iBOAT: Isolation-Based Online Anomalous Trajectory Detection

Chao Chen, Daqing Zhang, *Member, IEEE*, Pablo Samuel Castro,
Nan Li, Lin Sun, Shijian Li, and Zonghui Wang

***Abstract*—Trajectories obtained from Global Position System (GPS)-enabled taxis grant us an opportunity not only to extract meaningful statistics, dynamics, and behaviors about certain urban road users but also to monitor adverse and/or malicious events. In this paper, we focus on the problem of detecting anomalous routes by comparing the latter against time-dependent historically "normal" routes. We propose an online method that is able to detect anomalous trajectories "on-the-fly" and to identify which parts of the trajectory are responsible for its anomalousness. Furthermore, we perform an in-depth analysis on around 43 800 anomalous trajectories that are detected out from the trajectories of 7600 taxis for a month, revealing that most of the anomalous trips are the result of conscious decisions of greedy taxi drivers to commit fraud. We evaluate our proposed isolation-based online anomalous trajectory *(iBOAT)* through extensive experiments on large-scale taxi data, and it shows that *iBOAT* achieves state-of-the-art performance, with a remarkable performance of the area under a curve (AUC) $\geq$ 0.99.**

***Index Terms*—Anomalous trajectory detection, Global Positioning System (GPS) traces, isolation, online.**

I. Introduction

THE RECENT advances in sensing, communication, storage, and computing have revolutionized the way that we interact with the world. Many of these interactions leave behind digital traces of our actions, providing a rich source of information for understanding social behaviors and community dynamics in different contexts [42]. This results in improved services in various areas, including public safety, urban planning, and transportation management [7], [16], [19], [29], [39].

Global Positioning Systems (GPS) have become a tool present in most vehicles for localization and navigation. The traces left behind by GPS-enabled vehicles provide us with an unprecedented window into the dynamics of a city's road network. This information has been analyzed to uncover traffic patterns [26], city dynamics [37], driving directions [33], a city's "hot spots" [8], [38], finding vacant taxis around a city [31], and good taxi operation patterns [21], [28].

Recent years have witnessed an increasing interest in automatically detecting anomalous trajectories [6], [14], [20]. Although several aspects of abnormality have been used for automatic detection by previous works, a few of them have been analyzed with respect to the practical applications, which they may serve. We would like to use one potential application to motivate the work presented in this paper.

Example: Many passengers are victims of fraud caused by greedy taxi drivers who overcharge passengers by deliberately taking unnecessary detours [1]. The detection of these fraudulent behaviors is essential to ensure high-quality taxi service. These frauds are currently detected by manual inspection from experienced staff, based on complaints from passengers. This is rather costly and not very effective as most frauds are not even noticed by passengers if they are unfamiliar with the city. Given that anomalous traces usually deviate significantly from "normal" traces, it is possible to automatically detect them by comparing them against a large collection of historical trajectories.

To effectively support the application, a successful anomaly detection method should possess the following characteristics.

1) **Accurate classifications**: This implies that the method should have high detection accuracy and a low false-alarm rate.
2) **Subtrajectory specificity**: In addition to labeling trajectories as anomalous, it can inform which parts or *subtrajectories* are responsible for the trajectory's anomalousness.
3) **Real-time response**: This implies that the method should detect anomalous trajectories in real time. Alerts can be provided once anomaly is detected while the trip is still ongoing.
4) **Characterizing the anomaly degree**: This implies that the method should provide a score quantifying the degree

Manuscript received July 20, 2012; revised October 25, 2012; accepted December 25, 2012. This work was supported in part by the Institut Mines-Télécom through the "Futur et ruptures" Program and the Systematic Paris-Region Smart City "AQUEDUC" Program, by the National Science Foundation of China under Grant 60975043, Grant 60803109, and Grant 61105043, by JiangsuSF under Grant BK2011566, and by the Ministry of Science and Technology of the People's Republic of China through the 836 Program under Grant 2011AA010104. The work of C. Chen was supported by the Government of China. The Associate Editor for this paper was S. Tang. *(Corresponding Author: D. Zhang.)*

C. Chen is with the Department of Network and Services, Institut Mines-Télécom/Telecom SudParis, Evry 91011, France, and also with the Department of Computer, Pierre and Marie Curie University, Paris 75005, France (e-mail: chao.chen@it-sudparis.eu).

D. Zhang and L. Sun are with the Department of Network and Services, Institut Mines-Télécom/Telecom SudParis, Evry 91011, France (e-mail: daqing.zhang@it-sudparis.eu; lin.sun@it-sudparis.eu).

P. S. Castro was with the Department of Network and Services, Institut Mines-Télécom/Telecom SudParis, Evry 91011, France. He is now with Google Pittsburgh, Pittsburgh, PA 15206 USA (e-mail: pablosamuelcastro@gmail.com).

N. Li is with the National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China, and also with the School of Mathematical Sciences, Soochow University, Suzhou 215006, China (e-mail: lin@lamda.nju.edu.cn).

S. Li and Z. Wang are with the College of Computer Science, Zhejiang University, Hangzhou 310027, China (e-mail: shijianli@zju.edu.cn; zjuzhwang@zju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2013.2238531

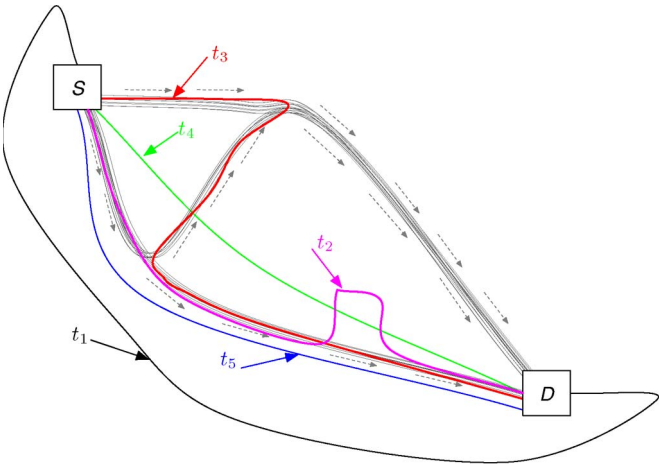


Fig. 1. Example taxi trajectories between S and D.

of anomalousness for each trajectory. This score can be used to rank a collection of trajectories.

A set of trajectories are considered "normal" with respect to a particular travel itinerary (i.e., from a specified point to another). We must then specify source (S) and destination (D) areas, and we consider only those trajectories traveling from S to D.

Consider the three groups of "normal" trajectories between S and D, along with five anomalous trajectories (t_1 through t_5), displayed in Fig. 1. The anomalous trajectories are labeled so because they are infrequent and differ from the majority of other trajectories. Not only those trajectories that follow a completely different route (t_1, t_4, t_5) should be considered anomalous but also those that detour for a part of the trajectory (t_2, t_3). The anomalous trajectories can be long detours made by greedy taxi drivers (t_1 and t_2 in Fig. 1), or they can be short cuts or new routes taken by experienced drivers (t_4 and t_5 in Fig. 1). Detecting these anomalous trajectories is no trivial task due to the following challenging issues.

1) As shown in Fig. 1, there may be many different normal routes between S and D, and these clusters are usually with different densities and separated from each other. Traditional anomaly detection techniques [6], [14], [20], which are based on differences in distance or density, may be difficult to identify all the anomalies.
2) Multiple normal routes also mean different driving distances. If we model driving distance, it is not able to discover those anomalies whose driving distance is close to that of the normal trajectories (such as t_3 and t_5).
3) Anomalous trajectories can be diverse. Similar to t_1, t_2, t_3, t_4, and t_5 in Fig. 1, they are regarded as anomalous due to quite different reasons. Then, it is not straightforward to characterize them with a single method.
4) The concept of anomalous trajectory might drift over time because the road network may change (i.e., newly built or blocked roads). Hence, it is important to be able to capture these changes and incorporate them into the model. Moreover, GPS traces often suffer from the low-sampling-rate problem since GPS devices usually send data at a low and changing frequency.

In this paper, we aim to propose a novel anomalous trajectory detection method that addresses the given four challenges. First, we extract valid taxi rides from all the taxi GPS traces, divide the city map into grid cells of equal size, group all the taxi rides crossing the same S–D cell pair, and augment and represent each taxi trajectory in each S–D pair as an ordered sequence of traversed cell symbols. In such a way, the problem of anomalous trajectory detection is converted to that of finding anomalous trajectories from all the trajectories with the same S–D cell pair. Second, for all the taxi trajectories between a certain S–D cell pair, we define those trajectories that are "few" and "different" from the normal trajectory clusters as anomalies. We then propose an isolation-based online anomalous trajectory (*iBOAT*) detection method that exploits the property that anomalies are susceptible to a mechanism called isolation [27]. Third, we perform an empirical evaluation comparing *iBOAT* and other state-of-the-art methods with real-world taxi GPS data. Finally, we show how *iBOAT* can be used to effectively support real-world applications. In summary, the main contributions of this paper include the following.

1) We present an *iBOAT* detection method that successfully addresses all the challenges listed earlier while still possessing the four characteristics previously mentioned.
2) We evaluate *iBOAT* with real-world GPS traces collected from 7600 taxis for one month. We demonstrate the remarkable accuracy of our method, its ability to identify which subtrajectories are anomalous, and its low computational cost. We also show that *iBOAT* outperforms the state-of-the-art anomalous trajectory detection methods.
3) After detecting the anomalous trajectories, we perform an analysis revealing that most of the anomalous trips are the result of conscious decisions of greedy taxi drivers to commit fraud. In addition, we further provide evidence to deny possible excuses that some cunning drivers may use.

