a region (described as a set of road segments), they always traverse in this region in a given time period. Thus in this experiment, we create a grid-based representation of the city (based on the road segments) and investigate the effectiveness of TODMIS by applying it on the collected trajectories of the drivers, and then comparing the identified groups with the real groups defined by the company (ground truth). To compare our detected trajectory community (trajectory-based) against a traditional social network-based clustering approach, we collected the social contact information of the above taxi drivers, consisting of the phone call (from a communicator on taxi) records including caller taxi ID, callee taxi ID, start time, end time, start location and end location.

Experiment environment: Our experiments and latency observations are conducted on a standard server (Linux), with four Intel Core Quad CPUs, Q9550 2.83 GHz and 32 GB main memory.

Baseline methods: For the trajectory data, we compare our method with BU algorithm [24], M-Atlas [16, 5] and Multifeature [1]. BU algorithm is used to detect traveling companions from streaming trajectories. M-Atlas is used to create and navigate a catalog of the mobility behaviors of a territory (GPS data). Multifeature is used to estimate common patterns of behaviors and isolates outliers in video data. For the social connection data, we compare our method with DSHRINK (one of the latest community detection algorithms), a distance-based clustering algorithm for detecting communities in incomplete information networks with missing edges [12]. The parameters for the various baseline methods will be described shortly.

Evaluation metrics: In our experiments, we use the total execution time (at different scales) to evaluate the *computational efficiency*. We utilize the precision and recall to evaluate *effectiveness*. The detailed definitions of the above evaluation metrics are explained in the experimental results. The parameters of all the kernels for TODMIS are tuned strictly according to the methods described in the previous sections.

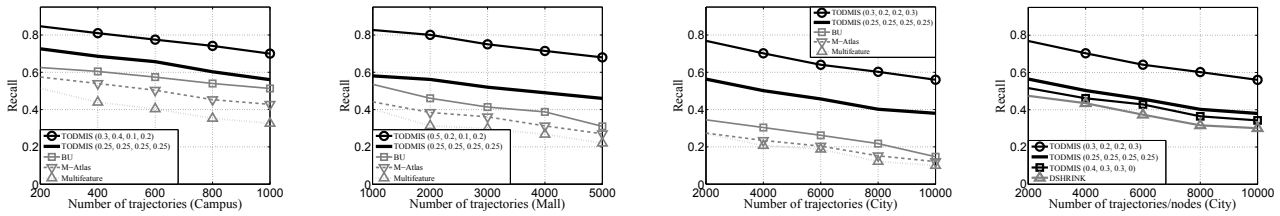
7.2 Effectiveness evaluation

In this experiment, *precision* is defined as the fraction of retrieved communities that are relevant to the search. The *recall* is defined as the fraction of the communities that are relevant to the query that are successfully retrieved. The *F1* score can be easily calculated based on *precision* and *recall*, and we omit it here.

Single kernel: We conduct experiments to validate the advantage of using multiple information sources over single information source. We compare TODMIS using average kernel weight ($\alpha_m = 0.25, m = 1, ..., 4$, which has been widely accepted in multiple-feature fusion paradigm) with TODMIS using single kernel corresponding to $\alpha_m = 1, m = 1, ..., 4$, respectively. For example, in the *Mall* setting with 1,000 trajectories, the precision of TODMIS using average weight is 0.61, which outperforms the best performance of the single kernel version (the best precision is 0.45 achieved by using semantic kernel K_s). The recall of TODMIS using average weight is 0.58, which outperforms the best recall of the single kernel version (the best recall is 0.41 achieved by using K_s). For the *City* scenario, with 2,000 trajectories, the precision of TODMIS using average weight is 0.59, which outperforms the best performance of the single kernel version (the best precision is 0.39 achieved by using velocity kernel K_h). The recall of TODMIS using average weight is 0.58, which outperforms the best recall of the single kernel version (the best recall is 0.35 achieved by using K_h). Our experiments thus provide strong evidence of the benefit of using multiple information sources.