The rest of this paper is organized as follows. We begin by reviewing related work in Section II. After introducing related background in Section III, we describe our method in Section IV. An empirical evaluation along with an analysis of its differences with related methods are presented in Section V. We analyze the different types of anomalous trajectories and provide evidence to deny possible excuses for fraudulent behaviors in Section VI. Finally, we present concluding remarks in Section VII.

II. Related Work

Here, we will review the related work, which can be categorized into two groups. The first group consists of the work on analyzing or exploiting GPS traces for purposes other than anomaly detection, whereas the second group focuses on anomaly detection methods that are related to this paper.

A. Trajectory Pattern Mining

There have been many studies on mining GPS traces for a number of different applications. Liao *et al.* [24] and Patterson *et al.* [30] studied the problem of predicting a user's mode of transportation and daily routine to provide reminders

when needed, whereas Li *et al.* [21] and Liu *et al.* [28] uncovered taxi drivers' operating patterns. Giannotti *et al.* [15] provided concise descriptions of frequent movement patterns in terms of space and time. Alvares *et al.* [3] developed a model to enrich the raw trajectory data with semantic geographical information and discovered knowledge based on *semantic* trajectory data. Other studies show how to predict the route and the destination based on historical GPS traces [12], [18], [40], in addition to providing driving directions by exploiting taxi drivers' knowledge [33]. GPS traces have also been used for uncovering interesting "hot spots" and recommending travel routes for tourists [10], [36], [38], for uncovering the attractiveness of commercial centers [34], for passengers searching for vacant taxis [31], or for classifying the social functions of different regions in a city [29].

B. Anomalous Trajectory Detection

In the literature, some solutions of anomalous trajectory detection have already been reported, each addressing certain aspects of abnormality. For instance, Lee *et al.* [20] split a trajectory into various partitions (at equal intervals), and a hybrid of distance- and density-based approaches was used to classify each partition as anomalous or not; however, as we previously mentioned, solely using distance and density can fail to correctly classify some trajectories as anomalous. Bu *et al.* [6] presented an outlier detection framework for monitoring outliers over continuous trajectory streams, whose key idea was to build local clusters upon trajectory streams and to detect outliers by a cluster join mechanism. Ge *et al.* [14] studied a similar problem of detecting evolving trajectory outliers, and they computed the anomaly score based on evolving direction and density of trajectories. Somewhat related, but addressing a different problem, Li *et al.* [23] identified outlier road segments by detecting drastic changes between current data and historical trends. Their approach detected what can be labeled as *global* anomalous events. These were events that affect many taxis; thus, their method would not be able to detect anomalous behaviors on an *individual* level. Balan *et al.* [4] reported trajectories with extremely long traveling distances as anomalous; as we previously mentioned, this rather simplistic approach may fail to detect other types of anomalous behaviors. Ge *et al.* [13] identified fraudulent taxi trajectories by using a model combining two forms of evidence: distance and density characteristics. Specifically, they first computed the independent components (using independent component analysis) of a set of trajectories, and they computed the *coding cost* (which is essentially the entropy) of a trajectory's independent components. Once this was done, they determined the expected distance for the most common routes, and they computed how much a trajectory's distance differs from the norm. These two pieces of evidence were combined using the Dempster–Schafer theory. Finally, some recent studies have used learning methods to identify anomalous trajectories [2], [22], [25], [32]. However, these last methods usually required training data, which are expensive to label. After reviewing existing studies on anomalous trajectory detection, it is not difficult to find that we are investigating a different problem from previous studies. That is, given all the taxi trajectories between a certain S–D pair, our objective is to discover those few that take very *different* routes from the majority.

Fig. 2. Traces of a taxi in Hangzhou city during a month, where red or blue indicates the taxi is occupied or vacant.

Most of these methods identify anomalous trajectories based on their physical distance to "normal" clusters or their orientations [5], [17]. Based on the idea of isolating anomalies [27], our previous work [9], [35] proposed a method that identifies trajectories as anomalous when they follow paths that are rare with respect to historical trajectories. This paper builds on them but differs in the following respects. First, we introduce a novel anomaly scoring method that considers both the anomalous subtrajectory and the number of trajectories "supporting" it. Anomalous trajectories with a longer anomalous subtrajectory and less support would be ranked higher. Second, the effect of the anomaly threshold and the size of the set of historical trajectories on the detection performance are investigated. This paper allows developers to trade off between the detection accuracy and the cost (i.e., computation time and memory). Finally, motivations behind the anomalous behaviors are analyzed. Different applications corresponding to this motivation could be developed, leveraging the proposed *iBOAT* method.

III. Preliminaries and Problem Definition

A taxi's GPS trace consists of a sequence of timestamped GPS points (i.e., latitude/longitude, the estimated speed, and vacant/occupied state) generated by a GPS device. Our data set consists of the GPS trajectories for 7600 taxis in Hangzhou, China, where each GPS record is received at a rate of around once per minute. Fig. 2 shows the trajectories for one taxi during a month; the red lines indicate when the taxi is occupied, whereas the blue lines indicate when it is vacant. In this paper, we will only use occupied trajectories since fraud detection is one of the motivations for this paper, and fraud can only be committed with a passenger.

Definition 1: A *trajectory* t consists of a sequence of points $\langle p_1, p_2 \ldots, p_n \rangle$, where $p_i \in \mathbb{R}^2$ is the physical location (i.e., latitude/longitude). We will use t_i to reference position i in t, and for any $1 \leq i < j \leq n$, $t_{i \to j}$ denotes the *subtrajectory* $\langle p_i, \ldots, p_j \rangle$.

Points p_i exist in a continuous domain; therefore, directly dealing with them is difficult. To mitigate this problem, we assume that we have access to a finite decomposition of the

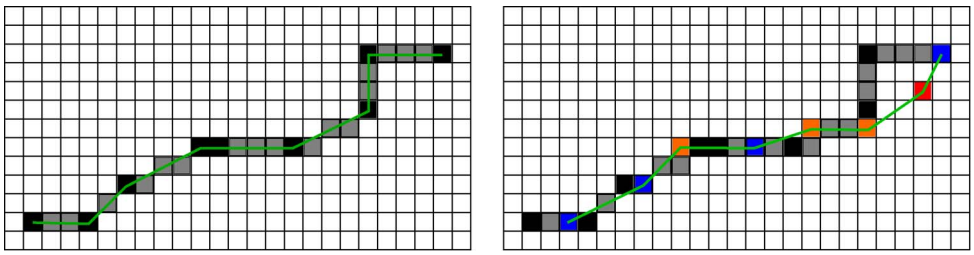


Fig. 3. (Left) Example of a trajectory with augmented cells. (Right) Comparing existing trajectory with a new trajectory.

area of interest. Specifically, we decompose the city area into a matrix G of grid cells, and we define $\rho : \mathbb{R}^2 \rightarrow G$ as a function that maps locations to grid cells. The criterion for choosing the grid cell size is to ensure the accuracy of the anomalous trajectory detection while maximizing the grid cell size. We experimented with different grid cell sizes and found that 250 m $\times$ 250 m is the biggest grid size with the set detection accuracy.

Definition 2: Mapped trajectory $\bar{t}$, which is obtained from trajectory t, consists of a sequence of cells $\langle g_1, g_2 \ldots, g_n \rangle$, where for all $1 \leq i \leq n$, $g_i \in G$, and $\bar{t}_i = \rho(t_i)$. We will write $g \in \bar{t}$ when $\bar{t}_i = g$ for some $1 \leq i \leq n$.

Henceforth, we will only deal with mapped trajectories; therefore, we will drop the *mapped* qualifier. Because of the rate at which GPS entries are received and the small size of our grid cells, the mapped points (black squares in Fig. 3) may not be adjacent, thereby leaving gaps. We *augment* all the trajectories to ensure that there are no gaps in the trajectories by (roughly) following the line segment (green line in left panel) between the two cells in question and "coloring" the cells underneath (gray cells in figure). Whereas the original trajectory consisted only of the black grids in Fig. 3, the augmented trajectory consists of both the black and gray grids.

Let $\mathbb{T}$ denote the set of all mapped and augmented trajectories. Define function pos : $\mathbb{T} \times G \rightarrow \mathbb{N}^+$, given that trajectory t and element g returns the first index in t that is equal to g, as follows:

$$\text{pos}(t,\, g) = \begin{cases} \arg\min_{i \in \mathbb{N}^+} \{t_i = g\}, & \text{if } g \in t \\ \infty, & \text{otherwise.} \end{cases} \tag{1}$$

For example, if $t = \langle g_1, g_2, g_3, g_5, g_3, g_8 \rangle$, then $\text{pos}(t, g_3) = 3$ and $\text{pos}(t, g_7) = \infty$.

We will be comparing an ongoing trajectory against a set of trajectories T. Because of the low sampling rates, two taxis following the same path may have points mapping to disjoint cells. In the right panel of Fig. 3, we display the augmented trajectory from the left panel, along with a new trajectory (colored squares and green line). Some of the grid cells of the new trajectory fall on the augmented path (blue squares), whereas others fall in "empty" grid cells (orange and red cells). Because of the simplicity of the augmentation method, there is the possibility that the augmented path was not completely accurate; therefore, we must account for this type of error. If a grid cell of the new trajectory is adjacent to one of the augmented cells, we consider it as if it were along the same path (orange cells), whereas if it is not adjacent to any augmented cells, we consider it as following a different path (red cell). For this purpose, we define $N : G \rightarrow \mathcal{P}(G)$ as a function that returns the adjacent neighbors of a grid cell (each nonborder grid cell will thus have nine neighbors, including itself). For a grid cell g and trajectory t, we let $N(g) \in t$ denote the fact that at least one of the neighbors of g is in t, and $\text{pos}(t, N(g))$ return the first index in t that is equal to one of the *neighbors* of g. For instance, given the grid cells in Fig. 4 and sample trajectory $t = \langle g_1, g_2, g_3, g_4, g_{11}, g_{12} \rangle$, we would obtain $\text{pos}(t, N(g_9)) = 2$ (since $g_9 \in N(g_2)$).

g1	g2	g3	g4	g5	g6
g7	g8	g9	g10	g11	g12
g13	g14	g15	g16	g17	g18
g19	g20	g21	g22	g23	g24
g25	g26	g27	g28	g29	g30
g31	g32	g33	g34	g35	g36

Fig. 4. Sample trajectory used to illustrate a cell's neighbors.

Problem statement: We say that subtrajectory t is anomalous with respect to T (and the fixed S–D pair) if the path that it follows rarely occurs in T. Given a fixed S–D pair (S, D) with a set of trajectories T between them and an ongoing trajectory $t = \langle g_1, g_2, \ldots, g_n \rangle$ going from S to D, we would like to verify whether t is *anomalous* with respect to T. Furthermore, we would like to identify which parts of the trajectory are anomalous.

Definition 3: We define function hasPath : $\mathcal{P}(\mathbb{T}) \rightarrow \mathcal{P}(\mathbb{T})$ (where $\mathcal{P}(X)$ is the power set of X) that returns the set of trajectories that contain all of the points in t in the correct order. Note, however, that the points need not be *sequential*; it suffices that they appear in the same order, as given in the following:

$$\text{hasPath}(T,\, t) = \left\{ t' \in T \,\middle|\, \begin{array}{l} (i)\, \forall 1 \leq i \leq n.\ N(g_i) \in t' \\ (ii)\, \forall 1 \leq i < j \leq n. \\ \text{pos}\,(t', N(g_i)) < \text{pos}\,(t', N(g_j)) \end{array} \right\}. \tag{2}$$

For instance, if $T = \{t1, t2, t3\}$, where $t1 = \langle g_1, g_2, g_3, g_4, g_5, g_8, g_9, g_{10} \rangle$, $t2 = \langle g_1, g_2, g_4, g_5, g_6, g_8, g_{10} \rangle$, and $t3 = \langle g_1, g_3, g_4, g_3, g_6, g_8, g_{10} \rangle$, and an ongoing trajectory $t = \langle g_1, g_2, g_5, g_8 \rangle$, then $\text{hasPath}(T, t) = \{t1, t2\}$. Given these definitions, we can specify when two trajectories are identical, given our augmentation method.

Definition 4: Given threshold $0 \leq \theta \leq 1$, trajectory t is $\boldsymbol{\theta}$*-anomalous* with respect to a set of trajectories T if

$$\text{support}(T,\, t) = \frac{|\text{hasPath}(T,\, t)|}{|T|} < \theta. \tag{3}$$

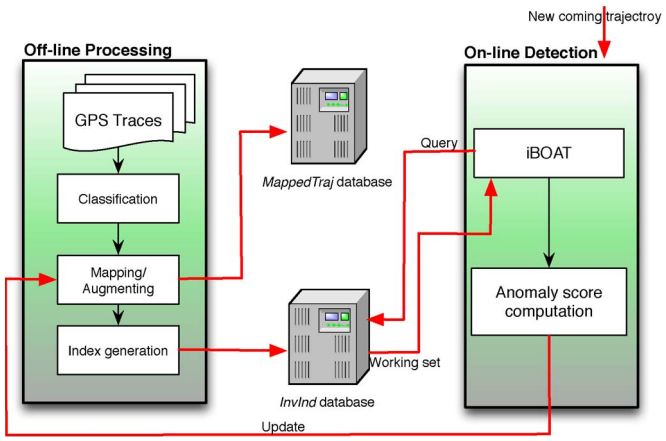


Fig. 5. Overview of our approach.

IV. Isolation-Based Online Anomalous Trajectory Detection

Having defined the necessary preliminaries, we are ready to present our method for anomalous trajectory detection. The process is split into an offline preprocessing phase and an online detection phase (see Fig. 5). In the offline phase, we receive a set of historical trajectories, which we classify and index using a sophisticated but highly efficient method. This allows us to respond to the online algorithm's queries in real time. In the online phase, we process a series of incoming GPS points from each occupied taxi and provide an indication as to whether each point is anomalous or not. Once this ongoing trajectory is completed, we add it to our historical database.

A. Offline Preprocessing

The offline phase is in charge of collecting and classifying a set of historical trajectories, which will be used to determine "normal" routes between an S–D pair. These historical trajectories must be accessible in an efficient manner to provide a real-time response.

We begin by grouping the trajectories according to S–D pairs and the time of occurrence. It is important to separate trajectories according to the time of occurrence since the "normalcy" of routes may depend on traffic patterns. We index the set of historical trajectories using a triple $\langle \text{sg}, \text{eg}, \text{time} \rangle$, where sg is the starting grid cell, eg is the end grid cell, and time is the time at which the trajectory occurred. Note that to avoid the unnecessarily fine granularity of time, we divide time into coarser bins. Each set indexed by a triple may contain both normal and anomalous trajectories. Once the trajectories have been classified, we map and augment them. For each trajectory, we store the resulting mapped grid cells (in the correct order) in a record in the MappedTraj database, and we index the records by their (unique) trajectory number and the time of occurrence.

To determine the anomalousness of a new mapped GPS point, we must be able to access *all* trajectories that contain this mapped point (or some point in its neighborhood) in the same time bin. Using MappedTraj for this purpose would be terribly inefficient as it would imply searching through all trajectories for each new point. Instead, we make use of the inverted index mechanism [41] for fast retrieval of relevant trajectories. For this mechanism, we maintain a second database where we maintain a record for each possible grid cell; the elements of each record are trajectory–position pairs, indicating the trajectories where the indexing grid cell appears, along with its position in that trajectory. For instance, consider the following trajectories:

$$t_1 : g_1 \to g_5 \to g_8 \to g_{10}$$
$$t_2 : g_1 \to g_2 \to g_5 \to g_8 \to g_5 \to g_9$$
$$t_3 : g_2 \to g_8 \to g_9.$$

In the inverted index database InvInd, the record indexed by grid cell g_1 will be $\text{InvInd}(g_1) = \{(t_1, 1), (t_2, 1)\}$, the record indexed by g_5 will be $\text{InvInd}(g_5) = \{(t_1, 2), (t_2, 3), (t_2, 5)\}$, and the record indexed by g_9 will be $\text{InvInd}(g_9)\{(t_2, 6), (t_3, 3)\}$. Thus, if a new GPS point maps to g_9, by accessing $\text{InvInd}(g_9)$, we will immediately know that this grid cells occurs in trajectories t_2 and t_3.

We now have an efficient mechanism for accessing the trajectories that contain a particular grid cell. In Section V, we will incrementally use this (as new GPS points arrive) to determine the anomalousness of an ongoing trajectory.

B. iBOAT

Our *iBOAT* detection method is based on the idea of *isolating* trajectories. Anomalous (sub-)trajectories will be isolated from the majority of routes, whereas normal trajectories will be *supported* by a large number of trajectories. The less support a trajectory has, the higher its degree of anomalousness would be. In [35], Zhang *et al.* determine the anomalousness of a trajectory once the trajectory is completed. This is unfortunate, since it prevents one from providing alerts to the passenger while a trajectory is ongoing. On the other hand, using purely density-based methods as described earlier will most likely result in inaccurate classifications. We aim to overcome this problem by using an *adaptive working window* that provides us with historical contexts to better determine the anomalousness of the incoming trajectory. We will use the definition of θ-anomalousness presented in Section III to describe our proposed algorithm.

1) Basic idea: The basic idea of *iBOAT* is to maintain an *adaptive working window* of the latest incoming GPS points to compare against the set of historical trajectories. As a new incoming point is added to the *adaptive working window*, the set of historical trajectories is pruned by removing any trajectories that are *inconsistent* with the subtrajectory in the *adaptive working window*. New points continue to be added to the *working window* as long as the support of the subtrajectory in the *adaptive working window* is above θ. If the support drops below θ, then the *adaptive working window* is reduced to contain only the latest GPS point. We outline this approach in Algorithm 1.

We maintain a *working set* of trajectories (initially equal to T) and an *adaptive working window* w. After $i-1$ entries are received, our partial trajectory t (*adaptive working window*) consists of $\langle p_1, p_2, \ldots, p_{i-1} \rangle$, and we have a *working set* T_{i-1}. Upon arrival of point p_i, we map it to grid cell g_i (line 8) and

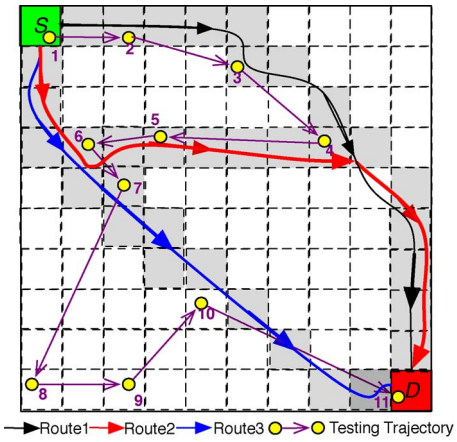


Fig. 6. Running example for *iBOAT*.