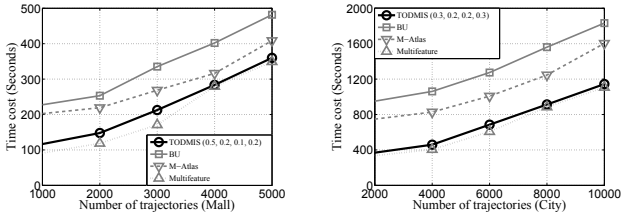
Multiple kernels: Based on the results from Figure 5 and Figure 6, TODMIS performs better precision and recall than the baseline methods. Note that even if we simplistically weigh the semantic, temporal, spatial and velocity measures *equally*, TODMIS still performs better. From the parameter tuning, we can find that different measures impact the precision and recall in different scenarios. For the *Campus* scenario, we observe that the grouping performance is more sensitive to the semantic and temporal similarity measures (rather than the spatial and velocity measures). Intuitively, this is due to the fact that groups perform different activities in different spaces (e.g., lectures in lecture halls, project work in group study rooms) and for different durations (e.g., lectures for 1.5 hours, project work for longer durations), whereas the spatial and velocity information are less discriminative (especially because the floor is very crowded). In the *Mall* scenario, the semantic similarity measure (each store is treated as a distinct semantic label) is obviously important as, intuitively, members of a group are likely to visit the same stores in a similar sequence. However, unlike the campus, we cannot perform any significant state-space reduction during post-processing, as it is highly unlikely to find two outlets of the same store in the mall. Also, similar to the *Campus* scenario, the spatial and velocity similarity measures in the *Mall* are less significant for clustering.

For the *City* scenario, the observations are quite different: for taxi traffic, the semantic information of different sites and the velocity information of drivers are more discriminative than the other two measures (although the temporal and spatial measures do play a role). Note that TODMIS is less accurate in our experiment on taxi drivers' trajectories, compared to the *Campus* and *Mall* scenarios. The reason is that even though we utilize the road segment to identify each taxi driver's trajectory, taxi drivers do not traverse the roads in groups (unlike students on campus or shoppers in the mall). But TODMIS still achieves promising effectiveness as shown in Figure 5 (c) and Figure 6 (c). In Figure 5 (d) and Figure 6 (d), we



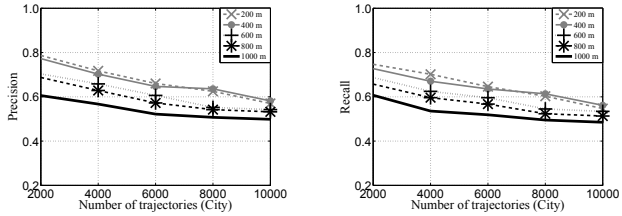
(a) Recall in campus test (b) Recall in mall test (c) Recall in city test (d) Recall in social test

Figure 6: Recall evaluation (TODMIS performs the best in trajectory and social connection based community detection).



(a) Test in mall (b) Test in city

Figure 7: Efficiency evaluation. TODMIS is efficient and scalable in different scenarios.



(a) Precision (b) Recall

Figure 8: Sensitivity evaluation. TODMIS is tolerant to different trajectory granularities.

report the experiment results of community detection based on social links (phone calls between drivers). The precision and recall tests are based on the real group (community) information from the company. It is obvious that our method, even without velocity kernel, is still better than DSHRINK (close to but not better than TODMIS without velocity kernel). From the results, we notice that the real social connection based community is actually not the same as the community information from taxi companies. On investigating the reasons for the poor performance of DSHRINK, we found that 4.5% taxi drivers actually make less than 5 calls per day and 18% taxi drivers make more out-going calls to other communities (in the same company) than to their own community. For TODMIS, we not only capture the calling behavior between drivers, but also their semantic, temporal, spatial and velocity similarities from the actual movement, which is a better indicator of the community as opposed to the call records.

7.3 Efficiency evaluation

In Figure 7, we report the total execution time of the clustering algorithm, as a function of the number of trajectories. We see that TODMIS's computational efficiency is close to Multi-feature but better than the other two baseline methods. Moreover, TODMIS shows a close-to-linear relationship between execution time and the number of trajectories.