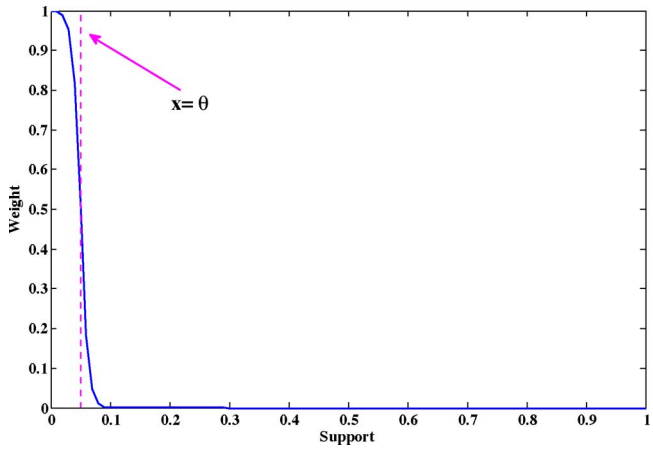


Fig. 7. Weighting function σ.

concatenate g_i to the *adaptive working window* w (line 9). We then compute support(T_{i-1}, w) (line 10). If its value is less than θ, the trajectory points contained in the *adaptive working window* are said to have anomalies; then, point p_i is considered anomalous. Therefore, it is added to the set of anomalous points χ (line 13), and we reset the *working set* (line 14) and the *adaptive working window* (line 15); otherwise, we set $T_i =$ hasPath(T_{i-1}, w) (line 11). This procedure is repeated as long as the trip does not reach the destination. Note that $T_0 = T$ and that, every time an anomalous point is encountered, the *working set* is reset to the original trajectory set T. This resetting is what enables our adaptive algorithm to accurately detect anomalous subtrajectories in real time with finer granularity than the fixed window approach (with $k > 1$). Additionally, by reducing the *working set* with each incoming point, the adaptive approach has a computational advantage over the fixed window approach.

To illustrate the process, we will use a running example, as shown in Fig. 6. We assume there are three common routes that drivers take when delivering passengers from S to D. There are 100 taxi drivers who have taken *Route 1* (in black), 200 taxi drivers have taken *Route 2* (in red), and 150 drivers have taken *Route 3* (in blue). The test trajectory is depicted using the numbered yellow circles and the purple line (indicating the order of arrival of the points). We can immediately see that, although the test trajectory visits only "common" cells in the initial part, it does so in reverse order between points 4 and 7. In the beginning, the *adaptive working window* will grow to contain points $\langle g_1, g_2, g_3, g_4\rangle$ since this subtrajectory has enough support. However, when g_5 is added to the *adaptive working window*, the support of this subtrajectory drops below the threshold; thus, g_5 is considered as an anomalous point, and the new *adaptive working window* contains only g_5. The size of the *adaptive working window* would not increase (only containing the single latest GPS point) until receiving g_7; now, the *adaptive working window* will be $\langle g_6, g_7\rangle$. Again, the *working window* will shrink to contain only a single point throughout the anomalous section ($\langle g_8, g_9, g_{10}\rangle$). When the trajectory is completed, *iBOAT* will return $\chi = \{g_5, g_6, g_8, g_9, g_{10}\}$ as the set of anomalous points.

We can also consider a simple variant of *iBOAT*: maintaining a fixed-sized window. In this approach, the *sliding window* consists only of the most recent k points. Specifically, given a set of trajectories T and an ongoing trajectory $t = \langle p_1, p_2, \ldots, p_n\rangle$, we verify whether the last k-sized subtrajectory from t occurs with enough frequency in T to determine if it is anomalous. Note that, when $k = 1$, we have the density method used for comparison in [25]. Following the example in Fig. 6, we have the following results for different values of k:

$$\chi = \begin{cases} \{g_8, g_9\}, & \text{if } k = 1 \\ \{g_5, g_6, g_8, g_9, g_{10}\}, & \text{if } k = 2 \\ \{g_5, g_6, g_7, g_8, g_9, g_{10}, g_{11}\}, & \text{if } k = 3. \end{cases}$$

Note that the size of χ depends on the value of k. For the same anomalous trajectory, the larger k is, the larger χ would be. While the size of χ for $k = 2$ and *iBOAT* is closer to that of the real anomaly segments, it produces larger χ when k increases, leading to excessive counting of the anomaly segments. However, in the case of $k = 1$, the size of χ is much smaller than that of the real anomaly segments. This explains why the anomaly detection algorithm with $k = 2$ and *adaptive window* outperforms that with $k = 1$ or $k \geq 3$. However, in some specific cases, as shown in Section V, the proposed *iBOAT* method with an adaptive window can detect certain anomalous trajectories that the fixed sliding window methods are not able to detect, making *iBOAT* the most effective anomaly detection approach.

2) Anomaly score: As the trajectory is ongoing, we maintain an anomalous score, which will be used to provide alerts and to rank the trajectories once they are completed. Intuitively, a trajectory with smaller *support* and longer anomalous distance should be ranked higher; therefore, we compute this score based on the length of the anomalous subsection, and the density in each anomalous subsection, rather than only summing the length of each anomalous part [9]. We weigh the support according to function σ, which is a logistic function (shown in Fig. 7), as follows:

$$\sigma(x) = \frac{1}{1 + e^{\lambda(x-\theta)}}.$$

Here, λ is a temperature parameter and θ is the aforementioned threshold. For our experiments, we choose $\lambda = 150$. This function will assign a larger weight to very low supports, and the weight will drop to zero for values above θ. The advantage

of using this weighting function is that it *smoothes* the cutoff point imposed by the chosen threshold θ; in a sense, it plays a similar role as sigmoid functions in neural networks. For each incoming point p_i, we compute its score as shown in line 17 of Algorithm 1. We add the score for the previous point to the distance just traveled multiplied by the weighted support (note that we also do for the fixed window approach).

Given the way the ongoing score is computed, once the trajectory is completed after n steps, we have the final score as given by the following equation, which is a weighted sum of the distance between points:

$$\text{score} = \text{score}(n) = \sum_{i=2}^{n} \frac{1}{1+e^{\lambda(\text{support}(i)-\theta)}} \text{dist}(p_i, p_{i-1}) \quad (4)$$

where dist : $\mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is the standard sphere distance between two points.

Algorithm 1 *iBOAT* with adaptive window

Input: incoming trajectory $- t = \langle p_1, p_2, \ldots \rangle$
T - set of mapped and augmented historical trajectories
θ - anomaly threshold
Output: score; χ- set of anomalous points
1: $\chi \leftarrow \emptyset$ // initialization
2: $T_0 \leftarrow T$
3: $i \leftarrow 0$ // Position in incoming trajectory
4: $w \leftarrow \emptyset$ // Adaptive window from t
5: score$(0) \leftarrow 0$
6: **while** the testing trajectory is not completed **do**
7: $i \leftarrow i + 1$
8: $g_i = \rho(p_i)$
9: $w \leftarrow w \cdot g_i$
10: support$(i) = |\text{hasPath}(T_{i-1}, w)|/|T_{i-1}|$
11: $T_i \leftarrow \text{hasPath}(T_{i-1}, w)$ // working set reduced
12: **if** $support(i) < \theta$ **then**
13: $\chi \leftarrow \chi \cup p_i$
14: $T_i \leftarrow T$ // reset the working set
15: $w \leftarrow g_i$
16: **end if**
17: score$(i) = \text{score}(i-1) + \sigma(\text{support}(i)) *$ dist(p_{i-1}, p_i)
18: **end while**

V. Empirical Evaluation

Here, we provide an empirical evaluation and analysis of our proposed approaches. All the experiments are run in Matlab on an Intel Xeon W3500 PC with 12-GB RAM running Windows 7.

A. Data sets

Out of the 7.35 million of trajectories extracted from the one-month GPS records of 7600 taxis, we picked nine S–D pairs[1] (T-1 through T-9) with sufficient trajectories between them (at least 450, but on average over 1000) and asked volunteers to *manually* label whether the trajectories are anomalous or not. On average, about 5.1% of the trajectories are labeled as anomalous. We summarize the information for each data set in Table I.

[1]The S and D areas are twice as big as the regular grid cells.

TABLE I
Data Sets Used in Our Experiments

	#Trajectories	#Anomalousness(%)
T-1	453	15(3.3%)
T-2	1494	57(3.8%)
T-3	528	43(8.1%)
T-4	946	58(6.1%)
T-5	1018	68(6.7%)
T-6	1369	72(5.3%)
T-7	1310	67(5.1%)
T-8	1216	71(5.8%)
T-9	1254	24(1.9%)

B. Evaluation Criteria

A classified trajectory will fall into one of four scenarios: 1) True positive (TP), when an anomalous trajectory is correctly classified as anomalous; 2) false positive (FP), when a normal trajectory is incorrectly classified as anomalous; 3) false negative (FN), when an anomalous trajectory is incorrectly classified as normal; and 4) true negative (TN), when a normal trajectory is correctly classified as normal. The TP rate (TPR) measures the proportion of correctly labeled anomalous trajectories, and it is defined as

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (5)$$

The FP rate (FPR) measures the proportion of *false alarms* (i.e., normal trajectories that are labeled as anomalous) and is defined as

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \quad (6)$$

A perfect classifier will have TPR $= 1$ and FPR $= 0$. In a receiver operating characteristic (ROC) [11] curve, we plot FPR on the x-axis and TPR on the y-axis, which indicates the tradeoff between false alarms and accurate classifications. By measuring the area under a curve (AUC), we can quantify this tradeoff.

C. Results

To test *iBOAT*, we selected trajectory t as an ongoing trajectory from data set T and used both *iBOAT* and *fixed window* approaches with $\theta = 0.05$. In Section V-D1, we will discuss the effect that the choice of θ has on the performance. In the left panel of Fig. 8, we display the output of our method for a test trajectory from T-6, where we plot the set of trajectories $T - \{t\}$ in light blue; for the test trajectory (t), the anomalous points are drawn in red and the rest (normal points) in dark blue. As shown, our method can accurately detect which parts of a trajectory are anomalous and which are normal. In the middle and right panels of Fig. 8, we plot support$(T - \{t\}, t)$ [see (3)] and the score [see (4)] for the ongoing trajectory t. We can see that the value of support is a clear indication of when

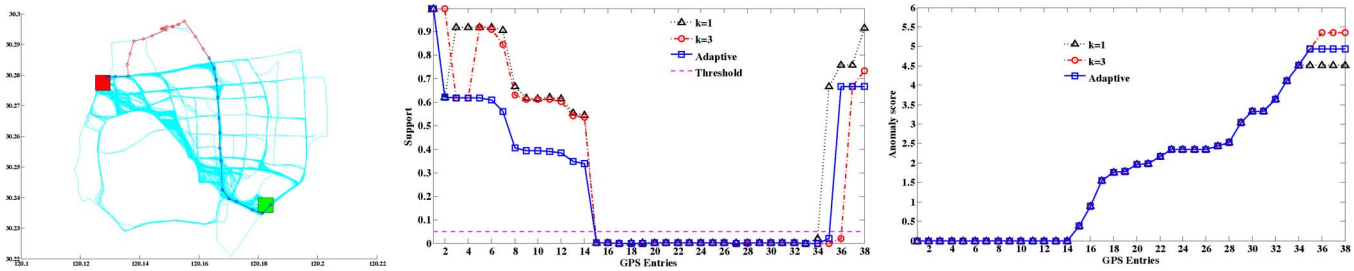


Fig. 8. (Left) Detected anomalous subtrajectories from T-6 using *iBOAT*. (Middle) Plot of ongoing support. (Right) Plot of ongoing score.

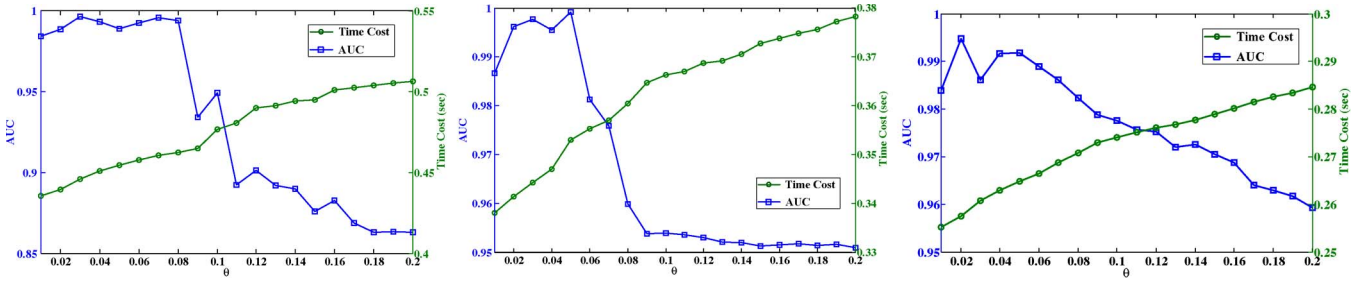


Fig. 9. (Blue) AUC value and (green) average time under varying θ. (Left) T-2. (Middle) T-6. (Right) T-7.

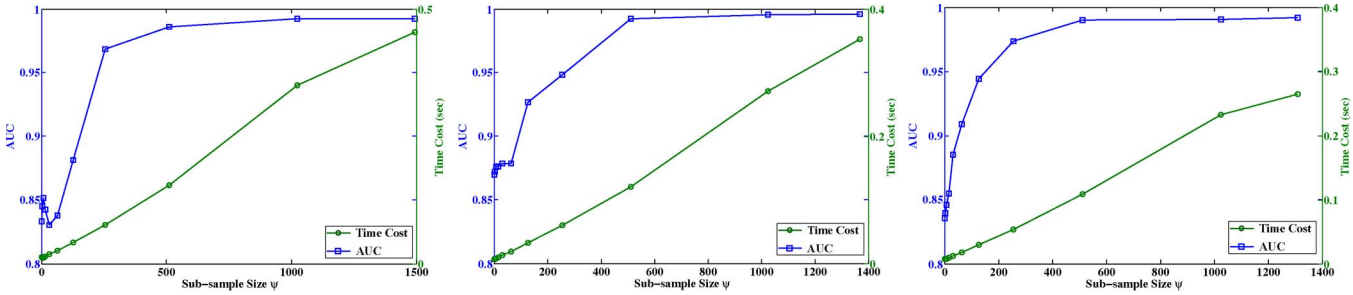


Fig. 10. (Blue) AUC value and (green) average time under varying n. (Left) T-2. (Middle) T-6. (Right) T-7.

trajectories become anomalous and that there is little difference between the different variants of *iBOAT*. Note, however, that there is a trailing *lag* for the fixed window approach, which is equal to k. This is because the last anomalous point in an anomalous subtrajectory will be included in the following k subtrajectories. Although setting $k = 1$ will solve the lag problem, this minimal window size contains no contextual information on the trajectory and will therefore have poor prediction quality. This was observed in [35] (therein referred to as the density method) and will be evident in the following figures.

D. Varying Parameters

To better understand *iBOAT*, we conduct experiments to study its performance (in terms of running time and accuracy) under different parameter settings. We choose the three largest data sets (T-2, T-6, and T-7). We begin by varying the choice of θ in Section V-D1. In Section V-D2 we vary the size of the data sets; specifically, for each of the three data sets, we choose $n = \{2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, \ldots\}$ trajectories randomly to serve as the historical trajectories. We measure the *average time*, which is the average amount of processing time per *completed* trajectory.

1) Varying θ: Since θ is the threshold value for determining anomalousness, it is important to investigate its effect on the performance of the algorithm. We study the effect on performance when θ ranges between 0.01 and 0.2.

In Fig. 9, we plot the AUC and the average time for different values of θ. We can see that θ should not be set any higher than 0.1 since beyond this the performance would significantly decrease. The average time increases with θ. This is because as θ becomes larger, the working set is reset more frequently, resulting in larger working sets on average. We can also see that our choice of $\theta = 0.05$ is reasonable as it has good accuracy with low average time.

2) Varying n: It is evident that the average time will be longer with larger values of n since there are more comparisons necessary for each incoming GPS point; on the other hand, if n is too small, then more trajectories will be isolated since there are fewer trajectories to support it. It is thus important to investigate how many trajectories are necessary between two endpoints for *iBOAT* to return accurate results. In Fig. 10, we plot the AUC value and average time for different values of n. We can see that *iBOAT* achieves remarkable performance even with a small subsample size; the figure suggests that data sets have around 500 trajectories to guarantee a reasonable performance.

TABLE II
AUC VALUES OF THE DIFFERENT ALGORITHMS

	$k=1$	$k=2$	$k=3$	*Adaptive*
T-1	0.9635	0.9904	0.9811	0.9985
T-2	0.9367	0.9902	0.9887	0.9952
T-3	0.8140	0.9733	0.9152	0.9962
T-4	0.9005	0.9586	0.9575	0.9890
T-5	0.9323	0.9885	0.9821	0.9967
T-6	0.9227	0.9912	0.9840	0.9952
T-7	0.8806	0.9853	0.9849	0.9937
T-8	0.9438	0.9739	0.9724	0.9937
T-9	0.9788	0.9991	0.9987	0.9995

The given analysis suggests that if we maintain a fixed number of trajectories, we can ensure good performance at a low computing and storage cost. Trajectories can be maintained in a first-in–first-out (FIFO) queue; as new trajectories are coming and processed, they can replace the oldest trajectories in the queue. This technique can also *capture* the change of distribution of trajectories.

E. Adaptive Versus Fixed Window Approach

We display the AUC values of the different approaches on the nine data sets in Table II. While the density approach ($k = 1$) has the worst performance, our proposed *iBOAT* method slightly outperforms the fixed sliding window approach with $k = 2$, and the fixed sliding window method with $k = 2$ is better than that with $k = 1$ and $k \geq 3$. As explained in Section IV-B, the fixed sliding window method with $k = 1$ is worse than that of $k = 2$ because the anomalies detected are *fewer* than the actual anomalies, whereas the fixed sliding window method with $k = 3$ is worse than that of $k = 2$ because the anomalies detected are much *more* than the actual anomalies. The performance of the fixed window approach with $k = 2$ and that of the adaptive approach are nearly identical. This is because, for the anomalous sections, the adaptive approach ends up using a window with a size of 2, just as $k = 2$. The advantage of the fixed window approach is that it requires a very small amount of memory for real-time anomalous detection, whereas the adaptive method requires memory proportional to the size of the longest "normal" subtrajectory. In practice, this difference is negligible. In the following, we will use an example to demonstrate that the adaptive approach has an advantage over the fixed window approach due to its use of longer historical "contexts."