7.4 Sensitivity evaluation

The spatial granularity of trajectory data has great impact on the clustering result/ community discovery quality. To study this, we run TODMIS on the *City* scenario, with different trajectory (spatial grid) granularities. More specifically, we experiment with different road segment lengths (where each road segment defines a sample point in the trajectory data). As shown in Figure 8, TODMIS is tolerant to different trajectory granularities (different road segment lengths). More specifically, the results for segment lengths of 200 meters and 400 meters are very similar, but the precision begins to decrease if resolution gets coarser (segment lengths of 600 meters and higher). Intuitively, we note that 55.4% of the true road segments within our test area have length between 200-400 meters, implying that larger values of segment lengths (e.g., 600 meters) causes TODMIS to miss out features distinct to each individual true segments. However, in general, we find that TODMIS is fairly robust to different trajectory granularity values.

7.5 Discussions

Outliers in trajectories: Since the trajectories are collected from real-life scenarios, it is obvious that some outliers will be included in the database. These outliers act differently with all the other trajectories within the detected communities, so they cannot be included into any of them. For example, for the taxi driver's trajectories, the taxi may encounter with traffic accident and stay in certain location for a very long time to wait for the traffic police, or some drivers may interrupt their business due to unpredictable emergency. These trajectories act differently and they are not similar from other trajectories in all the information sources. Therefore, they are automatically assigned to the "background" by our algorithm and the average similarity of the background are very low (usually less than 0.2).

Automatically learning kernel weights: Besides manually setting, the weights for different kernels can be automatically learned towards certain objective. For example, given labels describing if the trajectories belong to the same community or different communities, we can learn the kernel weights to maximize the performance of community discovery. In future, we plan to apply to techniques such as metric learning on multiple kernels [26] to learn similarity weights better.

Noise tolerance and heterogeneity: Noise is ubiquitous due to inaccurate signal sensing and localization. In our algorithm, the risk of model degradation brought by noise can be alleviated by the weighted combination of the information in multiple kernels. For example, if the noise level of spatial kernel $\mathbf{K}_p$ is unexpectedly high, the influenced can be reduced by decreasing its weight and the missing information can be complemented by other ker-

nels such as the semantic kernel. Moreover, the trajectories are collected from different people in real-life; hence, the community behavior can be influenced by many factors, such as age, nationality, culture and gender. Therefore, the compactness and average similarity levels are not consistent from community to community. Our proposed method is robust to such kind of heterogeneity, since we can choose the sub-graphs with top ranked average similarity levels as the discovered community, or we just choose the top c' sub-graphs where there is a significant drop between the average similarity level of c'-th and $(c'+1)$-th sub-graph.

8. CONCLUSIONS

In this paper, we proposed TODMIS, a trajectory-based community detection technique, consisting of three phases: 1) modeling semantic information of locations, temporal, spatial and velocity information with kernels; 2) creating a unified similarity measure via weighted combination of multiple information markers; 3) identifying communities via dense sub-graph mining. Extensive experiments were conducted on three real life data sets: customers in a shopping mall, students in a campus building and taxi drivers in a city. The results demonstrated that TODMIS outperformed other clustering algorithms in detecting the correct groups from different trajectory data.

In future work, we plan to enhance our work on trajectory-based community discovery as follows: 1) we will collect more real-life trajectory data; 2) we will attempt to construct a trajectory dictionary from the underlying data, where each element in the dictionary represents a canonical user behavior; 3) we will try to develop automated learning of kernel parameters and multi-marker weights given some level of specific community information; 4) we will study the usefulness of community-aware recommendation techniques, especially for targeted mobile advertising.

9. ACKNOWLEDGEMENTS

This research was supported by the T-SET University Transportation Center sponsored by US DoT Grant No. DTRT12-G-UTC11, the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office and National Basic Research Program of China (973 Program): 2012CB316400. The authors also thank Lionel Ni for valuable discussions and support regarding this work.

10. REFERENCES

[1] N. Anjum and A. Cavallaro. Multifeature object trajectory clustering for video analysis. *IEEE TCSVT*, 2008.

[2] M. Cuturi. Fast global alignment kernels. In *ICML'11*.

[3] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In *KDD'99*.