In Fig. 11, we display an anomalous trajectory that "switches" from one normal route to another. The fixed window method with $k = 2$ is not able to detect this anomalous switch. Going from point 19 to point 20 seems normal since this sequence occurs in route A, and going from point 20 to point 21 also seems normal since it occurs in route B. On the other hand, *iBOAT* would maintain the entire route up to the point when the driver switches routes and would immediately detect it as an anomalous point. Although this example is specific to window sizes equal to 2, similar situations (with longer overlaps between routes) will produce a similar effect.

F. iBOAT Versus iBAT

iBAT is a recent anomaly detection method introduced in [35] that is similar to our approach. To determine whether a trajectory is anomalous, *iBAT* picks cells from the testing trajectory at random to split the collection of trajectories into those that contain the cell and those that do not. This process is repeated until the trajectory is isolated or until there are no more cells in the trajectory. Usually the number of cells required to isolate anomalous trajectories will be much less than the number of cells in the trajectory. This isolation procedure is repeated a number of times, and $\mathbb{E}(n(t))$, i.e., the average number of cells required to isolate a trajectory, is used to compute the score, which is proportional to $2^{-\mathbb{E}(n(t))}$.

Our proposed method is a clear improvement over *iBAT* on two levels. First of all, we are able to determine which *parts* of a trajectory are anomalous, in contrast to *iBAT* that only classifies *full* trajectories as anomalous. Second of all, our method works in *real time*. We can detect anomalous sections as soon as they occur and do not require a full trajectory as an input.

In Fig. 12, we show an example where a road block has forced a taxi to retrace its path and search for another route to its destination. We focus on the first part of the trajectory where the taxi retraces its steps. In the right panel of Fig. 12, we can see that the support is accurately identifying the anomalous section of the trajectory. We determined what anomalous ranking (based on the scores) both methods assign this partial trajectory in comparison with all other trajectories.[2] Out of 1418 trajectories, *iBOAT* ranked this trajectory in the 48th place, whereas *iBAT* ranked it in the 831st place. Furthermore, *iBAT* assigned this trajectory a score of 0.4342, which is below their usual threshold of 0.5. Thus, while *iBAT* is unable to detect that this trajectory is anomalous, *iBOAT* has ranked it among the top 3% of anomalous trajectories, as well as identifying which part is anomalous. The reason *iBAT* fails in this example is that their method does not take the *order* that the points appear in into consideration; despite the fact that the taxi is retracing its steps and actually going *away* from the destination, it is only visiting "normal" grid cells.

Now, consider the hypothetical example in Fig. 13, which highlights the differences in the two scoring functions. In this simple situation, the value $\mathbb{E}(n(t))$ for *iBAT* is just the expected number of times that their algorithm must pick cells before an anomalous cell (in red) is picked. This is essentially a Bernoulli trial with "success" probability p equal to the proportion of anomalous cells to total number of cells in the trajectory. It is well known that the expected number of trials before reaching success in a Bernoulli trial is given by $1/p$. Let n be the number of cells in the straight line between S and D; then, trajectories of the form on the left will have $2n-2$ anomalous cells and $5n-4$ total cells, whereas trajectories of the form on the right will have $2n-2$ anomalous cells and $2n+2$ total cells. It follows that, for trajectories of the form on the left, $\mathbb{E}(n(t)) = (5n-4)/(2n-2) \to 5/2 \Rightarrow score \approx 0.1768$, whereas for trajectories of the form on the right, $\mathbb{E}(n(t)) = (2n+2)/(2n-2) \to 1 \Rightarrow score = 0.5$. Thus, *iBAT* will qualify trajectories of the form on the right as more anomalous than those on the left. This runs contrary to intuition, which would perceive trajectories such as the one on the left at least

[2] A higher ranking means a higher degree of anomalousness.

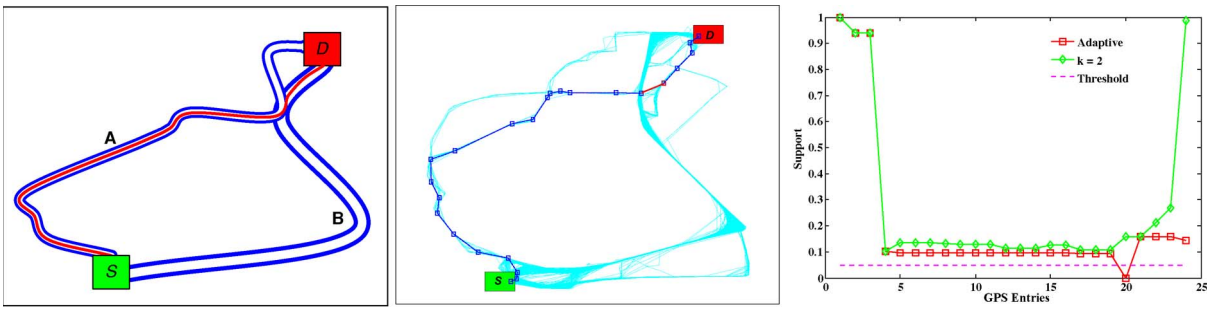


Fig. 11. Situation the fixed window method ($k = 2$) fails to classify as anomalous. Two normal routes (route A and B) are in dark blue; an anomalous trajectory (in red) switches from route A to route B at their intersection. (Left) Illustration of situation. (Middle) Real trajectories. (Right) Ongoing support from *iBOAT*.

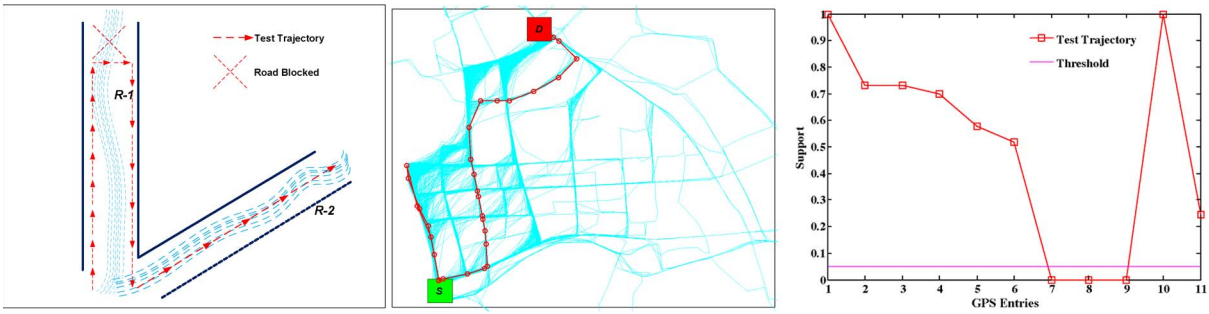


Fig. 12. Trajectory where the taxi had to retrace its path due to a blocked route. (Left) Illustration of situation. (Middle) Real trajectory. (Right) Ongoing support from *iBOAT*.

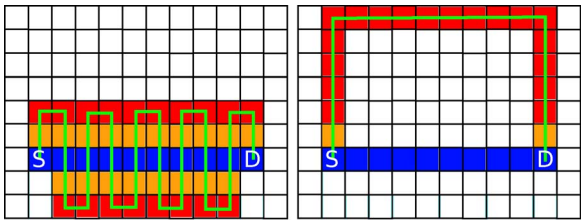


Fig. 13. Two anomalous trajectories of different types. The normal trajectory between S and D is in blue, cells adjacent to normal cells are in orange, and anomalous cells are in red.

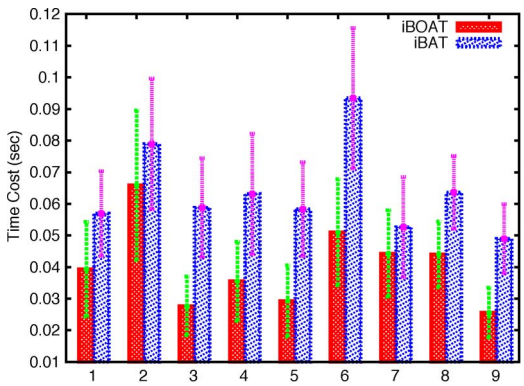


Fig. 14. Running times of *iBOAT* and *iBAT* on all the data sets.

as anomalous as the one on the right, given that the path taken is much longer and they are clearly taking longer routes than necessary. *iBOAT*'s scoring method, on the other hand, would assign the left trajectory an anomalous score around 33% higher than the one on the right.

Finally, we compared the running time of both algorithms on all the data sets, and we display the results in Fig. 14. We computed the running time for checking each trajectory in each data set and averaged over the size of the data set. Although *iBAT* will usually check fewer grid cells than *iBOAT* (since one anomalous cell is enough to classify the trajectory as anomalous), *iBAT* is based on random cell selections; therefore, they must average over m runs; as in [35], we set $m = 50$. We can see that *iBOAT* is consistently faster than *iBAT* on all data sets.

VI. Fraud Behavior Analysis

Here, we first perform an in-depth statistical study of detected anomalous trajectories. Then, we further provide evidence to deny possible excuses for fraud behaviors because some cunning taxi drivers may use detour reasons such as traffic accidents on roads as excuses.

A. Statistical Study

One main motivation for this paper is fraud detection and the ability to alert passengers to fraudulent behaviors. Travel distance and time are two crucial parameters to judge if a certain taxi trajectory is a long detouring trip committed by fraudulent behaviors. Thus, here, we perform an analysis of the anomalous trajectories to attempt to discover whether anomalous behaviors are the result of conscious decisions to commit fraud, by visualizing where most of the anomalous trips begin and comparing the average distance and travel time of anomalous routes with that of normal routes.

For this analysis, we collected around 441 million records in March 2010, and detected about 438 000 anomalous trajectories out of 7.35 million trips. This provides us with an opportunity to perform a statistical analysis of the anomalous trajectories,

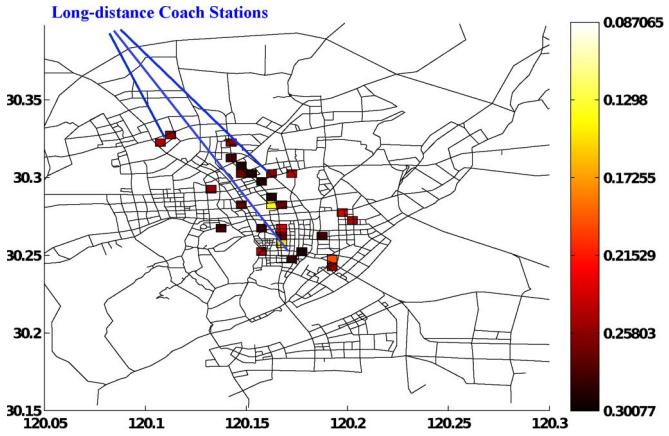


Fig. 15. Areas where most of the anomalous trips began.

in the hope of uncovering common characteristics of the trajectories and driving "trends" of those responsible for anomalous behaviors.

In Fig. 15, we display the areas where most of the anomalous trips began. We can see that many of the places are long-distance coach stations, where tourists would generally arrive. It is not surprising that they are responsible for a large fraction of the anomalous trajectories. This provides strong evidence that anomalous behaviors are conscious decisions. Note that it is possible that some passengers who are not familiar with the city cannot provide the detailed address of destination. This might have a certain impact on choosing the best route for drivers. However, as we group the historical trajectories with vague S–D (in an area with around 500 m $\times$ 500 m), and judge the ongoing trajectory by comparing it with historical ones with the same S–D it will not thus cause problems when the system shows the reasonably longer trajectory traveled to the unfamiliar passengers.

In most research revolving around detecting anomalous taxi driving behaviors, one is mainly interested in detecting fraudulent activities. We believe that many of these fraudulent trips will take passengers along routes that are much longer than what is considered normal. Given our database of historical trajectories, we can determine the length of the longest normal trip between S and D; we can safely say that an anomalous trip is *detouring* if the trip distance is longer than this maximal distance. For an S–D pair, we denote $\max D$ and $\min D$ as the maximal and minimal lengths among the normal trips. It may be the case that a longer trip is actually a faster route, placing in doubt whether the driver's actions were fraudulent. We could try to determine $\max T$ and $\min T$ for the traveling time taken between two points, but due to varying traffic conditions, these values have high variability. Because of this, for each S–D pair, we compute the mean time among the normal trajectories μ_T and the standard deviation σ_T. We then define our boundaries as $\max T = \mu_T + \sigma_T$ and $\min T = \mu_T - \sigma_T$. In Table III, we display the distribution of the anomalous trips with respect to these classifications. We can see that over 60% of the anomalous trajectories are taking longer time and distance than the maximal normal trajectories, clearly suggesting that fraud is one of the main motivating factors behind anomalous taxi driving behaviors.

TABLE III
DISTRIBUTION OF ANOMALOUS TRAJECTORIES WITH RESPECT TO TRAVELING DISTANCE AND TIME

Trip length	Travel time $[0, minT)$	$[minT, maxT]$	$(maxT, \infty)$
$[0, minD)$	0.0013	0.0137	0.0117
$[minD, maxD]$	0.0062	0.1063	0.0881
$(maxD, \infty)$	0.0045	0.1522	0.6162

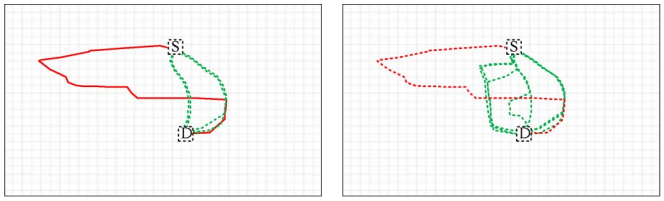


Fig. 16. Avoiding excuses for taxi driving fraud detection. (Left) (red solid) anomalous trajectory is compared with (green dashed) previous trips of the same driver. (Right) Anomalous trajectory is compared with (green dashed) trajectories in the same time slot.

B. Deny Possible Excuses

From the evidence provided in Table III, we can see that a large proportion of detected anomalous trajectories are actually due to detours. Some cunning drivers who take a detour may argue that: 1) they are unfamiliar with this area; or 2) unexpected car accidents or heavy traffic occurred. Sometimes passengers request a detour in order to pick up a friend or avoid a traffic jam. In these cases, the passenger will not complain, even when the system shows the longer detour trajectory to them.

To justify the driver's excuses, more evidence needs to be provided. To deal with the first excuse, if an anomalous trajectory is detected, we can get all previous trajectories of the corresponding driver to verify whether he truly has had little previous experience driving through this area. Note that one taxi may be operated by more than one driver; therefore, a mechanism for detecting driver shift change may be necessary. This is an interesting problem in itself but outside of the scope of this paper. For the second excuse, we can find all the trajectories that took place around the same time and area to check whether there is some traffic disturbance. In Fig. 16, we give an illustrative example. Suppose that we detect an anomalous trajectory (solid red line in the left panel of Fig. 16). We compare it with the driver's previous trips (dashed green lines) between the same S and D and can verify that this driver has experience driving between these two points. In parallel, we recall all the trips (dashed green lines in the right panel of Fig. 16) that happened around that same time slot since time-of-day has impact on the occurrence of anomalous routes. Since, in this example, we can see that many other drivers did not detour, it is unlikely that there is a traffic disturbance. Having discredited both types of excuses, we can be more confident in our assessment of fraud.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a new algorithm for fast *real-time* detection of anomalous trajectories obtained from GPS devices equipped in taxis. Rather than using time and distance to directly judge whether a test trajectory is anomalous or not,

we compare it against a set of sampled historical trajectories with the same S–D pair. In addition to classifying completed trip trajectories as anomalous or normal, *iBOAT* can work with ongoing trajectories and can determine which parts of a trajectory are responsible for its anomalousness. We validated *iBOAT* on a large data set of taxi GPS trajectories recorded over a month and found that our method achieved excellent performance (AUC ≥ 0.99 for all data sets), which is comparable to *iBAT*'s performance; however, we demonstrated a number of examples that highlight *iBOAT*'s advantage over *iBAT* and the sliding window method. We further showcased *iBOAT*'s use for fraudulent behavior analysis. The result suggests that most anomalous trajectories are in fact due to fraud. We also provide evidence to deny possible excuses for fraud behaviors.

In the future, we plan to broaden this paper in several directions. First, we plan to explore using statistical approaches to enhance detection performance and data processing efficiency. Second, we also plan to develop a real-life anomalous trajectory detection system with the proposed method. Third, to address the issue that some S–D pairs may not have enough samples, we would like to either cluster S and/or D areas in a *principled* way to "combine" trajectories from different S–D pairs, or simply collect more historical data, or partition the map into different grid sizes. Finally, we would like to conduct further analysis on the GPS traces obtained to better understand the motivations and characteristics of fraudulent activities.

ACKNOWLEDGMENT

The authors would like to thank Prof. Z.-H. Zhou for his suggestions.

REFERENCES

[1] C. Leocha, "NYC taxi drivers overcharge passengers $ 8.3 million," [Online]. Available: http://www.consumertraveler.com/today/nyc-taxi-drivers-overcharge-passengers-8-3-millioin/

[2] N. Abe, B. Zadrozny, and J. Langford, "Outlier detection by active learning," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2006, pp. 504–509.

[3] L. O. Alvares, V. Bogorny, B. Kuijpers, J. A. F. de Macedo, B. Moelans, and A. Vaisman, "A model for enriching trajectories with semantic geographical information," in *Proc. 15th ACM Int. Symp. Adv. Geograph. Inf. Syst.*, 2007, pp. 22:1–22:8.

[4] R. K. Balan, K. X. Nguyen, and L. Jiang, "Real-time trip information service for a large taxi fleet," in *Proc. 9th Int. Conf. Mobile Syst. Appl. Serv.*, 2011, pp. 99–112.

[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2000, pp. 93–104.

[6] Y. Bu, L. Chen, A. W.-C. Fu, and D. Liu, "Efficient anomaly monitoring over moving object trajectory streams," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 159–168.

[7] F. Calabrese, M. Colonna, P. Lovisolo, D. Parata, and C. Ratti, "Real-time urban monitoring using cell phones: A case study in Rome," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 141–151, Mar. 2011.

[8] H.-W. Chang, Y.-C. Tai, and J. Y.-J. Hsu, "Context-aware taxi demand hotspots prediction," *Int. J. Bus. Intell. Data Mining*, vol. 5, no. 1, pp. 3–18, Dec. 2010.

[9] C. Chen, D. Zhang, P. S. Castro, N. Li, L. Sun, and S. Li, "Real-time detection of anomalous taxi trajectories from GPS traces," in *Proc. 8th Int. ICST Conf. Mobile Ubiquitous Syst.*, 2012, pp. 63–74.

[10] Z. Chen, H. T. Shen, and X. Zhou, "Discovering popular routes from trajectories," in *Proc. 27th IEEE Int. Conf. Data Eng.*, 2011, pp. 900–911.