[4] Y. Ge, H. Xiong, Z. hua Zhou, H. Ozdemir, J. Yu, and K. C. Lee. Top-eye: top-k evolving trajectory outlier detection. In *CIKM'10*.

[5] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, C. Renso, S. Rinzivillo, and R. Trasarti. Unveiling the complexity of human mobility by querying and mining massive trajectory data. *VLDB J.*, 2011.

[6] D. Horowitz and S. D. Kamvar. The anatomy of a large-scale social search engine. In *WWW'10*.

[7] J. Huang, H. Sun, J. Han, H. Deng, Y. Sun, and Y. Liu. Shrink: a structural clustering algorithm for detecting hierarchical communities in networks. In *CIKM'10*.

[8] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR (2)'06*.

[9] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD'07*.

[10] J. Leskovec, K. J. Lang, and M. W. Mahoney. Empirical comparison of algorithms for network community detection. In *WWW'10*.

[11] X. Li, W. Hu, and W. Hu. A coarse-to-fine strategy for vehicle motion trajectory clustering. In *ICPR (1)'06*.

[12] W. Lin, X. Kong, P. S. Yu, Q. Wu, Y. Jia, and C. Li. Community detection in incomplete information networks. In *WWW'12*.

[13] H. Liu, L. J. Latecki, and S. Yan. Fast detection of dense subgraphs with iterative shrinking and expansion. *IEEE TPAMI*, 2013.

[14] S. Liu, Y. Liu, L. M. Ni, J. Fan, and M. Li. Towards mobility-based clustering. In *KDD'10*, 2010.

[15] B. Morris and M. M. Trivedi. Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. In *CVPR'09*.

[16] M. Nanni and D. Pedreschi. Time-focused clustering of trajectories of moving objects. *J. Intell. Inf. Syst.*, 2006.

[17] M. Newman. Detecting community structure in networks. *EPJ B*, 2004.

[18] R. T. Ng and J. Han. Clarans: A method for clustering objects for spatial data mining. *IEEE TKDE*, 2002.

[19] A. N. Papadopoulos. Trajectory retrieval with latent semantic analysis. In *SAC'08*.

[20] N. Pelekis, I. Kopanakis, E. E. Kotsifakos, E. Frentzos, and Y. Theodoridis. Clustering uncertain trajectories. *Knowl. Inf. Syst.*, 2011.

[21] C. Piciarelli, G. L. Foresti, and L. Snidaro. Trajectory clustering and its applications for video surveillance. In *AVSS'05*.

[22] C. Shi, P. S. Yu, Y. Cai, Z. Yan, and B. Wu. On selection of objective functions in multi-objective community detection. In *CIKM'11*.

[23] S. Spaccapietra, C. Parent, M. L. Damiani, J. A. F. de Macedo, F. Porto, and C. Vangenot. A conceptual view on trajectories. *Data Knowl. Eng.*, 2008.

[24] L. A. Tang, Y. Zheng, J. Yuan, J. Han, A. Leung, C.-C. Hung, and W.-C. Peng. On discovery of traveling companions from streaming trajectories. In *ICDE'12*.

[25] M. Vlachos, D. Gunopulos, and G. Kollios. Discovering similar multidimensional trajectories. In *ICDE'02*.

[26] S. Wang, Q. Huang, S. Jiang, and Q. Tian. S^3MKL: scalable semi-supervised multiple kernel learning for real-world image applications. *IEEE trans. on multimedia*, 14(4), 2012.

[27] X. Wang, K. T. Ma, G. W. Ng, and W. E. L. Grimson. Trajectory analysis and semantic region modeling using nonparametric hierarchical bayesian models. *IJCV*, 2011.

[28] X. Wang, K. Tieu, and E. Grimson. Learning semantic scene models by trajectory analysis. In *ECCV (3)'06*.

[29] T. Xiang and S. Gong. Beyond tracking: Modelling activity and understanding behaviour. *IJCV*, 2006.

[30] Z. Yan, D. Chakraborty, C. Parent, S. Spaccapietra, and K. Aberer. Semantic trajectories: Mobility data computation and annotation. *ACM TIST*, 2012.

[31] H. Zhu, J. Su, and O. H. Ibarra. Trajectory queries and octagons in moving object databases. In *CIKM'02*.