[11] T. Fawcett, "An introduction to roc analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

[12] J. Froehlich and J. Krumm, "Route prediction from trip observations," presented at the Soc. Automotive Engineers (SAE) World Congr., Detroit, MI, 2008, Paper 2008-01-0201.

[13] Y. Ge, H. Xiong, C. Liu, and Z.-H. Zhou, "A taxi driving fraud detection system," in *Proc. IEEE Int. Conf. Data Mining*, 2011, pp. 181–190.

[14] Y. Ge, H. Xiong, Z.-H. Zhou, H. Ozdemir, J. Yu, and K. C. Lee, "Top-EYE: Top-k evolving trajectory outlier detection," in *Proc. 19th ACM Int. Conf. Inf. Knowl. Manage.*, 2010, pp. 1733–1736.

[15] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory pattern mining," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2007, pp. 330–339.

[16] M. Gonzalez, C. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, Jun. 2008.

[17] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognit. Lett.*, vol. 24, no. 9/10, pp. 1641–1650, Jun. 2003.

[18] J. Krumm and E. Horvitz, "Predestination: Where do you want to go today?" *Computer*, vol. 40, no. 4, pp. 105–107, Apr. 2007.

[19] D. Lazer, A. S. Pentland, L. Adamic, S. Aral, A. L. Barabasi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, T. Jebara, G. King, M. Macy, D. Roy, and M. V. Alstyne, "Life in the network: The coming age of computational social science," *Science*, vol. 323, no. 5915, pp. 721–723, Feb. 2009.

[20] J.-G. Lee, J. Han, and X. Li, "Trajectory outlier detection: A partition-and-detect framework," in *Proc. 24th IEEE Int. Conf. Data Eng.*, 2008, pp. 140–149.

[21] B. Li, D. Zhang, L. Sun, C. Chen, S. Li, G. Qi, and Q. Yang, "Hunting or waiting? discovering passenger-finding strategies from a large-scale real-world taxi dataset," in *Proc. 9th IEEE Int. Conf. PERCOM Workshops*, 2011, pp. 63–68.

[22] X. Li, X. Li, J. Han, S. Kim, and H. Gonzalez, "ROAM: Rule- and motif-based anomaly detection in massive moving object data sets," in *Proc. 7th SIAM Int. Conf. Data Ming*, 2007, pp. 273–284.

[23] X. Li, Z. Li, J. Han, and J.-G. Lee, "Temporal outlier detection in vehicle traffic data," in *Proc. 25th IEEE Int. Conf. Data Eng.*, 2009, pp. 1319–1322.

[24] L. Liao, D. Patterson, D. Fox, and H. Kautz, "Learning and inferring transportation routines," *Artif. Intell.*, vol. 171, no. 5/6, pp. 311–331, Apr. 2007.

[25] Z. Liao, Y. Yu, and B. Chen, "Anomaly detection in GPS data based on visual analytics," in *Proc. IEEE Symp. Visual Anal. Sci. Technol.*, 2010, pp. 51–58.

[26] M. Lippi, M. Bertini, and P. Frasconi, "Collective traffic forecasting," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discov. Databases: Part II*, 2010, pp. 259–273.

[27] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Trans. Knowl. Discov. Data*, vol. 6, no. 1, pp. 3:1–3:39, Mar. 2012.

[28] L. Liu, C. Andris, and C. Ratti, "Uncovering cabdrivers' behavior patterns from their digital traces," *Comput. Environ. Urban Syst.*, vol. 34, no. 6, pp. 541–548, Nov. 2010.

[29] G. Pan, G. Qi, Z. Wu, D. Zhang, and S. Li, "Land-use classification using taxi GPS traces," *IEEE Trans. Intell. Transp. Syst.*, 2012, to be published.

[30] D. J. Patterson, L. Liao, D. Fox, and H. Kautz, "Inferring high-level behavior from low-level sensors," in *Proc. 5th Int. Conf. Ubiquitous Computi.*, 2003, pp. 73–89.

[31] S. Phithakkitnukoon, M. Veloso, C. Bento, A. Biderman, and C. Ratti, "Taxi-aware map: Identifying and predicting vacant taxis in the city," in *Proc. Ambient Intell.*, 2010, pp. 86–95.

[32] R. R. Sillito and R. B. Fisher, "Semi-supervised learning for anomalous trajectory detection," in *Proc. Brit. Mach. Vis. Conf.*, 2008, pp. 1035–1044.

[33] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang, "T-drive: Driving directions based on taxi trajectories," in *Proc. 18th SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2010, pp. 99–108.

[34] Y. Yue, H. D. Wang, B. Hu, Q. Q. Li, Y. G. Li, and A. G. Yeh, "Exploratory calibration of a spatial interaction model using taxi GPS trajectories," *Comput., Environ. Urban Syst.*, vol. 36, no. 2, pp. 140–153, Mar. 2012.

[35] D. Zhang, N. Li, Z.-H. Zhou, C. Chen, L. Sun, and S. Li, "iBAT: Detecting anomalous taxi trajectories from GPS traces," in *Proc. 13th Int. Conf. Ubiquitous Comput.*, 2011, pp. 99–108.

[36] V. Zheng, B. Cao, Y. Zheng, X. Xie, and Q. Yang, "Collaborative filtering meets mobile recommendation: A user-centered approach," in *Proc. 24th AAAI Conf. Artif. Intell.*, 2010, pp. 236–241.

[37] Y. Zheng, Y. Liu, J. Yuan, and X. Xie, "Urban computing with taxicabs," in *Proc. 13th Int. Conf. Ubiquitous Comput.*, 2011, pp. 89–98.

[38] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proc. 18th Int. Conf. World Wide Web*, 2009, pp. 791–800.
[39] X. Zhou and H. Mahmassani, "Dynamic origin-destination demand estimation using automatic vehicle identification data," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 105–114, Mar. 2006.
[40] B. D. Ziebart, A. L. Maas, A. K. Dey, and J. A. Bagnell, "Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior," in *Proc. 10th Int. Conf. Ubiquitous Comput.*, 2008, pp. 322–331.
[41] J. Zobel and A. Moffat, "Inverted files for text search engines," *ACM Comput. Surv.*, vol. 38, no. 2, pp. 1–55, 2006.
[42] D. Zhang, B. Guo, and Z. Yu, "The emergence of social and community intelligence," *Computer*, vol. 44, no. 7, pp. 21–28, 2011.

Chao Chen received the B.Sc. and M.Sc. degrees in control science and control engineering from Northwestern Polytechnical University, Xi'an, China, in 2007 and 2010, respectively. He is currently working toward the Ph.D. degree with Pierre and Marie Curie University, Paris, France, and with Institut Mines-Télécom/Telecom SudParis, Evry, France.

In 2009, he worked as a Research Assistant with Hong Kong Polytechnic University, Kowloon, Hong Kong. His research interests include pervasive computing, social network analysis, and data mining from large-scale taxi data.

Daqing Zhang (M'11) received the Ph.D. degree from the University of Rome La Sapienza, Rome, Italy, and the University of L'Aquila, L'Aquila, Italy, in 1996.

He is currently a Professor with the Institut Mines-Télécom/Telecom SudParis, Evry, France. His research interests include context-aware computing, Ambient Assisted Living, large-scale data mining, and urban computing. All of his research has been motivated by practical applications in digital cities, mobile social networks, and elderly care. He is the author of more than 160 referred journal and conference papers.

Dr. Zhang is the Associate Editor for four leading journals, including the *ACM Transactions on Intelligent Systems and Technology*. He has been a frequent Invited Speaker at various international events on ubiquitous computing and has also served as a General or Program Chair for many conferences.

Pablo Samuel Castro received the Ph.D. degree from McGill University, Montreal, QC, Canada, in 2011.

He was previously a Postdoctoral Researcher with Institut Mines-Télécom/Telecom SudParis, Evry, France. He is currently a Software Engineer with Google Pittsburgh, Pittsburgh, PA. His research interests include fully and partially observable Markov decision processes, reinforcement learning, artificial intelligence, and ubiquitous computing.

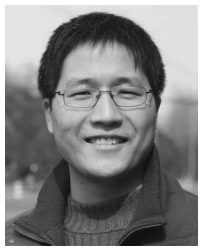
Nan Li received the M.Sc. degree in computer science from Nanjing University, Nanjing, China, in 2008. He is currently working toward the Ph.D. degree with the Department of Computer Science and Technology, Nanjing University.

Since 2008, he has been a faculty member with the School of Mathematical Sciences, Soochow University, Suzhou, China. His research interests include machine learning, data mining, and ubiquitous computing.

Mr. Li was a co-recipient of the Grand Prize (Open Category) in the 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining Competition.

Lin Sun received the B.S. and M.S. degrees from Northwestern Polytechnical University, Xi'an, China. He is currently working toward the Ph.D. degree with Institut Mines-Télécom/Telecom SudParis, Evry, France.

His research interests include pervasive/ubiquitous computing and context-aware systems, particularly in activity recognition with multimodality sensors and reality mining from large-scale real-life taxi Global Positioning System data.



Shijian Li received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2006.

He is currently an Associate Professor with the College of Computer Science, Zhejiang University. He is the author of over 60 papers. His research interests include ubiquitous computing and social computing.

Zonghui Wang was born in March 1979. He received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2007.

He is currently a Lecturer with the College of Computer Science and Engineering, Zhejiang University. His research interests include intelligence transportation, cloud computing, and computer architecture.